Kinetis KM34 Sub-Family Reference Manual

Supports: MKM34Z256VLL7 MKM34Z256VLQ7

Document Number: KM34P144M75SF0RM
Rev. 2, May 2015

freescale™

Contents

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

Chapter 1
About This Document

Chapter 2
Introduction

Chapter 6
Reset and Boot

Chapter 7
Power Management

Chapter 8
Security

Chapter 9
Debug

Chapter 10
Pinouts and Packaging

Chapter 11
Port Control and Interrupts (PORT)

Chapter 12
System Integration Module (SIM)

Chapter 13
Inter-Peripheral Crossbar Switch (XBAR)

Chapter 14
Memory Mapped Arithmetic Unit (MMAU)

Chapter 15
System Mode Controller (SMC)

Chapter 16
Low-Leakage Wakeup Unit (LLWU)

Chapter 17
Power Management Controller (PMC)

Chapter 18
Reset Control Module (RCM)

Chapter 19
Miscellaneous Control Module (MCM)

Chapter 20
Bit Manipulation Engine (BME)

Chapter 21
Micro Trace Buffer (MTB)

Chapter 22
Crossbar Switch Lite (AXBS-Lite)

Chapter 23
Peripheral Bridge (AIPS-Lite)

Chapter 24
DMA Controller Module

Chapter 25
Direct Memory Access Multiplexer (DMAMUX)

Chapter 26
Memory Protection Unit (MPU)

Chapter 27
Flash Memory Module (FTFA)

Chapter 28
Flash Memory Controller (FMC)

Chapter 29
Watchdog Timer (WDOG)

Chapter 30
External Watchdog Monitor (EWM)

Chapter 31
Analog Front End (AFE)

Chapter 32
Analog-to-Digital Converter (ADC)

Chapter 33
Comparator (CMP)

Chapter 34
Voltage Reference (VREF)

Chapter 35
Multipurpose Clock Generator (MCG)

Chapter 36
Oscillator (OSC)

Chapter 37
RTC Oscillator (OSC32K)

Chapter 38
Independent Real Time Clock (iRTC)

Chapter 39
Low-Power Timer (LPTMR)

Chapter 40
Periodic Interrupt Timer (PIT)

Chapter 41
Quad Timer (QTMR)

Chapter 42
Programmable Delay Block (PDB)

Chapter 43
Inter-Integrated Circuit (I2C)

Chapter 44
Serial Peripheral Interface (SPI)

Chapter 45
Universal Asynchronous Receiver/Transmitter (UART)

Chapter 46
Low-Power Universal Asynchronous Receiver/Transmitter (LPUART)

Chapter 47
Memory Mapped Cryptographic Acceleration Unit (MMCAU)

Chapter 48
Cyclic Redundancy Check (CRC)

Chapter 49
Random Number Generator Accelerator (RNGA)

Chapter 50
Segment LCD Controller (SLCD)

Chapter 51
General-Purpose Input/Output (GPIO)

Chapter 1
About This Document

1.1 Overview

1.1.1 Purpose

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

1.1.2 Audience

This document is primarily for system architects and software application developers who are using or considering using the KM3x_256 microcontroller in a system.

1.2 Conventions

1.2.1 Numbering systems

The following suffixes identify different numbering systems:

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

1.2.2 Typographic notation

The following typographic notation is used throughout this document:

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

1.2.3 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.

Chapter 2
Introduction

2.1 KM3x_256 family introduction

KM3x_256 family of devices are 32-bit MCUs in 90nm Thin Film Storage (TFS) embedded flash technology. These devices are primarily focused to serve the metering markets for smart three-phase energy meters (India and China) and three-phase meters (US, Japan, and Europe). KM3x_256 family targets EN 50470-1, EN 50470-3, IEC 62053-21, IEC 62053-22, and IEC 62053-23 class of meters.

KM3x_256 device are based on 32-bit ARM®Cortex® M0+ core with integrated Analog Front End (AFE). CPU clock rates on these devices can reach up to 75 MHz. The KM3x_256 family of devices includes memory-mapped arithmetic unit (MMAU), highly accurate Sigma Delta (SD) ADC, Programmable Gain Amplifier (PGA), high precision internal voltage reference, flash, RAM, phase compensation logic block and other peripherals. KM3x_256 family also provides tamper detection and accurate real time clock on all devices.

2.2 Detailed block diagram

The following figure shows the detailed block diagram for KM3x_256 family of microcontrollers:

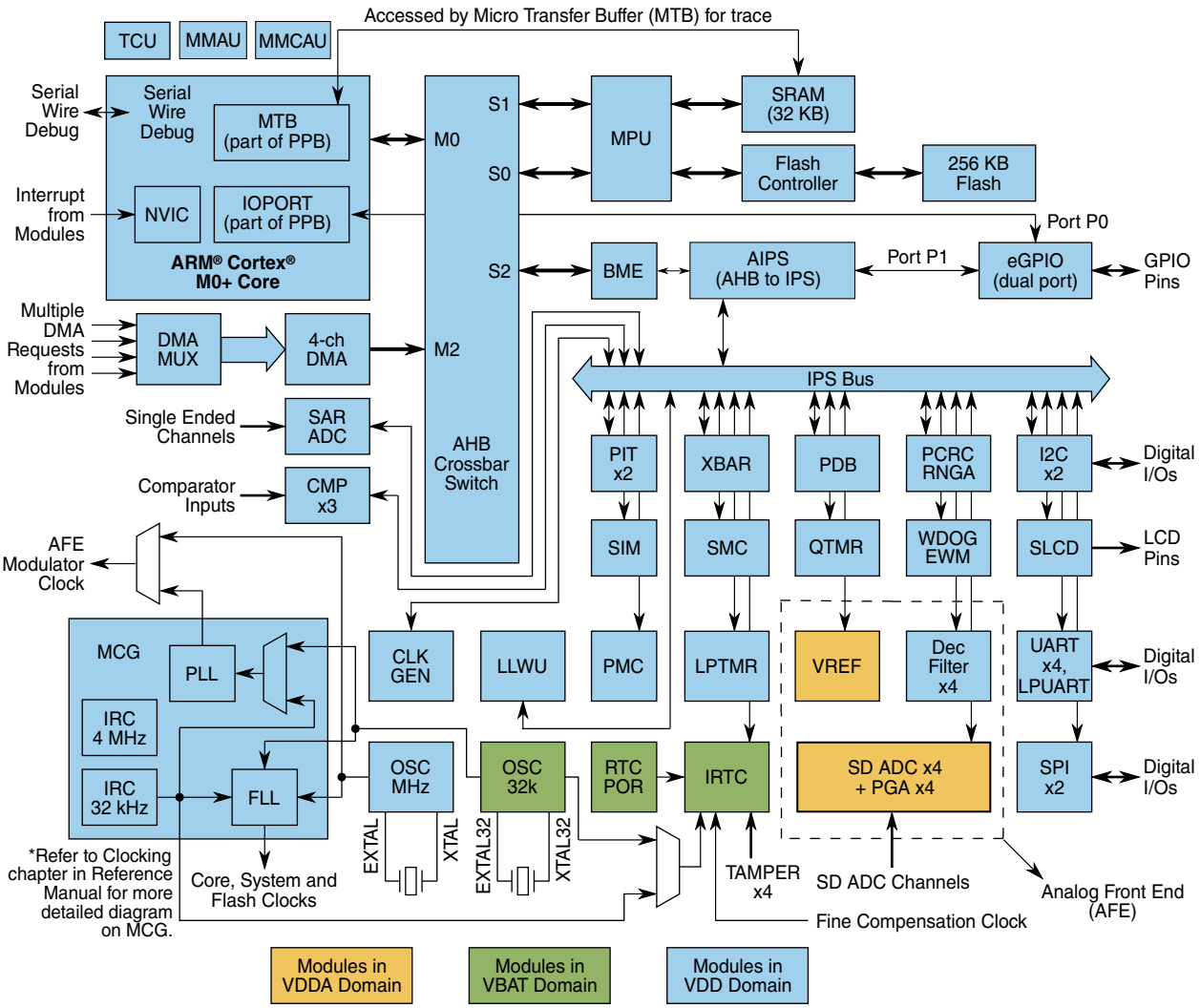


Figure 2-1. Detailed block diagram

2.3 KM3x_256 feature set

KM3x_256 family of devices have the following features:

Table 2-1. KM3x_256 feature set

Features	Description
Operating Characteristics	• Voltage range: 1.71 V to 3.6 V (without AFE) • Voltage range: 2.7 V to 3.6 V (with AFE) • Flash programming voltage from 1.71 V to 3.6 V • iRTC battery supply range 1.71 V to 3.6 V • Temperature range (T_A) –40°C to 105°C • Flexible modes of operation
Core	• High Performance ARM®Cortex® M0+ Core • Up to 75 MHz of Core Clock Frequency

Table continues on the next page...

Table 2-1. KM3x_256 feature set (continued)

Features	Description

	• Supports v6-M Instruction Set Architecture (ISA) including all 16-bit v7-M instructions plus a number of 32-bit Thumb-2 instructions
	• 100% compatible with Cortex® M0
	• 0.95 DMIPS per MHz performance when executing from internal RAM
	• Nested Vectored Interrupt Controller
	  • 32 vectored interrupts
	  • 4 programmable priority levels

Memory Mapped Arithmetic Unit (MMAU)	• Multiplication and Accumulation • Division • Root Square
Memory Mapped Cryptographic Acceleration Unit (MMCAU)	• DES • 3DES • AES • MD5 • SHA-1/SHA-256

Clocks	• MHz Oscillator
	  • Mid Range: 1 MHz to 8 MHz
	  • High Range: 8 MHz to 32 MHz
	• 32.768 kHz crystal oscillator in iRTC power domain
	• Two internal trim-able clock references
	  • 32 kHz
	  • 4 MHz
	• Internal 1 kHz low power oscillator
	• PLL to generate clocks for AFE
	  • Input range: 31.25 kHz to 39.0625 kHz
	  • Output range: 11.72 MHz to 14.65 MHz
	• FLL to generate core, system and flash clocks
	  • Input range: 31.25 kHz to 39.0625 kHz
	  • Output Range: 20 MHz to 75 MHz
	• Clock ratio
	  • Core:Bus:Flash = 3:1:1 for core clock > 50 MHz
	  • Core:Bus:Flash = 2:1:1 for core clock ≤ 50 MHz
	  • Core:Bus:Flash = 1:1:1 for core clock ≤ 25 MHz
	  • Core:Bus:Flash = 4:1:1, depending on application need
	NOTE: Bus clock must be configured to be higher than or equal to 64 kHz.

Analog	• 24 bit Sigma Delta ADC • Can operate in RUN, VLPR, STOP, PSTOP1 and VLPS modes • PGA with gains from 1 to 32 • 1.2 V VREF • 16-bit SAR ADC • CMP with 6-bit DAC
System, protection and power management features	• Various stop, wait, and run modes to provide lower power based on application needs • AWIC to wakeup from STOP and VLPS modes • Peripheral clock enable register can disable clocks to unused modules, thereby reducing currents • Low voltage warning and detection with selectable trip points

Table continues on the next page...

Table 2-1. KM3x_256 feature set (continued)

Features	Description
	• Illegal opcode and illegal address detection with reset • Hardware Programmable CRC module to support fast cyclic redundancy checks (CRC) • Random Number Generator (RNGA) • 128-bit unique chip identifier • Software and hardware watchdog with external monitor pin • 3 Tamper inputs for Tamper Detection (Part of iRTC) • Flash security and block protection • Peripheral crossbar to allow internal signal remapping for flexibility
Debug	• 2-pin Serial Wire Debug (SWD) - Standard ARM® debug interface
DMA	• 4 channel DMA • No support for Scater/Gather • Support asynchronous DMA transfer
Timers	• Independent Real Time Clock (iRTC) independently powered by battery and on-chip crystal clock drift compensation. • Quad Timer (4 channels) • Programmable Interrupt Timer (PIT) • Low-power timer (LPTMR) • Programmable delay block (PDB)

Communications	• Universal asynchronous receiver/transmitter (UART) (all supporting hardware flow control)
	  • with ISO 7816 (on 2 UARTs)
	  • IrDA capability (via CMP0 & XBAR) on all UARTs (one can be used at a time)
	  • with DMA (all UARTs)
	  • 1 LPUART
	• Low-power UART (LPUART0)
	• Serial Peripheral Interface (SPI) with FIFO (SPI1) and without FIFO (SPI0)
	• Inter-Integrated Circuit (I^2C) ×2, with SMBus protocol support

Input/Output	• Up to 13 eGPIOs ports and 1 port with digital glitch filters
	• Pin interrupt/DMA request capability
	• eGPIO pins connected to the processor's local 32-bit platform bus (RGPIO)
	  • eGPIO pins also accessible via IPS bus accesses (protected via AIPS)
	  • eGPIO module to have inbuilt access protection

Memory	• 256 KB single array Flash
	• Flash memory read and write down to 1.71 V
	• No FlexMemory
	• Core:Flash frequency is
	  • 3:1 for high core frequency (> 50 MHz)
	  • 2:1 for medium core frequency (≤ 50 MHz)
	  • 1:1 for low core frequency (≤ 25 MHz)
	  • 4:1, depending on application need
	• 32 KB of single access RAM

Display	• 4×60 Segment LCD

Table continues on the next page...

Table 2-1. KM3x_256 feature set (continued)

Features	Description
	• 6×58 Segment LCD • 8×56 Segment LCD • 3 V LCD glass supported with segment fault detection
Power Consumption	• RUN Mode (all modules OFF): 6.0 mA • I_{DD} Equation: 1.14 mA + 97 µA/MHz Please refer to Power Modes section for more details.

NOTE

Features will vary from device to device. See "Device configuration" section for exact details.

2.4 Device configuration

The following table lists the device configuration for KM3x_256 microcontroller:

Table 2-2. Device configuration

Configuration/Module	144 LQFP	100 LQFP
Maximum CPU frequency (MHz)	75	75
Dimensions (mm^2)	20 × 20	14 × 14
MMAU	1	1
MMCAU	1	1
DMA	4 ch	4 ch
MPU	1	1
Peripheral XBAR (number of IO supported)	up to 11 inputs and 11 outputs	up to 9 inputs and 9 outputs
Single Wire Debug (SWD)	1	1
Cyclic redundancy check (CRC)	1	1
Random Number Generator (RNGA)	1	1
Watchdog timer (WDOG)	1	1
External Watchdog Monitor (EWM)	1	1
Passive Tamper Pins	3	3
Flash memory (KB)	256	256
RAM (KB)	32	32
MCG	FLL + Internal OSC (32 kHz or 4 MHz) PLL (32.768 kHz reference)	FLL + Internal OSC (32 kHz or 4 MHz) PLL (32.768 kHz reference)
Real Time Clock (32 kHz OSC)	1	1

Table continues on the next page...

Table 2-2. Device configuration (continued)

Configuration/Module	144 LQFP	100 LQFP
Quad Timer (TMR)	1 with 4 channels	1 with 4 channels
Low-power Timer (LPTMR)	1	1
Periodic Interrupt Timer (PIT)	2	2
Programmable Delay Block (PDB)	1	1
UART[1]	4	4
LPUART	1	1
SPI (16-bit)	2	2
I2C	2	2
24-bit Analog Front End[2] • Total dedicated SD ADC differential channels (for current measurement) with PGA	Up to 4	Up to 4
16-bit SAR ADC	1	1
Total Single Ended Channels for SAR ADC	16	12
1.2 V VREF	1	1
CMP (Number of Channels)	3 (18)	3 (18)
Segmented LCD	4×60 (8×56)	4×40 (8×36)
Total GPIO pins	99	72
eGPIO	All GPIO pins (when accessed via Core's IOPORT)	All GPIO pins (when accessed via Core's IOPORT)

1. Any UART can be selected for IrDA. But only one IrDA communication is supported (enforced via XBAR).
2. See "Supported Packages" section for exact number of supported channels.

NOTE

See "Supported Packages" section for exact number of SD channels, memory and LCD options.

2.4.1 Supported Packages

The following table shows a list of supported packages.

Table 2-3. Supported Packages

Part Number	Part Description	Package	LCD	SD[1]	Flash (KB)	RAM (KB)
MKM34Z256VLQ7	KM3x_256	144 LQFP	Yes	4	256	32
MKM34Z256VLL7	KM3x_256	100 LQFP	Yes	4	256	32

1. Number of SD ADC Channels supported.

2.5 Modules on the device

2.5.1 Platform modules

The following table lists all platform modules that are available on this device:

Table 2-4. Platform modules

Module	Description	Reference

ARM®Cortex®-M0+ core	The Cortex®-M0+ processor is an entry-level 32-bit ARM®Cortex® processor designed for a broad range of embedded applications.
	• Up to 75 MHz of Core Clock Frequency
	• Supports v6-M Instruction Set Architecture (ISA) including all 16-bit v7-M instructions plus a number of 32-bit Thumb-2 instructions
	• 100% compatible with Cortex® M0
	• 0.95 DMIPS per MHz performance when executing from internal RAM
	• Nested Vectored Interrupt Controller
	  • 32 vectored interrupts
	  • 4 programmable priority levels
	• User and privileged mode execution
	• Single-cycle 32-bit hardware multiplier
	• Low latency, high-speed peripheral I/O port
	• A Vector Table Offset Register
		For ARM® processor documentation, refer to arm.com

Nested Vectored Interrupt Controller (NVIC)	External interrupt signals connect to the NVIC, and the NVIC prioritizes the interrupts. Software can set the priority of each interrupt. The NVIC and the Cortex®-M0+ processor core are closely coupled, providing low latency interrupt processing and efficient processing of late arriving interrupts. All NVIC registers are only accessible using word transfers. Any attempt to read or write a halfword or byte individually is Unpredictable. NVIC registers are always little-endian. Processor accesses are correctly handled regardless of the endian configuration of the processor.	Full documentation for this module is provided by ARM® and can be found at http://www.arm.com
Asynchronous Wake-up Interrupt Controller (AWIC)	The primary function of the Asynchronous Wake-up Interrupt Controller (AWIC) is to detect asynchronous wake-up events in stop modes and signal to clock control logic to resume system clocking. After clock restart, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.	Full documentation for this module is provided by ARM® and can be found at http://www.arm.com
Debug Interfaces	Two pin Serial Wire Debug (SWD) - Standard ARM® debug interface	For ARM® debug documentation, refer to *ARM® Debug Interface* -

Table 2-4. Platform modules

Module	Description	Reference
		IHI0031A_ARM_debug_interface_v5.pdf at http://www.arm.com

2.5.2 System modules

The following system modules are available on this device:

Table 2-5. System modules

Module	Description
DMA Controller (DMA)	• 4 independently programmable DMA controller channels provide the means to directly transfer data between system memory and I/O peripherals • DMA controller can operate in run and wait modes • Dual-address (source, destination) transfers via 32-bit master connection to the system bus • Data transfers in 8-, 16-, or 32-bit accesses • Continuous-mode or cycle-steal transfers from software initiated or peripheral paced transfers • Programmable request per channel selected from 16 possible sources
Crossbar Switch (AXBS)	• Hardware interconnect matrix interfacing bus masters to bus slaves • Supports 2 masters and 3 slaves • Two-stage pipelined AMBA-AHB system bus protocol • Support for concurrent data transfers to slave targets • Programmable fixed priority or round robin arbitration
Memory Protection Unit (MPU)	• Concurrently monitors and evaluates access controls for 2 slave transfers • Supports 8 memory region descriptors, each 128 bits in size defining start and end addresses and access rights • Detects access protection errors if a memory reference does not hit in any memory region, or if the reference is illegal in all memory regions hit. If an access error occurs, the reference is terminated with an error response, and the MPU inhibits the bus cycle being sent to the targeted slave device. • 64-bit error registers (one per slave port) capture the last faulting address, attributes, and other information • Global MPU enable/disable control bit
Low-Leakage Wakeup Unit (LLWU)	The LLWU module allows the device to wake from low leakage power modes (VLLS) through various internal peripheral and external pin sources.
Power Management Controller (PMC)	• Separate digital (regulated) and analog (referenced to digital) supply outputs • Programmable power saving modes • No output supply decoupling capacitors required • Available wake-up from power saving modes via RTC and external inputs • Integrated Power-on Reset (POR) • Integrated Low Voltage Detect (LVD) with reset (brownout) capability • Selectable LVD trip points • Programmable Low Voltage Warning (LVW) interrupt capability • Buffered bandgap reference voltage output • Factory programmed trim for bandgap and LVD • 1 kHz Low Power Oscillator (LPO)
System Integration Module (SIM)	• System clocking configuration

Table continues on the next page...

Table 2-5. System modules (continued)

Module	Description
	• System clock divide values • Architectural clock gating control • LPTPM clock selection • Flash and System RAM size configuration • LP Timer external clock and input capture selection • Compiled memory self-time control • QuadTimer clock selection • DMA Done flag selection • Selectable clock for CLKOUT pin • UART Receive/Transmit select • PIT0 or PIT1 output selection

External Watchdog Monitor (EWM)	• Independent 1 kHz LPO clock source
	• Programmable time-out period specified in terms of number of EWM LPO clock cycles
	• Windowed refresh option
	  • Provides robust check that program flow is faster than expected
	  • Programmable window
	  • Refresh outside window leads to assertion of EWM_out
	• Robust refresh mechanism
	  • Write values of 0xB4 and 0x2C to EWM Refresh Register within (EWM_service_time) peripheral bus clock cycles
	• One output port, EWM_out, when asserted is used to reset or place the external circuit into safe mode
	• One Input port, EWM_in, allows an external circuit to control the EWM_out signal

Watchdog Timer (WDOG)	• IEC 60730 compliant watchdog monitor • Independent, configurable clock source input • Write-once control bits with unlock sequence • Programmable timeout period • Ability to test watchdog timer and reset • Windowed refresh option • Robust refresh mechanism • Cumulative count of watchdog resets between power-on resets • Configurable interrupt on timeout
Inter-Peripheral Crossbar Switch (XBAR)	• 33 identical 33-input muxes with individual select fields • Edge detection with associated interrupt or DMA request generation for a subset of mux outputs • Memory mapped registers with IPBus interface for select and control fields • Register write protection input signal

2.5.3 Clock

The following clocks are available on this device:

Table 2-6. Clock modules

Module	Description
Clock	• Frequency-Locked Loop (FLL)

Table 2-6. Clock modules

Module	Description
	• Digitally controlled oscillator (DCO) with programmable frequency range • Option to program DCO frequency for a 32.768 kHz external reference clock source • Internal or external reference clock can be used to control the FLL • Phase-Locked Loop (PLL) • Fixed multiplier to get Voltage-Controlled Oscillator (VCO) frequency for a 31.25 kHz to 39.0625 kHz (typ. 32.768 kHz) external reference clock source input range • VCO frequency range 11.71875 MHz @ 31.25 kHz to 14.6484375 MHz @ 39.0625 kHz (typ. 12.288 MHz @ 32.768 kHz) • Modulo VCO frequency divider phase/frequency detector • Integrated loop filter • Primarily drives the AFE clock • Internal Reference Clock (IRC) generator • 32 kHz low range clock with 9 trim bits for accuracy • 2 MHz fast clock with 3 trim bits • Low range clock can be used to control the FLL • Low range or fast clock can be selected as MCU's clock source • Can be used as a clock source for other on-chip peripherals • External Clock (ERCLK) from the Crystal Oscillator (XOSC) • Can be used as the FLL and/or PLL source • Can be selected as the clock source for the MCU. This can impact AFE accuracy and is not preferred when using AFE for metering. • External clock monitor with reset request capability • Lock detector with interrupt request capability for use with the PLL • Auto Trim Machine (ATM) for trimming both the low range and fast internal reference clocks • Reference dividers for FLL are provided • Clock source selected can be divided down by 1, 2, 4, 8, or 16

2.5.4 Security modules

The following security modules are available on this device:

Table 2-7. Security modules

Module	Description
Programmable CRC (PCRC)	• Hardware CRC generator circuit using 16/32-bit shift register • User Configurable 16/32 bit CRC • Programmable Generator Polynomial • Error detection for all single, double, odd, and most multi-bit errors • Programmable initial seed value • High-speed CRC calculation • Optional feature to transpose input data and CRC result via transpose register, required on applications where bytes are in LSB format
Random Number Generator (RNGA)	• RNGA can be used along with software to comply with *NIST SP800-90* standards • National Institute of Standards and Technology (NIST)-approved pseudo-random number generator http://csrc.nist.gov

Table 2-7. Security modules

Module	Description
	• Supports the key generation algorithm defined in the *Digital Signature Standard*http://www.itl.nist.gov/fipspubs/fip186.htm • Integrated entropy sources capable of providing the PRNG with entropy for its seed

2.5.5 Analog modules

The following analog modules are available on this device:

Table 2-8. Analog modules

Module	Description

Analog Front End (AFE)	• 4 independent 16-bit 2nd-order Sigma Delta (SD) ADC for simultaneous voltage and current sampling
	• 24-bit AFE dynamic range for energy calculations (after averaging)
	• 4 Continuous Time Programmable Gain Amplifiers (PGA)
	• PGA with gains from 1x to 32x (1, 2, 4, 8, 16, and 32)
	• Fixed Gain of 32 for shunt Resistor
	• Final SD ADC sample rate selectable from 3 kHz to 96 kHz (3 kHz, 6 kHz, 12 kHz, 24 kHz, 48 kHz, or 96 kHz) (OSR in range 2048x down to 64x)
	• Modes of operation
	  • Normal mode (higher accuracy)
	  • Low power mode (lower accuracy)
	• On-chip phase compensation and decimation
	• Separate controls to enable or disable PGA, ADC, filters individually
	• Synchronized start of operation
	• Supports current transformer, shunt resistor, Rogowski coil
	• Support for interfacing with external Sigma Delta Modulators
	• ±250 mV (1 Vpp differential, 0.5 Vpp single ended), input range for both differential and single ended inputs
	• Temperature Stable 1.2 V Voltage Reference @ 15 ppm with trim capability
	• Conversion ready signals asserted when conversion results are moved to the corresponding result registers
	• AFE clocked from dedicated PLL. Option to run AFE with other clock sources but at reduced accuracy.
	• Optimized for *IEC50470 Class C: 0.15-5(150)A* and *ANSI C 12.20 Class 0.1, 3-50(320)A*

High Speed Comparator (CMP)	• 6-bit DAC programmable reference generator output
	• Up to 7 selectable comparator inputs; each input can be compared with any input by any polarity sequence
	• Differential input on 2 comparator channels (CMP1) to support zero crossing detection on voltage channels of AFE. These can be used as stand alone channels also. CMP0 channels can be used to implement IrDA communication along with UART.
	• Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output
	• Comparator output supports:
	  • Sampled
	  • Windowed (ideal for certain PWM zero-crossing-detection applications
	  • Digitally filtered using external sample signal or scaled peripheral clock

Table continues on the next page...

Table 2-8. Analog modules (continued)

Module	Description

	• Two performance modes:
	  • Shorter propagation delay at the expense of higher power
	  • Low power, with longer propagation delay
	• Operational in all MCU power modes

Voltage Reference (VREF)	• Programmable trim register with 0.5 mV steps, automatically loaded with room temp value upon reset
	• Programmable mode selection:
	  • Off
	  • Bandgap out (or stabilization delay)
	  • Low-power buffer mode
	  • Tight-regulation buffer mode
	• 1.2 V output with 15 ppm at room temperature
	• Dedicated output pin

Analog-to-Digital Converter (SAR ADC)	• 16-bit resolution • Up to 32 kHz sampling rate • Dedicated internal input for offset calibration • Single or continuous conversion modes • Programmable clock source (bus or alternate) • Programmable voltage reference (internal or external) • DMA support

2.5.6 Timer modules

The following timer modules are available on this device:

Table 2-9. Timer modules

Module	Description
Periodic Interrupt Timer (PIT)	• Up to 2 general purpose interrupt timers • Up to 2 interrupt timers for triggering ADC conversions • 32-bit counter resolution • Clocked by system clock frequency • DMA support

Low Power Timer (LPTMR)	• Operation as timer or pulse counter
	• Selectable clock for prescaler/glitch filter
	  • 1 kHz internal LPO
	  • External low power crystal oscillator
	  • Internal reference clock (not available in low leakage power modes)
	  • Secondary external reference clock (for example, 32 kHz crystal)
	• Configurable glitch filter or prescaler
	• Interrupt generated on timer compare
	• Hardware trigger generated on timer compare

Independent Real Time Clock (iRTC)	• Independent Power Supply, POR with brownout detection and 32 kHz Crystal Oscillator • On-chip Hardware Calendaring • Programmable alarm with interrupt • Hardware compensation to compensate the 1 Hz clock

Table continues on the next page...

Table 2-9. Timer modules (continued)

Module	Description
	• Support for Coarse and Fine Compensation • 32 bytes Battery backed Memory • Support for up-to 3 Passive external tampers • Up to 2 Active Tamper pins • Write Protection Mechanism • Power Supply Glitch Detector • Tamper Queue for storing up-to 4 tamper events along with time stamp • iRTC operation from 1.71 V to 3.6 V • Ultra low power iRTC operation (< 1µA)
Quad Timer (TMR)	• Four 16-bit counters/timers • Counters are cascadable • Max count rate equals to peripheral clock for internal clocks • Max count rate equals peripheral clock divided by 2 for external clocks • Programmable count modulo • Count once or repeatedly • Separate prescaler for each counter • Each counter has capture and compare capability
Programmable Delay Block (PDB)	PDB provides controllable delays from either an internal or an external trigger, or a programmable interval tick, to the hardware trigger inputs of ADC.

2.5.7 Communication interfaces

The following communication interfaces are available on this device:

Table 2-10. Communication modules

Module	Description
Serial Peripheral Interface (SPI)	• Master and slave mode • Full-duplex, three-wire synchronous transfers • Programmable transmit bit rate • Double-buffered transmit and receive data registers • Serial clock phase and polarity options • Slave select output • Mode fault error flag with CPU interrupt capability • Control of SPI operation during wait mode • Selectable MSB-first or LSB-first shifting • Programmable 8-bit or 16-bit data transmission length • Receive data buffer hardware match feature • 64-bit FIFO mode for high speed transfers of large amounts of data (SPI0 only) • 5V AMR support on one SPI (SPI1) • Support for both transmit and receive by DMA
Inter-Integrated Circuit (I2C)	• Compatible with I2C bus standard and *SMBus Specification Version 2* features • Up to 100 kbps with maximum bus loading • Multi-master operation • Software programmable for one of 64 different serial clock frequencies • Programmable slave address and glitch input filter • Interrupt or DMA driven byte-by-byte data transfer

Table continues on the next page...

Table 2-10. Communication modules (continued)

Module	Description
	• Arbitration lost interrupt with automatic mode switching from master to slave • Calling address identification interrupt • Bus busy detection broadcast and 10-bit address extension • Address matching causes wake-up when processor is in low power mode

Universal Asynchronous Receiver/ Transmitter (UART)	• Support for *ISO 7816* protocol for interfacing with smart cards
	• Full-duplex operation
	• Standard mark/space non-return-to-zero (NRZ) format
	• 13-bit baud rate selection with fractional division of 32
	• Programmable 8-bit or 9-bit data format
	• Separately enabled transmitter and receiver
	• Programmable transmitter output polarity
	• Programmable receive input polarity
	• 13-bit break character option
	• 11-bit break character detection option
	• Parameterizable buffer support for one dataword for each transmit and receive
	• Independent FIFO structure for transmit and receive
	• Two receiver wakeup methods:
	  • Idle line wakeup
	  • Address mark wakeup
	• Address match feature in receiver to reduce address mark wakeup ISR overhead
	• Ability to select MSB or LSB to be first bit on wire
	• Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
	• Interrupt-driven operation with 11 flags:
	  • Transmitter data buffer at or below watermark
	  • Transmission complete
	  • Receiver data buffer at or above watermark
	  • Idle receiver input
	  • Receiver overrun
	  • Receiver data buffer underflow
	  • Noise error
	  • Framing error
	  • Parity error
	  • Active edge on receive pin
	• Receiver framing error detection
	• Hardware parity generation and checking
	• 1/16 bit-time noise detection
	• DMA requests

Low-power UART (LPUART)	Low-power UART module that retains functionality in stop modes. • Supports basic UART with DMA interface function and ×4 to ×32 oversampling of baud-rate • Keeps functional in VLPS mode provided the clock it is using remains enabled • Combines all standard UART interrupt sources into a single interrupt request. It supports MODEM and IrDA. • Supports LIN slave operation

2.5.8 Human-machine interfaces

The following Human-Machine Interfaces (HMI) are available on this device:

Table 2-11. HMI modules

Module	Description

Segment LCD (SLCD)	• LCD waveforms functional in low-power modes
	• Up to 44 LCD pins with selectable front plane/back plane configuration
	  • Generate up to 40 front plane signals
	  • Generate up to 8 back planes signals
	• Programmable LCD frame frequency
	• Programmable blink modes and frequency
	  • All segments blank during blink period
	  • Alternate display for each LCD segment in x4 or less mode
	  • Blink operation in low-power modes
	• Programmable LCD power supply switch, making it an ideal solution for battery-powered and board-level applications
	  • Charge pump requires only four external capacitors
	  • Internal LCD power using VDD
	  • Internal VIREG regulated power supply option for 3 V LCD glass
	  • External VLL3 power supply option (3 V)
	• Internal-regulated voltage source with a 4-bit trim register to apply contrast control
	• Integrated charge pump for generating LCD bias voltages; Hardware-configurable to drive 3 V LCD panels; On-chip generation of bias voltages.
	• Waveform storage registers
	• Backplane reassignment to assist in vertical scrolling on dot-matrix displays
	• Software-configurable LCD frame frequency interrupt

enhanced General Purpose Input/Output (eGPIO)	• Programmable glitch filter on up to 8 input pins and interrupt with selectable polarity on all input pins
	• Independent pin value register to read logic level on digital pin
	• Configurable data direction & pin enable on all pins
	• Protected access to all GPIO registers
	• Dual access to GPIO registers
	  • Single cycle access with zero wait states (IOPORT access)
	  • Access via AIPS slave port (non-zero wait states)

Chapter 3
Core Overview

3.1 Introduction

The enhanced ARM®Cortex® M0+ is the member of the Cortex®-M Series of processors targeting micro-controller cores focused on very cost sensitive, low power applications. It has a single 32-bit AMBA AHB-Lite interface and includes an NVIC component. It also has hardware debug functionality including support for simple program trace capability. The processor supports the ARMv6-M instruction set (Thumb) architecture including all but three 16-bit Thumb opcodes (52 total) plus seven 32-bit instructions. It is upward compatible with other Cortex-M profile processors. It also has an I/O port which supports the single cycle loads and stores to tightly-coupled peripherals (e.g. GPIO). Target of load/store transactions determined via external address decode logic.

3.1.1 Buses, interconnects, and interfaces

The ARM®Cortex®-M0+ core has two system bus interfaces:

- Single 32-bit AMBA-3 AHB-Lite system interface that provides connections to all system memory (flash, RAM) and peripherals;
- Single 32-bit I/O Port bus interfacing to the device pads (all GPIO pins) with 1-cycle loads and stores. This port is connected to eGPIO module.

3.1.2 System Tick Timer

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

- The CLKSOURCE bit in SysTick Control and Status register is always set to select the core clock.

- Because the timing reference (FCLK) is a variable frequency, the TENMS bit in the SysTick Calibration Value Register is always zero.
- The NOREF bit in SysTick Calibration Value Register is always set, implying that FCLK is the only available source of reference timing.

3.1.3 Debug facilities

This device support the standard ARM® 2-pin Serial Wire Debug (SWD) debug port only, which provides a more efficient pin interface.

3.1.4 Core privilege levels

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

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

3.2 Nested vectored interrupt controller (NVIC)

3.2.1 Interrupt priority levels

This device supports 4 priority levels for interrupts. Therefore, in the NVIC each source in the IPR registers contains 2 bits. For example, IPR0 is shown below:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IRQ3		0	0	0	0	0	0	IRQ2		0	0	0	0	0	0	IRQ1		0	0	0	0	0	0	IRQ0		0	0	0	0	0	0
W																																

3.2.2 Non-maskable interrupt

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

3.2.3 Interrupt Channel Assignments

The interrupt vector assignments are defined in the following table.

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

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

Table 3-2. Interrupt vector assignments

Address	Vector	IRQ number	Source module	Source description
ARM® Core System Handler Vectors				
0x0000_0000	0	–	ARM® core	Initial Stack Pointer
0x0000_0004	1	–	ARM® core	Initial Program Counter
0x0000_0008	2	–	ARM® core	Non-Maskable Interrupt (NMI)
0x0000_000C	3	–	ARM® core	Hard Fault
0x0000_0010	4	–	—	—
0x0000_0014	5	–	—	—
0x0000_0018	6	–	—	—
0x0000_001C	7	–	—	—
0x0000_0020	8	–	—	—
0x0000_0024	9	–	—	—
0x0000_0028	10	–	—	—
0x0000_002C	11	–	ARM® core	Supervisor call (SVCall)
0x0000_0030	12	–	—	—
0x0000_0034	13	–	—	—
0x0000_0038	14	–	ARM® core	Pendable request for system service (PendableSrvReq)
0x0000_003C	15	–	ARM® core	System tick timer (SysTick)
Non-Core On Platform Vectors				
0x0000_0040	16	0	DMA	DMA channel 0 transfer complete
0x0000_0044	17	1	DMA	DMA channel 1 transfer complete
0x0000_0048	18	2	DMA	DMA channel 2 transfer complete
0x0000_004C	19	3	DMA	DMA channel 3 transfer complete
Off Platform IRQ Vectors				
0x0000_0050	20	4	SPI0/SPI1	OR'ed Interrupt SPI0 and SPI1
0x0000_0054	21	5	PDB0	OR'ed Interrupt of PDB0
0x0000_0058	22	6	PMC	Low-voltage detect, low-voltage warning
0x0000_005C	23	7	TMR0	TMR0 of Quad Timer
0x0000_0060	24	8	TMR1	TMR1 of Quad Timer
0x0000_0064	25	9	TMR2	TMR2 of Quad Timer

Table continues on the next page...

Table 3-2. Interrupt vector assignments (continued)

Address	Vector	IRQ number	Source module	Source description
0x0000_0068	26	10	TMR3	TMR3 of Quad Timer
0x0000_006C	27	11	PIT0/PIT1	OR'ed Interrupt of PIT0 and PIT1
0x0000_0070	28	12	LLWU	Low Leakage Wakeup
0x0000_0074	29	13	Flash	OR'ed interrupt for Flash Command Complete/Read collision
0x0000_0078	30	14	ACMP0/ACMP1/ ACMP2	OR'ed Interrupt of 3 ACMP
0x0000_007C	31	15	SLCD	OR'ed Interrupt of SLCD
0x0000_0080	32	16	ADC	OR'ed Interrupt from SAR
0x0000_0084	33	17	PTx	OR'ed Interrupt from all GPIO (PTx) ports
0x0000_0088	34	18	RNGA	OR'ed Interrupt from RNGA
0x0000_008C	35	19	UARTx	OR'ed Interrupt UART0,1,2,3
0x0000_0090	36	20	MMAU	Memory Mapped Arithmetic Unit interrupt
0x0000_0094	37	21	AFE_CH0	AFE Channel 0 OR'ed Interrupt
0x0000_0098	38	22	AFE_CH1	AFE Channel 1 OR'ed Interrupt
0x0000_009C	39	23	AFE_CH2	AFE Channel 2 OR'ed Interrupt
0x0000_00A0	40	24	AFE_CH3	AFE Channel 3 OR'ed Interrupt
0x0000_00A4	41	25	iRTC	iRTC Interrupt
0x0000_00A8	42	26	I2C0/I2C1	OR'ed I2C interrupt
0x0000_00AC	43	27	LPUART0	OR'ed Interrupt of LPUART0
0x0000_00B0	44	28	MCG	MCG Loss of Clock, Loss of Lock
0x0000_00B4	45	29	WDOG/EWM	OR'ed WDOG Interrupt and External Watchdog Monitor
0x0000_00B8	46	30	LPTMR	OR'ed LPTMR Interrupt
0x0000_00BC	47	31	XBAR	Peripheral XBAR OR'ed Interrupt

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

Suppose you need to configure the TMR3 interrupt. The following table is an excerpt of the TMR3 row from Interrupt Channel Assignments.

Table 3-3. Interrupt vector assignments

Address	Vector	IRQ[1]	NVIC IPR register number[2]	Source module	Source description
0x0000_0068	26	10	2	TMR3	TMR3 of Quad Timer

1. Indicates the NVIC's interrupt source number.
2. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4.

- The NVIC registers you would use to configure the interrupt are:
  - NVICIPR2
- To determine the particular IRQ's field location within these particular registers:
  - NVICIPR2 field starting location = 8 × (IRQ mod 4) + 6 = 22

  Since the NVICIPR fields are 2-bit wide (4 priority levels), the NVICIPR2 field range is 22–23.

Therefore, the following field locations are used to configure the TMR3 interrupts:
- NVICIPR2[23:22]

3.3 AWIC introduction

The primary function of the AWIC block is to detect asynchronous wake-up events in stop modes and signal to clock control logic to resume system clocking. After clock restart, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.

3.3.1 Wake-up sources

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

Table 3-4. AWIC Stop and VLPS Wake-up Sources

Wake-up source	Description
Available system resets	$\overline{\text{RESET}}$ pin and WDOG when LPO is its clock source
Low-voltage detect	Mode Controller
Low-voltage warning	Mode Controller[1]
Pin interrupts	Port Control Module - Any enabled pin interrupt is capable of waking the system
ADC	The ADC is functional when using internal clock source
CMP	Since no system clocks are available, functionality is limited
I^2C	Address match wakeup (on all I^2Cs)
UARTx	Active edge on RXD (on all UARTs)
LPTMR0	Functional in Stop/VLPS modes
iRTC Interrupt	Functional in Stop/VLPS modes (this includes the tamper event)
iRTC Alarm	Functional in Stop/VLPS modes
AFE	Optionally active in Stop/VLPS modes
NMI	NMI_B functionality enabled on PTA4 pin
MCG	MCG Loss of clock and loss of lock interrupts, if configured appropriately.[2]
LPUART	Functional when using clock source which is active in Stop and VLPS modes

Table continues on the next page...

Table 3-4. AWIC Stop and VLPS Wake-up Sources (continued)

Wake-up source	Description
XBAR	Active edge on XBAR interrupt request channel
SPI	Slave mode interrupt
SLCD	Segment LCD interrupt

1. Only in STOP mode. LVD, LVW disabled in VLPS mode.
2. Only in STOP mode.

NOTE

Any module that is Optionally active in STOP/VLPS modes and capable of generating asynchronous interrupt can wakeup the system. The above table lists the sources which must be available.

Chapter 4
System Memory Map

4.1 Introduction

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

4.2 System Memory Map

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

Table 4-1. System memory map

System 32-bit Address Range	Destination Slave	Size	Access
0x0000_0000–0x0003_FFFF	Flash Memory (Includes exception vectors in first 192 bytes)	256 KB	All masters
0x0004_0000–0x1FFF_DFFF	Reserved	–	–
0x1FFF_E000–0x1FFF_FFFF	SRAM_L: Lower SRAM	8 KB	All masters
0x2000_0000–0x2000_5FFF	SRAM_U: Upper SRAM	24 KB	All masters
0x2000_6000–0x3FFF_FFFF	Reserved	–	–
0x4000_0000–0x4007_EFFF	AIPS peripherals - See AIPS peripheral slot assignment for more details	508 KB	All masters
0x4007_F000–0x400F_EFFF	Reserved	–	–
0x400F_F000–0x400F_FFFF	eGPIO Ports	4 KB	All masters
0x4010_0000–0x43FF_FFFF	Reserved	–	–
0x4400_0000–0x5FFF_FFFF	Peripheral Bit Manipulation Engine	448 MB	Cortex® M0+ core only
0x6000_0000 – 0xDFFF_FFFF	Reserved	–	–
0xE000_0000-0xE00F_FFFF	Private Peripherals	1000 KB	Cortex® M0+ core only
0xE010_0000-0xEFFF_FFFF	Reserved	–	–
0xF000_0000–0xFFFF_FFFF	Private Peripheral Bus (PPB) - See Private Peripheral Bus (PPB) memory map for more details	256 MB	Cortex® M0+ core and DMA

NOTE

- Exception vectors are included in first 192 bytes of the Flash Memory space.
- Accesses to invalid memory ranges should cause a transfer error exception.
- The 8 KB region of SRAM_U from 0x2000_0000 to 0x2000_1FFF block is the region of RAM left powered on in low-power mode VLLS2.
- There are no bitband regions supported by M0+ (neither for RAM nor peripheral).

4.3 Flash Memory Map

The flash array and the flash registers are located at different base addresses as shown in the following figure. The base address for each is specified in System Memory Map.

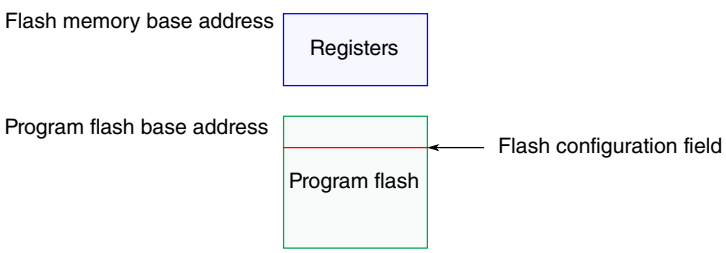


Figure 4-1. Flash memory map

Addresses for flash read accesses is determined according to the overall system memory as shown in System Memory Map.

4.4 SRAM memory map

The on-chip RAM is split amongst SRAM_L and SRAM_U in the ratio of 1:3. The RAM is implemented such that the SRAM_L and SRAM_U ranges from a contiguous block in the memory map.

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

4.5 Peripheral bridge (AIPS-Lite) memory map

The peripheral memory map is accessible via one slave port on the crossbar in the 0x4000_0000–0x4007_FFFF region. The device implements one peripheral bridge that defines a 512 KB address space.

NOTE

The eGPIO registers is present in the 0x400F_F000 - 0x400F_FFFF address range.

Slave peripheral space is divided into three regions:

- A 128 KB region, partitioned as 32 spaces, each 4 KB in size and reserved for on-platform peripheral devices. The AIPS controller generates unique module enables for all 32 spaces.
- A 380 KB region, partitioned as 95 spaces, each 4 KB in size and reserved for off-platform modules. The AIPS controller generates unique module enables for all 95 spaces.
- A 4 KB region left reserved to have compatibility with Kinetis.

The AIPS controller supports disabling of individual slots via a bus of input control signals. Unimplemented slots have their control signal tied off to disabled (logic 1). Modules that have clock gate control bits (implemented in the SIM's SCGCx registers) to disable system clocking to identified peripherals should also use these control bits to enable/disable associated AIPS slots.

NOTE

When the associated SCGCx bit is logic 0, the disable input is driven to logic 1.

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

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

The eGPIO module has two ports. The first port is connected to IOPORT bus of the core to allow direct and fast access to GPIO pins from the core. The other port of eGPIO is active when access to 0x400F_F000 - 0x400F_FFFF address space via the AIPS is done.

4.6 AIPS peripheral slot assignment

NOTE

- Slots 0-79 are 32-bit data width modules;
- Slots 80-95 are 16-bit data width modules;
- Slots 96-126 are 8-bit data width modules.
- Peripherals generate transfer error for reserved addresses in their assigned slots.

Table 4-2. Peripheral bridge slot assignments

System 32-bit base address	Slot number	Module
0x4000_0000	0	AIPS Lite
0x4000_1000	1	Reserved for Platform Modules
0x4000_2000	2	Reserved for Platform Modules
0x4000_3000	3	Reserved for Platform Modules
0x4000_4000	4	—
0x4000_5000	5	—
0x4000_6000	6	—
0x4000_7000	7	—
0x4000_8000	8	DMA Controller (DMA)
0x4000_9000	9	—
0x4000_A000	10	Memory Protection Unit (MPU)
0x4000_B000	11	—
0x4000_C000	12	—
0x4000_D000	13	—
0x4000_E000	14	—
0x4000_F000	15	eGPIO [1]
0x4001_0000	16	—
0x4001_1000	17	—
0x4001_2000	18	—
0x4001_3000	19	—
0x4001_4000	20	—
0x4001_5000	21	—
0x4001_6000	22	—
0x4001_7000	23	—
0x4001_8000	24	—
0x4001_9000	25	—
0x4001_A000	26	—
0x4001_B000	27	—

Table continues on the next page...

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

System 32-bit base address	Slot number	Module
0x4001_C000	28	—
0x4001_D000	29	—
0x4001_E000	30	—
0x4001_F000	31	—
Off-platform Modules		
0x4002_0000	32	Flash Registers
0x4002_1000	33	DMA MUX
0x4002_2000	34	
0x4002_3000	35	
0x4002_4000	36	
0x4002_5000	37	—
0x4002_6000	38	—
0x4002_7000	39	—
0x4002_8000	40	—
0x4002_9000	41	Random Number Generator (RNGA)
0x4002_A000	42	LPUART0
0x4002_B000	43	SAR ADC (ADC0)
0x4002_C000	44	—
0x4002_D000	45	Programmable Interval Timer 0 (PIT0)
0x4002_E000	46	Programmable Interval Timer 1 (PIT1)
0x4002_F000	47	—
0x4003_0000	48	Analog Front End (AFE) [2]
0x4003_1000	49	—
0x4003_2000	50	—
0x4003_3000	51	—
0x4003_4000	52	Programmable CRC (CRC)
0x4003_5000	53	—
0x4003_6000	54	Programmable Delay Block (PDB0)
0x4003_7000	55	Port J Pin MUX Control (PORT J)
0x4003_8000	56	Port K Pin MUX Control (PORT K)
0x4003_9000	57	Port L Pin MUX Control (PORT L)
0x4003_A000	58	Port M Pin MUX Control (PORT M)
0x4003_B000	59	—
0x4003_C000	60	Low Power Timer (LPTMR)
0x4003_D000	61	—
0x4003_E000	62	System Integration Module Low Power Logic (SIM_LP)
0x4003_F000	63	System Integration Module High Power Logic (SIM_HP)
0x4004_0000	64	—
0x4004_1000	65	—

Table continues on the next page...

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

System 32-bit base address	Slot number	Module
0x4004_2000	66	—
0x4004_3000	67	Segment LCD (SLCD)
0x4004_4000	68	—
0x4004_5000	69	—
0x4004_6000	70	Port A Pin MUX Control (PORT A)
0x4004_7000	71	Port B Pin MUX Control (PORT B)
0x4004_8000	72	Port C Pin MUX Control (PORT C)
0x4004_9000	73	Port D Pin MUX Control (PORT D)
0x4004_A000	74	Port E Pin MUX Control (PORT E)
0x4004_B000	75	Port F Pin MUX Control (PORT F)
0x4004_C000	76	Port G Pin MUX Control (PORT G)
0x4004_D000	77	Port H Pin MUX Control (PORT H)
0x4004_E000	78	Port I Pin MUX Control (PORT I)
0x4004_F000	79	—
0x4005_0000	80	Independent Real Time Clock (iRTC)
0x4005_1000	81	iRTC 32 Byte Standby RAM (iRTC_RAM)
0x4005_2000	82	—
0x4005_3000	83	Watchdog (WDOG)
0x4005_4000	84	—
0x4005_5000	85	Peripheral Crossbar (XBAR)
0x4005_6000	86	—
0x4005_7000	87	Quad Timer (QTMR) timer channel 0
0x4005_8000	88	Quad Timer (QTMR) timer channel 1
0x4005_9000	89	Quad Timer (QTMR) timer channel 2
0x4005_A000	90	Quad Timer (QTMR) timer channel 3
0x4005_B000	91	—
0x4005_C000	92	—
0x4005_D000	93	—
0x4005_E000	94	—
0x4005_F000	95	—
0x4006_0000	96	—
0x4006_1000	97	External Watchdog Monitor (EWM)
0x4006_2000	98	—
0x4006_3000	99	—
0x4006_4000	100	Module Clock Generator (MCG)
0x4006_5000	101	—
0x4006_6000	102	MHz Oscillator (OSC)
0x4006_7000	103	I^2C0
0x4006_8000	104	I^2C1

Table continues on the next page...

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

System 32-bit base address	Slot number	Module
0x4006_9000	105	
0x4006_A000	106	UART0
0x4006_B000	107	UART1
0x4006_C000	108	UART2
0x4006_D000	109	UART3
0x4006_E000	110	—
0x4006_F000	111	1.2 V Voltage Reference (VREF)
0x4007_0000	112	—
0x4007_1000	113	—
0x4007_2000	114	Comparator (CMP) [3]
0x4007_3000	115	—
0x4007_4000	116	—
0x4007_5000	117	SPI0
0x4007_6000	118	SPI1
0x4007_7000	119	—
0x4007_8000	120	—
0x4007_9000	121	—
0x4007_A000	122	—
0x4007_B000	123	Reset Control Module (RCM)
0x4007_C000	124	Low Leakage Wakeup Unit (LLWU)
0x4007_D000	125	Power Management Controller (PMC)
0x4007_E000	126	System Mode Controller (SMC)
0x4007_F000	127	Reserved
0x400F_F000	—	eGPIO (All ports: PTA to PTM) [4]

1. Alias of the eGPIO module.
2. All channels covered in one module.
3. This module has three comparators each with DAC and 8 channels.
4. This address range is for accessing GPIO through AIPS. For single cycle access, IOPORT should be used.

4.7 Private peripherals

Table 4-3. Private Peripherals

System 32-bit Address Range	Destination Slave
0xE000_0000–0xE000_DFFF	Reserved
0xE000_E000–0xE000_EFFF	System Control Space(SCS)
0xE000_E000–0xE000_E00F	Reserved
0xE000_E010–0xE000_E0FF	SysTick

Table continues on the next page...

Table 4-3. Private Peripherals (continued)

System 32-bit Address Range	Destination Slave
0xE000_E100–0xE000_ECFF	NVIC
0xE000_ED00– 0xE000_ED8F	System Control Block
0xE000_ED90–0xE000_EDEF	Reserved
0xE000_EDF0–0xE000_EEFF	Debug
0xE000_EF00–0xE000_EFFF	Reserved
0xE000_F000–0xE00F_EFFF	Reserved
0xE00F_F000 - 0xE00F_FFFF	M0+ Core ROM Table

4.8 Private Peripheral Bus (PPB) memory map

The PPB is part of the defined ARM bus architecture and provides access to select processor-local modules. Except MMAU, these resources are only accessible from the core; other system masters do not have access to them. The M0+ core does not support a true "External PPB". It has an IOPORT which is used to access the eGPIO.

All of the resources shown in the table below are standard ARM supplied blocks supporting the ARM Cortex-M0+ except for the miscellaneous control modules.

Table 4-4. PPB memory map

System 32-bit Address Range	Resource
0xF000_0000–0xF000_0FFF	Micro Trace Buffer (MTB) for program trace
0xF000_1000–0xF000_1FFF	MTB Data Watchpoint and Trace
0xF000_2000–0xF000_2FFF	Debug ROM Table
0xF000_3000–0xF000_3FFF	Miscellaneous Control Module (MCM)
0xF000_4000–0xF000_4FFF	Memory Mapped Arithmetic Unit (MMAU)
0xF000_5000–0xF000_5FFF	Memory Mapped Cryptographic Acceleration Unit (MMCAU)
0xF000_6000–0xF7FF_FFFF	Reserved
0xF800_0000–0xFFFF_FFFF	IOPORT (connected to eGPIO)[1]

1. This port accesses one port of eGPIO while the other port of eGPIO is connected to AIPS as normal IPS connection.

NOTE

The platform provides access control registers in the eGPIO module thereby securing the accesses to GPIO. Please refer to eGPIO chapter for more details.

4.8.1 GPIO accessibility in the memory map

The GPIO is multi-ported and can be accessed directly by the core with zero wait states at base address 0xF800_0000. It can also be accessed by the core and DMA masters through the cross bar/AIPS interface at 0x400F_F000 and at an aliased slot (15) at address 0x4000_F000. All BME operations to the GPIO space can be accomplished referencing the aliased slot (15) at address 0x4000_F000. Only some of the BME operations can be accomplished referencing GPIO at address 0x400F_F000.

Chapter 5
Clock Distribution

5.1 Introduction

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

NOTE

Refer to SCGCx register in SIM chapter for granular shutoff of modules.

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

In addition, there are various other module-specific clocks that have other alternate sources. Clock selection for most modules is controlled by the SOPT registers in the SIM module.

5.2 High-level clocking diagram

The following diagram shows how the overall clock generation structure operates.

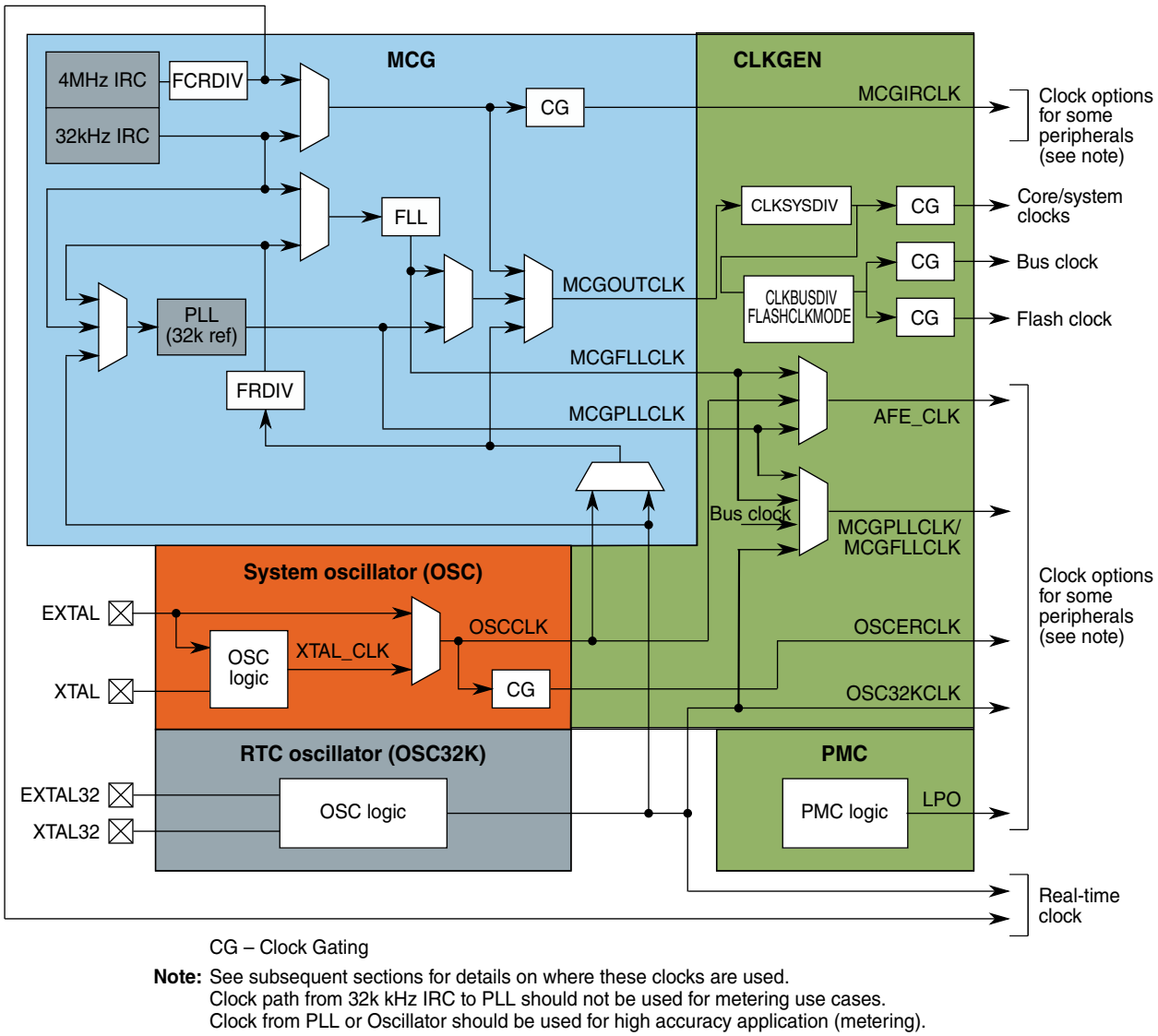


Figure 5-1. Clocking Diagram

Device's clocking architecture comprises of the following clock generators:

- 4 MHz and 32 kHz IRC Oscillators
- FLL
- PLL
- 1 kHz LPO

Additionally, the clocking logic provides for selecting any of the clock source or bypassing them to use the clocks directly from the oscillator. More details on configuring this logic is provided in the individual block chapters.

AFE Modulator Clock Generation: The PLL will drive the AFE modulator clock for metering applications and optionally drive the core, flash and system clocks for non-metering applications. When PLL is driving AFE clock, this clock should not be used for any other module including the core, bus, flash and other peripherals as this is necessary

to ensure good accuracy from AFE. If PLL clock is driving core, bus and flash clocks, there could be some degradation in performance from AFE. The AFE can optionally be clocked from MHz oscillator clock (provided it is less than 32 MHz - for crystal clock and 50 MHz - externally supplied clock) and FLL clock when AFE accuracy is not important (PLL can be disabled to save power in these cases). The final clock output (MCGPLLCLK for AFE) is prescaled (from 1 to 256) inside the AFE Wrapper to generate a clock with 50% duty cycle (must for AFE). See "AFE" section for more details.

NOTE

The 32 kHz IRC is not preferred to drive the PLL input reference when device is used for metering applications (IRC & FLL clocks are inaccurate clock sources for AFE). For general purpose use, this IRC & FLL can be used.

For neutral disconnect case, FLL & PLL can be optionally disabled to save power and IRC clock can be divided down to drive the AFE at reduced accuracy.

System Clocks Generation: The core, flash and bus clocks are generated using either clock source that is generating the MCGOUTCLK (PLL can optionally drive both MCGCLKOUT and clock to AFE; however, for best accuracy in AFE, PLL should drive clock to AFE only).

5.3 Clock definitions

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

Clock name	Description
Core clock	MCGOUTCLK divided by CLKDIVSYS clocks the ARM Cortex-M0+ core
System clock	MCGOUTCLK divided by CLKDIVSYS clocks the crossbar switch and bus masters directly connected to the crossbar. In addition, this clock is used for SPI0, SPI1, UART1 and UART3.
Bus clock	System clock divided by CLKDIVBUS clocks the bus slaves and peripheral (excluding memories) [1]
Flash clock	System clock divided by (CLKDIVBUS + FLASHCLKMODE) clocks the flash memory [2]
MCGIRCLK	MCG output of the slow or fast internal reference clock
MCGOUTCLK	MCG output of either IRC, MCGFLLCLK, MCGPLLCLK, or MCG's external reference clock (MHz OSC clock or iRTC OSC clock) that sources the core, system, bus and flash clock. It is also an option for the debug trace clock.

Table continues on the next page...

Clock name	Description
MCGPLLCLK	MCG output for AFE. Either of PLL clock, MHz OSC clock or FLL clock can be used to drive this clock output.
OSCCLK	System oscillator output of the internal MHz oscillator or sourced directly from EXTAL
OSCERCLK	System oscillator output sourced from OSCCLK that may clock some on-chip modules (e.g. WDOG, etc.)
ERCLK32K	System oscillator 32 kHz output for low range operation
LPO	PMC 1 kHz output
OSC32KCLK	RTC Oscillator 32 kHz output for iRTC

1. Bus clock can be system divided by 1, 2, 3 or 4.
2. Flash clock equal to Bus clock if FLASHCLKMODE = 0b0, or Bus clock divided by 2 if FLASHCLKMODE = 0b1.

5.3.1 Device clock summary

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

Table 5-1. Clock Summary

Clock name	Run mode clock frequency	VLPR mode clock frequency	Clock source	Clock is disabled when…
MCGOUTCLK	Up to 75 MHz	Up to 4 MHz	MCG	In all stop modes
Core clock	Up to 75 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all wait and stop modes
System clock	Up to 75 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all stop modes
Bus clock	Up to 25 MHz	Up to 1MHz	MCGOUTCLK clock divider	In all stop modes
Flash clock	Up to 25 MHz	Up to 1 MHz	MCGOUTCLK clock divider	In all stop modes
AFE Modulator Clock	30 kHz to 6.5 MHz	30 kHz to 6.5 MHz[1, 2]	Output of Prescaler in AFE Wrapper. See AFE section for details on all sources.	AFE is disabled
AFE Clock (Externally supplied clock)	30 kHz to 50 MHz	30 kHz to 6.5 MHz	AFE_CLK pin	AFE is disabled
Internal reference (MCGIRCLK)	4 MHz/32 kHz[3]	4 MHz/32 kHz[3]	MCG	MCG_C1[IRCLKEN] cleared, (clock disabled in any mode) or Stop mode and MCG_C1[IREFSTEN] cleared, or VLLS mode
External reference (OSCERCLK)	Up to 50 MHz (bypass), 32-40 kHz (crystal), or 4 -32 MHz (crystal)	Up to 8 MHz (bypass), 32-40 kHz (low-range crystal)	System OSC	System OSC's OSC_CR[ERCLKEN] cleared (clock disabled in any mode), or

Table continues on the next page...

Table 5-1. Clock Summary (continued)

Clock name	Run mode clock frequency	VLPR mode clock frequency	Clock source	Clock is disabled when…
				Stop mode and OSC_CR[EREFSTEN] cleared
External reference 32kHz (ERCLK32K)	30-40 kHz	30-40 kHz	System oscillator 32 kHz output for low range operation.	System OSC's OSC_CR[ERCLKEN] cleared
LPO	1 kHz	1 kHz	PMC	in VLLS0
SLCD clock	Up to 25 MHz	—	MCGOUTCLK, 32k IRC or OSC32KCLK as alternate clock	SLCD is disabled

1. When AFE is in low power mode, the maximum modulator clock can be 1.6 MHz
2. When AFE clock source is IRCLK, max frequency is 4 MHz.
3. Clock selectable by software. Default: 2 MHz (internally divided by 2 after reset)

5.4 Internal clocking requirements

The clock dividers are programmed via the SIM module's CLKDIV registers. Each divider is programmable from a divide-by-1 through divide-by-4 setting. The following requirements must be met when configuring the clocks for this device:

1. The core and system clock frequencies must be 75 MHz or slower.
2. The bus/flash clock frequency must be programmed to 25 MHz or less and an integer divide of the core clock.
3. The lowest frequency configured for bus clock should always be more than 32.768 kHz clock (preferably twice or more).
4. The PLL clock will be 14.65 MHz or less for the AFE. This clock is prescaled internal to AFE before being used.

The clock ratio of core:bus that will be support is 4:1, 3:1, 2:1 or 1:1. The clock ratio of bus : flash can be 1:1 or 2:1.

5.4.1 Clock divider values after reset

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

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

This gives the user flexibility for a fast boot with higher power consumption vs lower frequency and low-power boot option. The flash erased state defaults to fast clocking mode, because where the low power boot (FTFL_FOPT[LPBOOT]) bit resides in flash is at logic 1 in the flash erased state.

To enable the slow boot option program FTFL_FOPT[LPBOOT] to zero. After power up LPBOOT can be configured. Upon any system reset, the clock dividers return to this configurable reset state.

- Clock divider setting can be user-programmed on-the-fly without clock glitches. Control logic handles stalling and restarting the clocks. Supports divide-by-1 through divide-by-16.

5.4.2 VLPR mode clocking

They must be programmed prior to entering VLPR mode to guarantee:

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

5.4.3 Enable PLL in VLPR or VLPR and PSTOP1

The following steps are used to enable PLL in VLPR mode (after VLPR boot):

- Device boots in VLPR mode.
- Enable PMC band-gap by configuring corresponding register in D_PMC.
- Set SIM_CTRL_REG[PLLVLPEN]
- CLEAR the MCG LP bit MCG_C2[1]=0 and set the MCG PLLCLKEN bit
- Wait for MCG LOCK bit to set
- Once PLL clock is available configure AFE and start conversion

Steps to enable PLL in VLPR mode (after RUN boot):

- Device boots in RUN mode.
- Program SMC to enter VLPR mode
- Enable PMC band-gap by configuring corresponding register in D_PMC.
- Set SIM_CTRL_REG[PLLVLPEN] field

- CLEAR the MCG LP bit MCG_C2[1]=0 and set the MCG PLLCLKEN bit
- Wait for MCG LOCK bit to set
- Once PLL clock is available configure AFE and start conversion

Steps to keep PLL enabled in Partial Stop 2 mode:

- Set PLL_STPEN bit in MCG before entering Stop mode.
- Program STOPM = 000 and PSTOPO = 01 in SMC
- Set SLEEPDEEP bit in ARM to enter Partial stop Mode.
- Use async DMA feature for the use-case if required to read result registers of AFE

NOTE

- If PLL is already enabled in RUN and MCU is made to enter VLPR mode, the PLL will remain active.
- Some restrictions that the user software needs to care about:
  - If SIM_CTRL_REG[PLLVLPEN] field is set, software should ensure proper clock configuration before entering VLP mode as hardware checking is overridden.
  - In case user wants to switch to any low power mode (VLLSx, VLPS) from VLP mode, SIM_CTRL_REG[PLLVLPEN] field need to be cleared prior to mode transition to VLLSx, VLPS mode.
  - In VLP mode, when PLL is enabled, user should not disable the PMC bandgap.
  - RUN->VLPR(also with PSTOP1)->RUN
  - RUN->VLPR(+PSTOP1)->RESET are valid transition with the feature enabled.
  - If the feature is not required or if FLL needs to be enabled, SIM_CTRL_REG[PLLVLPEN] needs to be cleared.

5.5 Clock Gating

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

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

5.6 Module clocks

The following table summarizes the clocks associated with each module.

Table 5-2. Module clocks

Module	Bus interface clock	Internal clocks/ Protocol clocks	I/O interface clocks
Core modules			
ARM Cortex-M0+ core	System clock	Core clock	—
MMCAU	System clock	Core clock	
MMAU	System clock	Core clock	
NVIC	System clock	—	—
Single Wire Debug	—	—	SWD_CLK
System modules			
DMA	System clock	—	—
DMA Mux	Bus clock	—	—
Port control	Bus clock	LPO	—
Crossbar Switch	System clock	—	—
Peripheral bridges	System clock	Bus clock	—
MPU	System clock	—	—
LLWU, PMC, SIM	Flash Clock	LPO	—
Mode controller	Flash Clock	—	—
EWM	Bus clock	LPO/MCGIRCLK/ OSC32KCLK/ERCLK32K	—
Watchdog timer	Bus clock	LPO/OSC32KCLK/ MCGIRCLK/ERCLK32K	—
Peripheral Crossbar (XBAR)	Bus clock	—	—
Clocks			
MCG	Flash clock	MCGOUTCLK, MCGPLLCLK, MCGFLLCLK, MCGIRCLK, OSCERCLK	—
OSC	Bus clock	OSCERCLK	—
Memory and memory interfaces			
Flash Controller	System clock	Flash clock	—
Flash memory	Flash clock	Flash Oscillator Clock[1]	—
Security			
PCRC	Bus clock	—	—
RNGA	Bus clock	—	—
Analog			

Table continues on the next page...

Table 5-2. Module clocks (continued)

Module	Bus interface clock	Internal clocks/ Protocol clocks	I/O interface clocks
AFE	Bus clock	MCGPLLCLK, MCGFLLCLK, MCGIRCLK, TMR_0 output[2]	AFE_CLK
CMP	Bus clock	—	—
VREF	Flash clock	—	—
SAR ADC	Bus Clock	OSCERCLK	—
Timers			
PIT	Bus clock	—	—
LPTMR	Flash clock	LPO, OSCERCLK, MCGIRCLK, ERCLK32K	—
iRTC	Bus clock	EXTAL32, Fine Compensation Clock[3]	—
Quad Timer	Bus clock	Selectable via XBAR	—
PDB	Bus clock		
Communication interfaces			
SPI	System clock	—	SCK
I^2C	Bus clock	—	SCL
UART1, UART3	System clock	—	—
UART0, UART2	Bus clock	—	—
LPUART0	Bus clock	MCGPLLCLK, MCGFLLCLK, MCGIRCLK, OSCERCLK, OSC32KCLK	
Human-machine interfaces			
GPIO	System clock	—	—
Segment LCD	Bus Clock	OSC32KCLK, MCGIRCLK	—

1. Internal clock switches to Flash Clock when booting in VLPR and/or NVOPT[5] bit is asserted. In this case Flash clock is 1 MHz
2. Selection done inside AFE Wrapper
3. This clock can be 2 MHz or 4 MHz. Selection done via register bit in MCG

Chapter 6
Reset and Boot

6.1 Reset

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

6.1.1 System resets and sources

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

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

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

The various system reset sources are discussed in the following sub-sections.

6.1.1.1 Power-on reset (POR)

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

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

6.1.1.2 External pin reset ($\overline{\text{RESET}}$)

On this device, $\overline{\text{RESET}}$ pin has an internal pull-up device. Asserting $\overline{\text{RESET}}$ wakes the device from any low power mode.

NOTE

Using the LLWU module, RESET is always enabled as a wakeup source in VLLS mode (if RESET pin is not configured as GPIO). The wakeup will always cause a CPU reset flow on exit from these power modes.

6.1.1.2.1 Reset pin filter

The $\overline{\text{RESET}}$ pin supports digital filtering in all modes of operation. For VLLS3/2/1 modes, the LLWU provides an optional fixed digital filter running off the 1 kHz LPO clock. See the LLWU chapter for operation of this filter. During non-low leakage operation, there are two clock options for the $\overline{\text{RESET}}$ pin filter – the 1kHz LPO clock and the bus clock.

This $\overline{\text{RESET}}$ pin filter implemented in RCM logic includes a separate filter for each clock source. In Stop and VLPS operation this logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected.

There are several modes defined – See the RCM_RPFC register description in "RCM" module chapter for more details. Both filters are reset on POR, LVD, and wakeup from VLLS. The reset value for each filter defaults to off (non-detect).

The LPO filter is simple with a fixed filter value count of 3. There is also a synchronizer on the input signal that results in an associated latency (2 cycles). As such, it takes 5 cycles to complete a transition from low-to-high or high-to-low. The LPO Filter initializes to off (logic 1) when the LPO filter is not enabled.

The Bus Filter initializes to off (logic 1) when the Bus Filter not enabled. When the Bus Filter is enabled, the number of counts is controlled by RCM_RPFW[RSTFLTSEL[4:0].

6.1.1.3 Low-voltage detect (LVD) reset

This device includes a system to protect against low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and a low-voltage detect (LVD) circuit with a user-selectable trip voltage, either high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by the LVDSC1[LVDV] bits. The LVD system is always enabled in normal run, wait, and stop modes. The LVD system is disabled in VLPx, and VLLSx modes. Refer to Power Management Controller (PMC) chapter for more details.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting LVDSC1[LVDRE]. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage rises above the low voltage detection threshold.

The SRSL[LVD] bit is set following an LVD reset or POR.

6.1.1.4 Watchdog reset

The watchdog timer monitors the operation of the system by expecting periodic communication from the software, generally known as servicing (or refreshing) the watchdog. If this periodic refreshing does not occur, the watchdog issues a system reset. The WDOG reset causes the SRSL[WDOG] bit to set.

6.1.1.5 Low leakage wakeup (LLWU) reset

The LLWU allows 24 external pins, the $\overline{\text{RESET}}$ and NMI pin, and up to six internal peripherals to wake the MCU from VLLSx power modes. The LLWU module is only functional in VLLSx power modes. In these modes, any VLLS mode exits via a wakeup or reset event, the SRSL[WAKEUP] bit in mode controller module is set indicating the low leakage mode was active prior to the last system reset flow. Using the $\overline{\text{RESET}}$ pin to trigger an exit from VLLS results in the SRSL[PIN] bit being set as well. Refer to the mode controller chapter for more details.

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

NOTE

Pin wakeup and error condition flags are cleared in the LLWU and module wakeup flags are required to be cleared in the peripheral module. Refer to the individual peripheral specifications for more information.

6.1.1.6 Stop acknowledgement reset

Prior to entering any low power mode where module clocks are being gated, the SMC expects every module to provide a stop_ack signal which indicates that the module can be put into low power mode. If such a response is expected from a module and is not received till a certain timeout period, a reset is generated and the reset sequence is executed and chip starts over again. This reset generation due to non-receipt of stop_ack is call the "stop ack reset".

Stop acknowledgement is generated by modules that want to delay entry into low power mode until their current operation is complete. Peripherals that may generate stop_ack signal are:

- I2C—Will complete current transfer and enter low power mode if configured to halt in the low power mode
- Other modules include—Flash, Port control blocks, PIT, DMA

6.1.1.7 Multipurpose clock generator loss-of-clock (LOC) reset

The MCG includes a clock monitor. The clock monitor resets the device when the following conditions are met:

- The clock monitor is enabled (MCG_C6[CME] = 1)
- The MCG's external reference clock falls outside of the expected frequency range, depending on the MCG_C2[RANGE] bit

The RCM_SRSL[LOC] bit is set to indicate the error.

6.1.1.8 Software reset (SW)

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

6.1.1.9 Lockup reset (LOCKUP)

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

The LOCKUP condition causes a system reset and also causes SRSH[LOCKUP] bit to set.

6.1.1.10 Debug reset (DAP reset)

The chip can be reset while in debug mode by generating a DAP reset. The reset command is sent to the DAP (Debug Access Port) which when decoded generates a system reset and the chip executes the reset sequence. Refer to Debug chapter for more details on this reset.

6.2 Boot

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

6.2.1 Boot sources

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

6.2.2 Boot options

The device boots up in functional mode by default after any reset or power up. The device boots up using the internal flash as the boot source.

6.2.3 FOPT boot options

The flash option register (FOPT) in flash memory module (FTFL) allows the user to customize the operation of the MCU at boot time. The register contains read-only bits that are loaded from the NVM's option byte in the flash configuration field. The user can

reprogram the option byte in flash to change the FOPT values that are used for subsequent resets. For more details on programming the option byte, refer to the flash memory chapter.

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

Table 6-1. Flash Option Register (FTFL_FOPT) Bit Definitions

Bit Num	Field	Value	Definition
7-6	Reserved	-	Reserved for future expansion
5	CLK_SRC[1]	1	Internal clock source used by Flash
		0	Externally supplied clock used by Flash
4	Reserved	-	Reserved for future expansion
3	EXE_MODE[2]	1	Execution Mode is VLPR Mode
		0	Execution Mode is RUN Mode
2	NMI_EN	1	NMI_B pin is enabled. NMI functionality is enabled
		0	NMI_B pin is disabled. NMI functionality is disabled
1	Reserved	-	Reserved for future expansion
0	LPBOOT	0	Slow boot: SYSDIV and SYSCLKMODE values in SIM_CLKDIV1 register are auto-configured at reset exit for higher divide values that produce lower power consumption at reset exit. • Core and system clock divider (SYS_DIV) is set to 0x7 (divide by 8)
		1	Fast boot: SYSDIV and SYSCLKMODE values in SIM_CLKDIV1 register are auto-configured at reset exit for higher frequency values that produce faster operating frequencies at reset exit. • Core and system clock divider (SYS_DIV) is set to 0x0 (divide by 1)

1. This bit has effect only in RUN mode boot up. Ignored in VLPR boot.
2. This bit will not change mode for boot automatically and is different from the RTC Boot Override bit. This bit is for software so that a startup code can change mode based on setting of this bit whenever it is executed on every boot up

6.2.4 Boot sequence

6.2.4.1 Boot Up Process

NOTE

The steps described are common to boot in RUN or boot in VLPR unless explicitly mentioned for that step.

NOTE

Boot will happen in RUN mode unless overridden using RTC bit (Hardware override - takes effect from next reset) or

NVOPT bit (EXE_MODE—Software override). EXE_MODE does not change mode automatically but software can check the setting of this bit and change mode accordingly.

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

1. A system reset is held on internal logic, the $\overline{\text{RESET}}$ pin is driven out low
2. SMC enters RUN or VLPR mode (depending on override bit in RTC) keeping LVD enabled in either case.
3. The MCG is enabled in its default clocking mode in which this is needed to ensure low startup current during Missing Neutral Scenario Recommendations.
   - System will boot-up in BLPI (Bypass Low Power Internal) with 4 MHz IRC being used and FLL disabled.
4. All module clocks except RTC clock are disabled. iRTC is the only module that can function after POR as it is in separate power domain (VBAT). SMC and modules needed for boot will be active.
5. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Reset Control logic continues to drive the RESET pin out low
6. Early in reset sequencing the NVM option byte is read and stored to FOPT. If the CLK_SRC bit is programmed clear, the Flash initialization switches to slower clock (when in RUN mode). When in VLPR mode, the Flash initialization happens on slower clock from beginning and does not depend on CLK_SRC bit value. If the bits associated with LPBOOT are programmed for an alternate clock divider reset value, the system/core clock is switched to an even slower clock speed.
7. When Flash Initialization completes, the $\overline{\text{RESET}}$ pin is observed. If $\overline{\text{RESET}}$ continues to be asserted (an indication of a slow rise time on the $\overline{\text{RESET}}$ pin or external drive in low), the system continues to be held in reset. Once the $\overline{\text{RESET}}$ pin is detected high, the system is released from reset.
8. At release of system reset, LVD is disabled provided boot happens in VLPR mode (LVD not disabled in RUN mode boot).
9. When the system exits reset, the processor sets up the stack, program counter (PC), and link register (LR). The processor reads the start SP (SP_main) from vector-table offset 0. The core reads the start PC from vector-table offset 4. LR is set to 0xFFFF_FFFF.
11. If the NMI input is low and the NMI_EN bit equals 1, this results in an NMI interrupt. The processor executes an Exception Entry and reads the NMI interrupt handler address from vector-table offset 8. The CPU begins execution at the NMI interrupt handler.

12. If NMI input is high when the system exits reset then the CPU begins execution at the PC location.

Subsequent system resets follow this reset and boot flow.

Chapter 7
Power Management

7.1 Introduction

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

7.2 Power Modes

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

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

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

Depending on the stop requirements of the user application, a variety of stop modes are available that provide state freeze (by clock gating), partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation.

The following table compares the various power modes available.

Table 7-1. Chip power modes

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

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
Normal Wait (WAIT) - via WFI	Allows peripherals to function while the core is in sleep mode, thus reducing power. NVIC remains sensitive to interrupts; peripherals continue to be clocked. Flash can be optionally kept off to reduce leakage current further.	Sleep	Interrupt
Partial STOP2 (PSTOP2)	Core and System clocks are gated while Bus clock is active. Allows peripherals to function. Clock generators and regulators are fully functional. AWIC is used to wake up from async. interrupt. Less power consumption than WAIT. Can be entered from both RUN & VLPR modes.	Sleep Deep	Interrupt
Partial STOP1 (PSTOP1)	Core, System and Bus clocks are gated. Peripherals working on different clock source will continue to operate. Clock generators and regulators are fully active. AWIC is used to wake up from async. interrupt. Less power consumption than PSTOP2 but higher than STOP. Can be entered from both RUN & VLPR modes.	Sleep Deep	Interrupt
Normal Stop (STOP) - via WFI	Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. NVIC is disabled; AWIC is used to wake up from interrupt; peripheral clocks are stopped.	Sleep Deep	Interrupt
Very Low Power Run (VLPR)	On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; internal oscillator provides a low power 4 MHz source for the core, the bus and the peripheral clocks.	Run	Interrupt
Very Low Power Wait (VLPW) - via WFI	Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.	Sleep	Interrupt
Very Low Power Stop (VLPS) - via WFI	Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. Peripheral clocks are stopped, but LPTimer, iRTC, CMP, can be used. NVIC is disabled (FCLK = OFF); AWIC is used to wake up from interrupt. On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. All SRAM is operating (content retained and I/O states held).	Sleep Deep	Interrupt
VLLS3 (Very Low Leakage Stop3)	Most peripherals are powered off but LLWU, LPTMR, iRTC, CMP can be used. NVIC is disabled; LLWU is used to wake up. Full SRAM remains powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[1]
VLLS2 (Very Low Leakage Stop2)	Most peripherals are powered off but LLWU, LPTMR, iRTC, CMP can be used. NVIC is disabled; LLWU is used to wake up. 8 KB of SRAM is powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[1]
VLLS1 (Very Low Leakage Stop1)	Most peripherals are powered off but LLWU, LPTMR, iRTC, CMP can be used. NVIC is disabled; LLWU is used to wake up; Internal regulator is disabled; LPO & Brown-out are powered All of SRAM is powered off. The 32-byte VBAT register file remain powered for customer-critical data.	Sleep Deep	Wakeup Reset[1]
VLLS0 (Very Low Leakage Stop 0)	LLWU, iRTC can be used. All SRAM powered off. NVIC disabled; Internal regulator is disabled; LPO is OFF. Brown-out detection is configurable	Sleep Deep	Wakeup Reset[1]

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
BAT (backup battery only)	The chip is powered down except for the VBAT supply. The iRTC and the 32-byte VBAT register file for customer-critical data remain powered. This mode can be entered when VDD gets removed.	Off	Power-up Sequence

1. Follows the reset flow with the LLWU interrupt flag set for the NVIC.

7.3 Entering and exiting power modes

The WFI instruction invokes wait and stop modes for the chip. The processor exits the low-power mode via an interrupt (Reset causes chip to go into Run/VLPR mode - current boot-up power mode). The NVIC controls what peripherals can cause interrupts.

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

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

NOTE

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

The oscillator cannot be re-enabled before the ACKISO bit is cleared and must be reconfigured after the hold is released.

7.4 Power mode transitions

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

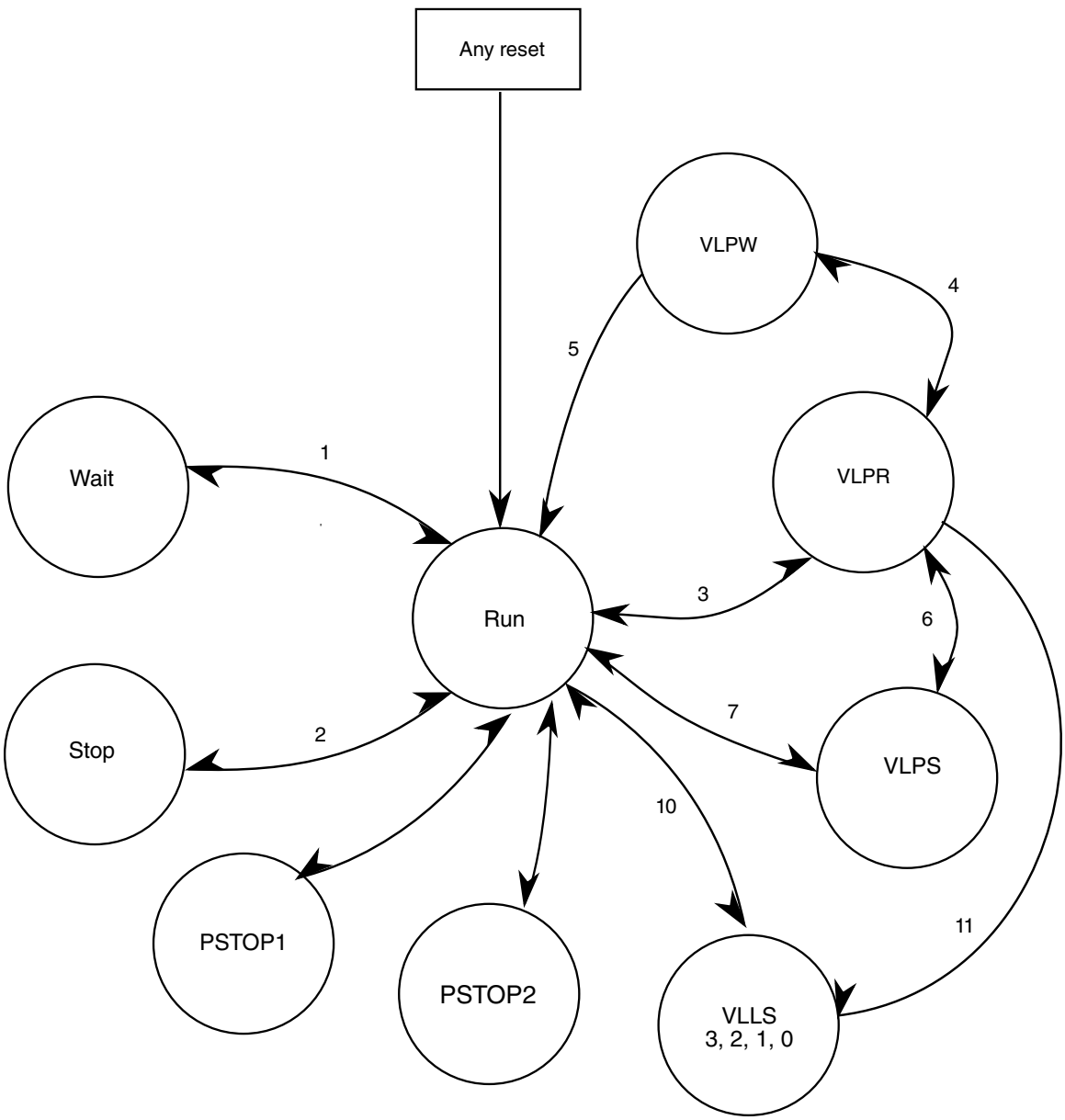


Figure 7-1. Power mode state transition diagram

7.5 Power modes shutdown sequencing

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

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

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

- Shuts off Core Clock and System Clock to the ARM Cortex-M0+ core immediately.
- Polls stop acknowledge indications from the non-core crossbar master (DMA), supporting peripherals and the Flash Controller for indications that System Clock, Bus Clock and/or Flash Clock need to be left enabled to complete a previously initiated operation, effectively stalling entry to the targeted low power mode. When all acknowledges are detected, System Clock, Bus Clock and Flash Clock are turned off at the same time.
- MCG and Mode Controller shut off clock sources and/or the internal supplies driven from the on-chip regulator as defined for the targeted low power mode.

In wait modes, most of the system clocks are not affected by the low power mode entry. The Core Clock to the ARM Cortex-M0+ core is shut off. Some modules support stop-in-wait functionality and have their clocks disabled under these configurations.

7.6 Module Operation in Low Power Modes

The following table illustrates the functionality of each module while the chip is in each of the low power modes. Number ratings (such as 4 MHz and 1 Mbps) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- FF = Full Functionality. In VLPR and VLPW the system frequency is limited, limited peripherals can be chosen to work and hence mentioned as Optionally Active.
- static = Module register states and associated memories are retained (clock to those blocks are gated).
- powered = Memory is powered to retain contents.
- low power = Flash/Memory or Regulator has a low power state that retains configuration registers to support faster wakeup.

- OFF = Modules are powered off; module is in reset state upon wakeup (VLLSx modes)
- ON = Modules are powered on (VLLSx modes)
- wakeup = Modules can serve as a wakeup source for the chip.
- Optionally Active = Software can configure if module will be "FF" or "static" in particular low power mode

Table 7-2. Module operation in low power modes

Modules	WAIT/PSTOP2	STOP/PSTOP1	VLPR	VLPW	VLPS	VLLSx
Core modules						
NVIC	FF	static	FF	FF	static	OFF
SWD	Optionally Active	Optionally Active	Optionally Active	Optionally Active	Optionally Active	OFF
MMAU	Optionally Active	Optionally Active	Optionally Active	Optionally Active	Optionally Active	OFF
MMCAU	Optionally Active	Optionally Active	Optionally Active	Optionally Active	Optionally Active	OFF
System modules						
Mode Controller	FF	FF	FF	FF	FF	ON
AWIC	static	FF	static	static	FF	OFF
LLWU[1]	static	static	static	static	static	ON[2]
Regulator	ON	ON	low power	low power	low power	low power in VLLS2/3, OFF in VLLS0/1
LVD	ON	ON	disabled	disabled	disabled	disabled
Brown-out Detection	ON	ON	ON	ON	ON	ON in VLLS1/2/3, optionally disabled in VLLS0[3]
DMA[4]	Optionally active	Optionally active[5]	Optionally active	Optionally active	Optionally active[5]	OFF
Watchdog	FF	Optionally active	FF	Optionally active	Optionally active	OFF
EWM	Optionally active	static	Optionally active	Optionally active	static	OFF
XBAR	Optionally active	Optionally active	Optionally active	Optionally Active	Optionally active[6]	OFF
Clocks						
1kHz LPO	FF	FF	FF	FF	FF	ON in VLLS1/2/3, OFF in VLLS0
System Oscillator (OSC)	Optionally active OSCERCLK	Optionally active OSCERCLK	Optionally active OSCERCLK	Optionally active OSCERCLK	Optionally active OSCERCLK	limited to low range/low power in VLLS1/2/3, OFF in VLLS0[7]
MCG[8]	FF	static - IRCLK optional; FLL optionally on but gated	4 MHz IRC	4 MHz IRC	static - no clock output	OFF
PLL 32k	Optionally active	Optionally active	Optionally active	Optionally active	Optionally active	OFF

Table continues on the next page...

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

Modules	WAIT/PSTOP2	STOP/PSTOP1	VLPR	VLPW	VLPS	VLLSx
Core clock	static	static	4 MHz max	static	static	OFF
System clock	FF[9]	static	4 MHz max	4 MHz max	static	OFF
Bus clock	FF	static	1 MHz max	1 MHz max	static	OFF
Memory and Memory Interfaces						
Flash	Optionally in low power or FF (no program)	powered	1 MHz max access - no program	low power	low power	OFF
SRAM	low power	low power	low power	low power	low power	low power in VLLS3, 8 KB powered in VLLS2, OFF in VLLS0/1
iRTC register file[10]	powered	powered	powered	powered	powered	powered
Communication interfaces						
UART	Optionally active[11]	static, wakeup on edge	Optionally active 125 kbps for UART1/UART3 if active 62.5 kbps for UART0/UART2 if active	Optionally active 125 kbps for UART1/UART3 if active 62.5 kbps for UART0/UART2 if active	static, wakeup on edge	OFF
LPUART	Optionally active	Async Operation	Optionally active	Optionally active	Async Operation	OFF
SPI	Optionally active[12]	static, slave mode receive	Optionally active 1 Mbps if active	Optionally active 1 Mbps if active	static, slave mode receive	OFF
I^2C	Optionally active	static, address match wakeup	Optionally active 100 kbps if active	Optionally active 100 kbps if active	static, address match wakeup	OFF
Security						
CRC	Optionally active	static	Optionally active	Optionally active	static	OFF
RNGA	Optionally active	static	Optionally active	Optionally active	static	OFF
Timers						
PIT	Optionally active	static	Optionally active	Optionally active	static	OFF
LPTMR	Optionally active	Optionally active	Optionally active	Optionally active	Optionally active	FF[13]
TMR	Optionally active	Optionally active[14]	Optionally active	Optionally active	Optionally active	OFF
RTC, POR and 32kHz OSC[10]	FF	FF	FF	FF	FF	FF
PDB	Optionally active	static	Optionally active	Optionally active	static	OFF
Analog						
16-bit SAR ADC	Optionally active	Optionally active	Optionally active	Optionally active	Optionally active	OFF

Table continues on the next page...

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

Modules	WAIT/PSTOP2	STOP/PSTOP1	VLPR	VLPW	VLPS	VLLSx
		ADC internal clock only			ADC internal clock only	
CMP[15]	Optionally active	HS or LS compare	Optionally active	Optionally active	HS or LS compare	LS compare in VLLS1/2/3, OFF in VLLS0
AFE	Optionally active	Optionally active[16]	Optionally active[17]	Optionally active	Optionally active[18]	OFF
VREF[19]	Optionally active	Optionally active[20]	Optionally active[20]	Optionally active[20]	Optionally active[20]	OFF
Human-machine interfaces						
GPIO	wakeup	wakeup	Optionally active	wakeup	wakeup	OFF, pins latched
Segment LCD[21]	Optionally active	Optionally active	Optionally active	Optionally active	Optionally active	Optionally active.[22, 23]

1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
2. Since LPO clock source is disabled, filters will be bypassed during VLLS0
3. The VLLSCTRL[PORPO] bit in the SMC module controls this option.
4. Async DMA request allows DMA to operate in low power modes without waking up CPU.
5. DMA can be used if a module generates Async DMA request (e.g. AFE)
6. Configured signal paths are fully functional. Support interrupt and async DMA request generation.
7. In VLLSx mode, design does not guarantee operation of system oscillator in high freq/gain.
8. SOC cannot enter in VLPR mode until the LP bit in MCG_C2 register is set to 1.
9. System clock, clock used by core will be gated. Enabled only for active peripherals
10. These components remain powered in VBAT power mode.
11. UART1 and UART3 cannot function under PSTOP2 mode
12. SPI0 and SPI1 cannot function under PSTOP2 mode
13. System OSC and LPO clock sources are not available in VLLS0
14. TMR module logic keep operation if the clock source is enabled. TMR register access may be limited without BUSCLK.
15. CMP in stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in VLLSx only supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled & filtered modes of operation are not available while in stop, VLPS, or VLLSx modes.
16. As long as modulator clock is not gated, AFE can work.
17. Will operate in low power mode of AFE for missing neutral scenario.
18. PLL is not available in VLPS. AFE can run off other clock sources.
19. VREF can be configured to operate in low power buffer mode (cannot drive VREF pin)
20. Can be configured to work in low power buffer mode
21. LCD IOs will continue to drive last value being driven before entering low power mode.
22. End of Frame wakeup not supported in VLLSx.
23. In VLPS and VLLSx modes, SLCD can be configured to continue displaying the current LCD panel contents.

7.7 Clocking modes

Information found here describes the various clocking modes supported on this device.

7.7.1 Partial Stop

Partial Stop is a clocking option that can be taken instead of entering Stop mode and is configured in the SMC Stop Control Register (SMC_STOPCTRL). The Stop mode is only partially entered, which leaves some additional functionality alive at the expense of higher power consumption. Partial Stop can be entered from either Run mode or VLP Run mode.

When configured for PSTOP2, only the core and system clocks are gated and the bus clock remains active. The bus masters and bus slaves clocked by the system clock enter Stop mode, but the bus slaves clocked by bus clock remain in Run (or VLP Run) mode. The clock generators in the MCG and the on-chip regulator in the PMC also remain in Run (or VLP Run) mode. Exit from PSTOP2 can be initiated by a reset, an asynchronous interrupt from a bus master or bus slave clocked by the system clock, or a synchronous interrupt from a bus slave clocked by the bus clock. If configured, a DMA request (using the asynchronous DMA wakeup) can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP2.

When configured for PSTOP1, both the system clock and bus clock are gated. All bus masters and bus slaves enter Stop mode, but the clock generators in the MCG and the on-chip regulator in the PMC remain in Run (or VLP Run) mode. Exit from PSTOP1 can be initiated by a reset or an asynchronous interrupt from a bus master or bus slave. If configured, an asynchronous DMA request can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP1.

PSTOP1 is functionally similar to Stop mode, but offers faster wake-up at the expense of higher power consumption. Another benefit is that it keeps all of the MCG clocks enabled, which can be useful for some of the asynchronous peripherals that can remain functional in Stop modes.

7.7.2 DMA Wakeup

The DMA can be configured to wake the device on a DMA request whenever it is placed in Stop mode. The wake-up is configured per DMA channel and is supported in Compute Operation, PSTOP, STOP, and VLPS low power modes.

When a DMA wake-up is detected in PSTOP, STOP or VLPS then the device will initiate a normal exit from the low power mode. This can include restoring the on-chip regulator and internal power switches, enabling the clock generators in the MCG, enabling the system and bus clocks (but not the core clock) and negating the stop mode signal to the bus masters and bus slaves. The only difference is that the CPU will remain in the low power mode with the CPU clock disabled.

During Compute Operation, a DMA wake-up will initiate a normal exit from Compute Operation. This includes enabling the clocks and negating the stop mode signal to the bus masters and bus slaves. The core clock always remains enabled during Compute Operation.

Since the DMA wakeup will enable the clocks and negate the stop mode signals to all bus masters and slaves, software needs to ensure that bus masters and slaves that are not involved with the DMA wake-up and transfer remain in a known state. That can be accomplished by disabling the modules before entry into the low power mode or by setting the Doze enable bit in selected modules.

Once the DMA request that initiated the wake-up negates and the DMA completes the current transfer, the device will transition back to the original low-power mode. This includes requesting all non-CPU bus masters to enter Stop mode and then requesting bus slaves to enter Stop mode. In STOP and VLPS modes, MCG and PMC would then also enter their appropriate modes.

NOTE

If the requested DMA transfer cannot cause the DMA request to negate, then the device will remain in a higher power state until the low power mode is fully exited.

An enabled DMA wake-up can cause an aborted entry into the low power mode, if the DMA request asserts during the stop mode entry sequence (or reentry if the request asserts during a DMA wakeup) and can cause the SMC to assert its Stop Abort flag. Once the DMA wake-up completes, entry into the low power mode will restart.

An interrupt that occurs during a DMA wake-up will cause an immediate exit from the low power mode (this is optional for Compute Operation) without impacting the DMA transfer.

A DMA wake-up can be generated by either a synchronous DMA request or an asynchronous DMA request. Not all peripherals can generate an asynchronous DMA request in stop modes, although in general if a peripheral can generate synchronous DMA requests and also supports asynchronous interrupts in stop modes, then it can generate an asynchronous DMA request.

7.7.3 Compute Operation

Compute Operation is an execution or compute-only mode of operation that keeps the CPU enabled with full access to the SRAM and Flash read port, but places all other bus masters and bus slaves into their stop mode. Compute Operation can be enabled in either Run mode or VLP Run mode.

NOTE

Do not enter any Stop mode without first exiting Compute Operation.

Because Compute Operation reuses the Stop mode logic (including the staged entry with bus masters disabled before bus slaves), any bus master or bus slave that can remain functional in Stop mode also remains functional in Compute Operation, including generation of asynchronous interrupts and DMA requests. When enabling Compute Operation in Run mode, module functionality for bus masters and slaves is the equivalent of STOP mode. When enabling Compute Operation in VLP Run mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. The MCG, PMC, SRAM, and Flash read port are not affected by Compute Operation, although the Flash register interface is disabled.

During Compute Operation, the AIPS peripheral space is disabled and attempted accesses generate bus errors. The private peripheral space remains accessible during Compute Operation, including the MCM, NVIC, IOPORT, and SysTick. Although access to the GPIO registers via the IOPORT is supported, the GPIO Port Data Input registers do not return valid data since clocks are disabled to the Port Control and Interrupt modules. By writing to the GPIO Port Data Output registers, it is possible to control those GPIO ports that are configured as output pins.

Compute Operation is controlled by the CPO register in the MCM (MCM_CPO), which is only accessible to the CPU. Setting or clearing MCM_CPO[CPOREQ] initiates entry or exit into Compute Operation. Compute Operation can also be configured to exit automatically on detection of an interrupt, which is required in order to service most interrupts. Only the core system interrupts (exceptions, including NMI and SysTick) and any edge-sensitive interrupts can be serviced without exiting Compute Operation.

- When entering Compute Operation, the CPOACK status field in the CPO register of MCM module (MCM_CPO[CPOACK]) indicates when entry has completed.
- When exiting Compute Operation in Run mode, MCM_CPO[CPOACK] negates immediately.
- When exiting Compute Operation in VLP Run mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means that MCM_CPO[CPOACK] is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wake-up is also supported during Compute Operation and causes MCM_CPO[CPOACK] to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wake-up, the device transitions back into Compute Operation.

7.7.4 Peripheral Doze

Several peripherals support a Peripheral Doze mode, where a register bit can be used to disable the peripheral for the duration of a low-power mode. The flash memory can also be placed in a low-power state during Peripheral Doze via a register bit in the SIM.

Peripheral Doze is defined to include all of the modes of operation listed below.

- The CPU is in Wait mode.
- The CPU is in Stop mode, including the entry sequence and for the duration of a DMA wakeup.
- The CPU is in Compute Operation, including the entry sequence and for the duration of a DMA wakeup.

Peripheral Doze can therefore be used to disable selected bus masters or slaves for the duration of WAIT or VLPW mode. It can also be used to disable selected bus slaves immediately on entry into any stop mode (or Compute Operation), instead of waiting for the bus masters to acknowledge the entry as part of the stop entry sequence. Finally, it can be used to disable selected bus masters or slaves that should remain inactive during a DMA wakeup.

If the flash memory is not being accessed during WAIT and PSTOP modes, then the Flash Doze mode can be used to reduce power consumption, at the expense of a slightly longer wake-up when executing code and vectors from flash. It can also be used to reduce power consumption during Compute Operation when executing code and vectors from SRAM.

7.8 Clock Gating

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

Chapter 8
Security

8.1 Introduction

This device has been designed to consider aspects of system reliability that includes robust watchdog mechanism with enhanced features to satisfy safely standards IEC 60730 for the appliance market. These are also useful for other applications like medical where fault tolerance is important.

This micro-controller also includes an external watchdog monitor that can monitor external events like power supply.

8.2 External Watchdog Monitor

This device includes additional watchdog Monitor apart from the main Watchdog (WDOG) to monitor external circuits as well as the MCU software flow. This provides a back-up mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM if allowed to time-out, provides an independent output pin (EWM_out) that, when asserted, resets or places an external circuit into a safe mode. This can be very useful for example to monitor external supplies if they don't behave in a specific way, switch off a relay to make a critical signal line go open circuit.

EWM in conjunction with the standard internal watchdog that is included on the MCU, provides additional safety for critical circuits.

8.2.1 EWM counter

The 8-bit EWM counter if fed from a clock source that is independent of the CPU clock source. As the preferred time-out is between 1 ms and 100 ms, the actual clock source should be in the kHz range. The clock source comes from the same 1 kHz LPO clock that feeds the independent clocked watchdog.

The 8-bit ripple counter is reset to 0x00 after a CPU reset or an EWM refresh cycle. The 8-bit ripple counter value is not accessible to the CPU.

The EWM 8-bit compare registers are write once after a CPU reset and cannot be modified until another CPU reset occurs. The EWM compare registers are used to create a time window, when the CPU services/refreshes the EWM module.

If the CPU services the EWM when the counter value lies between EWM Compare Low Value (EWMxCMPL) and EWM Compare High Value (EWMxCMPH) value, the counter is reset to 0x00.

If the CPU executes a EWM service/refresh outside this time window, EWM_out is asserted.

8.2.2 EWM_out signal

The EWM_out is a digital output signal used to gate an external circuit (application specific) that control safety critical functions. For example, the EWM_out can be connected to the high voltage transistors circuits that control an AC motor in large appliance or to the power supply on board.

The EWM_out signal is de-asserted when the EWM is being regularly serviced by the CPU, indicating that the application code is executing as expected. When an error condition occurs, the EWM_out is asserted.

Once the EWM_out pin is asserted, it can only be de-asserted by forcing a MCU reset.

8.2.3 EWM_in signal

The EWM_in is a digital input signal that allows an external circuit to control the EWM_out signal. For example, in the application, an external circuit monitors a safety specific feature, and if there is fault with this circuit’s behavior, it can then actively initiate the EWM_out signal that controls the gating circuit. If the EWM is connected to external Power supply, EWM_In can indicate failure or problem with the supply thus asserting the feedback signal (EWM_out). Application may choose to reset in this case.

8.3 Robust Watchdog for Improved System Reliability

For the metering application, it is of utmost importance that the system must display a high degree of fault tolerance, so that if and when faults like software crashes happen, it is able to recover quickly and bring itself into a safe state. With the introduction of the IEC 60730 standards, it is required that even automatic electronic controls in household appliances ensure safe and reliable operation of their component.

Applications like metering, factory automation, and appliance can have severe EMI or noisy environment that can result in data corruption or even system failure. When such data corruption occurs, program execution can be affected as the program counter might have been modified. Modification of the program code memory or a read of wrong data from code memory can result in a totally different and unintended instruction getting executed. Thus, program flow or the program code itself gets modified, i.e. code runaway, and the system can enter an unknown state where its behavior is unpredictable.

Once the program flow takes an unexpected branch, the system can start behaving unpredictably, which is unacceptable for a safety critical system. Thus a lot of standards now require system operation to be immune to EMI so as to have reliable system operation. Watchdog needs to monitor the system so as to detect such system failures and take action to bring the system into a safe/known state.

There have been some unique features added to the WDOG timer.

8.3.1 32-bit programmable timeout Period

For system flexibility it is important to have programmable timeout period. To generate timeout values ranging from 1 ms to 1 second, the length of the watchdog timer has to be chosen carefully. The frequency of the clock source for the watchdog could vary widely from a few kHz (say an on-chip RTC oscillator) to hundreds of MHz (system clock) that may affect the width of counter. The following figure shows timeout values possible with 8, 16, 24 and 32 bit timers, for different, practical clock frequencies.

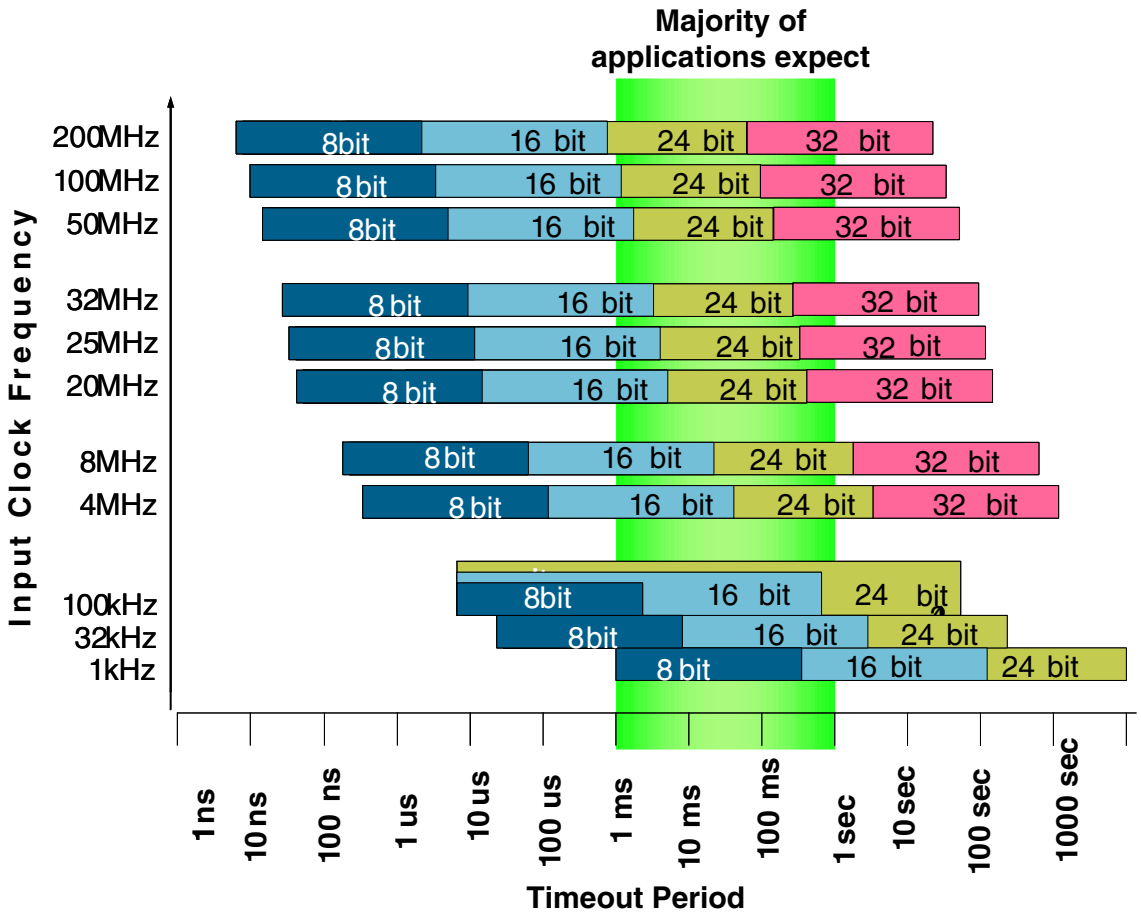


Figure 8-1. Possible Timeouts (Log scale)

The vertical green band marks out a range of timeouts which cover the 1 ms to 1 second range. As can be observed, a 32 bit counter is required to cover all clock frequencies and the expected range of watchdog timeouts.

8.3.2 Independent Clock Source

The Watchdog module clock can be sourced from either 1 kHz LPO clock or ERCLK32K.

8.3.3 Write Protection

Watchdogs generally have several control and configuration register bits, which are used to influence its operation, for example a bit to disable or enable the watchdog. Since these bits have a direct impact on the watchdog's operation, it is of prime interest to make sure they are not modified unintentionally. To achieve this objective a write protection scheme is implemented.

One of the better write-protection schemes is to have a password style protection on the register bits, where the password is a sequence of two particular values. However, this scheme allows any amount of time to elapse in between the write of the two values, which means that the chances of runaway code managing to accidentally replicate the password are high. If the writes of the two values are spaced far apart in the code, it could happen that after the write of the first value the code runs away in an unintended direction, causes havoc, and then after enough number of iterations, branches to the location of the write of the second value.

The WDOG places a restriction on the time gap between the writes of the two values, thereby reducing chances of runaway code being able to "unlock" the registers for writing and possibly disabling the watchdog. By placing a limit on the time gap, where the limit is just equal to the time it takes for the CPU to fetch and execute the write instruction for the second value, the user is forced to place the write instructions for the two values one after the other in the code (as assembly instructions). Now if there is a runaway after the execution of the first write, there is no time left for the code to possibly return and execute the instruction writing the second value of the sequence. This makes the refresh sequence more unique because it minimizes the chance of the sequence being replicated by runaway code.

If the gap between the two words of the password is more than the programmable number (default 10) of system bus clock cycles, the watchdog infers an exception and resets the system. In addition, the amount of time for which the registers stay "unlocked" is limited too and roughly equal to the time it takes for these registers to be configured once, after which they are "locked" again. This write protection is in effect from right after system reset, leaving no room for runaway code to "sneak in" and change the watchdog's configuration.

NOTE

To prevent unintended modification of the watchdog's control and configuration register bits, the user is allowed to update them only within a period of N bus clock cycles after unlocking. This window period is known as the Watchdog Configuration Time (WCT).

8.3.4 Robust Refresh Mechanism

The WDOG's refresh scheme is a sequence of two values (0xA602 followed by 0xB480), but the difference from watchdogs being used on existing S08/Coldfire parts is that it places a limit on the time that can elapse between the write of the two values. If the first value of the sequence is written and not followed by the second one within a certain number of system bus clock cycles, the watchdog infers an exception and resets the system.

8.3.5 Windowed Refresh

The WDOG has an option for a windowed refresh, as opposed to the normal refresh. The principle behind the windowed refresh is that watchdog can be refreshed only in a particular window of its timeout period. This window is defined by points in time, in between the timeout period and at the end of the timeout period. If the refresh takes place outside the window, this is a sign that the program code execution is taking place faster than expected and hence points to something abnormal in program code execution. The following figure illustrates the concept of windowed refreshing.

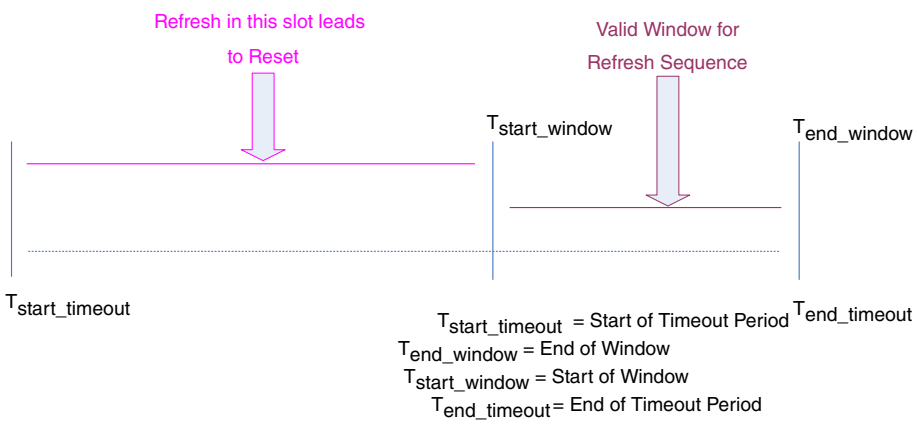


Figure 8-2. Windowed Refresh Mode

8.3.6 Fast Response to Code Runaway

As has been emphasized before, it is imperative that the response of the watchdog to a runaway code be as fast as possible. Code runaway is a state in which the system acts indeterministically and so it should be brought out of that state as fast as possible. The WDOG takes a proactive approach to this problem.

While the method of running a timer in the watchdog and interpreting its timeout as a sign of system failure (due to runaway code or system clock failure) is time tested; it does however have one shortcoming. If code runaway happens in the early stages of the watchdog timer period, it takes a lot of time before safety measures (like resetting the system) kick in, because the watchdog waits for its timer to timeout. In some applications, this delay in the watchdog reacting, might be as large as 1 second (the watchdog's timeout period). The WDOG seeks to do this by recognizing the signs of runaway code early on and resetting the system immediately, without waiting for a timeout of its internal timer. These signs are:

- Presence of a value, other than the two bonafide values of the refresh sequence or the register-unlock password, in the watchdog's refresh or unlock register - The user's software code would only contain instruction writing the said sets of two values to these registers. Thus, the presence of a third value indicates something abnormal happening in the code, probably due to a runaway.
- Failure to write to configuration registers within a small, fixed amount of time after unlocking them - Again, this indicates something abnormal as a normal user code would contain at least one watchdog configuration operation following the instructions which unlock the registers.
- Failure to write to at least one of the configuration registers within a small, fixed amount of time after system reset de-assertion - This might seem an overkill but by forcing the user to do so, it is ensured that the user doesn't forget to properly configure the watchdog and get it up and running, as per the system's needs, as soon as possible after reset, in the midst all the other boot up tasks that are required by the system.

When a timeout takes place, the logic generating a reset to the system is run off the fast system clock (in the range of tens to hundreds of MHz), rather than the watchdog's dedicated, slow clock (in the range of a few kHz to a few MHz). If the reset were to be generated off the slow clock, say 1 kHz (LPO clock), it could take the watchdog almost 1ms to reset the system, after timeout, leaving too much time for run away code to cause havoc. One risk in generating the reset off the system clock is that in the event this clock fails, the watchdog timer's timeout would go unacknowledged and wouldn't reset the system. To take care of such a situation, a backup circuit in the WDOG waits for second consecutive timeout of timer and passes it on as reset to the system, as shown in the following figure.

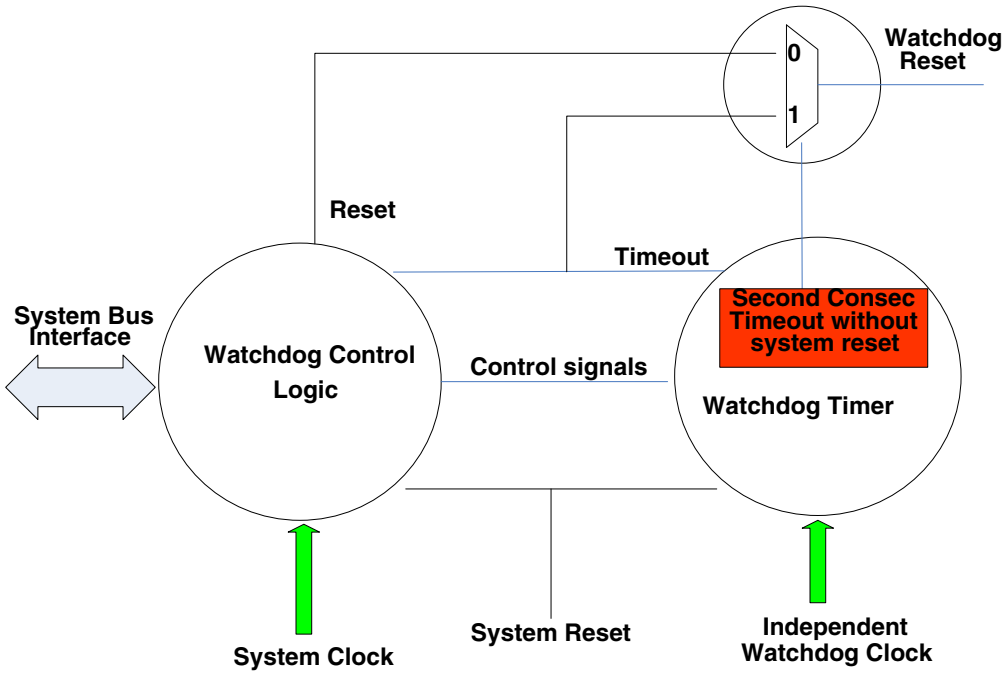


Figure 8-3. Reset Generation Logic

8.4 Watchdog configuration

Table 8-1. Watchdog configuration

Description	Value
Default Value for WDOG Timer High	0x004C
Default Value for WDOG Timer Low	0x4B40
Watchdog Configuration Time (in number of bus clocks)	256
Time allowed in between the two writes of the unlock and refresh sequences.	20

8.5 iRTC Write Protect State Machine

For additional system reliability, the iRTC includes a state machine that protects iRTC registers and standby RAM from any spurious updates that can happen due to run-away code.

After a power on reset, the write-protect mechanism is disabled, allowing the user code to calibrate the iRTC clock, set the time in the clock registers, and set the date in the calendar registers. Once calibration and time & date settings are done, the user code should enable write protection mode. If not, the registers are put into write protect mode 15 seconds after power on. In case the write protect mode is unlocked to update registers, then the write protect mode is automatically enabled 2 seconds after unlock, if not already done by CPU.

The protection mechanism works on the values written to write enable bits of the iRTC control register. By default unconditional write access is allowed to these bits only.

To enable write protection, write 0b10 to these bits. To disable write protection, write the sequence 00, 01, 11, 10 to these bits.

Any access made to the register space when write protection is enabled (i.e. registers in locked mode) will cause the transfer error signal to be asserted.

8.6 iRTC Tamper Detection Mechanism

iRTC supports various tamper detection mechanisms as described below. These tampers can be classified as internal and external tampers. External tampers are detected outside the MCU while Internal tampers are detected within the MCU. All tamper events are recorded with the time stamp to indicate the time and date of the tamper event.

8.6.1 Internal Tamper Condition 1: Battery removed when MCU is powered OFF

The detection of battery removal during system power off is done using a flop that is asserted on power-on reset. Since there is no difference between a proper shutdown and this tamper condition, this is considered as a tamper always. It is up to the firmware to differentiate between the tamper condition and normal power up. One way is to ignore this tamper interrupt when the SoC or application is in Service mode and simply reset the tamper interrupt. For other conditions it will be taken as a tamper.

8.6.2 Internal Tamper Condition 2: Battery removed when MCU is powered ON

The analog circuitry monitoring the battery voltage indicates when the battery voltage is removed. This signal is used by the tamper circuit to indicate a tamper provided MCU power in ON.

8.6.3 External Tamper Condition: Off Chip Tamper Indication

An external off chip tamper switch is also used to monitor tampering external to the SoC. For example, a signal can be used to indicate that the case housing the SoC/board is opened or not. Since these events are detected off chip, these tamper inputs are pre-condition in the SoC logic (i.e. conversion to digital or level shifting, etc) before being input to RTC.

External tamper switches are prone to noise and can cause false tamper indication if not filtered. Noise filtering is present in RTC for all external tamper inputs. The duration for which a tamper signal should be asserted to be indicated as tamper is programmable by user software in the register space. This duration is programmable from 64 μs to 125 ms to support a variety of tamper switches. The filtered signals are then used to generate the tamper status and interrupt signals. The polarity of the tamper inputs can be configured to be active high or active low.

8.6.4 Tamper Detection Flow

Following steps describe how tamper is logged in the registers and how should software acknowledge the tamper status indication:

- POR asserts tamper status bit 8 in register space (RTC_TTSR_SCR[11]) and tamper interrupt status bit (in RTC_ISR) to indicate tamper on power up of device.
- Tamper Interrupt Enable bit is also asserted on POR and an interrupt indication to CPU.
- When CPU acknowledges the tamper interrupt by writing 1’b1 to the tamper interrupt status bit (RTC_TTSR_SCR[11]), the tamper status bits are cleared.
- Internal tamper event (detected by SoC logic) are simply stored in their corresponding status bits and time stamp is captured.
- Any tamper signal asserted externally are filtered in the tamper block.
- Tamper detect signal is asserted when the filter counter matches the programmed filter duration.
- Based on the tamper detect signal the appropriate status bit is asserted in the tamper status register (RTC_TTSR_SCR[15:8]) irrespective the tamper control bit is enabled or not. Time stamp is also logged for the latest tamper event. The time stamp does not indicate which tamper occurred but the time when the latest tamper event took place.
- The tamper interrupt status bit is asserted for the tamper status bits that are enabled by asserting the corresponding tamper control bit in RTC_TTSR_SCR[7:0].
- If interrupt is enabled CPU is interrupted by assertion of the interrupt signal.

- Tamper interrupt status bit (in RTC_ISR) is cleared when all tamper status bits (in RTC_TTSR_SCR[15:8]) are cleared by writing 1 to them.
- Tamper interrupt can be armed or disabled by writing to the interrupt enable register bit (in RTC_IER[0]) or the individual tamper control bits (RTC_TTSR_SCR[7:0]) through the CPU register programming interface.

Chapter 9
Debug

9.1 Introduction

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

It provides register and memory accessibility from the external debugger interface, basic run/halt control plus 2 breakpoints and 2 watchpoints.

Only one debug interface is supported:

- Serial Wire Debug (SWD)

9.2 Debug port pin descriptions

The debug port pins default after POR to their SWD functionality.

Table 9-1. Serial wire debug pin description

Pin Name	Type	Description
SWD_CLK	Input	Serial Wire Clock. This pin is the clock for debug logic when in the Serial Wire Debug mode. This pin is pulled down internally.
SWD_IO	Input / Output	Serial wire debug data input/output. The SWD_IO pin is used by an external debug tool for communication and device control. This pin is pulled up internally.

9.3 SWD status and control registers

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

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

Table 9-2. MDM-AP Register Summary

Address	Register	Description
0x0100_0000	Status	See MDM-AP Status Register
0x0100_0004	Control	See MDM-AP Control Register
0x0100_00FC	IDR	Read-only identification register that always reads as 0x001C_0020

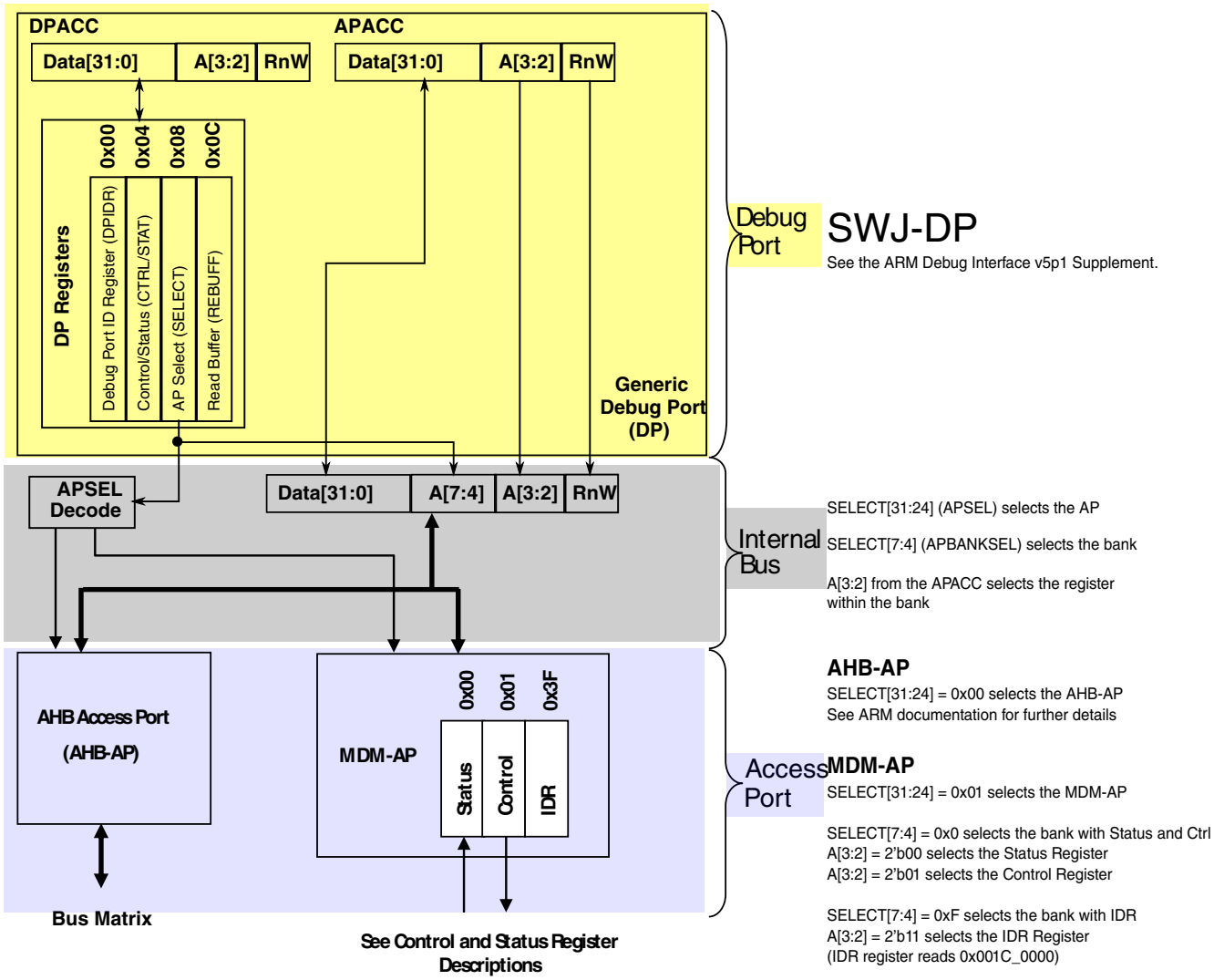


Figure 9-1. MDM AP Addressing

9.3.1 MDM-AP Control Register

Table 9-3. MDM-AP Control register assignments

Bit	Name	Secure[1]	Description
0	Flash Mass Erase in Progress	Y	Set to cause mass erase. Cleared by hardware after mass erase operation completes. When mass erase is disabled (via MEEN and SEC settings), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset.
1	Debug Disable	N	Set to disable debug. Clear to allow debug operation. When set it overrides the C_DEBUGEN bit within the DHCSR and force disables Debug logic.
2	Debug Request	N	Set to force the Core to halt.

Table continues on the next page...

Table 9-3. MDM-AP Control register assignments (continued)

Bit	Name	Secure[1]	Description
			If the Core is in a stop or wait mode, this bit can be used to wakeup the core and transition to a halted state.
3	System Reset Request	N	Set to force a system reset. The system remains held in reset until this bit is cleared.
4	Core Hold Reset	N	Configuration bit to control Core operation at the end of system reset sequencing. 0 Normal operation - release the Core from reset along with the rest of the system at the end of system reset sequencing. 1 Suspend operation - hold the Core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the Core from reset and CPU operation begins.
5	VLLSx Debug Request (VLLDBGREQ)	N	Set to configure the system to be held in reset after the next recovery from a VLLSx mode. This bit holds the in reset when VLLSx modes are exited to allow the debugger time to re-initialize debug IP before the debug session continues. The Mode Controller captures this bit logic on entry to VLLSx modes. Upon exit from VLLSx modes, the Mode Controller will hold the in reset until VLLDBGACK is asserted. The VLLDBGREQ bit clears automatically due to the POR reset generated as part of the VLLSx recovery.
6	VLLSx Debug Acknowledge (VLLDBGACK)	N	Set to release a being held in reset following a VLLSx recovery This bit is used by the debugger to release the system reset when it is being held on VLLSx mode exit. The debugger re-initializes all debug IP and then assert this control bit to allow the Mode Controller to release the from reset and allow CPU operation to begin. The VLLDBGACK bit is cleared by the debugger or can be left set because it clears automatically due to the POR reset generated as part of the next VLLSx recovery.
7	VLLSx Status Acknowledge	N	Set this bit to acknowledge the DAP VLLS Status bits have been read. This acknowledge automatically clears the status bits. This bit is used by the debugger to clear the sticky VLLSx mode entry status bits. This bit is asserted and cleared by the debugger.

1. Command available in secure mode

9.3.2 MDM-AP Status Register

Table 9-4. MDM-AP Status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge bit is cleared after any system reset. The bit is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress bit in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation. When mass erase is disabled (via MEEN and SEC settings), an erase request due to setting of Flash Mass Erase in Progress bit is not acknowledged.
1	Flash Ready	Indicate Flash has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger.
2	System Security	Indicates the security state. When secure, the debugger does not have access to the system bus or any memory mapped peripherals. This bit indicates when the part is locked and no system bus access is possible.
3	System Reset	Indicates the system reset state. 0 System is in reset 1 System is not in reset
4	Reserved	
5	Mass Erase Enable	Indicates if the MCU can be mass erased or not 0 Mass erase is disabled 1 Mass erase is enabled **NOTE:**
6	Backdoor Access Key Enable	Indicates if the MCU has the backdoor access key enabled. 0 Disabled 1 Enabled
7	LP Enabled	Decode of LPLLSM control bits to indicate that VLPS or VLLSx are the selected power mode the next time the ARM Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled 1 Low Power Stop Mode is enabled Usage intended for debug operation in which Run to VLPS is attempted. Per debug definition, the system actually enters the Stop state. A debugger should interpret deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjunction with this bit asserted as the debugger-VLPS status indication.
8	Very Low Power Mode	Indicates current power mode is VLPx. This bit is not ‘sticky’ and should always represent whether VLPx is enabled or not. This bit is used to throttle JTAG TCK frequency up/down.
9	Reserved	Always read 0.
10	VLLSx Modes Exit	This bit indicates an exit from VLLSx mode has occurred. The debugger will lose communication while the system is in VLLSx (including access to this register). Once communication is reestablished, this bit indicates that the system had been in VLLSx. Since the debug modules lose their state during VLLSx modes, they need to be reconfigured.

Table continues on the next page...

Table 9-4. MDM-AP Status register assignments (continued)

Bit	Name	Description
		This bit is set during the VLLSx recovery sequence. The VLLSx Mode Exit bit is held until the debugger has had a chance to recognize that a VLLS mode was exited and is cleared by a write of 1 to the VLLSx Status Acknowledge bit in MDM AP Control register.
11 – 15	Reserved for future use	Always read 0.
16	Core Halted	Indicates the Core has entered debug halt mode
17	Core SLEEPDEEP	Indicates the Core has entered a low power mode
18	Core SLEEPING	SLEEPING==1 and SLEEPDEEP==0 indicates wait or VLPW mode. SLEEPING==0 and SLEEPDEEP==1 indicates stop or VLPS mode.
19 – 31	Reserved for future use	Always read 0.

9.4 Debug resets

The debug system receives the following sources of reset:

- System POR reset

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

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

9.5 Micro Trace Buffer (MTB)

The Micro Trace Buffer (MTB) provides a simple execution trace capability for the Cortex®-M0+ processor. When enabled, the MTB records changes in program flow reported by the Cortex®-M0+ processor, via the execution trace interface, into a configurable region of the SRAM. Subsequently an off-chip debugger may extract the trace information, which would allow reconstruction of an instruction flow trace. The MTB does not include any form of load/store data trace capability or tracing of any other information.

In addition to providing the trace capability, the MTB also operates as a simple AHB-Lite SRAM controller. The system bus masters (including the processor) have read/write access to all of the SRAM via the AHB-Lite interface, allowing the memory to also be used to store program and data information. The MTB simultaneously stores the trace information into an attached SRAM and allows bus masters to access the memory. The MTB ensures that trace information write accesses to the SRAM take priority over accesses from the AHB-Lite interface.

The MTB includes trace control registers for configuring and triggering the MTB functions. The MTB also supports triggering via TSTART and TSTOP control functions in the MTB DWT module.

9.6 Debug in low-power modes

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

9.7 Debug and security

When flash security is enabled, the debug port capabilities are limited in order to prevent exploitation of secure data. In the secure state the debugger still has access to the status register and can determine the current security state of the device. In the case of a secure device, the debugger only has the capability of performing a mass erase operation.

Chapter 10
Pinouts and Packaging

10.1 Package Types

KM3x_256 family of devices shall support the following packages options:

- 144-pin LQFP (20 × 20 mm^2)
- 100-pin LQFP (14 × 14 mm^2)
- 64-pin LQFP (10 × 10 mm^2) [1]

NOTE

Pin muxing selection between TAMPER0 and WKUP is done using control bit in RTC_CTRL2 register.

NOTE

All pin muxing configurations reset to default value on any reset assertion (reset asserts on VLLSx mode exit).

When RESET pin is used as GPIO and pulled low; an internal reset (e.g. VLLSx mode exit or WDOG reset, etc) will make this pin function as RESET (default function) and since it is pulled low, it will appear as if pin reset is asserted and will cause full chip reset.

1. This package for the product is not yet available. However, it is included in Package Your Way program for Kinetis MCUs. Visit Freescale.com/KPYW for more details.

10.2 Port control and interrupt module features

Table 10-1. Ports summary

Feature	Port A	Port B	Port C	Port D	Port E	Port F	Port G	Port H	Port I	Port J	Port K	Port L	Port M
Pull select control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pull select at reset	PTA4 = pull up; others = pull down	Pull down	Pull down	Pull down	PTE6/ PTE1 = pull up; others = pull down	Pull down	Pull down	Pull down	Pull down	Pull down	Pull down	Pull down	Pull down
Pull enable control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pull enable at reset	PTA4 = enabled; others = disabled	Disabled	Disabled	Disabled	PTE7/ PTE6/ PTE1 = enabled; others = disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Slew rate enable control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Slew rate enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Passivie filter enable control	No	No	No	No	No	No	No	No	No	No	No	No	No
Passivie filter enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Open drain enable control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Open drain enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Drive strength enable control	No	No	No	No	No	No	No	No	No	No	No	No	No

Table continues on the next page...

Table 10-1. Ports summary (continued)

Feature	Port A	Port B	Port C	Port D	Port E	Port F	Port G	Port H	Port I	Port J	Port K	Port L	Port M
Drive strength enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Pin mux control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pin mux at reset	PTA4 = ALT7; others = ALT0	ALT0	ALT0	ALT0	PTE7/ PTE6/ PTE1 = ALT7; others = ALT0	ALT0	ALT0	ALT0	ALT0	ALT0	ALT0	ALT0	ALT0
Lock bit	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Interrupt and DMA request	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Digital glitch filter	No	No	No	No	Yes	No	No	No	No	No	No	No	No

10.3 KM3x_256 Signal multiplexing and pin assignments

144 QFP	100 QFP	Pin Name	DEFAULT	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
1	—	NC	NC								
2	—	NC	NC								
3	—	PTI5	Disabled	LCD_P45	PTI5						LCD_P45
4	1	PTA0/ LLWU_P16	Disabled	LCD_P23	PTA0/ LLWU_P16						LCD_P23
5	2	PTA1	Disabled	LCD_P24	PTA1						LCD_P24
6	—	PTI6	Disabled	LCD_P46	PTI6	UART2_RX					LCD_P46
7	—	PTI7	Disabled	LCD_P47	PTI7	UART2_TX					LCD_P47
8	3	PTA2	Disabled	LCD_P25	PTA2						LCD_P25
9	4	PTA3	Disabled	LCD_P26	PTA3						LCD_P26
10	5	PTA4/ LLWU_P15	NMI_b	LCD_P27	PTA4/ LLWU_P15					LCD_P27	NMI_b
11	6	PTA5	Disabled	LCD_P28	PTA5	CMP0_OUT					LCD_P28
12	7	PTA6/ LLWU_P14	Disabled	LCD_P29	PTA6/ LLWU_P14	XBAR0_IN0					LCD_P29
13	8	PTA7	Disabled	LCD_P30	PTA7	XBAR0_OUT0					LCD_P30
14	—	PTJ0	Disabled	LCD_P48	PTJ0	I2C1_SDA					LCD_P48
15	—	PTJ1	Disabled	LCD_P49	PTJ1	I2C1_SCL					LCD_P49
16	9	PTB0	Disabled	LCD_P31	PTB0						LCD_P31

144 QFP	100 QFP	Pin Name	DEFAULT	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
17	—	PTJ2	Disabled	LCD_P50	PTJ2						LCD_P50
18	10	VDD	VDD	VDD							
19	11	VSS	VSS	VSS							
20	12	PTB1/ LLWU_P17	Disabled	LCD_P32	PTB1/ LLWU_P17						LCD_P32
21	13	PTB2	Disabled	LCD_P33	PTB2						LCD_P33
22	14	PTB3	Disabled	LCD_P34	PTB3						LCD_P34
23	15	PTB4	Disabled	LCD_P35	PTB4						LCD_P35
24	16	PTB5	Disabled	LCD_P36	PTB5						LCD_P36
25	17	PTB6	Disabled	LCD_P37/ CMP1_IN0	PTB6						LCD_P37
26	18	PTB7	Disabled	LCD_P38	PTB7	AFE_CLK					LCD_P38
27	19	PTC0	Disabled	LCD_P39	PTC0	UART3_RTS_b	XBAR0_IN1	PDB0_EXTRG			LCD_P39
28	20	PTC1	Disabled	LCD_P40/ CMP1_IN1	PTC1	UART3_CTS_b					LCD_P40
29	21	PTC2	Disabled	LCD_P41	PTC2	UART3_TX	XBAR0_OUT1				LCD_P41
30	22	PTC3/ LLWU_P13	Disabled	LCD_P42/ CMP0_IN3	PTC3/ LLWU_P13	UART3_RX					LCD_P42
31	23	PTC4	Disabled	LCD_P43	PTC4						LCD_P43
32	24	VBAT	VBAT	VBAT							
33	25	XTAL32	XTAL32	XTAL32							
34	26	EXTAL32	EXTAL32	EXTAL32							
35	—	NC	NC								
36	—	NC	NC								
37	—	NC	NC								
38	—	NC	NC								
39	27	VSS	VSS	VSS							
40	28	TAMPER2	TAMPER2	TAMPER2							
41	29	TAMPER1	TAMPER1	TAMPER1							
42	30	TAMPER0	TAMPER0	TAMPER0							
43	31	AFE_VDDA	AFE_VDDA	AFE_VDDA							
44	32	AFE_VSSA	AFE_VSSA	AFE_VSSA							
45	33	AFE_SDADP0	AFE_SDADP0	AFE_SDADP0							
46	34	AFE_SDADM0	AFE_SDADM0	AFE_SDADM0							
47	35	AFE_SDADP1	AFE_SDADP1	AFE_SDADP1							
48	36	AFE_SDADM1	AFE_SDADM1	AFE_SDADM1							
49	37	VREFH	VREFH	VREFH							
50	38	VREFL	VREFL	VREFL							
51	39	AFE_SDADP2/ CMP1_IN2	AFE_SDADP2/ CMP1_IN2	AFE_SDADP2/ CMP1_IN2							
52	40	AFE_SDADM2/ CMP1_IN3	AFE_SDADM2/ CMP1_IN3	AFE_SDADM2/ CMP1_IN3							
53	41	VREF	VREF	VREF							

144 QFP	100 QFP	Pin Name	DEFAULT	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
54	42	AFE_SDADP3/ CMP1_IN4	AFE_SDADP3/ CMP1_IN4	AFE_SDADP3/ CMP1_IN4							
55	43	AFE_SDADM3/ CMP1_IN5	AFE_SDADM3/ CMP1_IN5	AFE_SDADM3/ CMP1_IN5							
56	—	NC	NC								
57	—	NC	NC								
58	44	PTC5/ LLWU_P12	Disabled	ADC0_SE0/ CMP2_IN0	PTC5/ LLWU_P12	UART0_RTS_b					
59	45	PTC6	Disabled	ADC0_SE1/ CMP2_IN1	PTC6	UART0_CTS_b	QTMR0_TMR1	PDB0_EXTRG			
60	46	PTC7	Disabled	ADC0_SE2/ CMP2_IN2	PTC7	UART0_TX	XBAR0_OUT2				
61	47	PTD0/ LLWU_P11	Disabled	CMP0_IN0	PTD0/ LLWU_P11	UART0_RX	XBAR0_IN2				
62	—	PTJ3	Disabled		PTJ3	LPUART0_ RTS_b	CMP2_OUT				
63	—	PTJ4	Disabled		PTJ4	LPUART0_ CTS_b					
64	48	PTD1	Disabled		PTD1	UART1_TX	SPI0_PCS0	XBAR0_OUT3	QTMR0_TMR3		
65	49	PTD2/ LLWU_P10	Disabled	CMP0_IN1	PTD2/ LLWU_P10	UART1_RX	SPI0_SCK	XBAR0_IN3			
66	—	PTJ5	Disabled		PTJ5	LPUART0_TX					
67	—	PTJ6/ LLWU_P18	Disabled		PTJ6/ LLWU_P18	LPUART0_RX					
68	—	PTJ7	Disabled		PTJ7						
69	50	PTD3	Disabled		PTD3	UART1_CTS_b	SPI0_MOSI				
70	—	PTK0	Disabled	ADC0_SE12	PTK0						
71	—	NC	NC								
72	—	NC	NC								
73	—	NC	NC								
74	—	NC	NC								
75	—	PTK1	Disabled	ADC0_SE13	PTK1						
76	51	PTD4/ LLWU_P9	Disabled	ADC0_SE3	PTD4/ LLWU_P9	UART1_RTS_b	SPI0_MISO				
77	52	PTD5	Disabled	ADC0_SE4a	PTD5	LPTMR0_ALT3	QTMR0_TMR0	UART3_CTS_b			
78	53	PTD6/ LLWU_P8	Disabled	ADC0_SE5a	PTD6/ LLWU_P8	LPTMR0_ALT2	CMP1_OUT	UART3_RTS_b			
79	54	PTD7/ LLWU_P7	Disabled	CMP0_IN4	PTD7/ LLWU_P7	I2C0_SCL	XBAR0_IN4	UART3_RX			
80	55	PTE0	Disabled		PTE0	I2C0_SDA	XBAR0_OUT4	UART3_TX	CLKOUT		
81	—	PTK2	Disabled	ADC0_SE14	PTK2	UART0_TX					
82	—	PTK3/ LLWU_P19	Disabled	ADC0_SE15	PTK3/ LLWU_P19	UART0_RX					
83	56	PTE1	RESET_B		PTE1						RESET_b
84	57	PTE2	EXTAL0	EXTAL0	PTE2	EWM_IN	XBAR0_IN6	I2C1_SDA			

144 QFP	100 QFP	Pin Name	DEFAULT	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
85	58	PTE3	XTAL0	XTAL0	PTE3	EWM_OUT_b	AFE_CLK	I2C1_SCL			
86	59	VSS	VSS	VSS							
87	60	VSSA	VSSA	VSSA							
88	61	VDDA	VDDA	VDDA							
89	62	VDD	VDD	VDD							
90	63	PTE4	Disabled		PTE4	LPTMR0_ALT1	UART2_CTS_b	EWM_IN			
91	64	PTE5/ LLWU_P6	Disabled		PTE5/ LLWU_P6	QTMR0_TMR3	UART2_RTS_b	EWM_OUT_b			
92	65	PTE6/ LLWU_P5	SWD_DIO	CMP0_IN2	PTE6/ LLWU_P5	XBAR0_IN5	UART2_RX		I2C0_SCL		SWD_DIO
93	66	PTE7	SWD_CLK	ADC0_SE6a	PTE7	XBAR0_OUT5	UART2_TX		I2C0_SDA		SWD_CLK
94	67	PTF0/ LLWU_P4	Disabled	ADC0_SE7a/ CMP2_IN3	PTF0/ LLWU_P4	RTC_CLKOUT	QTMR0_TMR2	CMP0_OUT			
95	68	PTF1	Disabled	LCD_P0/ ADC0_SE8/ CMP2_IN4	PTF1	QTMR0_TMR0	XBAR0_OUT6				LCD_P0
96	69	PTF2	Disabled	LCD_P1/ ADC0_SE9/ CMP2_IN5	PTF2	CMP1_OUT	RTC_CLKOUT				LCD_P1
97	—	PTK4	Disabled	LCD_P51	PTK4	XBAR0_IN9	AFE_CLK				LCD_P51
98	—	PTK5	Disabled		PTK5	UART1_RX					
99	—	PTK6	Disabled		PTK6	UART1_TX					
100	70	PTF3/ LLWU_P20	Disabled	LCD_P2	PTF3/ LLWU_P20	SPI1_PCS0	LPTMR0_ALT2	UART0_RX			LCD_P2
101	71	PTF4	Disabled	LCD_P3	PTF4	SPI1_SCK	LPTMR0_ALT1	UART0_TX			LCD_P3
102	72	PTF5	Disabled	LCD_P4	PTF5	SPI1_MISO	I2C1_SCL				LCD_P4
103	73	PTF6/ LLWU_P3	Disabled	LCD_P5	PTF6/ LLWU_P3	SPI1_MOSI	I2C1_SDA				LCD_P5
104	74	PTF7	Disabled	LCD_P6	PTF7	QTMR0_TMR2	CLKOUT	CMP2_OUT			LCD_P6
105	—	PTK7	Disabled	LCD_P52	PTK7	I2C0_SCL	XBAR0_OUT9				LCD_P52
106	—	PTL0	Disabled	LCD_P53	PTL0	I2C0_SDA					LCD_P53
107	—	NC	NC								
108	—	NC	NC								
109	—	NC	NC								
110	75	PTG0	Disabled	LCD_P7	PTG0	QTMR0_TMR1	LPTMR0_ALT3				LCD_P7
111	76	PTG1/ LLWU_P2	Disabled	LCD_P8/ ADC0_SE10	PTG1/ LLWU_P2		LPTMR0_ALT1				LCD_P8
112	77	PTG2/ LLWU_P1	Disabled	LCD_P9/ ADC0_SE11	PTG2/ LLWU_P1	SPI0_PCS0					LCD_P9
113	78	PTG3	Disabled	LCD_P10	PTG3	SPI0_SCK	I2C0_SCL				LCD_P10
114	79	PTG4	Disabled	LCD_P11	PTG4	SPI0_MOSI	I2C0_SDA				LCD_P11
115	80	PTG5	Disabled	LCD_P12	PTG5	SPI0_MISO	LPTMR0_ALT2				LCD_P12
116	81	PTG6/ LLWU_P0	Disabled	LCD_P13	PTG6/ LLWU_P0		LPTMR0_ALT3				LCD_P13

144 QFP	100 QFP	Pin Name	DEFAULT	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
117	82	PTG7	Disabled	LCD_P14	PTG7						LCD_P14
118	83	PTH0	Disabled	LCD_P15	PTH0	LPUART0_ CTS_b					LCD_P15
119	84	PTH1	Disabled	LCD_P16	PTH1	LPUART0_ RTS_b					LCD_P16
120	85	PTH2	Disabled	LCD_P17	PTH2	LPUART0_RX					LCD_P17
121	86	PTH3	Disabled	LCD_P18	PTH3	LPUART0_TX					LCD_P18
122	87	PTH4	Disabled	LCD_P19	PTH4						LCD_P19
123	88	PTH5	Disabled	LCD_P20	PTH5						LCD_P20
124	89	PTH6	Disabled		PTH6	UART1_CTS_b	SPI1_PCS0	XBAR0_IN7			
125	90	PTH7	Disabled		PTH7	UART1_RTS_b	SPI1_SCK	XBAR0_OUT7			
126	91	PTI0/ LLWU_P21	Disabled	CMP0_IN5	PTI0/ LLWU_P21	UART1_RX	XBAR0_IN8	SPI1_MISO	SPI1_MOSI		
127	92	PTI1 (This pin is true open drain pad. External pull-up resistor should be added.)	Disabled		PTI1	UART1_TX	XBAR0_OUT8	SPI1_MOSI	SPI1_MISO		
128	—	PTL1	Disabled	LCD_P54	PTL1	XBAR0_IN10					LCD_P54
129	—	PTL2	Disabled	LCD_P55	PTL2	XBAR0_ OUT10					LCD_P55
130	93	PTI2/ LLWU_P22	Disabled	LCD_P21	PTI2/ LLWU_P22	LPUART0_RX					LCD_P21
131	94	PTI3	Disabled	LCD_P22	PTI3	LPUART0_TX	CMP2_OUT				LCD_P22
132	95	VSS	VSS	VSS							
133	—	VDD	VDD	VDD							
134	96	VLL3	VLL3	VLL3							
135	97	VLL2	VLL2	VLL2/ LCD_P60	PTM0						LCD_P60
136	98	VLL1	VLL1	VLL1/ LCD_P61	PTM1						LCD_P61
137	99	VCAP2	VCAP2	VCAP2/ LCD_P62	PTM2						LCD_P62
138	100	VCAP1	VCAP1	VCAP1/ LCD_P63	PTM3						LCD_P63
139	—	PTL3	Disabled	LCD_P56	PTL3	EWM_IN					LCD_P56
140	—	PTL4	Disabled	LCD_P57	PTL4	EWM_OUT_b					LCD_P57
141	—	PTL5/ LLWU_P23	Disabled	LCD_P58	PTL5/ LLWU_P23						LCD_P58
142	—	PTL6	Disabled	LCD_P59	PTL6						LCD_P59
143	—	PTI4	Disabled	LCD_P44	PTI4						LCD_P44
144	—	NC	NC								

10.4 KM3x_256 Family Pinouts

10.4.1 100-pin LQFP

The following figure represents the KM3x_256 100 LQFP pinouts:

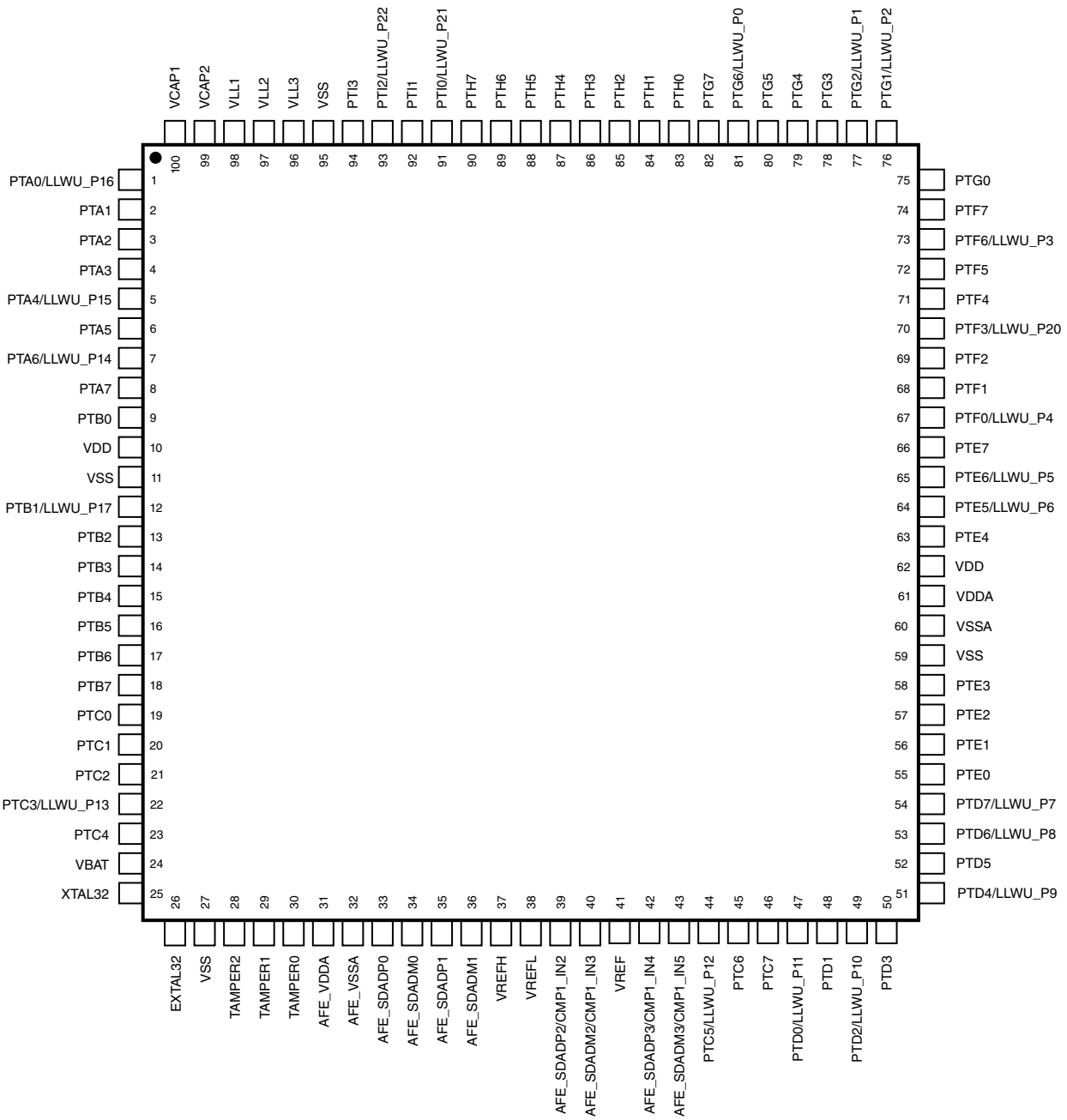


Figure 10-1. 100-pin LQFP Pinout Diagram

10.4.2 144-pin LQFP

The following figure represents the KM3x_256 144 LQFP pinouts:

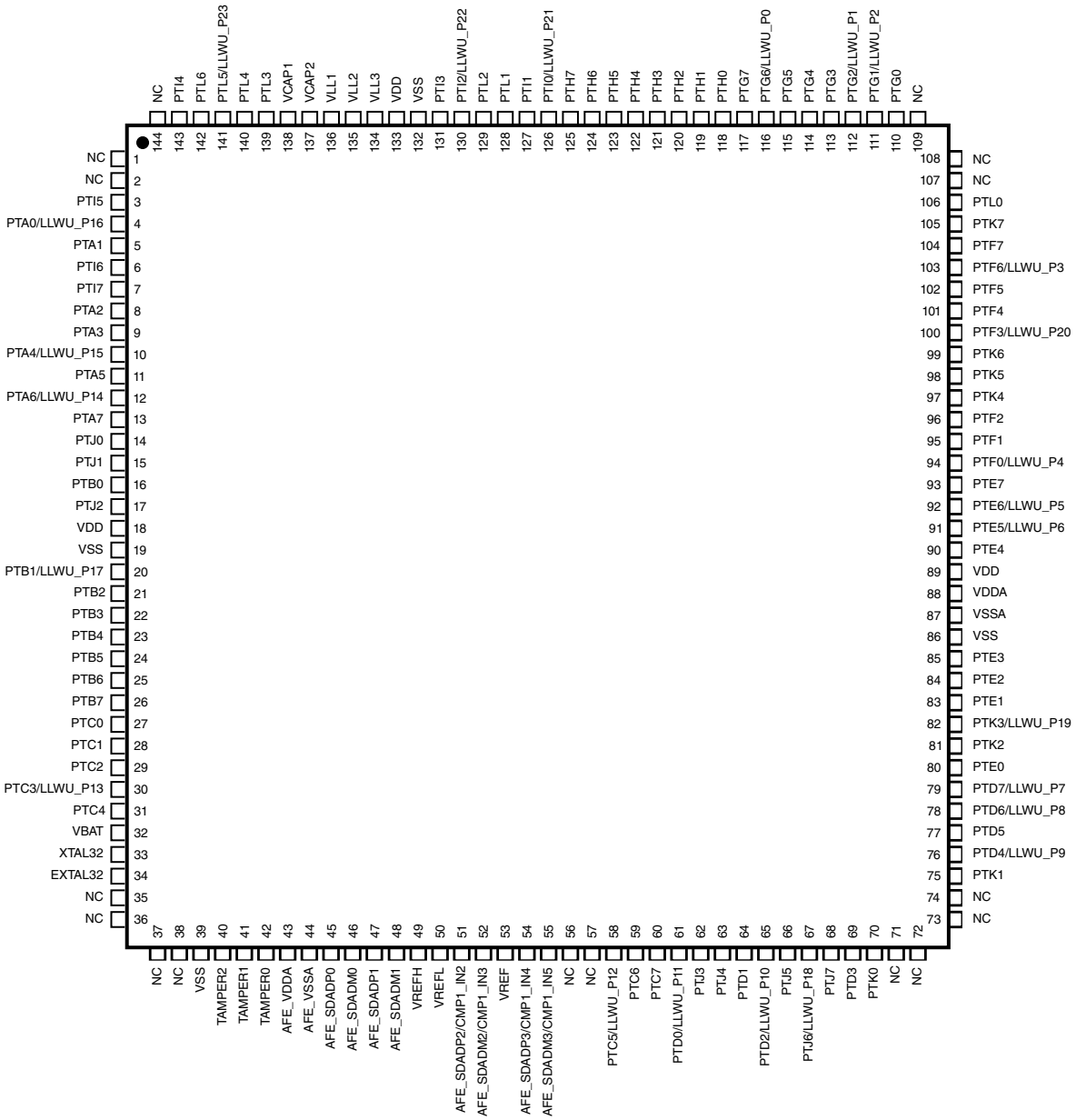


Figure 10-2. 144-pin LQFP Pinout Diagram

10.5 Module Signal Description Tables

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

10.5.1 Core Modules

Table 10-2. SWD Signal Descriptions

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

10.5.2 System Modules

Table 10-3. System Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{NMI}}$	—	Non-maskable interrupt NOTE: Driving the NMI signal low forces a non-maskable interrupt, if the NMI function is selected on the corresponding pin.	I
$\overline{\text{RESET}}$	—	Reset bidirectional signal	I/O
VDD	—	MCU power	I
VSS	—	MCU ground	I
VBAT	—	Battary power	I

Table 10-4. LLWU Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LLWU_Pn	LLWU_Pn	Wakeup inputs (n = 0, 1, …, 23)	I

Table 10-5. EWM Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EWM_IN	EWM_IN	EWM input for safety status of external safety circuits. The polarity of EWM_in is programmable using the EWM_CTRL[ASSIN] bit. The default polarity is active-low.	I
$\overline{\text{EWM\_OUT}}$	$\overline{\text{EWM\_OUT}}$	EWM reset out signal	O

Table 10-6. XBAR Signal Descriptions

Chip signal name	Module signal name	Description	I/O
XBAR_IN*n*	XBAR_IN*n*	Xbar input, n = 0, 1, … , 10	I
XBAR_OUT*n*	XBAR_OUT*n*	Xbar ouput, n = 0, 1, … , 11	O

10.5.3 Clock Modules

Table 10-7. OSC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EXTAL1	EXTAL	External clock/Oscillator input	I
XTAL1	XTAL	Oscillator output	O

Table 10-8. RTC Oscillator (OSC32K) Signal Descriptions

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

10.5.4 Analog

Table 10-9. ADC0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
ADC0_SE*n*	AD*n*	Single-Ended Analog Channel Inputs	I
VREFH	V_{REFSH}	Voltage Reference Select High	I
VREFL	V_{REFSL}	Voltage Reference Select Low	I
VDDA	V_{DDA}	Analog Power Supply	I
VSSA	V_{SSA}	Analog Ground	I

Table 10-10. CMP*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CMP*n*_IN[5:0]	IN[5:0]	Analog voltage inputs	I
CMP*n*_OUT	CMPO	Comparator output	O

Table 10-11. VREF Signal Descriptions

Chip signal name	Module signal name	Description	I/O
VREF	VREF_OUT	Internally-generated Voltage Reference output	O

Table 10-12. AFE Signal Descriptions

Chip signal name	Module signal name	Description	I/O
AFE_CLK	AFE_CLK	AFE external clock	I/O
AFE_SDADM[3:0]	AFE_SDADM[3:0]	Channel 3-0 -ve input	I
AFE_SDADP[3:0]	AFE_SDADP[3:0]	Channel 3-0 +ve input	I
AFE_VDDA	AFE_VDDA	Analog Power supply	I
AFE_VSSA	AFE_VSSA	Analog Ground	I

10.5.5 Timer Modules

Table 10-13. LPTMR Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPTMR_ALT[3:1]	LPTMR_ALT*n*	Pulse Counter Input pin	I

Table 10-14. RTC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
RTC_CLKOUT	RTC_CLKOUT	1 Hz square-wave output or OSCERCLK	O
TAMPER*n*	TAMPER*n*	Tamper detect, *n* = 0, 1, 2	I/O

Table 10-15. QTMR Signal Descriptions

Chip signal name	Module signal name	Description	I/O
QTMR0_TMR*n*	QTMR_CH*n*	QTMR channel *n* input/output, *n* = 0, 1, 2, 3	I/O

Table 10-16. PDB Signal Descriptions

Chip signal name	Module signal name	Description	I/O
PDB0_EXTRG	EXTRG	External Trigger Input Source If the PDB is enabled and external trigger input source is selected, a positive edge on the EXTRG signal resets and starts the counter.	I

10.5.6 Communication Interfaces

Table 10-17. PDB Signal Descriptions

Chip signal name	Module signal name	Description	I/O
SPI*n*_MISO	MISO	Master Data In, Slave Data Out	I/O
SPI*n*_MOSI	MOSI	Master Data Out, Slave Data In	I/O
SPI*n*_SCK	SPSCK	SPI Serial Clock	I/O
SPI*n*_PCS0	$\overline{SS}$	Slave Select	I/O

Table 10-18. I2C*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
I2C*n*_SCL	SCL	Bidirectional serial clock line of the I2C system.	I/O
I2C*n*_SDA	SDA	Bidirectional serial data line of the I2C system.	I/O

Table 10-19. LPUART0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPUART0_TX	TxD	Transmit data	O
LPUART0_RX	RxD	Receive data	I
$\overline{\text{LPUART0\_CTS}}$	$\overline{CTS}$	Clear to send	I
$\overline{\text{LPUART0\_RTS}}$	$\overline{RTS}$	Request to send	O

Table 10-20. UART*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
UART*n*_TX	TxD	Transmit data	O
UART*n*_RX	RxD	Receive data	I
$\overline{\text{UARTn\_CTS}}$	$\overline{CTS}$	Clear to send	I
$\overline{\text{UARTn\_RTS}}$	$\overline{RTS}$	Request to send	O

10.5.7 Human-Machine Interfaces (HMI)

Table 10-21. GPIO Signal Descriptions

Chip signal name	Module signal name	Description	I/O
PTA[7:0]	PORTA7–PORTA0	General-purpose input/output	I/O
PTB[7:0]	PORTB7–PORTB0	General-purpose input/output	I/O
PTC[7:0]	PORTC7–PORTC0	General-purpose input/output	I/O
PTD[7:0]	PORTD7–PORTD0	General-purpose input/output	I/O
PTE[7:0]	PORTE7–PORTE0	General-purpose input/output	I/O
PTF[7:0]	PORTF7–PORTF0	General-purpose input/output	I/O
PTG[7:0]	PORTG7–PORTG0	General-purpose input/output	I/O
PTH[7:0]	PORTH7–PORTH0	General-purpose input/output	I/O
PTI[7:0]	PORTI7–PORTI0	General-purpose input/output	I/O
PTJ[7:0]	PORTJ7–PORTJ0	General-purpose input/output	I/O
PTK[7:0]	PORTK7–PORTK0	General-purpose input/output	I/O
PTL[6:0]	PORTL6–PORTL0	General-purpose input/output	I/O
PTM[3:0]	PORTM3–PORTM0	General-purpose input/output	I/O

Table 10-22. LCD Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LCD_P[63:0]	LCD_P[63:0] . 64 LCD front plane/back plane	Configurable front plane/back plane driver that connects directly to the display. LCD_P[63:0] can operate as GPIO pins	O
VLL1, VLL2, VLL3	VLL1, VLL2, VLL3. LCD bias voltages	LCD bias voltages (requires external capacitors when charge pump is used).	I/O
Vcap1, Vcap2	Vcap1, Vcap2. LCD charge pump capacitance.	Charge pump capacitor pins.	O

Chapter 11
Port Control and Interrupts (PORT)

11.1 Introduction

11.2 Overview

The Port Control and Interrupt (PORT) module provides support for port control, digital filtering, and external interrupt functions.

Most functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin muxing state.

There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

11.2.1 Features

The PORT module has the following features:

- Pin interrupt on selected pins
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wake-up in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
- Digital input filter
  - Digital input filter for each pin, usable by any digital peripheral muxed onto the pin
  - Individual enable or bypass control field per pin

    - Selectable clock source for digital input filter with a five bit resolution on filter size
    - Functional in all digital pin multiplexing modes
- Port control
    - Individual pull control fields with pullup, pulldown, and pull-disable enable support on selected pins
    - Individual drive strength field supporting high and low drive strength on selected pins
    - Individual slew rate field supporting fast and slow slew rates on selected pins
    - Individual input passive filter field supporting enable and disable of the individual input passive filter on selected pins
    - Individual open drain field supporting enable and disable of the individual open drain output on selected pins
    - Individual mux control field supporting analog or pin disabled, GPIO, and up to six chip-specific digital functions
    - Pad configuration fields are functional in all digital pin muxing modes.

11.2.2 Modes of operation

11.2.2.1 Run mode

In Run mode, the PORT operates normally.

11.2.2.2 Wait mode

In Wait mode, PORT continues to operate normally and may be configured to exit the Low-Power mode if an enabled interrupt is detected. DMA requests are still generated during the Wait mode, but do not cause an exit from the Low-Power mode.

11.2.2.3 Stop mode

In Stop mode, the PORT can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

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

11.2.2.4 Debug mode

In Debug mode, PORT operates normally.

11.3 External signal description

The table found here describes the PORT external signal.

Table 11-1. Signal properties

Name	Function	I/O	Reset	Pull
PORTx[31:0]	External interrupt	I/O	0	-

NOTE

Not all pins within each port are implemented on each device.

11.4 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 11-2. PORT interface—detailed signal description

Signal	I/O	Description	
PORTx[31:0]	I/O	External interrupt.	
		State meaning	Asserted—pin is logic 1. Negated—pin is logic 0.
		Timing	Assertion—may occur at any time and can assert asynchronously to the system clock. Negation—may occur at any time and can assert asynchronously to the system clock.

11.5 Memory map and register definition

Any read or write access to the PORT memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states.

NOTE

Digital filter capability is only on PORTE. Ignore DFER, DFCR and DFWR registers on ports A, B, C, D, F, G, H, and I.

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_6000	Pin Control Register n (PORTA_PCR0)	32	R/W	See section	11.5.1/154
4004_6004	Pin Control Register n (PORTA_PCR1)	32	R/W	See section	11.5.1/154
4004_6008	Pin Control Register n (PORTA_PCR2)	32	R/W	See section	11.5.1/154
4004_600C	Pin Control Register n (PORTA_PCR3)	32	R/W	See section	11.5.1/154
4004_6010	Pin Control Register n (PORTA_PCR4)	32	R/W	See section	11.5.1/154
4004_6014	Pin Control Register n (PORTA_PCR5)	32	R/W	See section	11.5.1/154
4004_6018	Pin Control Register n (PORTA_PCR6)	32	R/W	See section	11.5.1/154
4004_601C	Pin Control Register n (PORTA_PCR7)	32	R/W	See section	11.5.1/154
4004_6080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_6084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_60A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_60C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_60C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_60C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_7000	Pin Control Register n (PORTB_PCR0)	32	R/W	See section	11.5.1/154
4004_7004	Pin Control Register n (PORTB_PCR1)	32	R/W	See section	11.5.1/154
4004_7008	Pin Control Register n (PORTB_PCR2)	32	R/W	See section	11.5.1/154
4004_700C	Pin Control Register n (PORTB_PCR3)	32	R/W	See section	11.5.1/154
4004_7010	Pin Control Register n (PORTB_PCR4)	32	R/W	See section	11.5.1/154
4004_7014	Pin Control Register n (PORTB_PCR5)	32	R/W	See section	11.5.1/154
4004_7018	Pin Control Register n (PORTB_PCR6)	32	R/W	See section	11.5.1/154
4004_701C	Pin Control Register n (PORTB_PCR7)	32	R/W	See section	11.5.1/154
4004_7080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_7084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_70A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_70C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_70C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_70C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_8000	Pin Control Register n (PORTC_PCR0)	32	R/W	See section	11.5.1/154
4004_8004	Pin Control Register n (PORTC_PCR1)	32	R/W	See section	11.5.1/154
4004_8008	Pin Control Register n (PORTC_PCR2)	32	R/W	See section	11.5.1/154
4004_800C	Pin Control Register n (PORTC_PCR3)	32	R/W	See section	11.5.1/154

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_8010	Pin Control Register n (PORTC_PCR4)	32	R/W	See section	11.5.1/154
4004_8014	Pin Control Register n (PORTC_PCR5)	32	R/W	See section	11.5.1/154
4004_8018	Pin Control Register n (PORTC_PCR6)	32	R/W	See section	11.5.1/154
4004_801C	Pin Control Register n (PORTC_PCR7)	32	R/W	See section	11.5.1/154
4004_8080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_8084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_80A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_80C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_80C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_80C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_9000	Pin Control Register n (PORTD_PCR0)	32	R/W	See section	11.5.1/154
4004_9004	Pin Control Register n (PORTD_PCR1)	32	R/W	See section	11.5.1/154
4004_9008	Pin Control Register n (PORTD_PCR2)	32	R/W	See section	11.5.1/154
4004_900C	Pin Control Register n (PORTD_PCR3)	32	R/W	See section	11.5.1/154
4004_9010	Pin Control Register n (PORTD_PCR4)	32	R/W	See section	11.5.1/154
4004_9014	Pin Control Register n (PORTD_PCR5)	32	R/W	See section	11.5.1/154
4004_9018	Pin Control Register n (PORTD_PCR6)	32	R/W	See section	11.5.1/154
4004_901C	Pin Control Register n (PORTD_PCR7)	32	R/W	See section	11.5.1/154
4004_9080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_9084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_90A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_90C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_90C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_90C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_A000	Pin Control Register n (PORTE_PCR0)	32	R/W	See section	11.5.1/154
4004_A004	Pin Control Register n (PORTE_PCR1)	32	R/W	See section	11.5.1/154
4004_A008	Pin Control Register n (PORTE_PCR2)	32	R/W	See section	11.5.1/154
4004_A00C	Pin Control Register n (PORTE_PCR3)	32	R/W	See section	11.5.1/154
4004_A010	Pin Control Register n (PORTE_PCR4)	32	R/W	See section	11.5.1/154
4004_A014	Pin Control Register n (PORTE_PCR5)	32	R/W	See section	11.5.1/154
4004_A018	Pin Control Register n (PORTE_PCR6)	32	R/W	See section	11.5.1/154
4004_A01C	Pin Control Register n (PORTE_PCR7)	32	R/W	See section	11.5.1/154

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_A084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_A0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_A0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_A0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_A0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_B000	Pin Control Register n (PORTF_PCR0)	32	R/W	See section	11.5.1/154
4004_B004	Pin Control Register n (PORTF_PCR1)	32	R/W	See section	11.5.1/154
4004_B008	Pin Control Register n (PORTF_PCR2)	32	R/W	See section	11.5.1/154
4004_B00C	Pin Control Register n (PORTF_PCR3)	32	R/W	See section	11.5.1/154
4004_B010	Pin Control Register n (PORTF_PCR4)	32	R/W	See section	11.5.1/154
4004_B014	Pin Control Register n (PORTF_PCR5)	32	R/W	See section	11.5.1/154
4004_B018	Pin Control Register n (PORTF_PCR6)	32	R/W	See section	11.5.1/154
4004_B01C	Pin Control Register n (PORTF_PCR7)	32	R/W	See section	11.5.1/154
4004_B080	Global Pin Control Low Register (PORTF_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_B084	Global Pin Control High Register (PORTF_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_B0A0	Interrupt Status Flag Register (PORTF_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_B0C0	Digital Filter Enable Register (PORTF_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_B0C4	Digital Filter Clock Register (PORTF_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_B0C8	Digital Filter Width Register (PORTF_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_C000	Pin Control Register n (PORTG_PCR0)	32	R/W	See section	11.5.1/154
4004_C004	Pin Control Register n (PORTG_PCR1)	32	R/W	See section	11.5.1/154
4004_C008	Pin Control Register n (PORTG_PCR2)	32	R/W	See section	11.5.1/154
4004_C00C	Pin Control Register n (PORTG_PCR3)	32	R/W	See section	11.5.1/154
4004_C010	Pin Control Register n (PORTG_PCR4)	32	R/W	See section	11.5.1/154
4004_C014	Pin Control Register n (PORTG_PCR5)	32	R/W	See section	11.5.1/154
4004_C018	Pin Control Register n (PORTG_PCR6)	32	R/W	See section	11.5.1/154
4004_C01C	Pin Control Register n (PORTG_PCR7)	32	R/W	See section	11.5.1/154
4004_C080	Global Pin Control Low Register (PORTG_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C084	Global Pin Control High Register (PORTG_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_C0A0	Interrupt Status Flag Register (PORTG_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_C0C0	Digital Filter Enable Register (PORTG_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_C0C4	Digital Filter Clock Register (PORTG_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_C0C8	Digital Filter Width Register (PORTG_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_D000	Pin Control Register n (PORTH_PCR0)	32	R/W	See section	11.5.1/154
4004_D004	Pin Control Register n (PORTH_PCR1)	32	R/W	See section	11.5.1/154
4004_D008	Pin Control Register n (PORTH_PCR2)	32	R/W	See section	11.5.1/154
4004_D00C	Pin Control Register n (PORTH_PCR3)	32	R/W	See section	11.5.1/154
4004_D010	Pin Control Register n (PORTH_PCR4)	32	R/W	See section	11.5.1/154
4004_D014	Pin Control Register n (PORTH_PCR5)	32	R/W	See section	11.5.1/154
4004_D018	Pin Control Register n (PORTH_PCR6)	32	R/W	See section	11.5.1/154
4004_D01C	Pin Control Register n (PORTH_PCR7)	32	R/W	See section	11.5.1/154
4004_D080	Global Pin Control Low Register (PORTH_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_D084	Global Pin Control High Register (PORTH_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_D0A0	Interrupt Status Flag Register (PORTH_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_D0C0	Digital Filter Enable Register (PORTH_DFER)	32	R/W	0000_0000h	11.5.5/158
4004_D0C4	Digital Filter Clock Register (PORTH_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_D0C8	Digital Filter Width Register (PORTH_DFWR)	32	R/W	0000_0000h	11.5.7/159
4004_E000	Pin Control Register n (PORTI_PCR0)	32	R/W	See section	11.5.1/154
4004_E004	Pin Control Register n (PORTI_PCR1)	32	R/W	See section	11.5.1/154
4004_E008	Pin Control Register n (PORTI_PCR2)	32	R/W	See section	11.5.1/154
4004_E00C	Pin Control Register n (PORTI_PCR3)	32	R/W	See section	11.5.1/154
4004_E010	Pin Control Register n (PORTI_PCR4)	32	R/W	See section	11.5.1/154
4004_E014	Pin Control Register n (PORTI_PCR5)	32	R/W	See section	11.5.1/154
4004_E018	Pin Control Register n (PORTI_PCR6)	32	R/W	See section	11.5.1/154
4004_E01C	Pin Control Register n (PORTI_PCR7)	32	R/W	See section	11.5.1/154
4004_E080	Global Pin Control Low Register (PORTI_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4004_E084	Global Pin Control High Register (PORTI_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4004_E0A0	Interrupt Status Flag Register (PORTI_ISFR)	32	w1c	0000_0000h	11.5.4/157
4004_E0C0	Digital Filter Enable Register (PORTI_DFER)	32	R/W	0000_0000h	11.5.5/158

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_E0C4	Digital Filter Clock Register (PORTI_DFCR)	32	R/W	0000_0000h	11.5.6/159
4004_E0C8	Digital Filter Width Register (PORTI_DFWR)	32	R/W	0000_0000h	11.5.7/159
4003_7000	Pin Control Register n (PORTJ_PCR0)	32	R/W	See section	11.5.1/154
4003_7004	Pin Control Register n (PORTJ_PCR1)	32	R/W	See section	11.5.1/154
4003_7008	Pin Control Register n (PORTJ_PCR2)	32	R/W	See section	11.5.1/154
4003_700C	Pin Control Register n (PORTJ_PCR3)	32	R/W	See section	11.5.1/154
4003_7010	Pin Control Register n (PORTJ_PCR4)	32	R/W	See section	11.5.1/154
4003_7014	Pin Control Register n (PORTJ_PCR5)	32	R/W	See section	11.5.1/154
4003_7018	Pin Control Register n (PORTJ_PCR6)	32	R/W	See section	11.5.1/154
4003_701C	Pin Control Register n (PORTJ_PCR7)	32	R/W	See section	11.5.1/154
4003_7080	Global Pin Control Low Register (PORTJ_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4003_7084	Global Pin Control High Register (PORTJ_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4003_70A0	Interrupt Status Flag Register (PORTJ_ISFR)	32	w1c	0000_0000h	11.5.4/157
4003_70C0	Digital Filter Enable Register (PORTJ_DFER)	32	R/W	0000_0000h	11.5.5/158
4003_70C4	Digital Filter Clock Register (PORTJ_DFCR)	32	R/W	0000_0000h	11.5.6/159
4003_70C8	Digital Filter Width Register (PORTJ_DFWR)	32	R/W	0000_0000h	11.5.7/159
4003_8000	Pin Control Register n (PORTK_PCR0)	32	R/W	See section	11.5.1/154
4003_8004	Pin Control Register n (PORTK_PCR1)	32	R/W	See section	11.5.1/154
4003_8008	Pin Control Register n (PORTK_PCR2)	32	R/W	See section	11.5.1/154
4003_800C	Pin Control Register n (PORTK_PCR3)	32	R/W	See section	11.5.1/154
4003_8010	Pin Control Register n (PORTK_PCR4)	32	R/W	See section	11.5.1/154
4003_8014	Pin Control Register n (PORTK_PCR5)	32	R/W	See section	11.5.1/154
4003_8018	Pin Control Register n (PORTK_PCR6)	32	R/W	See section	11.5.1/154
4003_801C	Pin Control Register n (PORTK_PCR7)	32	R/W	See section	11.5.1/154
4003_8080	Global Pin Control Low Register (PORTK_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4003_8084	Global Pin Control High Register (PORTK_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4003_80A0	Interrupt Status Flag Register (PORTK_ISFR)	32	w1c	0000_0000h	11.5.4/157
4003_80C0	Digital Filter Enable Register (PORTK_DFER)	32	R/W	0000_0000h	11.5.5/158
4003_80C4	Digital Filter Clock Register (PORTK_DFCR)	32	R/W	0000_0000h	11.5.6/159
4003_80C8	Digital Filter Width Register (PORTK_DFWR)	32	R/W	0000_0000h	11.5.7/159
4003_9000	Pin Control Register n (PORTL_PCR0)	32	R/W	See section	11.5.1/154
4003_9004	Pin Control Register n (PORTL_PCR1)	32	R/W	See section	11.5.1/154

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_9008	Pin Control Register n (PORTL_PCR2)	32	R/W	See section	11.5.1/154
4003_900C	Pin Control Register n (PORTL_PCR3)	32	R/W	See section	11.5.1/154
4003_9010	Pin Control Register n (PORTL_PCR4)	32	R/W	See section	11.5.1/154
4003_9014	Pin Control Register n (PORTL_PCR5)	32	R/W	See section	11.5.1/154
4003_9018	Pin Control Register n (PORTL_PCR6)	32	R/W	See section	11.5.1/154
4003_901C	Pin Control Register n (PORTL_PCR7)	32	R/W	See section	11.5.1/154
4003_9080	Global Pin Control Low Register (PORTL_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4003_9084	Global Pin Control High Register (PORTL_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4003_90A0	Interrupt Status Flag Register (PORTL_ISFR)	32	w1c	0000_0000h	11.5.4/157
4003_90C0	Digital Filter Enable Register (PORTL_DFER)	32	R/W	0000_0000h	11.5.5/158
4003_90C4	Digital Filter Clock Register (PORTL_DFCR)	32	R/W	0000_0000h	11.5.6/159
4003_90C8	Digital Filter Width Register (PORTL_DFWR)	32	R/W	0000_0000h	11.5.7/159
4003_A000	Pin Control Register n (PORTM_PCR0)	32	R/W	See section	11.5.1/154
4003_A004	Pin Control Register n (PORTM_PCR1)	32	R/W	See section	11.5.1/154
4003_A008	Pin Control Register n (PORTM_PCR2)	32	R/W	See section	11.5.1/154
4003_A00C	Pin Control Register n (PORTM_PCR3)	32	R/W	See section	11.5.1/154
4003_A010	Pin Control Register n (PORTM_PCR4)	32	R/W	See section	11.5.1/154
4003_A014	Pin Control Register n (PORTM_PCR5)	32	R/W	See section	11.5.1/154
4003_A018	Pin Control Register n (PORTM_PCR6)	32	R/W	See section	11.5.1/154
4003_A01C	Pin Control Register n (PORTM_PCR7)	32	R/W	See section	11.5.1/154
4003_A080	Global Pin Control Low Register (PORTM_GPCLR)	32	W (always reads 0)	0000_0000h	11.5.2/156
4003_A084	Global Pin Control High Register (PORTM_GPCHR)	32	W (always reads 0)	0000_0000h	11.5.3/157
4003_A0A0	Interrupt Status Flag Register (PORTM_ISFR)	32	w1c	0000_0000h	11.5.4/157
4003_A0C0	Digital Filter Enable Register (PORTM_DFER)	32	R/W	0000_0000h	11.5.5/158
4003_A0C4	Digital Filter Clock Register (PORTM_DFCR)	32	R/W	0000_0000h	11.5.6/159
4003_A0C8	Digital Filter Width Register (PORTM_DFWR)	32	R/W	0000_0000h	11.5.7/159

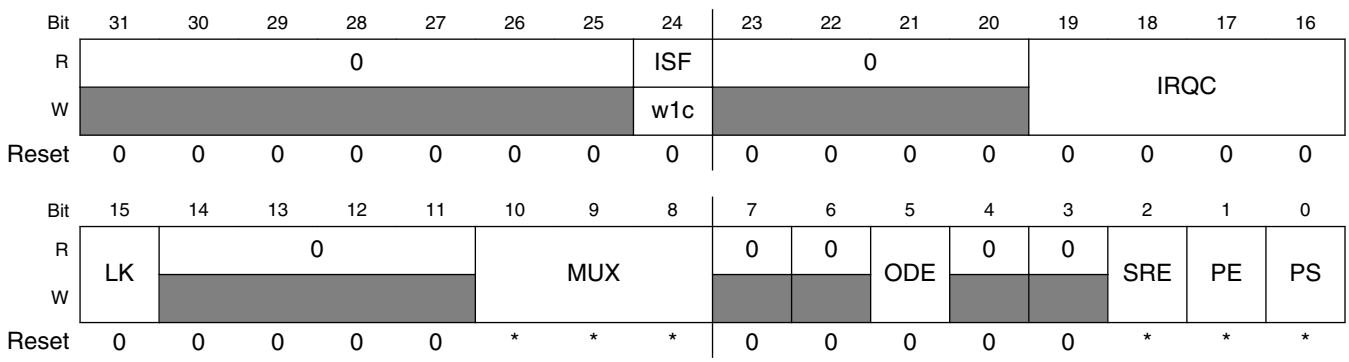
11.5.1 Pin Control Register n (PORTx_PCR*n*)

NOTE

See the Signal Multiplexing and Pin Assignment chapter for the reset value of this device.

See the GPIO Configuration section for details on the available functions for each pin.

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.



* Notes:
- MUX field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- SRE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PS field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.

PORTx_PCR*n* field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag This field is read-only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 IRQC	Interrupt Configuration This field is read-only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows: 0000 Interrupt Status Flag (ISF) is disabled. 0001 ISF flag and DMA request on rising edge. 0010 ISF flag and DMA request on falling edge. 0011 ISF flag and DMA request on either edge. 0100 Reserved. 0101 Reserved. 0110 Reserved. 0111 Reserved. 1000 ISF flag and Interrupt when logic 0. 1001 ISF flag and Interrupt on rising-edge. 1010 ISF flag and Interrupt on falling-edge. 1011 ISF flag and Interrupt on either edge. 1100 ISF flag and Interrupt when logic 1. 1101 Reserved. 1110 Reserved. 1111 Reserved.
15 LK	Lock Register 0 Pin Control Register fields [15:0] are not locked. 1 Pin Control Register fields [15:0] are locked and cannot be updated until the next system reset.
14–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 MUX	Pin Mux Control Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot. The corresponding pin is configured in the following pin muxing slot as follows: 000 Pin disabled (Alternative 0) (analog). 001 Alternative 1 (GPIO). 010 Alternative 2 (chip-specific). 011 Alternative 3 (chip-specific). 100 Alternative 4 (chip-specific). 101 Alternative 5 (chip-specific). 110 Alternative 6 (chip-specific). 111 Alternative 7 (chip-specific).
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

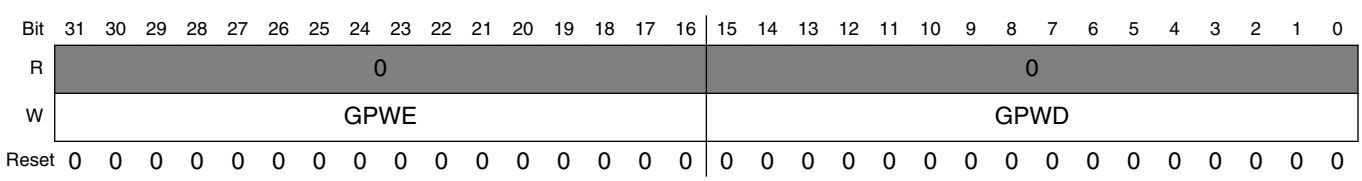
Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 ODE	Open Drain Enable This field is read-only for pins that do not support a configurable open drain output. Open drain configuration is valid in all digital pin muxing modes. 0 Open drain output is disabled on the corresponding pin. 1 Open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SRE	Slew Rate Enable This field is read-only for pins that do not support a configurable slew rate. Slew rate configuration is valid in all digital pin muxing modes. 0 Fast slew rate is configured on the corresponding pin, if the pin is configured as a digital output. 1 Slow slew rate is configured on the corresponding pin, if the pin is configured as a digital output.
1 PE	Pull Enable This field is read-only for pins that do not support a configurable pull resistor. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support a configurable pull resistor. Pull configuration is valid in all digital pin muxing modes. 0 Internal pullup or pulldown resistor is not enabled on the corresponding pin. 1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0 PS	Pull Select This bit is read only for pins that do not support a configurable pull resistor direction. Pull configuration is valid in all digital pin muxing modes. 0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding PE field is set. 1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding PE field is set.

11.5.2 Global Pin Control Low Register (PORTx_GPCLR)

Only 32-bit writes are supported to this register.



PORTx_GPCLR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (15 through 0) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

11.5.3 Global Pin Control High Register (PORTx_GPCHR)

Only 32-bit writes are supported to this register.

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

PORTx_GPCHR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (31 through 16) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

11.5.4 Interrupt Status Flag Register (PORTx_ISFR)

The corresponding bit is read only for pins that do not support interrupt generation.

The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Pin Control Register, and each flag can be cleared in either location.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_ISFR field descriptions

Field	Description
ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the field. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

11.5.5 Digital Filter Enable Register (PORTx_DFER)

The digital filter configuration is valid in all digital pin muxing modes.

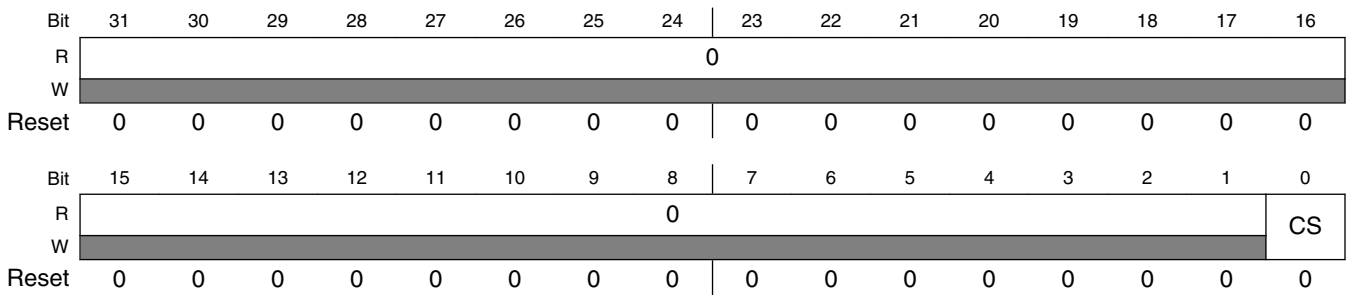
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DFE																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_DFER field descriptions

Field	Description
DFE	Digital Filter Enable The digital filter configuration is valid in all digital pin muxing modes. The output of each digital filter is reset to zero at system reset and whenever the digital filter is disabled. Each bit in the field enables the digital filter of the same number as the field. 0 Digital filter is disabled on the corresponding pin and output of the digital filter is reset to zero. 1 Digital filter is enabled on the corresponding pin, if the pin is configured as a digital input.

11.5.6 Digital Filter Clock Register (PORTx_DFCR)

The digital filter configuration is valid in all digital pin muxing modes.

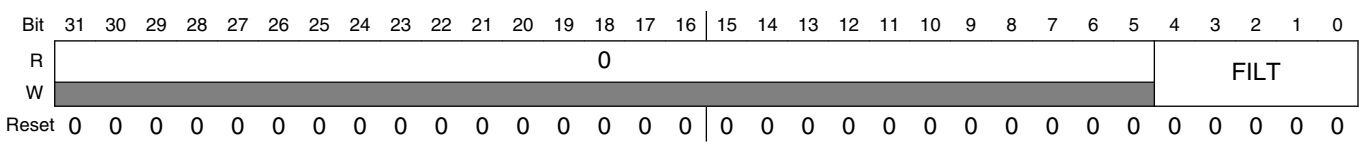


PORTx_DFCR field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CS	Clock Source The digital filter configuration is valid in all digital pin muxing modes. Configures the clock source for the digital input filters. Changing the filter clock source must be done only when all digital filters are disabled. 0 Digital filters are clocked by the bus clock. 1 Digital filters are clocked by the LPO clock.

11.5.7 Digital Filter Width Register (PORTx_DFWR)

The digital filter configuration is valid in all digital pin muxing modes.



PORTx_DFWR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FILT	Filter Length

Table continues on the next page...

PORTx_DFWR field descriptions (continued)

Field	Description
	The digital filter configuration is valid in all digital pin muxing modes. Configures the maximum size of the glitches, in clock cycles, that the digital filter absorbs for the enabled digital filters. Glitches that are longer than this register setting will pass through the digital filter, and glitches that are equal to or less than this register setting are filtered. Changing the filter length must be done only after all filters are disabled.

11.6 Functional description

11.6.1 Pin control

Each port pin has a corresponding Pin Control register, PORT_PCRn, associated with it.

The upper half of the Pin Control register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred. The LK bit (bit 15 of Pin Control Register PCR*n*) locks the lower 16 bits of each Pin Control register and blocks any writes to that register until the next system reset.

The lower half of the Pin Control register configures the following functions for each pin within the 32-bit port.

- Pullup or pulldown enable on selected pins
- Drive strength and slew rate configuration on selected pins
- Open drain enable on selected pins
- Passive input filter enable on selected pins
- Pin Muxing mode

The functions apply across all digital pin muxing modes and individual peripherals do not override the configuration in the Pin Control register. For example, if an I^2C function is enabled on a pin, that does not override the pullup or open drain configuration for that pin.

When the Pin Muxing mode is configured for analog or is disabled, all the digital functions on that pin are disabled. This includes the pullup and pulldown enables, output buffer enable, input buffer enable, and passive filter enable.

A lock field also exists that allows the configuration for each pin to be locked until the next system reset. When locked, writes to the lower half of that pin control register are ignored, although a bus error is not generated on an attempted write to a locked register.

The configuration of each Pin Control register is retained when the PORT module is disabled.

Whenever a pin is configured in any digital pin muxing mode, the input buffer for that pin is enabled allowing the pin state to be read via the corresponding GPIO Port Data Input Register (GPIO_PDIR) or allowing a pin interrupt or DMA request to be generated. If a pin is ever floating when its input buffer is enabled, then this can cause an increase in power consumption and must be avoided. A pin can be floating due to an input pin that is not connected or an output pin that has tri-stated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) will ensure a pin does not float when its input buffer is enabled; note that the internal pull resistor is automatically disabled whenever the output buffer is enabled allowing the Pull Enable bit to remain set. Configuring the Pin Muxing mode to disabled or analog will disable the pin's input buffer and results in the lowest power consumption.

11.6.2 Global pin control

The two global pin control registers allow a single register write to update the lower half of the pin control register on up to 16 pins, all with the same value. Registers that are locked cannot be written using the global pin control registers.

The global pin control registers are designed to enable software to quickly configure multiple pins within the one port for the same peripheral function. However, the interrupt functions cannot be configured using the global pin control registers.

The global pin control registers are write-only registers, that always read as 0.

11.6.3 External interrupts

The external interrupt capability of the PORT module is available in all digital pin muxing modes provided the PORT module is enabled.

Each pin can be individually configured for any of the following external interrupt modes:

- Interrupt disabled, default out of reset
- Active high level sensitive interrupt
- Active low level sensitive interrupt
- Rising edge sensitive interrupt
- Falling edge sensitive interrupt
- Rising and falling edge sensitive interrupt

- Rising edge sensitive DMA request
- Falling edge sensitive DMA request
- Rising and falling edge sensitive DMA request

The interrupt status flag is set when the configured edge or level is detected on the pin or at the output of the digital input filter, if the digital input digital filter is enabled. When not in Stop mode, the input is first synchronized to the bus clock to detect the configured level or edge transition.

The PORT module generates a single interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates after the interrupt status flags for all enabled interrupts have been cleared by writing a logic 1 to the ISF flag in either the PORT_ISFR or PORT_PCRn registers.

The PORT module generates a single DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that port. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

During Stop mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

11.6.4 Digital filter

The digital filter capabilities of the PORT module are available in all digital Pin Muxing modes if the PORT module is enabled.

The clock used for all digital filters within one port can be configured between the bus clock or the LPO clock. This selection must be changed only when all digital filters for that port are disabled. If the digital filters for a port are configured to use the bus clock, then the digital filters are bypassed for the duration of Stop mode. While the digital filters are bypassed, the output of each digital filter always equals the input pin, but the internal state of the digital filters remains static and does not update due to any change on the input pin.

The filter width in clock size is the same for all enabled digital filters within one port and must be changed only when all digital filters for that port are disabled.

The output of each digital filter is logic zero after system reset and whenever a digital filter is disabled. After a digital filter is enabled, the input is synchronized to the filter clock, either the bus clock or the LPO clock. If the synchronized input and the output of

the digital filter remain different for a number of filter clock cycles equal to the filter width register configuration, then the output of the digital filter updates to equal the synchronized filter input.

The minimum latency through a digital filter equals two or three filter clock cycles plus the filter width configuration register.

Chapter 12
System Integration Module (SIM)

12.1 Introduction

The System Integration Module (SIM) provides system control and chip configuration registers. The module is divided into two power domains: Low-Power SIM (SIM_LP) and High-Power SIM (SIM_HP). The SIM_HP registers are invalid during VLLSx modes and reset on exit from VLLSx modes, POR or LVD. Whereas, SIM_LP registers retain their content throughout VLLSx modes but reset on POR or LVD events. The register space belonging to 4003_E000h – 4003_EFFFh is part of SIM_LP and from 4003_F000h – 4003_FFFFh is part of SIM_HP.

12.2 Features

- System clocking configuration:
  - System clock divider values
  - Architectural clock gating control
  - LPTimer clock selection
- Flash size configuration
- System RAM power mode controls
- LPTMR external clock and input capture selection
- CLKOUT selection
- Peripheral XBAR input selection for scenarios where multiple peripheral exist to be used as input sources
- System device identification through SDID field
- ADC compensation values for –40℃, 25℃ and 105℃

12.3 Memory map and register definition

The SIM module contains many fields to select clock source and dividers for various module clocks.

SIM memory map

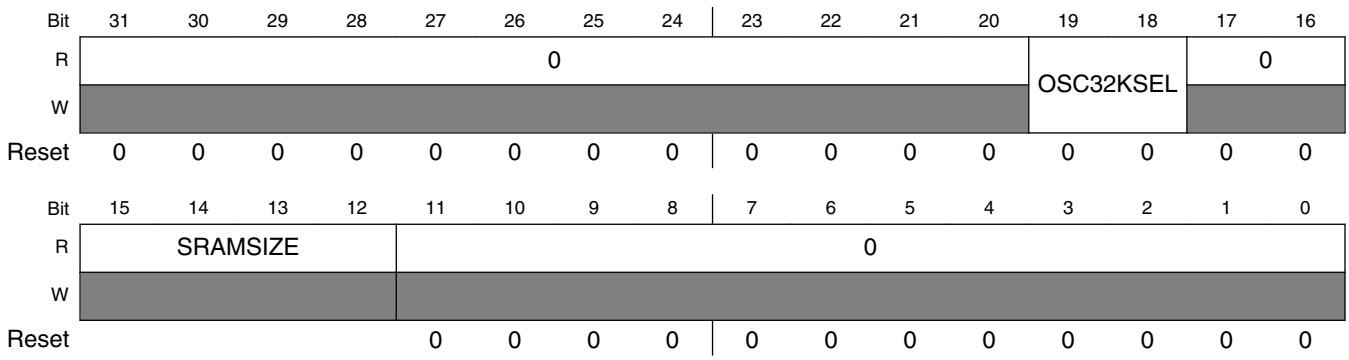
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_E000	System Options Register 1 (SIM_SOPT1)	32	R/W	0000_6000h	12.3.1/167
4003_E004	SOPT1 Configuration Register (SIM_SOPT1_CFG)	32	R/W	0000_0000h	12.3.2/168
4003_F004	System Control Register (SIM_CTRL_REG)	32	R/W	0000_0000h	12.3.3/169
4003_F024	System Device Identification Register (SIM_SDID)	32	R	Undefined	12.3.4/172
4003_F034	System Clock Gating Control Register 4 (SIM_SCGC4)	32	R/W	7800_8050h	12.3.5/174
4003_F038	System Clock Gating Control Register 5 (SIM_SCGC5)	32	R/W	000B_0000h	12.3.6/176
4003_F03C	System Clock Gating Control Register 6 (SIM_SCGC6)	32	R/W	C000_0001h	12.3.7/180
4003_F040	System Clock Gating Control Register 7 (SIM_SCGC7)	32	R/W	0000_0003h	12.3.8/183
4003_F044	System Clock Divider Register 1 (SIM_CLKDIV1)	32	R/W	0100_0000h	12.3.9/184
4003_F04C	Flash Configuration Register 1 (SIM_FCFG1)	32	R/W	See section	12.3.10/186
4003_F050	Flash Configuration Register 2 (SIM_FCFG2)	32	R	See section	12.3.11/188
4003_F054	Unique Identification Register High (SIM_UIDH)	32	R	Undefined	12.3.12/189
4003_F058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R	Undefined	12.3.13/189
4003_F05C	Unique Identification Register Mid-Low (SIM_UIDML)	32	R	Undefined	12.3.14/190
4003_F060	Unique Identification Register Low (SIM_UIDL)	32	R	Undefined	12.3.15/190
4003_F06C	Miscellaneous Control Register (SIM_MISC_CTL)	32	R/W	See section	12.3.16/190
4003_F0C8	ADC Compensation Register 0 (SIM_ADC_COMP0)	32	R	Undefined	12.3.17/195
4003_F0CC	ADC Compensation Register 1 (SIM_ADC_COMP1)	32	R	See section	12.3.18/196

12.3.1 System Options Register 1 (SIM_SOPT1)

NOTE

The SOPT1 register is only reset on POR or LVD.

Address: 4003_E000h base + 0h offset = 4003_E000h



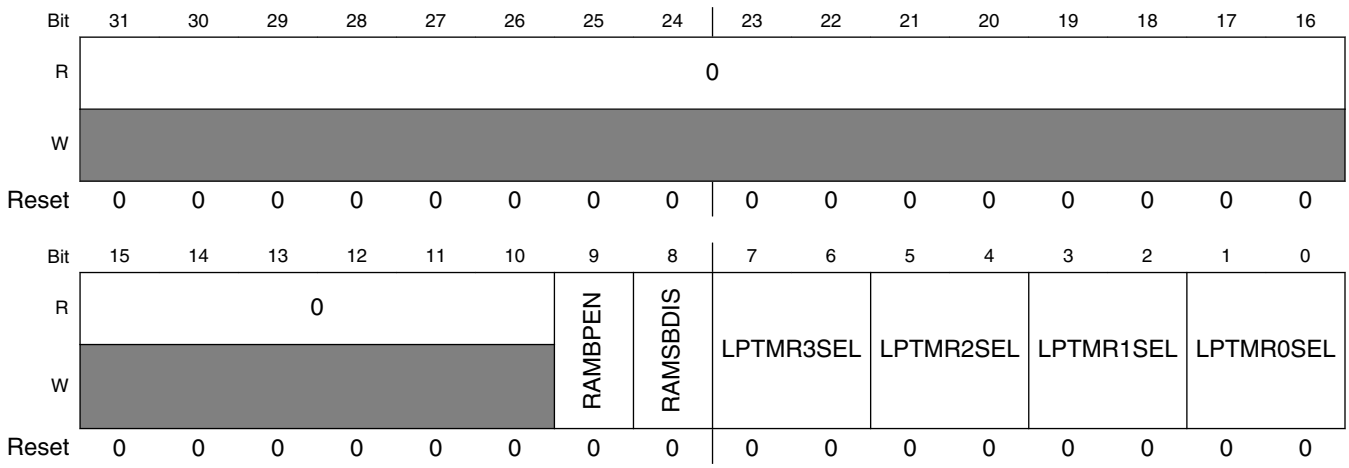
SIM_SOPT1 field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 OSC32KSEL	32K oscillator clock select Selects the 32 kHz clock source for LPTMR , CLKOUT , LCD , EWM, WDOG. This bit is reset only on POR/LVD. 00 OSC32KCLK (RTC Oscillator output) 01 ERCLK32K 10 MCGIRCLK 11 LPO
17–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 SRAMSIZE	Returns the size of the system RAM 0110 32 KB System RAM
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.2 SOPT1 Configuration Register (SIM_SOPT1_CFG)

The SOPT1CFG register is reset on system reset not VLLS.

Address: 4003_E000h base + 4h offset = 4003_E004h



SIM_SOPT1_CFG field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RAMBPEN	RAM Bitline Precharge Enable Enable System SRAM bitline precharge during VLPR and VLPW modes. 0 Bitline precharge of system SRAM disabled during VLPR and VLPW modes. 1 Bitline precharge of system SRAM enabled during VLPR and VLPW modes.
8 RAMSBDIS	Disable source bias of System SRAM arrays during VLPR and VLPW modes. 0 Source bias of System SRAM enabled during VLPR and VLPW modes. 1 Source bias of System SRAM disabled during VLPR and VLPW modes.
7–6 LPTMR3SEL	LP Timer Channel3 Select This field is used to select source for LP timer channel3. 00 Pad PTD5 01 Pad PTG0 10 Pad PTG6 11 Reserved
5–4 LPTMR2SEL	LP Timer Channel2 Select This field is used to select source for LP timer channel2. 00 Pad PTD6 01 Pad PTF3

Table continues on the next page...

SIM_SOPT1_CFG field descriptions (continued)

Field	Description
	10 Pad PTG5 11 Reserved
3–2 LPTMR1SEL	LP Timer Channel1 Select This field is used to select source for LP timer channel1. 00 Pad PTE4 01 Pad PTF4 10 Pad PTG1 11 Reserved
LPTMR0SEL	LP Timer Channel0 Select This field is used to select source for LP timer channel0. 00 CMP[0] output 01 CMP[1] output 10 CMP[2] output 11 Reserved

12.3.3 System Control Register (SIM_CTRL_REG)

NOTE

Writes to this register can only be performed in Supervisor mode

Address: 4003_E000h base + 1004h offset = 4003_F004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMRFREEZE	0	0		LPUARTSRC		0	AFEOUTCLKSEL	XBARCLKOUT			0			PLLFLLSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SPI1_INV3	SPI1_INV2	SPI1_INV1	SPI1_INV0	SPI0_INV3	SPI0_INV2	SPI0_INV1	SPI0_INV0	CLKOUT			ADCTRGSEL		0	PLLVLPEN	NMIDIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_CTRL_REG field descriptions

Field	Description
31 TMRFREEZE	QTMR counters Freeze control 0 QTMR counters operate normally. 1 QTMR counters and OFLAGs are reset. Clearing this bit will resume QTMR operation.
30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–26 LPUARTSRC	LPUART clock Source configuration 00 Clock disabled 01 MCGPLLCLK/MCGFLLCLK 10 OSCERCLK 11 MCGIRCLK
25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 AFEOUTCLKSEL	AFE clock output select 0 AFE output clock is divided by AFE clock prescaler. 1 AFE output clock is NOT divided by AFE clock prescaler.
23–21 XBARCLKOUT	XBAR clock out selection Selects the clock to XBAR_IN[5] 000 Disabled 001 Gated Core Clk 010 Bus Clk 011 LPO clock from PMC 100 IRC clock from MCG 101 MUXed 32 kHz source (please refer to SOPT1[19:18] for possible options) 110 MHz Oscillator external reference clock 111 PLL clock output from MCG
20–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 PLLFLLSEL	PLL/FLL selection This bitfield configures the MCGPLLCLK/MCGFLLCLK. 00 MCGFLLCLK 01 MCGPLLCLK 10 BUSCLK 11 OSC32KCLK (RTC Oscillator output)
15 SPI1_INV3	This bit inverts the SPI1 signal output. 0 not inverted 1 inverts MISO
14 SPI1_INV2	This bit inverts the SPI1 signal output.

Table continues on the next page...

SIM_CTRL_REG field descriptions (continued)

Field	Description
	0 not inverted 1 inverts MOSI
13 SPI1_INV1	This bit inverts the SPI1 signal output. 0 not inverted 1 inverts SCK
12 SPI1_INV0	This bit inverts the SPI1 signal output. 0 not inverted 1 inverts SS
11 SPI0_INV3	This bit inverts the SPI0 signal output. 0 not inverted 1 inverts MISO
10 SPI0_INV2	This bit inverts the SPI0 signal output. 0 not inverted 1 inverts MOSI
9 SPI0_INV1	This bit inverts the SPI0 signal output. 0 not inverted 1 inverts SCK
8 SPI0_INV0	This bit inverts the SPI0 signal output. 0 not inverted 1 inverts SS
7–5 CLKOUT	Clock out Select This field can be used to select one of the following clocks for CLKOUT pin. 000 Disabled 001 Gated Core Clk 010 Bus Clk 011 LPO clock from PMC 100 IRC clock from MCG 101 Muxed 32Khz source (please refer to SOPT1[19:18] for possible options) 110 MHz Oscillator external reference clock 111 PLL clock output from MCG
4–3 ADCTRGSEL	SAR ADC Trigger Clock Select Selects the clock used to generate the ADC triggers. 00 Bus Clock **NOTE:** During Low Power Modes such as stop, the Bus clock is not available for conversion and should not be selected in case a conversion needs to be performed while in stop. 01 ADC asynchronous Clock 10 ERCLK32K 11 OSCCLK

Table continues on the next page...

SIM_CTRL_REG field descriptions (continued)

Field	Description
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 PLLVLPEN	PLL VLP Enable When set, this bit enables the PLL in VLP modes. **NOTE:** • Valid operating mode transitions with PLLVLPEN enabled are: RUN->VLPR(also with PSTOP1)->RUN and RUN->VLPR(+PSTOP1)->RESET. • MCG must be set to BLPI mode by software, if PLLVLPEN is enabled before entering VLPR mode. • Prior to entering any low power mode, disable PLLVLPEN. • When in VLPR mode with PLL module operating, do not disable the PMC bandgap.
0 NMIDIS	NMI Disable This field is used as an alternate to FTFL_FOPT[2] to disable the NMI temporarily. It is also advised to program this bit while changing the state of the NMI pin from NMI to GPIO functionality. 0 NMI enabled 1 NMI disabled

12.3.4 System Device Identification Register (SIM_SDID)

Address: 4003_E000h base + 1024h offset = 4003_F024h

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

* Notes:
- x = Undefined at reset.

SIM_SDID field descriptions

Field	Description
31–28 FAMID	Metering family ID Indicates Presence of LCD module 0001 Device derivatives without LCD 0011 Device derivatives with LCD
27–24 SUBFAMID	Sub-Family ID Specifies the number of AFE channels supported 0000 Device derivatives with NO AFE enabled 0001 Device derivatives with 1 AFE enabled 0010 Device derivatives with 2 AFE enabled

Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
	0011 Device derivatives with 3 AFE enabled 0100 Device derivatives with 4 AFE enabled
23–20 SERIESID	Series ID Metering Series 0011 Metering Series
19–16 ATTR	Attribute ID Specifies type of Core on the device 0000 M0+ core
15–12 SRAMSIZE	SRAM Size Indicates the size of SRAM present on the device 0110 32 KB SRAM
11–8 REVID	Revision ID Marks the changes with every Mask Set 0000 First cut
7–4 DIEID	Die ID Marks the change in Base Layer 0000 First cut
PINID	Pincount identification Specifies the pincount of the device. 0000-0010 Reserved 0011 Reserved 0100 Reserved 0101 64-pin 0110 Reserved 0111 Reserved 1000 100-pin 1001 128-pin 1010 144-pin 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

12.3.5 System Clock Gating Control Register 4 (SIM_SCGC4)

NOTE

Writes to this register can only be performed in Supervisor mode

Address: 4003_E000h base + 1034h offset = 4003_F034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	1				0				SPI1	SPI0	0		CMP	0	
W																
Reset	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VREF	0	UART3	UART2	UART1	UART0	0	I2C1	I2C0	1	0	1	0		EWM	0
W																
Reset	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

SIM_SCGC4 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
26–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 SPI1	SPI1 Clock Gate Control This bit controls the clock gate to the SPI1 module. 0 Clock disabled 1 Clock enabled
21 SPI0	SPI0 Clock Gate Control This bit controls the clock gate to the SPI0 module. 0 Clock disabled 1 Clock enabled
20–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 CMP	High Speed Comparator Clock Gate Control.

Table continues on the next page...

SIM_SCGC4 field descriptions (continued)

Field	Description
	This bit controls the clock gate to the CMP module. 0 Clock disabled 1 Clock enabled
17–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 VREF	VREF Clock Gate Control This bit controls the clock gate to the VREF module. 0 Clock disabled 1 Clock enabled
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 UART3	UART3 Clock Gate Control This bit controls the clock gate to the UART3 module. 0 Clock disabled 1 Clock enabled
12 UART2	UART2 Clock Gate Control This bit controls the clock gate to the UART2 module. 0 Clock disabled 1 Clock enabled
11 UART1	UART1 Clock Gate Control This bit controls the clock gate to the UART1 module. 0 Clock disabled 1 Clock enabled
10 UART0	UART0 Clock Gate Control This bit controls the clock gate to the UART0 module. 0 Clock disabled 1 Clock enabled
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 I2C1	I2C1 Clock Gate Control This bit controls the clock gate to the I2C1 module. 0 Clock disabled 1 Clock enabled
7 I2C0	I2C0 Clock Gate Control This bit controls the clock gate to the I2C0 module.

Table continues on the next page...

SIM_SCGC4 field descriptions (continued)

Field	Description
	0 Clock disabled 1 Clock enabled
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 EWM	External Watchdog Monitor Clock gate control This bit controls the clock gate to the EWM module. 0 Clock disabled 1 Clock enabled
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.6 System Clock Gating Control Register 5 (SIM_SCGC5)

NOTE

Writes to this register can only be performed in Supervisor mode.

Address: 4003_E000h base + 1038h offset = 4003_F038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					TMR3	TMR2	TMR1	TMR0	0	XBAR	0	1	0	RTCREG	RTC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PORTI	PORTH	PORTG	PORTF	PORTE	PORTD	PORTC	PORTB	PORTA	0		SLCD	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_SCGC5 field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 TMR3	QaudTimer channel 3 Clock Gate Control This bit controls the clock gate to the QTMR0_TMR3. 0 Clock disabled 1 Clock enabled
25 TMR2	QaudTimer channel 2 Clock Gate Control This bit controls the clock gate to the QTMR0_TMR2. 0 Clock disabled 1 Clock enabled
24 TMR1	QaudTimer channel 1 Clock Gate Control This bit controls the clock gate to the QTMR0_TMR1. 0 Clock disabled 1 Clock enabled
23 TMR0	QaudTimer channel 0 Clock Gate Control This bit controls the clock gate to the QTMR0_TMR0. 0 Clock disabled 1 Clock enabled
22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 XBAR	Peripheral Crossbar Clock Gate Control This bit controls the clock gate to the XBAR module. 0 Clock disabled 1 Clock enabled
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 RTCREG	iRTC_REG_FILE Clock Gate Control This bit controls the clock gate to the iRTC_REG_FILE. 0 Clock disabled 1 Clock enabled
16 RTC	iRTC Clock Gate Control This bit controls the clock gate to the iRTC module.

Table continues on the next page...

SIM_SCGC5 field descriptions (continued)

Field	Description
	0 Clock disabled 1 Clock enabled
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 PORTI	PCTLI Clock Gate Control This bit controls the clock gate to the PCTLI module. 0 Clock disabled 1 Clock enabled
13 PORTH	PCTLH Clock Gate Control This bit controls the clock gate to the PCTLH module. 0 Clock disabled 1 Clock enabled
12 PORTG	PCTLG Clock Gate Control This bit controls the clock gate to the PCTLG module. 0 Clock disabled 1 Clock enabled
11 PORTF	PCTLF Clock Gate Control This bit controls the clock gate to the PCTLF module. 0 Clock disabled 1 Clock enabled
10 PORTE	PCTLE Clock Gate Control This bit controls the clock gate to the PCTLE module. 0 Clock disabled 1 Clock enabled
9 PORTD	PCTLD Clock Gate Control This bit controls the clock gate to the PCTLD module. 0 Clock disabled 1 Clock enabled
8 PORTC	PCTLC Clock Gate Control This bit controls the clock gate to the PCTLC module. 0 Clock disabled 1 Clock enabled
7 PORTB	PCTLB Clock Gate Control This bit controls the clock gate to the PCTLB module.

Table continues on the next page...

SIM_SCGC5 field descriptions (continued)

Field	Description
	0 Clock disabled 1 Clock enabled
6 PORTA	PCTLA Clock Gate Control This bit controls the clock gate to the PCTLA module. 0 Clock disabled 1 Clock enabled
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SLCD	Segmented LCD Clock Gate Control This bit controls the clock gate to the SLCD module. 0 Clock disabled 1 Clock enabled
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.7 System Clock Gating Control Register 6 (SIM_SCGC6)

NOTE

Writes to this register can only be performed in Supervisor mode

Address: 4003_E000h base + 103Ch offset = 4003_F03Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1		0		0						0		Reserved			
W				LPTMR		PORTM	PORTL	PORTK	PORTJ	PDB		CRC				AFE
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R				0					0							
W	Reserved	PIT1	PIT0		ADC	LPUART	RNGA								DMACHMUX	FTFA
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SIM_SCGC6 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 LPTMR	LPTMR Clock Gate Control This bit controls the clock gate to the LPTMR module. 0 Clock disabled 1 Clock enabled
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 PORTM	PCTLM Clock Gate Control This bit controls the clock gate to the PCTLM module. 0 Clock disabled 1 Clock enabled
25 PORTL	PCTLL Clock Gate Control This bit controls the clock gate to the PCTLL module. 0 Clock disabled 1 Clock enabled
24 PORTK	PCTLK Clock Gate Control This bit controls the clock gate to the PCTLK module. 0 Clock disabled 1 Clock enabled
23 PORTJ	PCTLJ Clock Gate Control This bit controls the clock gate to the PCTLJ module. 0 Clock disabled 1 Clock enabled
22 PDB	PDB Clock Gate Control This bit controls the clock gate to the PDB module. 0 Clock disabled 1 Clock enabled
21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 CRC	Programmable CRC Clock Gate Control This bit controls the clock gate to the PCRC module. 0 Clock disabled 1 Clock enabled

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
19–17 Reserved	This field is reserved.
16 AFE	AFE Clock Gate Control This bit controls the clock to all four AFE channels. 0 Clock disabled 1 Clock enabled
15 Reserved	Reserved. This field is reserved. This field is reserved and should always be written a value 0. This is reserved for future expansion and writing a 1 will cause unexpected behavior.
14 PIT1	PIT1 Clock Gate Control This bit controls the clock gate to the PIT1 module. 0 Clock disabled 1 Clock enabled
13 PIT0	PIT0 Clock Gate Control This bit controls the clock gate to the PIT0 module. 0 Clock disabled 1 Clock enabled
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 ADC	SAR ADC Clock Gate Control This bit controls the clock gate to the SAR ADC module. 0 Clock disabled 1 Clock enabled
10 LPUART	LPUART Clock Gate Control This bit controls the clock gate to the LPUART module. 0 Clock disabled 1 Clock enabled
9 RNGA	RNGA Clock Gate Control This bit controls the clock gate to the RNGA module. 0 Clock disabled 1 Clock enabled
8–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DMACHMUX	DMA Channel MUX Clock Gate Control This bit controls the clock gate to the DMA channel MUX module.

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
	0 Clock disabled 1 Clock enabled
0 FTFA	FTFA Clock Gate Control This bit controls the clock gate to the FTFA Module. 0 Clock disabled 1 Clock enabled

12.3.8 System Clock Gating Control Register 7 (SIM_SCGC7)

NOTE

Writes to this register can only be performed in Supervisor mode

Address: 4003_E000h base + 1040h offset = 4003_F040h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	CAU	DMA	MPU
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

SIM_SCGC7 field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CAU	CAU Clock Gate control. This bit controls the clock gate to the CAU module. 0 Clock disabled 1 Clock enabled
1 DMA	DMA Clock Gate control. This bit controls the clock gate to the DMA module.

Table continues on the next page...

SIM_SCGC7 field descriptions (continued)

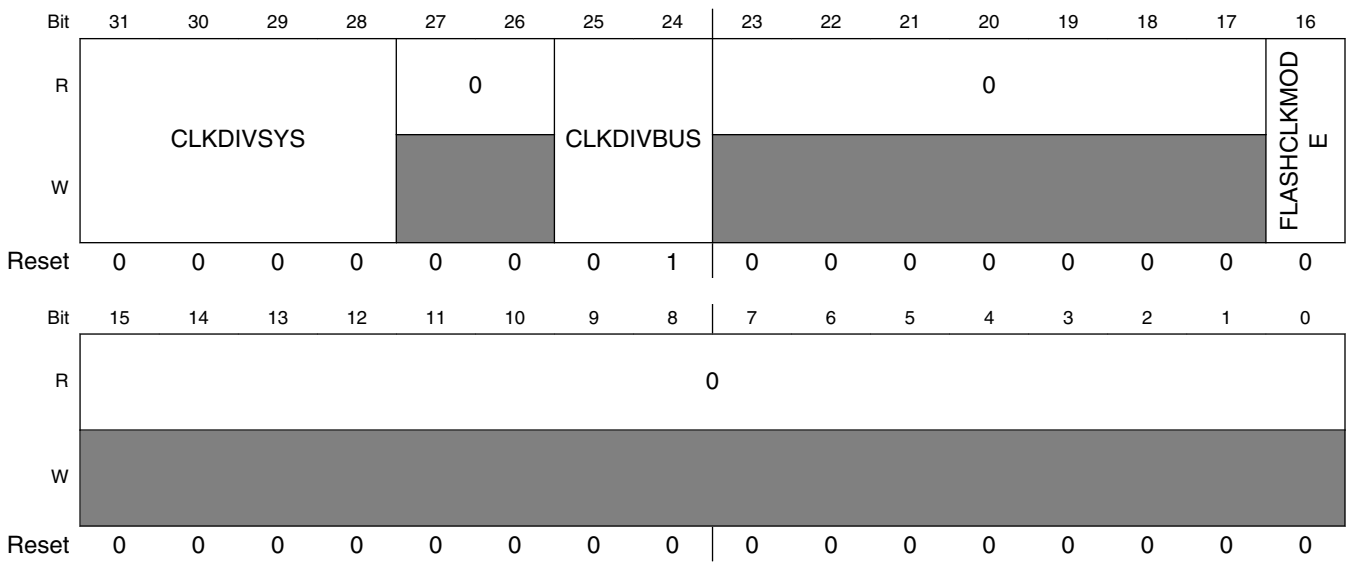
Field	Description
	0 Clock disabled 1 Clock enabled
0 MPU	MPU Clock Gate control. This bit controls the clock gate to the MPU module. 0 Clock disabled 1 Clock enabled

12.3.9 System Clock Divider Register 1 (SIM_CLKDIV1)

NOTE

Writes to this register can only be performed in Supervisor mode.

Address: 4003_E000h base + 1044h offset = 4003_F044h



SIM_CLKDIV1 field descriptions

Field	Description
31–28 CLKDIVSYS	System Clock divider This field can be used to program Core/Platform divider. 0000 Divide by 1 0001 Divide by 2

Table continues on the next page...

SIM_CLKDIV1 field descriptions (continued)

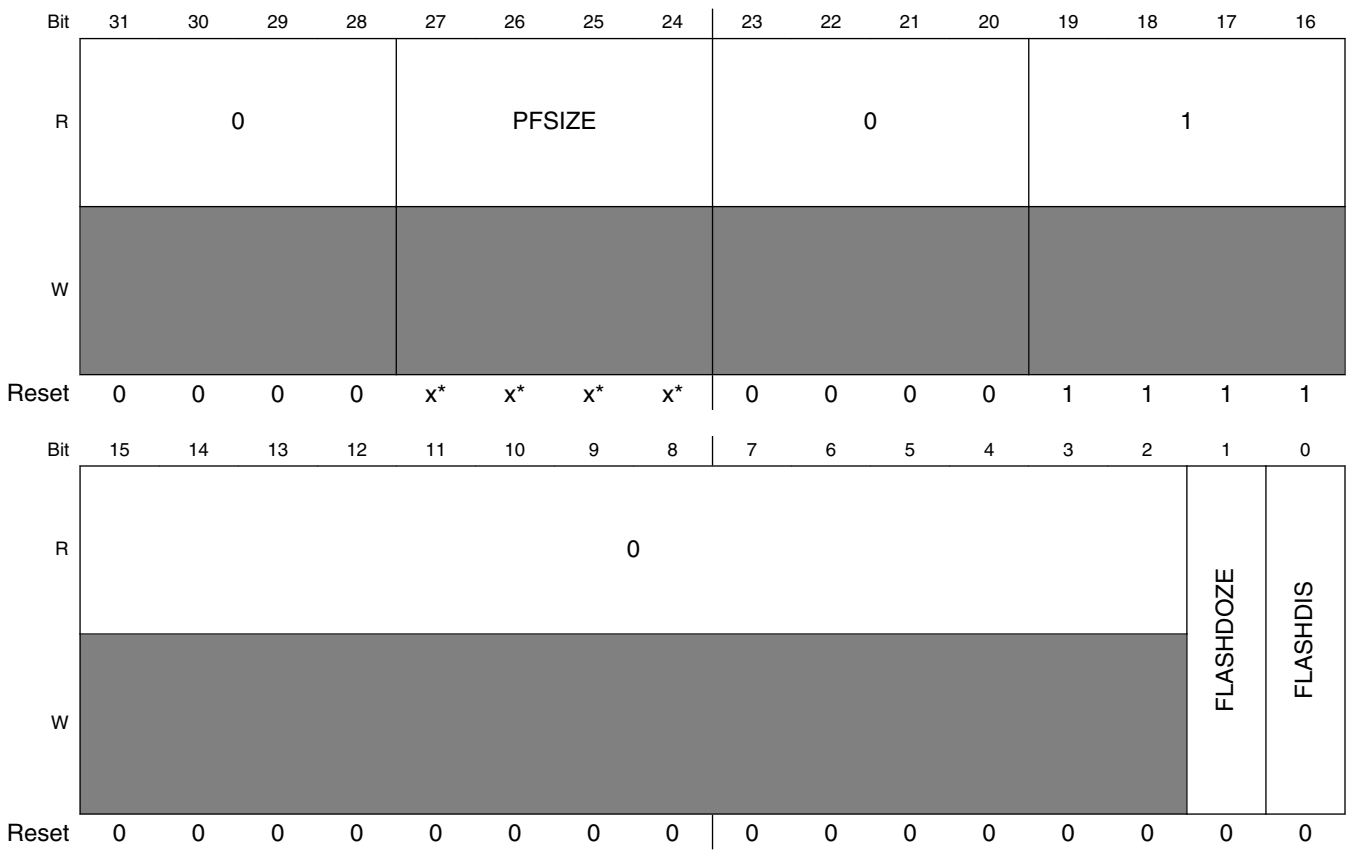
Field	Description
	0010 Divide by 3 0011 Divide by 4 and so on... If FOPT[0] is 0, the divider is set to div-by-8 after system reset is deasserted (after completion of system initialization sequence).
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 CLKDIVBUS	Bus Clock divider This field can be used to program Bus clock divider. 00 SYSCLK:BUSCLK = 1:1 01 SYSCLK:BUSCLK = 2:1 10 SYSCLK:BUSCLK = 3:1 11 SYSCLK:BUSCLK = 4:1
23–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 FLASHCLKMODE	Flash Clock Mode This field is used to select the Flash clock mode. 0 Flash Clock is the same as BUS clock. 1 Flash Clock is a half of BUS clock.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.10 Flash Configuration Register 1 (SIM_FCFG1)

NOTE

Writes to this register can only be performed in Supervisor mode

Address: 4003_E000h base + 104Ch offset = 4003_F04Ch



* Notes:
- x = Undefined at reset.

SIM_FCFG1 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 PFSIZE	Program flash size This field specifies the amount of program flash memory available on the device . Undefined values are reserved.

Table continues on the next page...

SIM_FCFG1 field descriptions (continued)

Field	Description
	0000 Reserved 0001 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 128 KB of program flash memory 1000 Reserved 1001 256 KB of program flash memory 1111 (Default)
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
15–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FLASHDOZE	Flash Doze When set, Flash memory is disabled for the duration of Wait mode. An attempt by the DMA or other bus master to access the Flash when the Flash is disabled will result in a bus error. This bit should be clear during VLP modes. The Flash will be automatically enabled again at the end of Wait mode so interrupt vectors do not need to be relocated out of Flash memory. The wakeup time from Wait mode is extended when this bit is set. 0 Flash remains enabled during Wait mode 1 Flash is disabled for the duration of Wait mode
0 FLASHDIS	Flash Disable Flash accesses are disabled (and generate a bus error) and the Flash memory is placed in a low power state. This bit should not be changed during VLP modes. Relocate the interrupt vectors out of Flash memory before disabling the Flash. 0 Flash is enabled 1 Flash is disabled

12.3.11 Flash Configuration Register 2 (SIM_FCFG2)

NOTE

Writes to this register can only be performed in Supervisor mode.

Address: 4003_E000h base + 1050h offset = 4003_F050h

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

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

* Notes:
- x = Undefined at reset.

SIM_FCFG2 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–24 MAXADDR	Max address This field concatenated with 13 trailing zeros indicates the first invalid address of each program flash block. For example, if MAXADDR0 = 0x20, the first invalid address of flash is 0x0004_0000. This would be the MAXADDR value for a device with 256 KB program flash in flash memory.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.12 Unique Identification Register High (SIM_UIDH)

Address: 4003_E000h base + 1054h offset = 4003_F054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID127_96																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UIDH field descriptions

Field	Description
UID127_96	Unique Identification UID[127:96] Unique identification for the device.

12.3.13 Unique Identification Register Mid-High (SIM_UIDMH)

Address: 4003_E000h base + 1058h offset = 4003_F058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID95_64																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UIDMH field descriptions

Field	Description
UID95_64	Unique Identification UID[95:64] Unique identification for the device.

12.3.14 Unique Identification Register Mid-Low (SIM_UIDML)

Address: 4003_E000h base + 105Ch offset = 4003_F05Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID63_32																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UIDML field descriptions

Field	Description
UID63_32	Unique Identification UID[63:32] Unique identification for the device.

12.3.15 Unique Identification Register Low (SIM_UIDL)

Address: 4003_E000h base + 1060h offset = 4003_F060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID31_0																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UIDL field descriptions

Field	Description
UID31_0	Unique Identification UID[31:0] Unique identification for the device.

12.3.16 Miscellaneous Control Register (SIM_MISC_CTL)

This register contains various fields to control miscellaneous features such as receive/ transmit select for UART, PIT output selection, EWM input selection, ClkOut selection, etc.

NOTE

Writes to this register can only be performed in Supervisor mode.

Address: 4003_E000h base + 106Ch offset = 4003_F06Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VREFBUFPD	VREFBUFINSEL	VREFBUFOUTEN	RTCCLKSEL	TMR3PCSSEL		TMR2PCSSEL		TMR1PCSSEL		TMR0PCSSEL		TMR3SCSEL	TMR2SCSEL	TMR1SCSEL	TMR0SCSEL
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR0PLLSEL	EWMINSEL	0		UART3IRSEL	UART2IRSEL	UART1IRSEL	UART0IRSEL	UARTMODTYPE	AFECLKPADDIR	AFECLKSEL		DMADONESEL		PDBADCTRG	OSCON
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x*

* Notes:
- x = Undefined at reset.

SIM_MISC_CTL field descriptions

Field	Description
31 VREFBUFPD	VrefBuffer Power Down Powers down VrefBuffer when set 0 Buffer Enabled 1 Buffer Powered Down
30 VREFBUFINSEL	VrefBuffer Input Select

Table continues on the next page...

SIM_MISC_CTL field descriptions (continued)

Field	Description
	Selects Between Internal 1.2v reference and external reference for SAR and DAC 0 Internal Reference selected as Buffer Input 1 External Reference selected as Buffer Input
29 VREFBUFOUTEN	VrefBuffer Output Enable Operates switch in VrefBuffer to enable the internal 1.2v reference to be driven to VREF pad. 0 Buffer does not drive PAD 1 Buffer drives selected voltage (selected by vref_buffer_sel) on pad
28 RTCCLKSEL	RTC Clock select Selects between MCGIRCLK and OSC_32K clk for RTC operation 0 RTC OSC_32K clock selected 1 MCGIRCLK selected
27–26 TMR3PCSSEL	Quadtimer Channel3 Primary Count Source Select This is used to select primary count source (clock) for Quadtimer Channel3. Configure TMR3_CTRL[PCS] = 1xxxb to route selected primary clock source to the timer Channel3. 00 Bus Clock 01 Peripheral Crossbar Output [9] 10 Peripheral Crossbar Output [10] 11 Disabled
25–24 TMR2PCSSEL	Quadtimer Channel2 Primary Count Source Select This is used to select primary count source (clock) for Quadtimer Channel2. Configure TMR2_CTRL[PCS] = 1xxxb to route selected primary clock source to the timer Channel2. 00 Bus Clock 01 Peripheral Crossbar Output [9] 10 Peripheral Crossbar Output [10] 11 Disabled
23–22 TMR1PCSSEL	Quadtimer Channel1 Primary Count Source Select This is used to select primary count source (clock) for Quadtimer Channel1. Configure TMR1_CTRL[PCS] = 1xxxb to route selected primary clock source to the timer Channel1. 00 Bus Clock 01 Peripheral Crossbar Output [9] 10 Peripheral Crossbar Output [10] 11 Disabled
21–20 TMR0PCSSEL	Quadtimer Channel0 Primary Count Source Select This is used to select primary count source (clock) for Quadtimer Channel0. Configure TMR0_CTRL[PCS] = 1xxxb to route selected primary clock source to the timer Channel0. 00 Bus Clock 01 Peripheral Crossbar Output [9]

Table continues on the next page...

SIM_MISC_CTL field descriptions (continued)

Field	Description
	10 Peripheral Crossbar Output [10] 11 Disabled
19 TMR3SCSEL	Quadtimer Channel3 Secondary Count source Select This field is used to select secondary count input source for Quadtimer Channel3. Configure TMR3_CTRL[SCS] = 11b to route selected secondary source to the timer Channel3. 0 Pad PTE5 or PTD1, depending upon PCTL configuration. 1 Peripheral Crossbar (XBAR) Output[8]
18 TMR2SCSEL	Quadtimer Channel2 Secondary Count source Select This field is used to select secondary count input source for Quadtimer Channel2. Configure TMR2_CTRL[SCS] = 10b to route selected secondary source to the timer Channel2. 0 Pad PTF7 or PTF0, depending upon PCTL configuration. 1 Peripheral Crossbar (XBAR) Output[7]
17 TMR1SCSEL	Quadtimer Channel1 Secondary Count source Select This field is used to select secondary count input source for Quadtimer Channel1. Configure TMR1_CTRL[SCS] = 01b to route selected secondary source to the timer Channel1. 0 Pad PTG0 or PTC6, depending upon PCTL configuration. 1 Peripheral Crossbar (XBAR) Output[6]
16 TMR0SCSEL	Quadtimer Channel0 Secondary Count source Select This field is used to select secondary count input source for Quadtimer Channel0. Configure TMR0_CTRL[SCS] = 00b to route selected secondary source to the timer Channel0. 0 Pad PTF1 or PTD5, depending upon PCTL configuration. 1 Peripheral Crossbar (XBAR) Output[5]
15 TMR0PLLSEL	Timer CH0 PLL clock Select 0 Selects Bus Clock as source for the Timer CH0 1 Selects the PLL_AFE clock as the source for Timer CH0. The PLL_AFE clock source is itself selected using the MISC_CTL[5:4]
14 EWMINSEL	External Watchdog Monitor Input Select This field is used to select input for external watchdog monitor. 0 Input from PAD (PTL[3], PTE[2] or PTE[3] as selected from Pinmux control) 1 Peripheral Crossbar (XBAR) Output[32]
13–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 UART3IRSEL	UART3 IrDA Select Exclusive selection with other UARTs. Do not select while another UART is selected for the same functionality 0 Pad RX input (PTC[3] or PTD[7], as selected in Pinmux control) selected for RX input of UART3 and UART3 TX signal is not used for modulation 1 UART3 selected for IrDA modulation. UART3 TX modulated by XBAR_OUT[14] and UART3 RX input connected to XBAR_OUT[13].

Table continues on the next page...

SIM_MISC_CTL field descriptions (continued)

Field	Description
	NOTE: UART*x*IRSEL cannot configure XBAR_OUT[14] and XBAR_OUT[13] automatically, and they need extra configuration in XBAR. User should configure XBAR[14:13] accordingly.
10 UART2IRSEL	UART2 IrDA Select Exclusive selection with other UARTs. Do not select while another UART is selected for the same functionality 0 Pad RX input PTI[6] or PTE[6] selected for RX input of UART2 and UART2 TX signal is not used for modulation 1 UART2 selected for IrDA modulation. UART2 TX modulated by XBAR_OUT[14] and UART2 RX input connected to XBAR_OUT[13]. **NOTE:** UART*x*IRSEL cannot configure XBAR_OUT[14] and XBAR_OUT[13] automatically, and they need extra configuration in XBAR. User should configure XBAR[14:13] accordingly.
9 UART1IRSEL	UART1 IrDA Select Exclusive selection with other UARTs. Do not select while another UART is selected for the same functionality 0 Pad RX input (PTD[2], PTI[0] or PTK[5], as selected in Pinmux control) selected for RX input of UART1 and UART1 TX signal is not used for modulation 1 UART1 selected for IrDA modulation. UART1 TX modulated by XBAR_OUT[14] and UART1 RX input connected to XBAR_OUT[13]. **NOTE:** UART*x*IRSEL cannot configure XBAR_OUT[14] and XBAR_OUT[13] automatically, and they need extra configuration in XBAR. User should configure XBAR[14:13] accordingly.
8 UART0IRSEL	UART0 IrDA Select Exclusive selection with other UARTs. Do not select while another UART is selected for the same functionality 0 Pad RX input (PTD[0], PTF[3] or PTK[3], as selected in Pinmux control) selected for RX input of UART0 and UART0 TX signal is not used for modulation 1 UART0 selected for IrDA modulation. UART0 TX modulated by XBAR_OUT[14] and UART0 RX input connected to XBAR_OUT[13]. **NOTE:** UART*x*IRSEL cannot configure XBAR_OUT[14] and XBAR_OUT[13] automatically, and they need extra configuration in XBAR. User should configure XBAR[14:13] accordingly.
7 UARTMODTYPE	UART Modulation Type Selects between TypeA and TypeB modulation for IrDA support 0 TypeA (OR'ed) Modulation selected for IrDA 1 TypeB (AND'ed) Modulation selected for IrDA
6 AFECLKPADDIR	AFE Clock Pad Direction Controls the direction of the AFE CLK pin 0 AFE CLK PAD is input 1 AFE CLK PAD is output
5–4 AFECLKSEL	AFE Clock Source Select (SIMAFECLK selection) Selects between PLL, FLL and OSC clock as the source for the PLL clock branch for AFE Clock Mux

Table continues on the next page...

SIM_MISC_CTL field descriptions (continued)

Field	Description
	00 MCG PLL Clock selected 01 MCG FLL Clock selected 10 OSC Clock selected 11 Disabled
3–2 DMADONESEL	DMA Done select This field can be used to select the DMA Done flag to drive XBAR_IN[32] 00 DMA0 01 DMA1 10 DMA2 11 DMA3
1 PDBADCTRG	PDB bypass XBAR as ADC trigger This field can make PDB to trigger ADC directly. 0 XBAR to trigger ADC 1 PDB output to trigger ADC
0 OSCON	RTC oscillator status This field shows the status of RTC oscillator. 0 RTC oscillator is disabled. 1 RTC oscillator is enabled.

12.3.17 ADC Compensation Register 0 (SIM_ADC_COMP0)

Address: 4003_E000h base + 10C8h offset = 4003_F0C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADCCOMPVAL1																ADCCOMPVAL0															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_ADC_COMP0 field descriptions

Field	Description
31–16 ADCCOMPVAL1	ADC Temperature Compensation Value 1 ADC Temperature Compensation Value at –40°C (ambient).
ADCCOMPVAL0	ADC Temperature Compensation Value 0 ADC Temperature Compensation Value at 25°C (ambient).

12.3.18 ADC Compensation Register 1 (SIM_ADC_COMP1)

Address: 4003_E000h base + 10CCh offset = 4003_F0CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																ADCCOMPVAL2																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_ADC_COMP1 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ADCCOMPVAL2	ADC Temperature Compensation Value 2 ADC Temperature Compensation Value at 105 ℃ (ambient).

12.4 Functional description

See Introduction section.

Chapter 13
Inter-Peripheral Crossbar Switch (XBAR)

13.1 Chip-specific XBAR information

13.1.1 Overview

The Peripheral Crossbar (XBAR) allows certain signals inside the device to be connected to specific peripherals thus allowing flexibility in system configuration. Certain pins of the device are connected to the XBAR outputs to allow certain signals to output from device (which otherwise do not have a dedicated pin) or certain pins to be connected to certain modules via XBAR inputs (which otherwise do not have a connection to those pins).

13.1.2 Instantiation Information

This device has one instance of the XBAR. This module has 52 inputs and 44 outputs. There are 4 interrupt/DMA request signal outputs from this crossbar.

13.1.3 Peripheral Interconnects

The XBAR has the following connections to various peripherals or device IOs. The XBAR Inputs can be configured to connect to XBAR Outputs as per application requirements.

Table 13-1. XBAR Interconnects

No.	XBAR Inputs	XBAR Outputs
0	Logic 1 (VDD)	XBAR DMA request or Interrupt 0[1]
1	Logic 0 (VSS)	XBAR DMA request or Interrupt 1[1]

Table continues on the next page...

Table 13-1. XBAR Interconnects (continued)

No.	XBAR Inputs	XBAR Outputs
2	AFE clock[2]	XBAR DMA request or Interrupt 2[1]
3	AFE modulator 0 data output	XBAR DMA request or Interrupt 3[1]
4	LPTimer Output	CMP2 sample window input
5	Clock Output[3]	Quad Timer channel 0 secondary input[4]
6	Quad Timer channel 0 output	Quad Timer channel 1 secondary input[4]
7	Quad Timer channel 1 output	Quad Timer channel 2 secondary input[4]
8	Quad Timer channel 2 output	Quad Timer channel 3 secondary input[4]
9	Quad Timer channel 3 output	Quad Timer primary clock input 1
10	iRTC Clock Output	Quad Timer primary clock input 2
11	CMP0 Output	CMP0 Sample Window input
12	CMP1 Output	CMP1 Sample Window input
13	iRTC Alarm Output	UART Rx Input[5]
14	UART Tx Output (after modulation)[6]	UART Tx Modulation Output[7]
15	EWM Output ($\overline{\text{EWM\_OUT}}$)	SAR ADC trigger select A pulse[8]
16	PIT 0 TIF0	SAR ADC trigger select B pulse[8]
17	XBAR Input pin 0	XBAR Output pin 0
18	XBAR Input pin 1	XBAR Output pin 1
19	XBAR Input pin 2	XBAR Output pin 2
20	XBAR Input pin 3	XBAR Output pin 3
21	XBAR Input pin 4	XBAR Output pin 4
22	XBAR Input pin 5	XBAR Output pin 5
23	XBAR Input pin 6	XBAR Output pin 6
24	XBAR Input pin 7	XBAR Output pin 7
25	XBAR Input pin 8	XBAR Output pin 8
26	OR'ed conversion complete flag for all SAR ADC channels	SAR ADC trigger select C pulse[8]
27	OR'ed conversion complete flag for all AFE channels[9]	SAR ADC trigger select D pulse[8]
28	AFE Channel 0 conversion complete	AFE Channel 0 Trigger
29	AFE Channel 1 conversion complete	AFE Channel 1 Trigger
30	AFE Channel 2 conversion complete	AFE Channel 2 Trigger
31	AFE Channel 3 conversion complete	AFE Channel 3 Trigger
32	DMA Done Signal[10]	EWM input (EWM_IN)
33	XBAR Input pin 9	XBAR Output pin 9
34	XBAR Input pin 10	XBAR Output pin 10
35	CMP2 Output	PDB0 Pre-trigger0 B2B Ack
36	PIT 0 TIF1	PDB0 Pre-trigger1 B2B Ack
37	PIT 1 TIF0	PDB0 Pre-trigger2 B2B Ack
38	PIT 1 TIF1	PDB0 Pre-trigger3 B2B Ack
39	AFE modulator 1 data output	PDB0 Trigger Input 14[11]
40	AFE modulator 2 data output	External modulator data input for AFE Channel 0[12]

Table continues on the next page...

Table 13-1. XBAR Interconnects (continued)

No.	XBAR Inputs	XBAR Outputs
41	AFE modulator 3 data output	External modulator data input for AFE Channel 1[12]
42	SAR ADC COCO A	External modulator data input for AFE Channel 2[12]
43	SAR ADC COCO B	External modulator data input for AFE Channel 3[12]
44	SAR ADC COCO C	Reserved
45	SAR ADC COCO D	Reserved
46	PDB0 CH0 Pre-trigger 0	Reserved
47	PDB0 CH0 Pre-trigger 1	Reserved
48	PDB0 CH0 Pre-trigger 2	Reserved
49	PDB0 CH0 Pre-trigger 3	Reserved
50	PDB0 CH0 Trigger	Reserved
51	PDB0 Pulse-Out 0	Reserved

1. DMA or Interrupt selectable inside XBAR.
2. The AFE clock can be selected through SIM_CTRL_REG[AFEOUTCLKSEL] bit. This is the modulator clock output from AFE.
3. Clock out is configurable to select one of the following clocks through SIM_CTRL_REG[XBARCLKOUT] :

NOTE

This selection does not affect the clock selection of CLKOUT pin and does not drive CLKOUT pin.

   - Core Clock (gated in WAIT)
   - Bus Clock
   - IRC Clock
   - PLL Clock
   - 32 kHz OSC clock
   - MHz OSC clock
   - 1 kHz LPO clock
4. The XBAR should not be configured to have any of Quad Timer channel's outputs driving any of the channel's secondary inputs.
5. Comparator 0 output routed to this output for IrDA. Connects to UART Rx module input which is selected by software. SIM_MISC_CTL[UART0IRSEL], SIM_MISC_CTL[UART1IRSEL], SIM_MISC_CTL[UART2IRSEL] and SIM_MISC_CTL[UART3IRSEL] provide for selection of UART for IrDA. Common selection for Rx & Tx.
6. This is user selectable. This is output via XBAR according to UART selected. Software should select the correct UART & XBAR output pin.
7. This signal is used to modulate the UART Tx output. The modulation type (ANDed or ORed) can be selected through SIM_MISC_CTL[UARTMODTYPE].
8. Generates the trigger to SAR. Separate trigger for each result register. The XBAR trigger to ADC will take effect when SIM_MISC_CTL[PDBADCTRG] bit is cleared.See SAR Chapter for details on triggering.
9. For cases where ΣΔs work independently.
10. User configurable to select DMA Done signal from any channel. DMA Done signal will not work when DMA transfer is initiated using DMA_DCRn[START] bit.
11. This XBAR output can be selected as PDB0 trigger input by configuring PDB0_SC[TRGSEL] = 0b1110.
12. Clock input for ΣΔ ADC will not be routed from XBAR.

NOTE

All interconnections required in XBAR must be done before any other peripheral is enabled.

13.2 Introduction

13.2.1 Overview

This module implements an array of M N-input combinational muxes. All muxes share the same N inputs in the same order, but each mux has its own independent select field.

The intended application of this module is to provide a flexible crossbar switch function that allows any input (typically from external GPIO or internal module outputs) to be connected to any output (typically to external GPIO or internal module inputs) under user control. This is used to allow user configuration of data paths between internal modules and between internal modules and GPIO.

A subset of the muxes can be configured to support edge detection and either interrupt or DMA request generation based on detected signal edges on the mux output. This allows signal transitions on the signals feeding the crossbar to trigger interrupts or initiate data transfers via DMA into or out of other system modules.

13.2.2 Features

The XBAR module design includes these distinctive features:

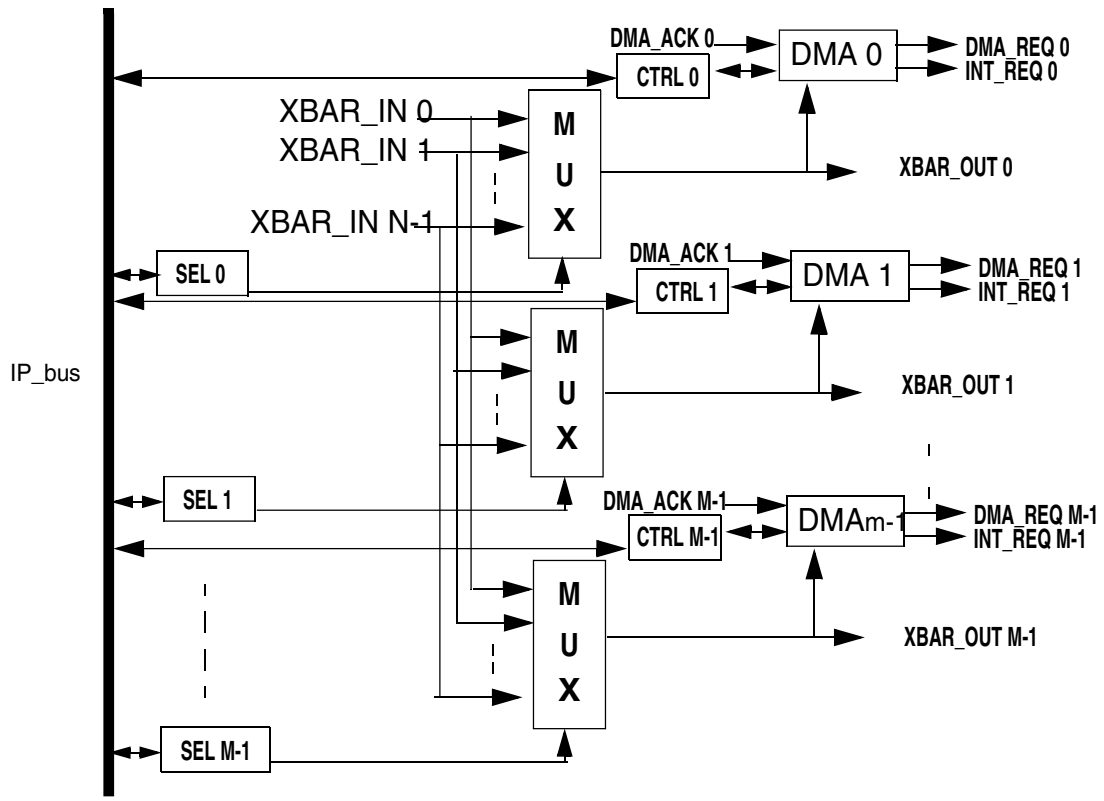
- M identical N-input muxes with individual select fields.
- Edge detection with associated interrupt or DMA request generation for a subset of mux outputs.
- Memory mapped registers with IPBus interface for select and control fields.
- Register write protection input signal.

13.2.3 Modes of Operation

The XBAR module design operates in only a single mode of operation: Functional Mode. The various counting modes are detailed in Functional Mode.

13.2.4 Block Diagram

The block diagram for XBAR is shown in -.



Note: Control field support can be restricted to a subset of outputs. CTRLn represents the memory mapped control fields in the XBAR_CTRL registers for each corresponding XBAR_OUT output. Control fields provide edge status, edge selection, and interrupt/DMA generation controls.

Figure 13-1. XBAR Block Diagram

13.3 Signal Descriptions

The following table summarizes the module's external signals.

Table 13-2. Control Signal Properties

Name	I/O Type	Function	Reset State	Notes
XBAR_OUT [0:NUMOUT-1]	O	Mux Outputs with configurable width	*	
XBAR_IN [0:NUMIN-1]	I	Mux Inputs with configurable width	*	
DMA_REQ	O	DMA request	0	
INT_REQ	O	Interrupt request	0	
DMA_ACK	I	DMA acknowledge	0	

At reset, each output XBAR_OUT[*] contains the reset value of the signal driving XBAR_IN[0].

13.3.1 XBAR_OUT[0:NUM_OUT-1] - MUX Outputs

This is a one-dimensional array of the mux outputs. The value on each output XBAR_OUT[n] is determined by the setting of the corresponding memory mapped register SELn such that XBAR_OUT[n] = XBAR_IN[SELn].

13.3.2 XBAR_IN[0:NUM_IN-1] - MUX Inputs

This is a one-dimensional array consisting of the inputs shared by each mux. All muxes share the same inputs in the same order.

13.3.3 DMA_REQ[n] - DMA Request Output(s)

DMA_REQ[n] is a DMA request to the DMA controller.

13.3.4 DMA_ACK[n] - DMA Acknowledge Input(s)

DMA_ACK[n] is a DMA acknowledge input from the DMA controller.

13.3.5 INT_REQ[n] - Interrupt Request Output(s)

INT_REQ[n] is an interrupt request output to the interrupt controller.

13.4 Memory Map and Register Descriptions

The XBAR module has select registers and control registers.

In the XBAR select registers, the SELn fields select which of the shared inputs (XBAR_IN[*]) is muxed to each mux output (XBAR_OUT[*]). There is one SELn field per mux and therefore one per XBAR_OUT output. Crossbar output XBAR_OUT[n]

presents the value of XBAR_IN[SELn]. Each select register contains two SELn fields. In the first select register, the LSBs contain the select field for mux 0, and the MSBs contain the select field for mux 1. The pattern repeats in subsequent select registers.

The actual signals connected to XBAR_IN and XBAR_OUT are application specific and are described in the Chip Configuration details.

The XBAR control register configures edge detection, interrupt, and DMA features for the XBAR_OUT[0] output.

XBAR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_5000	Crossbar Select Register 0 (XBAR_SEL0)	16	R/W	0000h	13.4.1/204
4005_5002	Crossbar Select Register 1 (XBAR_SEL1)	16	R/W	0000h	13.4.2/204
4005_5004	Crossbar Select Register 2 (XBAR_SEL2)	16	R/W	0000h	13.4.3/205
4005_5006	Crossbar Select Register 3 (XBAR_SEL3)	16	R/W	0000h	13.4.4/205
4005_5008	Crossbar Select Register 4 (XBAR_SEL4)	16	R/W	0000h	13.4.5/206
4005_500A	Crossbar Select Register 5 (XBAR_SEL5)	16	R/W	0000h	13.4.6/206
4005_500C	Crossbar Select Register 6 (XBAR_SEL6)	16	R/W	0000h	13.4.7/207
4005_500E	Crossbar Select Register 7 (XBAR_SEL7)	16	R/W	0000h	13.4.8/207
4005_5010	Crossbar Select Register 8 (XBAR_SEL8)	16	R/W	0000h	13.4.9/208
4005_5012	Crossbar Select Register 9 (XBAR_SEL9)	16	R/W	0000h	13.4.10/208
4005_5014	Crossbar Select Register 10 (XBAR_SEL10)	16	R/W	0000h	13.4.11/209
4005_5016	Crossbar Select Register 11 (XBAR_SEL11)	16	R/W	0000h	13.4.12/209
4005_5018	Crossbar Select Register 12 (XBAR_SEL12)	16	R/W	0000h	13.4.13/210
4005_501A	Crossbar Select Register 13 (XBAR_SEL13)	16	R/W	0000h	13.4.14/210
4005_501C	Crossbar Select Register 14 (XBAR_SEL14)	16	R/W	0000h	13.4.15/211
4005_501E	Crossbar Select Register 15 (XBAR_SEL15)	16	R/W	0000h	13.4.16/211
4005_5020	Crossbar Select Register 16 (XBAR_SEL16)	16	R/W	0000h	13.4.17/212
4005_5022	Crossbar Select Register 17 (XBAR_SEL17)	16	R/W	0000h	13.4.18/212
4005_5024	Crossbar Select Register 18 (XBAR_SEL18)	16	R/W	0000h	13.4.19/213
4005_5026	Crossbar Select Register 19 (XBAR_SEL19)	16	R/W	0000h	13.4.20/213
4005_5028	Crossbar Select Register 20 (XBAR_SEL20)	16	R/W	0000h	13.4.21/214
4005_502A	Crossbar Select Register 21 (XBAR_SEL21)	16	R/W	0000h	13.4.22/214
4005_502C	Crossbar Control Register 0 (XBAR_CTRL0)	16	R/W	0000h	13.4.23/214
4005_502E	Crossbar Control Register 1 (XBAR_CTRL1)	16	R/W	0000h	13.4.24/217

13.4.1 Crossbar Select Register 0 (XBAR_SEL0)

Address: 4005_5000h base + 0h offset = 4005_5000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL1						0		SEL0					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL0 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL1	Input (XBAR_INn) to be muxed to XBAR_OUT1 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL0	Input (XBAR_INn) to be muxed to XBAR_OUT0 (refer to Functional Description section for input/output assignment)

13.4.2 Crossbar Select Register 1 (XBAR_SEL1)

Address: 4005_5000h base + 2h offset = 4005_5002h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL3						0		SEL2					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL1 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL3	Input (XBAR_INn) to be muxed to XBAR_OUT3 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL2	Input (XBAR_INn) to be muxed to XBAR_OUT2 (refer to Functional Description section for input/output assignment)

13.4.3 Crossbar Select Register 2 (XBAR_SEL2)

Address: 4005_5000h base + 4h offset = 4005_5004h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL5						0		SEL4					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL2 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL5	Input (XBAR_INn) to be muxed to XBAR_OUT5 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL4	Input (XBAR_INn) to be muxed to XBAR_OUT4 (refer to Functional Description section for input/output assignment)

13.4.4 Crossbar Select Register 3 (XBAR_SEL3)

Address: 4005_5000h base + 6h offset = 4005_5006h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL7						0		SEL6					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL3 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL7	Input (XBAR_INn) to be muxed to XBAR_OUT7 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL6	Input (XBAR_INn) to be muxed to XBAR_OUT6 (refer to Functional Description section for input/output assignment)

13.4.5 Crossbar Select Register 4 (XBAR_SEL4)

Address: 4005_5000h base + 8h offset = 4005_5008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL9						0		SEL8					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL4 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL9	Input (XBAR_INn) to be muxed to XBAR_OUT9 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL8	Input (XBAR_INn) to be muxed to XBAR_OUT8 (refer to Functional Description section for input/output assignment)

13.4.6 Crossbar Select Register 5 (XBAR_SEL5)

Address: 4005_5000h base + Ah offset = 4005_500Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL11						0		SEL10					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL5 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL11	Input (XBAR_INn) to be muxed to XBAR_OUT11 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL10	Input (XBAR_INn) to be muxed to XBAR_OUT10 (refer to Functional Description section for input/output assignment)

13.4.7 Crossbar Select Register 6 (XBAR_SEL6)

Address: 4005_5000h base + Ch offset = 4005_500Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL13						0		SEL12					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL6 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL13	Input (XBAR_INn) to be muxed to XBAR_OUT13 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL12	Input (XBAR_INn) to be muxed to XBAR_OUT12 (refer to Functional Description section for input/output assignment)

13.4.8 Crossbar Select Register 7 (XBAR_SEL7)

Address: 4005_5000h base + Eh offset = 4005_500Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL15						0		SEL14					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL7 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL15	Input (XBAR_INn) to be muxed to XBAR_OUT15 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL14	Input (XBAR_INn) to be muxed to XBAR_OUT14 (refer to Functional Description section for input/output assignment)

13.4.9 Crossbar Select Register 8 (XBAR_SEL8)

Address: 4005_5000h base + 10h offset = 4005_5010h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL17						0		SEL16					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL8 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL17	Input (XBAR_INn) to be muxed to XBAR_OUT17 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL16	Input (XBAR_INn) to be muxed to XBAR_OUT16 (refer to Functional Description section for input/output assignment)

13.4.10 Crossbar Select Register 9 (XBAR_SEL9)

Address: 4005_5000h base + 12h offset = 4005_5012h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL19						0		SEL18					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL9 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL19	Input (XBAR_INn) to be muxed to XBAR_OUT19 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL18	Input (XBAR_INn) to be muxed to XBAR_OUT18 (refer to Functional Description section for input/output assignment)

13.4.11 Crossbar Select Register 10 (XBAR_SEL10)

Address: 4005_5000h base + 14h offset = 4005_5014h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL21						0		SEL20					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL10 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL21	Input (XBAR_INn) to be muxed to XBAR_OUT21 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL20	Input (XBAR_INn) to be muxed to XBAR_OUT20 (refer to Functional Description section for input/output assignment)

13.4.12 Crossbar Select Register 11 (XBAR_SEL11)

Address: 4005_5000h base + 16h offset = 4005_5016h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL23						0		SEL22					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL11 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL23	Input (XBAR_INn) to be muxed to XBAR_OUT23 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL22	Input (XBAR_INn) to be muxed to XBAR_OUT22 (refer to Functional Description section for input/output assignment)

13.4.13 Crossbar Select Register 12 (XBAR_SEL12)

Address: 4005_5000h base + 18h offset = 4005_5018h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL25						0		SEL24					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL12 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL25	Input (XBAR_INn) to be muxed to XBAR_OUT25 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL24	Input (XBAR_INn) to be muxed to XBAR_OUT24 (refer to Functional Description section for input/output assignment)

13.4.14 Crossbar Select Register 13 (XBAR_SEL13)

Address: 4005_5000h base + 1Ah offset = 4005_501Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL27						0		SEL26					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL13 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL27	Input (XBAR_INn) to be muxed to XBAR_OUT27 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL26	Input (XBAR_INn) to be muxed to XBAR_OUT26 (refer to Functional Description section for input/output assignment)

13.4.15 Crossbar Select Register 14 (XBAR_SEL14)

Address: 4005_5000h base + 1Ch offset = 4005_501Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL29						0		SEL28					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL14 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL29	Input (XBAR_INn) to be muxed to XBAR_OUT29 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL28	Input (XBAR_INn) to be muxed to XBAR_OUT28 (refer to Functional Description section for input/output assignment)

13.4.16 Crossbar Select Register 15 (XBAR_SEL15)

Address: 4005_5000h base + 1Eh offset = 4005_501Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL31						0		SEL30					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL15 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL31	Input (XBAR_INn) to be muxed to XBAR_OUT31 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL30	Input (XBAR_INn) to be muxed to XBAR_OUT30 (refer to Functional Description section for input/output assignment)

13.4.17 Crossbar Select Register 16 (XBAR_SEL16)

Address: 4005_5000h base + 20h offset = 4005_5020h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL33						0		SEL32					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL16 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL33	Input (XBAR_INn) to be muxed to XBAR_OUT33
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL32	Input (XBAR_INn) to be muxed to XBAR_OUT32

13.4.18 Crossbar Select Register 17 (XBAR_SEL17)

Address: 4005_5000h base + 22h offset = 4005_5022h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL35						0		SEL34					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL17 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL35	Input (XBAR_INn) to be muxed to XBAR_OUT35
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL34	Input (XBAR_INn) to be muxed to XBAR_OUT34

13.4.19 Crossbar Select Register 18 (XBAR_SEL18)

Address: 4005_5000h base + 24h offset = 4005_5024h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL37						0		SEL36					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL18 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL37	Input (XBAR_INn) to be muxed to XBAR_OUT37
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL36	Input (XBAR_INn) to be muxed to XBAR_OUT36

13.4.20 Crossbar Select Register 19 (XBAR_SEL19)

Address: 4005_5000h base + 26h offset = 4005_5026h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL39						0		SEL38					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL19 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL39	Input (XBAR_INn) to be muxed to XBAR_OUT39
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL38	Input (XBAR_INn) to be muxed to XBAR_OUT38

13.4.21 Crossbar Select Register 20 (XBAR_SEL20)

Address: 4005_5000h base + 28h offset = 4005_5028h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL41						0		SEL40					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL20 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL41	Input (XBAR_INn) to be muxed to XBAR_OUT41
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL40	Input (XBAR_INn) to be muxed to XBAR_OUT40

13.4.22 Crossbar Select Register 21 (XBAR_SEL21)

Address: 4005_5000h base + 2Ah offset = 4005_502Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL43						0		SEL42					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBAR_SEL21 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL43	Input (XBAR_INn) to be muxed to XBAR_OUT43
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL42	Input (XBAR_INn) to be muxed to XBAR_OUT42

13.4.23 Crossbar Control Register 0 (XBAR_CTRL0)

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT0 and XBAR_OUT1 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 0, the LSBs contain the control fields for XBAR_OUT0, and the MSBs contain the control fields for XBAR_OUT1.

Address: 4005_5000h base + 2Ch offset = 4005_502Ch

Bit	15	14	13	12	11	10	9	8
Read	0			STS1	EDGE1		IEN1	DEN1
Write				w1c				
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			STS0	EDGE0		IEN0	DEN0
Write				w1c				
Reset	0	0	0	0	0	0	0	0

XBAR_CTRL0 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 STS1	Edge detection status for XBAR_OUT1 This bit reflects the results of edge detection for XBAR_OUT1. This field is set to 1 when an edge consistent with the current setting of EDGE1 is detected on XBAR_OUT1. This field is cleared by writing 1 to it or by a DMA_ACK1 reception when DEN1 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT1, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT1 1 Active edge detected on XBAR_OUT1
11–10 EDGE1	Active edge for edge detection on XBAR_OUT1 This field selects which edges on XBAR_OUT1 cause STS1 to assert. 00 STS1 never asserts 01 STS1 asserts on rising edges of XBAR_OUT1 10 STS1 asserts on falling edges of XBAR_OUT1 11 STS1 asserts on rising and falling edges of XBAR_OUT1
9 IEN1	Interrupt Enable for XBAR_OUT1 This bit enables the interrupt function on the corresponding XBAR_OUT1 output. When the interrupt is enabled, the output INT_REQ1 reflects the value STS1. When the interrupt is disabled, INT_REQ1 remains low. The interrupt request is cleared by writing a 1 to STS1. **Restriction:** IEN1 and DEN1 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled

Table continues on the next page...

XBAR_CTRL0 field descriptions (continued)

Field	Description
8 DEN1	DMA Enable for XBAR_OUT1 This bit enables the DMA function on the corresponding XBAR_OUT1 output. When enabled, DMA_REQ1 presents the value STS1. When disabled, the DMA_REQ1 output remains low. **Restriction:** IEN1 and DEN1 should not both be set to 1. 0 DMA disabled 1 DMA enabled
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 STS0	Edge detection status for XBAR_OUT0 This bit reflects the results of edge detection for XBAR_OUT0. This field is set to 1 when an edge consistent with the current setting of EDGE0 is detected on XBAR_OUT0. This field is cleared by writing 1 to it or by a DMA_ACK0 reception when DEN0 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT0, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT0 1 Active edge detected on XBAR_OUT0
3–2 EDGE0	Active edge for edge detection on XBAR_OUT0 This field selects which edges on XBAR_OUT0 cause STS0 to assert. 00 STS0 never asserts 01 STS0 asserts on rising edges of XBAR_OUT0 10 STS0 asserts on falling edges of XBAR_OUT0 11 STS0 asserts on rising and falling edges of XBAR_OUT0
1 IEN0	Interrupt Enable for XBAR_OUT0 This bit enables the interrupt function on the corresponding XBAR_OUT0 output. When the interrupt is enabled, the output INT_REQ0 reflects the value STS0. When the interrupt is disabled, INT_REQ0 remains low. The interrupt request is cleared by writing a 1 to STS0. **Restriction:** IEN0 and DEN0 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
0 DEN0	DMA Enable for XBAR_OUT0 This bit enables the DMA function on the corresponding XBAR_OUT0 output. When enabled, DMA_REQ0 presents the value STS0. When disabled, the DMA_REQ0 output remains low. **Restriction:** IEN0 and DEN0 should not both be set to 1. 0 DMA disabled 1 DMA enabled

13.4.24 Crossbar Control Register 1 (XBAR_CTRL1)

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT2 and XBAR_OUT3 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 1, the LSBs contain the control fields for XBAR_OUT2, and the MSBs contain the control fields for XBAR_OUT3.

Address: 4005_5000h base + 2Eh offset = 4005_502Eh

Bit	15	14	13	12	11	10	9	8
Read	0			STS3	EDGE3		IEN3	DEN3
Write				w1c				
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			STS2	EDGE2		IEN2	DEN2
Write				w1c				
Reset	0	0	0	0	0	0	0	0

XBAR_CTRL1 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 STS3	Edge detection status for XBAR_OUT3 This bit reflects the results of edge detection for XBAR_OUT3. This field is set to 1 when an edge consistent with the current setting of EDGE3 is detected on XBAR_OUT3. This field is cleared by writing 1 to it or by a DMA_ACK3 reception when DEN3 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT3, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT3 1 Active edge detected on XBAR_OUT3
11–10 EDGE3	Active edge for edge detection on XBAR_OUT3 This field selects which edges on XBAR_OUT3 cause STS3 to assert. 00 STS3 never asserts 01 STS3 asserts on rising edges of XBAR_OUT3 10 STS3 asserts on falling edges of XBAR_OUT3 11 STS3 asserts on rising and falling edges of XBAR_OUT3
9 IEN3	Interrupt Enable for XBAR_OUT3

Table continues on the next page...

XBAR_CTRL1 field descriptions (continued)

Field	Description
	This bit enables the interrupt function on the corresponding XBAR_OUT3 output. When the interrupt is enabled, the output INT_REQ3 reflects the value STS3. When the interrupt is disabled, INT_REQ3 remains low. The interrupt request is cleared by writing a 1 to STS3. **Restriction:** IEN3 and DEN3 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
8 DEN3	DMA Enable for XBAR_OUT3 This bit enables the DMA function on the corresponding XBAR_OUT3 output. When enabled, DMA_REQ3 presents the value STS3. When disabled, the DMA_REQ3 output remains low. **Restriction:** IEN3 and DEN3 should not both be set to 1. 0 DMA disabled 1 DMA enabled
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 STS2	Edge detection status for XBAR_OUT2 This bit reflects the results of edge detection for XBAR_OUT2. This field is set to 1 when an edge consistent with the current setting of EDGE2 is detected on XBAR_OUT2. This field is cleared by writing 1 to it or by a DMA_ACK2 reception when DEN2 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT2, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT2 1 Active edge detected on XBAR_OUT2
3–2 EDGE2	Active edge for edge detection on XBAR_OUT2 This field selects which edges on XBAR_OUT2 cause STS2 to assert. 00 STS2 never asserts 01 STS2 asserts on rising edges of XBAR_OUT2 10 STS2 asserts on falling edges of XBAR_OUT2 11 STS2 asserts on rising and falling edges of XBAR_OUT2
1 IEN2	Interrupt Enable for XBAR_OUT2 This bit enables the interrupt function on the corresponding XBAR_OUT2 output. When the interrupt is enabled, the output INT_REQ2 reflects the value STS2. When the interrupt is disabled, INT_REQ2 remains low. The interrupt request is cleared by writing a 1 to STS2. **Restriction:** IEN2 and DEN2 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
0 DEN2	DMA Enable for XBAR_OUT2 This bit enables the DMA function on the corresponding XBAR_OUT2 output. When enabled, DMA_REQ2 presents the value STS2. When disabled, the DMA_REQ2 output remains low.

Table continues on the next page...

XBAR_CTRL1 field descriptions (continued)

Field	Description
	Restriction: IEN2 and DEN2 should not both be set to 1. 0 DMA disabled 1 DMA enabled

13.5 Functional Description

13.5.1 General

The XBAR module has only one mode of operation, functional mode.

13.5.2 Functional Mode

The value of each mux output is XBAR_OUT[n] = XBAR_IN[SELn]. The SELn select values are configured in the XBAR_SEL registers. All muxes share the same inputs in the same order.

A subset of XBAR_OUT[*] outputs has dedicated control fields in a Crossbar Control (XBAR_CTRL) register. Control fields provide the ability to perform edge detection on the corresponding XBAR_OUT output. Edge detection in turn can optionally be used to trigger an interrupt or DMA request. The intention is that, by detecting specified edges on signals propagating through the Crossbar, interrupts or DMA requests can be triggered to perform data transfers to or from other system components.

Control fields include an edge status field (STS), an detected edge type field (EDGE), and interrupt and DMA enable fields (DEN and IEN). STSn is set to 1 when an edge consistent with EDGEn occurs on XBAR_OUT[n]. STSn is cleared by writing 1 to it. Writing 0 as no effect. See Interrupts and DMA Requests for details on the use of STSn for DMA and interrupt request generation.

13.6 Resets

The XBAR module can be reset by only a hard reset, which forces all registers to their reset state.

13.7 Clocks

All sequential functionality is controlled by the Bus Clock.

13.8 Interrupts and DMA Requests

For each XBAR_OUT[*] output with XBAR_CTRL register support, DMA or interrupt functionality can be enabled by setting the corresponding XBAR_CTRL register bit DENn or IENn to 1. DENn and IENn should not be set to 1 at the same time for the same output XBAR_OUT[n].

Setting DENn to 1 enables DMA functionality for XBAR_OUT[n]. When DMA functionality is enabled, the output DMA_REQ[n] reflects the value of STSn. Thus the DMA request asserts when the edge specified by EDGEn is detected on XBAR_OUT[n]. Also, a rising edge on DMA_ACK[n] sets STSn to zero and thus clears the DMA request. When DEN is 0, DMA_REQ[n] is held low and DMA_ACK[n] is ignored.

Setting IENn to 1 enables interrupt functionality for XBAR_OUT[n]. When interrupt functionality is enabled, the output INT_REQ[n] reflects the value of STSn. Thus the interrupt request asserts when the edge specified by EDGEDENn is detected on XBAR_OUT[n]. The interrupt request is cleared by writing a 1 to STSn. When IENn is 0, INT_REQ[n] is held low.

DENn and IENn should not be set to 1 at the same time for the same output XBAR_OUT[n].

Chapter 14
Memory Mapped Arithmetic Unit (MMAU)

14.1 Chip-specific MMAU information

14.1.1 Overview

The Memory Mapped Arithmetic Unit (MMAU) provides acceleration to a set of math operations, including signed/unsigned multiplication and accumulation, division and root-square, and so on.

The MMAU clock source is the CPU/platform clock.

14.2 Introduction

ARM processor cores in the Cortex-M family implementing the ARMv6-M instruction set architecture do not include hardware support for many arithmetic operations like divide, square root, 64-bit multiply and fractional numeric computation. The affected processors include the Cortex-M0+ core. However, in certain deeply embedded application spaces such as metering or motor control, hardware support for this class of arithmetic operation is important to maximize system performance and minimize device power dissipation. Accordingly, the MMAU (**M**emory **M**apped **A**rithmetic **U**nit) is included in selected microcontrollers to serve as a memory-mapped co-processor located in a special address space within the system memory map accessible by the processor core or DMA.

The MMAU module supports execution of operations of divide, square root, multiply with option of accumulation. The supported numeric types of operands include unsigned integer, signed integer and signed fractional number in Q0.31 or Q0.63 format. It is also suitable for arithmetic computation of fractional numbers in Qm.n format other than Q0.31/Q0.63 with certain software interference to take care of the base changing.

14.2.1 Features

The key features of the MMAU include:

- Lightweight implementation of 32-bit/64-bit integer divide, square root and multiply arithmetic operations
  - Supports 32/32, 64/32, 64/64 divide computation on signed, unsigned integer and Q0.31 or Q0.63 fractional number
  - Supports 32x32, 64x32 multiply on signed, unsigned integer and Q0.31 or Q0.63 fractional number
  - Supports 32x32, 64x32 multiply with accumulation on signed, unsigned integer and Q0.31 or Q0.63 fractional number
  - Supports 32-bit and 64-bit square root calculation on unsigned integer and Q0.31 or Q0.63 fractional number
  - Optional to saturate the computation result when overflow occurs
- Simple programming model with seven registers
- Programming model interface optimized for activation from inline code or software library call
  - Decorated memory-mapped computation instruction launching mechanism
  - Optional DMA support to provide fast calculation launching and result fetching
  - Configurable computation status report on divide-by-zero or overflow through interrupt
  - Support lock on Supervisor only mode
- Memory-mapped co-processor based at 0xF000_4000 address space
- Pipelined design processes 2 bits per cycle with early termination for minimum execution time of divide and square root
- Fast implementation on supporting single cycle execution of multiply
- Architectural clock gating for power saving

14.2.2 Block diagram

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown in -. The MMAU module's location as a memory-mapped co-processor is highlighted.

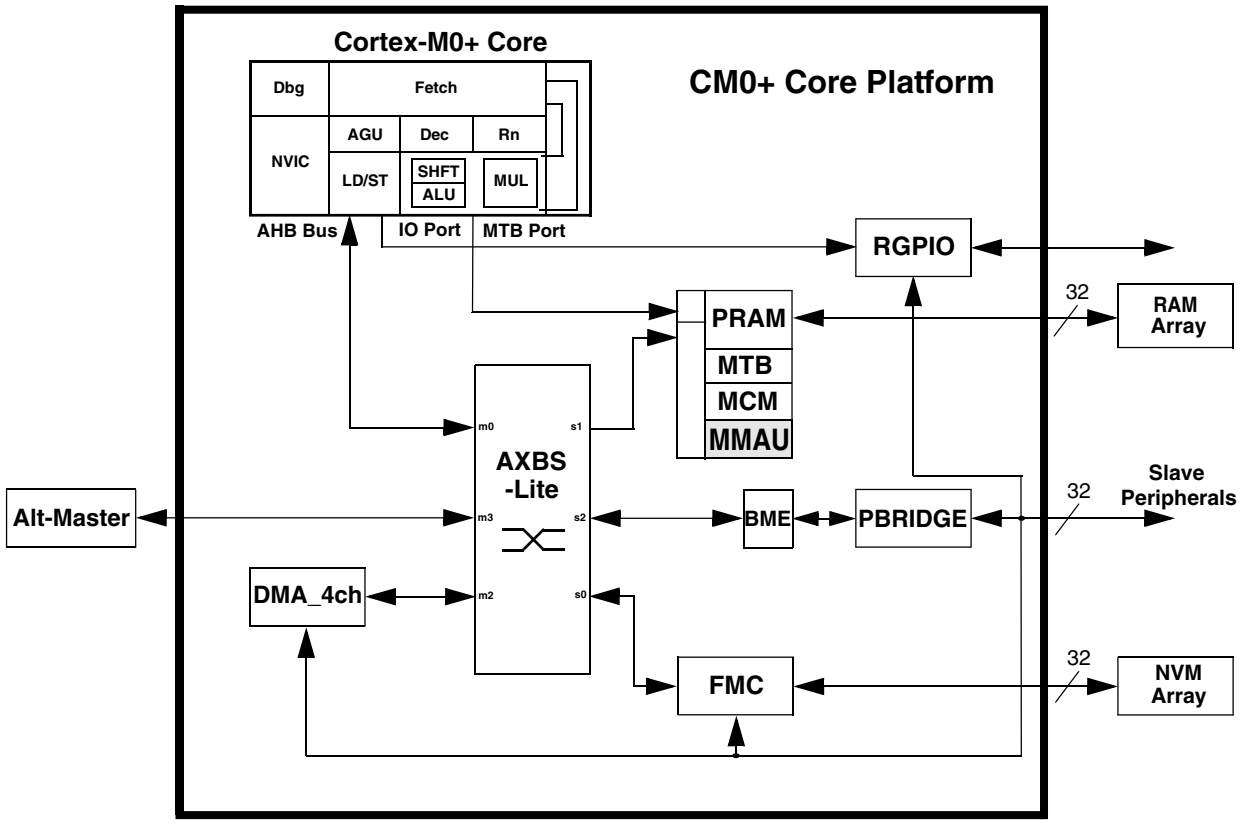


Figure 14-1. Generic Cortex-M0+ Core Platform Block Diagram

Next, a block diagram of the internal structure of the MMAU module is presented. See -.

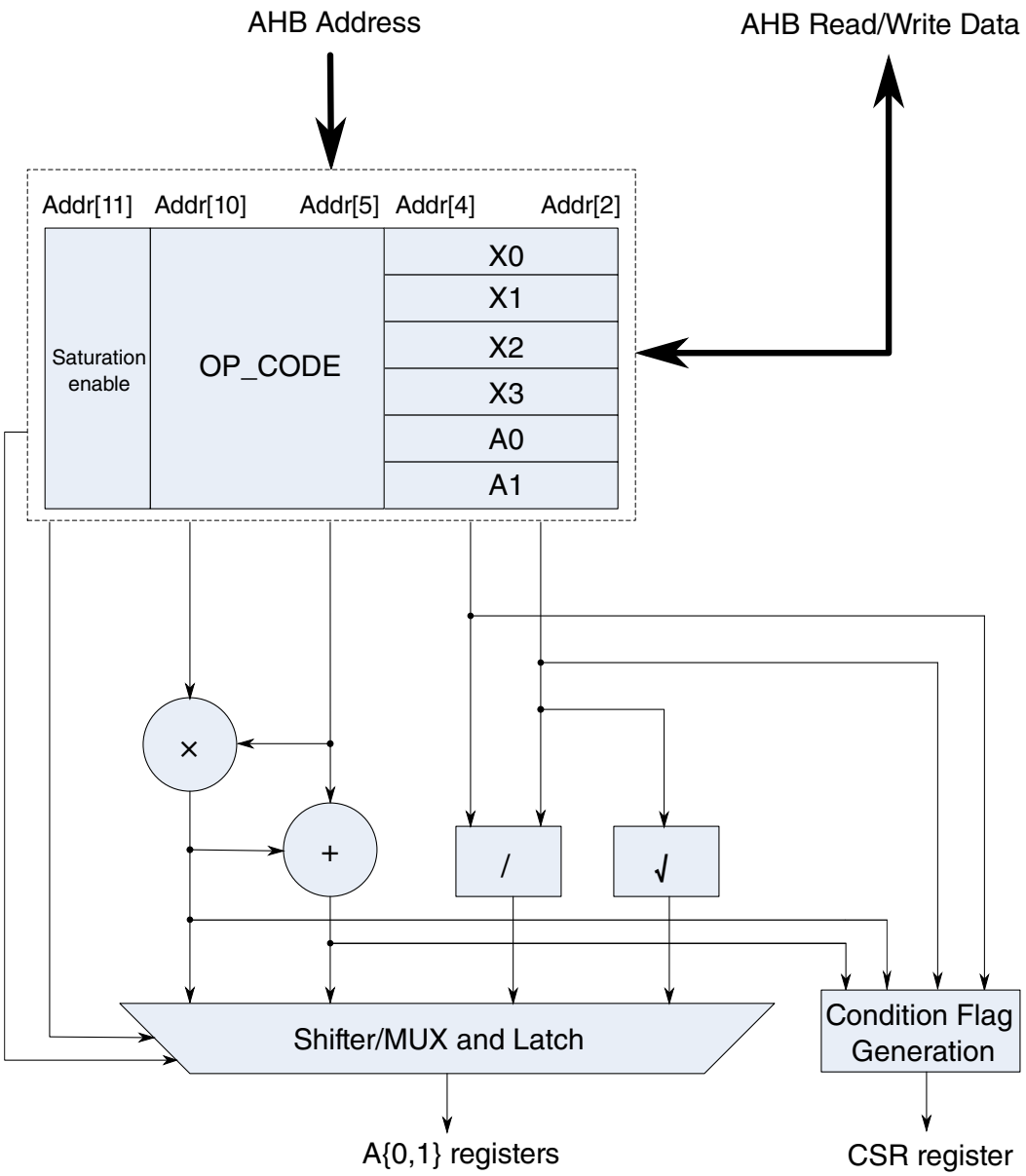


Figure 14-2. MMAU Block Diagram

14.2.3 Modes of operation

The MMAU module does not support any special modes of operation. As a memory-mapped device located on a crossbar slave AHB system bus port, MMAU responds based strictly on memory addresses to its programming model.

All functionality associated with the MMAU module resides in the core platform's clock domain; this includes its connections with the crossbar slave port. To minimize power dissipation, the design supports an architectural clock gate for the entire module, that is, the MMAU is only clocked when responding to bus requests to its programming model or is busy performing a calculation.

14.3 External signal description

The MMAU module does not directly support any external interfaces.

The internal interface includes a standard 32-bit AHB bus as shown in -.

14.4 Memory map and register definition

The MMAU module supports seven program-visible 32-bit registers including four for passing input operands, two accumulator and result registers for accumulation operation or retrieving the output results, and one configuration/status register. The four input operands registers are operand registers X0,X1,X2 & X3. The accumulation & result registers are A0, A1. The Configuration/Status register is specified as CSR.

The seven registers occupy lower 28 bytes of a standard 4 KB address IP slot. The MMAU programming model makes use of the higher address bits to define different arithmetic instructions of operation, which is called the decorated memory-mapped instruction launching. In the MMAU definition, the "decoration", that is, the additional semantic information such as instruction op_code, is encoded into the MMAU register's address to provide operation information to MMAU when access the registers. That means the seven registers are all mapped to multiple locations inside of MMAU memory map. It is for software easy of use this co-processor with minimum code or execution time required.

MMAU is an AHB slave with 4KB address space. When access the MMAU registers, Address bit[4:2] will choose which register is accessing from the seven registers, with the increasing address order from X0-3, A0-1 and CSR. Address bit[11:5] is the encoded "decoration code" provides operation type (op_code) for the instructions. Specifically the Address bit[11] defines if the saturation is enabled when there is an overflow in computation. When saturation is enabled, if the computation get an overflow result, the saturated result will be stored in A0 and A1. For instruction operation where overflow is not possible to happen, the saturation configuration (Address[11]) is ignored.

For operations required operands input, the instruction encoded in decoration code (Address bit[11:5]) when writing to X3 register will be started to execute immediately. For the operations that do not require operands from X0-3, such as square root calculation on A1-0, the read access to result register (which could be A0 or A1 depends on the numeric type returned) with its decodration code of Address bit[11:5] will trigger the start of the calculation and return the result.

For the list of supported instructions and their corresponding decoration code of address bit[11:5], please refer to Table 14-2 for detail information. Address[11] in the table is for saturation configuration for the instruction. The Hex code of the base address for each instruction is provided in this table for quick reference. For the address that is not listed in this table is reserved for future use (RFU). Any access to RFU regions will be terminated with an error by MMAU.

MMAU can only be accessed via word-sized (32 bit) accesses. Attempted accesses using smaller data sizes, or to reserved space are terminated with an error.

MMAU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_4000	Operand Register X0 (MMAU_X0)	32	R/W	0000_0000h	14.4.1/226
F000_4004	Operand Register X1 (MMAU_X1)	32	R/W	0000_0000h	14.4.2/227
F000_4008	Operand Register X2 (MMAU_X2)	32	R/W	0000_0000h	14.4.3/227
F000_400C	Operand Register X3 (MMAU_X3)	32	R/W	0000_0000h	14.4.4/228
F000_4010	Accumulator Register A0 (MMAU_A0)	32	R/W	0000_0000h	14.4.5/229
F000_4014	Accumulator Register A1 (MMAU_A1)	32	R/W	0000_0000h	14.4.6/230
F000_4018	Control/Status Register (MMAU_CSR)	32	R/W	0000_0000h	14.4.7/231
F000_401C	CSR Interrupt Flags Clearance Register (MMAU_CSR_IF_CLR)	32	W	0000_0000h	14.4.8/234

14.4.1 Operand Register X0 (MMAU_X0)

This register is loaded with the input X0 operand before a calculation operation is initiated. This register can be accessed (read/write) even when MMAU is busy with the current instruction execution. This is to allow pipeline in setting up next calculation operand. When access to this register, the MMAU address[4:0] need to be 0x00 and the MMAU decoration code bits (address bit[11:5]) are ignored.

Address: F000_4000h base + 0h offset = F000_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	X0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_X0 field descriptions

Field	Description
X0	Operand Register X0 This is the input X0 operand for the programmed arithmetic calculations.

14.4.2 Operand Register X1 (MMAU_X1)

This register is loaded with the input X1 operand before a calculation operation is initiated. This register can be accessed (read/write) even when MMAU is busy with the current instruction execution. This is to allow pipeline in setting up next calculation operand. When access to this register, the MMAU address[4:0] need to be 0x04 and the MMAU decoration code bits(address bit[11:5]) are ignored.

Address: F000_4000h base + 4h offset = F000_4004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	X1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_X1 field descriptions

Field	Description
X1	Operand Register X1 This is the input X1 operand for the programmed arithmetic calculations.

14.4.3 Operand Register X2 (MMAU_X2)

This register is loaded with the input X2 operand before a calculation operation is initiated. This register can be accessed (read/write) even when MMAU is busy with the current instruction execution. This is to allow pipeline in setting up next calculation operand. When access to this register, the MMAU address[4:0] need to be 0x08 and the MMAU decoration code bits (address bit[11:5]) are ignored.

Address: F000_4000h base + 8h offset = F000_4008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	X2																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_X2 field descriptions

Field	Description
X2	Operand Register X2 This is the input X2 operand for the programmed arithmetic calculations.

14.4.4 Operand Register X3 (MMAU_X3)

This register is loaded with the input X3 operand together with the instruction to be activated. When write access to X3, the MMAU Address[11:5] defines the decoration code, which is instruction type and the saturation configuration, and address[4:0] need to be 0x0C. Writing to Operand Register X3 will start the execution of the operation defined by the decoration code immediately. For multiply, the calculation completes in single clock cycle, and for divide and square root, the MMAU will be busy until the calculation completes. Any memory write access of the X3 register while the module is busy causes the access to be stalled (using wait states) until the calculation completes. Read access of this register returns the value immediately.

Address: F000_4000h base + Ch offset = F000_400Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	X3																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_X3 field descriptions

Field	Description
X3	Operand Register X3 This is the input X3 operand for the programmed arithmetic calculations.

14.4.5 Accumulator Register A0 (MMAU_A0)

This register is loaded with the lower 32-bit result of a calculation operation from divide, square root or multiply, or multiply with accumulation. It also serve as the lower 32-bit of the 64-bit operand for certain type of instruction. Any access (read or write) of this register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes. Together with A1, it provides a 64-bit result or operand. When access to this register, the MMAU address[4:0] need to be 0x10 and the MMAU decoration code bits (address bit[11:5]) are ignored if they are not matching QSQRDA. For unsigned square root calculation on A10 where there is no need input operands from X0-3, reading to A0 with decoration code Address[11:5] to be USQRDA will launching the calculation immediately and the read will be pending until the computation completes with the result returned.

Address: F000_4000h base + 10h offset = F000_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_A0 field descriptions

Field	Description
A0	Accumulator Register A0 This register holds the lower 32-bit of a 64-bit result from a calculation operation. It also serve as one operand in certain calculation.

14.4.6 Accumulator Register A1 (MMAU_A1)

This register is loaded with the higher 32-bit result of a calculation operation from divide, square root or multiply, or multiply with accumulation. It also serve as the higher 32-bit of the 64-bit operand for certain type of instruction. Any access (read or write) of this register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes. Together with A0, it provides a 64-bit result or operand in the operation involved 64-bit. When access to this register, the MMAU address[4:0] need to be 0x14 and the MMAU decoration code bits (address bit[11:5]) are ignored if they are not matching to QSQRDA. For fractional numeric square root calculation on A10 where there is no need input operands from X0-3, reading to A1 with decoration code Address[11:5] to be QSQRDA will launching the calculation immediately and the read will be pending until the computation completes with the result returned.

Address: F000_4000h base + 14h offset = F000_4014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

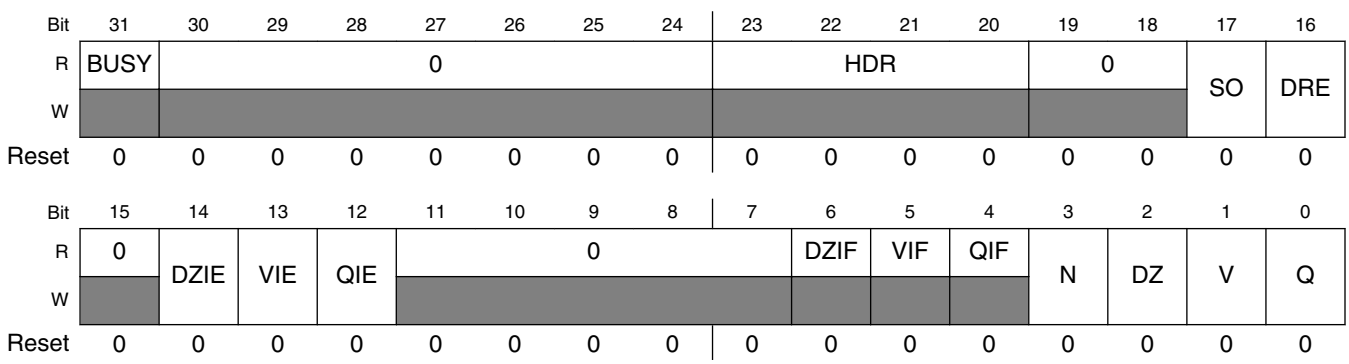
MMAU_A1 field descriptions

Field	Description
A1	Accumulator Register A1 This register holds the higher 32-bit of a 64-bit result from a calculation operation. It also serve as one operand in certain calculation.

14.4.7 Control/Status Register (MMAU_CSR)

This register defines the operating configuration and the computation result status of MMAU. It includes the busy status indicators, computation result status on overflow for multiply, divide and accumulation, error status flag for divide by zero. It has the Interrupt Flag bits of the overflow, divide by zero bits too. It also include supervisor lock bit, the MMAU can only be access in Supervisor mode when set. It has DMA enable, interrupt enable control bits too. A memory write access of the CSR register while the MMAU is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes, read to CSR return the register value immediately even when MMAU is in busy. When access to this register, the MMAU address[4:0] need to be 0x8C and the MMAU decoration code bits (address bit[11:5]) are ignored. The Interrupt Flag bits are read-only from the address offset of 0x18, MMAU provides another register entry (CSR_IF_CLR) of address[4:0]=0x1C for clearing CSR status and the Interrupt flag bits, where the interrupt flag bits can be cleared by W1C (write one to clear).

Address: F000_4000h base + 18h offset = F000_4018h



MMAU_CSR field descriptions

Field	Description
31 BUSY	BUSY This read-only bit is asserted when the MMAU is performing a divide or square root. Multiply operation completes in single cycle. When an operation is initiated, the hardware sets this flag. It remains asserted until the operation completes and the hardware automatically clears the indicator. This bit can be used to poll the MMAU's execution status. 0 MMAU is idle 1 MMAU is busy performing a divide or square root calculation
30–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–20 HDR	Hardware Revision Level This HDR field is read-only and tells the Hardware Revision Level of MMAU, it will changes with different version of MMAU.

Table continues on the next page...

MMAU_CSR field descriptions (continued)

Field	Description
	0000 Current Hardware Revision Level is 0000
19–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SO	Supervisor-Only This is the Supervisor Only bit. When this bit is set, CPU/DMA can only access MMAU in supervisor mode; when it is access through user mode, MMAU will terminate the access with an bus error. This bit can only be changed by CPU in supervisor mode, write access in User Mode will be ignored by MMAU. 0 MMAU registers can be access in both User Mode or Supervisor Mode 1 MMAU registers can only be access in Supervisor Mode
16 DRE	DMA Request Enable MMAU provide interface for DMA to configure the MMAU and launch the calculation. When MMAU completes the execution of instruction, it will be in IDLE state (not busy). It can be configured optional to generate the DMA request so that user can use DMA to fetch the result and program the new computation instruction. 0 MMAU will not generate DMA request when in IDLE (not busy) state 1 MMAU will generate DMA request when in IDLE (not busy) state
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DZIE	Divide-by-Zero Interrupt Enable This indicator configures the MMAU's interrupt when a divide-by-zero calculations is detected. If both CSR[DZIF] and CSR[DZE] are set, MMAU will generate an interrupt to CPU. 0 MMAU will not generate interrupt even CSR[DZIF]=1 1 If CSR[DZIF] = 1, MMAU will generate an interrupt to signal a divide-by-zero.
13 VIE	Divide/Multiply Overflow (V flag) Interrupt Enable This control bit is used to enable/disable VIF flag interrupt. 0 V flag (CSR[VIF]) set will not generate interrupt. 1 V flag (CSR[VIF]) set will generate interrupt to inform a divide or multiply overflow
12 QIE	Accumulation Overflow (Q flag) Interrupt Enable This control bit is used to enable/disable QIF flag interrupt. 0 Q flag (CSR[QIF]) set will not generate interrupt. 1 Q flag (CSR[QIF]) set will generate interrupt to inform an accumulation overflow
11–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DZIF	DZ Interrupt Flag: Divide by Zero The DZIF is the Divide by Zero Interrupt Flag bit, and it is the sticky version of CSR[DZ]. It can only be set by MMAU execution of instruction. Once it is set, it can only be cleared by user software code through writing "1" to this bit from register entry of CSR_IF_CLR at address offset 0x1C. When this bit is set, and if the Divide by Zero Interrupt is enable by setting CSR[DZIE], MMAU will issue interrupt to CPU.

Table continues on the next page...

MMAU_CSR field descriptions (continued)

Field	Description
	0 For divide, the divisor is not zero, or the calculation is not divide 1 For divide, the divisor is zero, which is a divide-by-zero error
5 VIF	V Interrupt Flag: Multiply or Divide overflow The VIF is the Multiply or Divide Overflow Interrupt Flag bit, and it is the sticky version of CSR[V]. It can only be set by MMAU execution of instruction. Once it is set, it can only be cleared by user software code through writing "1" to this bit from register entry of CSR_IF_CLR at address offset 0x1C. When this bit is set, and if the Multiply or Divide Overflow Interrupt is enable by setting CSR[VIE], MMAU will issue interrupt to CPU. 0 Calculation is not include divider or multiply, or the product/quotient does not overflow 1 Product in multiply or multiply with accumulation, or quotient of a divide overflows
4 QIF	Q Interrupt Flag: Accumulation Overflow Interrupt Status The QIF is the Accumulation Overflow Interrupt Flag bit, and it is the sticky version of CSR[Q]. It can only be set by MMAU execution of instruction. Once it is set, it can only be cleared by user software through writing "1" to this bit from register entry of CSR_IF_CLR at address offset 0x1C. When this bit is set, and if the accumulation interrupt is enable by setting CSR[QIE], MMAU will issue interrupt to CPU. 0 No accumulation operation or accumulation operation does not overflow 1 Accumulation overflows during a MAC instruction
3 N	N flag: Signed calculation result is negative For signed arithmetic calculation, if the raw computation result is negative (before overflows or saturation), this N flag bit will be set. It is updated on each calculation. It can be set/cleared by software code too. 0 Calculation raw result is zero or positive, or unsigned number 1 Calculation raw result is negative
2 DZ	DZ flag: Divide by Zero This DZ flag indicates that the divisor is zero for a divide operation when set. It is updated on each calculation. It can be set/cleared by user software code too. 0 For divide, the divisor is not zero, or the calculation is not divide 1 For divide, the divisor is zero, which is a divide-by-zero error
1 V	V flag: Multiply or Divide overflow For divide or multiply operation, if the divide quotient or multiply product computation overflows, this V flag bit will be set. It is updated on each calculation. It can be set/cleared by user software code too. 0 Calculation is not include divider or multiply, or the product/quotient does not overflow 1 Product in multiply or multiply with accumulation, or quotient of a divide overflows
0 Q	Q flag: Accumulation Overflow For computation of multiply with accumulation, this bit is set when the accumulation computation results overflow. It is updated on each calculation. It can be set/cleared by user software code too. 0 No accumulation operation or accumulation operation does not overflow 1 Accumulation overflows during a MAC instruction

14.4.8 CSR Interrupt Flags Clearance Register (MMAU_CSR_IF_CLR)

This write-only register provides the clearance entry address for the interrupt flag bits in CSR. CSR[QIF], CSR[VIF] & CSR[DZIF] can only be cleared by CSR_IF_CLR register entry through W1C (write one to clear). CSR[Q,V,DZ] can be cleared by writing '0' or set by writing '1' to corresponding bit. Read to CSR_IF_CLR returns zero. When access to this register, the MMAU address[4:0] need to be 0x1C and the MMAU decoration code bits (address bit[11:5]) are ignored. A memory write access to the CSR register through CSR_IF_CLR address while the MMAU is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes.

Address: F000_4000h base + 1Ch offset = F000_401Ch

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W										DZIF	VIF	QIF	N	DZ	V	Q
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMAU_CSR_IF_CLR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DZIF	DZ Interrupt Flag: Divide by Zero Write One to clear CSR[DZIF]. 0 Write "0" is ignored 1 Write "1" to clear CSR[DZIF]
5 VIF	V Interrupt Flag: Multiply or Divide overflow Write One to clear CSR[VIF]. 0 Write "0" is ignored 1 Write "1" to clear CSR[VIF]
4 QIF	Q Interrupt Flag: Accumulation Overflow Interrupt Status Write One to clear CSR[QIF]. 0 Write "0" is ignored 1 Write "1" to clear CSR[QIF]

Table continues on the next page...

MMAU_CSR_IF_CLR field descriptions (continued)

Field	Description
3 N	N flag: Signed calculation result is negative Write zero to clear CSR[N] and write one to set CSR[N]. 0 Write "0" to clear CSR[N] 1 Write "1" to set CSR[N]
2 DZ	DZ flag: Divide by Zero Write zero to clear CSR[DZ] and write one to set CSR[DZ]. 0 Write "0" will clear CSR[DZ] 1 Write "1" will set CSR[DZ]
1 V	V flag: Multiply or Divide overflow Write zero to clear CSR[V] and write one to set CSR[V]. 0 Write "0" will clear CSR[V] 1 Write "1" will set CSR[V]
0 Q	Q flag: Accumulation Overflow Write zero to clear CSR[Q] and write one to set CSR[Q]. 0 Write "0" will clear CSR[Q] 1 Write "1" will set CSR[Q]

14.4.9 MMAU register access in Busy State

When MMAU is executing the divide or square root computation, MMAU will be in the busy state with CSR[BUSY] set until the computation is finished and the result is stored into A10 registers. When MMAU is busy, A0, A1 read/write access, X3 or CSR write access will be stalled until the MMAU completes the on-going computation. The read/write to X0-2, read to X3 and CSR will not be stalled in any case. Table 14-1 summarised the register access status when MMAU is busy. For the cycles of wait when the access is stalled, please refer to Execution times.

Table 14-1. Summary of Register access status in Busy

Register Access	Access Status when MMAU in BUSY	
	Read	Write
X0	Not Stalled	Not Stalled
X1	Not Stalled	Not Stalled
X2	Not Stalled	Not Stalled
X3	Not Stalled	Stalled
A0	Stalled	Stalled

Table continues on the next page...

Table 14-1. Summary of Register access status in Busy (continued)

Register Access	Access Status when MMAU in BUSY	
	Read	Write
A1	Stalled	Stalled
CSR	Not Stalled	Stalled

14.5 Functional description

This section describes MMAU programming model, numeric types, arithmetic computation, the execution times and the software interface in detail.

14.5.1 MMAU Programming Model

The programming model of the MMAU is organized to be similar to the input arguments passed to software libraries for arithmetic operation.

The MMAU supports four types of arithmetic calculation: Divide, Square Root, Multiply, Multiply with Accumulation. Below is the Mnemonic used for describing the arithmetic instruction:

- DIV: Divide
- SQR: Square Root
- MUL: Multiply
- MAC: Multiply with Accumulation

There are three numeric types to be supported for the arithmetic computation: unsigned integer, signed integer and signed fractional number in Q0.31 or Q0.63 format. Single letter {U,S,Q} prefixes are used in the instruction mnemonic to specify the numeric type of the instruction :

- U: Prefix for Unsigned instruction, for example UMUL means unsigned Multiply
- S: Prefix for Signed instruction, for example SMUL means signed Multiply
- Q: Prefix for Fractional instruction, for example QMUL mean fractional Multiply

The MMAU is a 32-bit module but supports many 64-bit arithmetic computations. The registers and memory-map are in Little-Endian mode. Below are the mnemonics used in instruction description to specify the 64-bit operand for the arithmetic computation:

- X10: 64-bit data of {X1,X0}
- X21: 64-bit data of {X2,X1}

- X32: 64-bit data of {X3,X2}
- A10: 64-bit data of {A1,A0}

In the instruction descriptions, suffix of {D, DA} are used to indicate that the instruction operand bit width to be 64-bit (Double Word). By default without any suffix, the computation is on 32-bit operand. While 64-bit operand do have the suffix of "D" for operand register from Xn-Xn-1, "DA" means 64-bit operand register from A10. Some of the combination of the suffix used in instruction mnemonics are listed:

- D: first operand is 64bit from X32 or X21, second operand is 32bit from X3 (if there is a second operand)
- DD: both operand are 64bit from X32, X10.
- DA: the first operand is 64-bit and from A10, the second operand is 32bit
- DDA: the first operand is 64-bit and from A10, the second operand is 64-bit from Xn:n-1 registers

Table 14-2 lists all the MMAU instructions with its decorated address encoding. All the address that is not listed in this table is Reserved for Future Usage(RFU), MMAU will terminate the access with bus error if there is any access attempts to RFU regions.

Table 14-2. MMAU Decorated Memory-Map for Instruction Set

Instruction Numeric Type	Instruction Decoration Code[1] Address[11:2]							Reg Addr[2]			MMAU Instruction Base Address[11:0] (HEX)		Instruction Mnemonic	Operation Brief Description
	11	10	9	8	7	6	5	4	3	2	S[3]=0	S[3]=1		
NA	0	0	0	0	0	0	0	-	-	-	0x000	-	REG_RW[4]	Register RW
Unsigned Integer	-	0	0	0	0	0	1	-	-	-	0x020	-	UMUL	A10=X2*X3
	S	0	0	0	0	1	0	-	-	-	0x040	0x840	UMULD	A10=X21*X3
	S	0	0	0	0	1	1	-	-	-	0x060	0x860	UMULDA	A10=A10*X3
	S	0	0	0	1	0	1	-	-	-	0x0A0	0x8A0	UMAC	A10=X2*X3+A10
	S	0	0	0	1	1	0	-	-	-	0x0C0	0x8C0	UMACD	A10=X21*X3+A10
	S	0	0	0	1	1	1	-	-	-	0x0E0	0x8E0	UMACDA	A10=A10*X3+X21
	-	0	0	1	0	0	1	-	-	-	0x120	-	UDIV	A10=X2/X3
	-	0	0	1	0	1	0	-	-	-	0x140	-	UDIVD	A10=X21/X3
	-	0	0	1	0	1	1	-	-	-	0x160	-	UDIVDA	A10=A10/X3
	-	0	0	1	1	0	0	-	-	-	0x180	-	UDIVDD	A10=X10/X32
	-	0	0	1	1	0	1	-	-	-	0x1A0	-	UDIVDDA	A10=A10/X32
	-	0	1	0	0	0	1	-	-	-	0x220	-	USQR	A10=√(X3)
	-	0	1	0	0	1	0	-	-	-	0x240	-	USQRD	A10=√(X32)
	-	0	1	0	0	1	1	1	0	0	0x270[5]	-	USQRDA	A10=√(A10)
Fractional Number	-	0	1	0	1	0	1	-	-	-	0x2A0	-	QSQR	A10=√(X3)
	-	0	1	0	1	1	0	-	-	-	0x2C0	-	QSQRD	A10=√(X32)
	-	0	1	0	1	1	1	1	0	1	0x2F4[6]	-	QSQRDA	A10=√(A10)

Table continues on the next page...

Table 14-2. MMAU Decorated Memory-Map for Instruction Set (continued)

Instruction Numeric Type	Instruction Decoration Code[1]							Reg Addr[2]			MMAU Instruction Base Address[11:0] (HEX)		Instruction Mnemonic	Operation Brief Description
	Address[11:2]													
	11	10	9	8	7	6	5	4	3	2	S[3] =0	S[3] =1		
	S	0	1	1	0	0	1	-	-	-	0x320	0xB20	QDIV	A10=X2/X3
	S	0	1	1	0	1	0	-	-	-	0x340	0xB40	QDIVD	A10=X21/X3
	S	0	1	1	0	1	1	-	-	-	0x360	0xB60	QDIVDA	A10=A10/X3
	S	1	0	0	0	0	1	-	-	-	0x420	0xC20	QMUL	A10=X2*X3
	S	1	0	0	0	1	0	-	-	-	0x440	0xC40	QMULD	A10=X21*X3
	S	1	0	0	0	1	1	-	-	-	0x460	0xC60	QMULDA	A10=A10*X3
	S	1	0	0	1	0	1	-	-	-	0x4A0	0xCA0	QMAC	A10=X2*X3+A10
	S	1	0	0	1	1	0	-	-	-	0x4C0	0xCC0	QMACD	A10=X21*X3+A10
	S	1	0	0	1	1	1	-	-	-	0x4E0	0xCE0	QMACDA	A10=A10*X3+X21
Signed Integer	-	1	1	0	0	0	1	-	-	-	0x620	0xE20	SMUL	A10=X2*X3
	S	1	1	0	0	1	0	-	-	-	0x640	0xE40	SMULD	A10=X21*X3
	S	1	1	0	0	1	1	-	-	-	0x660	0xE60	SMULDA	A10=A10*X3
	S	1	1	0	1	0	1	-	-	-	0x6A0	0xEA0	SMAC	A10=X2*X3+A10
	S	1	1	0	1	1	0	-	-	-	0x6C0	0xEC0	SMACD	A10=X21*X3+A10
	S	1	1	0	1	1	1	-	-	-	0x6E0	0xEE0	SMACDA	A10=A10*X3+X21
	S	1	1	1	0	0	1	-	-	-	0x720	0xF20	SDIV	A10=X2/X3
	S	1	1	1	0	1	0	-	-	-	0x740	0xF40	SDIVD	A10=X21/X3
	S	1	1	1	0	1	1	-	-	-	0x760	0xF60	SDIVDA	A10=A10/X3
	S	1	1	1	1	0	0	-	-	-	0x780	0xF80	SDIVDD	A10=X10/X32
	S	1	1	1	1	0	1	-	-	-	0x7A0	0xFA0	SDIVDDA	A10=A10/X32

1. Decoration Code defines which operation to be performed by MMAU. For instructions other than QSRQDA or USQRDA, the writing to X3 with the decoration code will trigger the execution of the appropriate instruction immediately.
2. Address[4:2] will select which 32bit register in MMAU to be accessed.
3. S bit is the Saturation Enable bit, which is Address[11] in the decoration code.
4. The REG_RW instruction is provided to access the MMAU registers without triggering any computation operation.
5. USQRDA is triggered by reading to A0 with the decoration code (Address[11:5]) matching USQRDA, which makes the MMAU offset address to be 0x270. The reading will be stalled until the computation completes, and the result will be returned to the read access.
6. QSQRDA is triggered by reading to A1 with the decoration code (Address[11:5]) matching QSQRDA, which makes the MMAU offset address to be 0x2F4. The reading will be stalled until the computation completes, and the result will be returned to the read access.

As mentioned in Memory map and register definition and listed in above table, MMAU uses decorated memory-map for triggering the execution of a certain instruction. When accessing MMAU registers, the address bit[4:2] determine which register from the seven registers is accessing. For a write access to X3, the decoration code (address[11:5]) determine which instruction it will start. The address bit[11] will enable/disable the

saturation on the calculation. The address bit[11] (Saturation Enable) is not taking effect for the arithmetic calculation where the computation overflow is not possible to happen, and in such case it is ignored by MMAU.

The MMAU decorated memory-map for arithmetic instructions is designed for software to pass operand argument to MMAU and start the calculation in a simple and fast way. Making use of these advantages, user can develop the optimized inline arithmetic library for MMAU. Take "UDIVDD" as an example. This instruction performs a double word divide operation with both operands are in 64-bit, where X10 as dividend and X32 as divisor. User need to pass 4 32-bit operand register to MMAU to X0-3. The result is also 64-bit and returned in A10, user have to access A1-0 to fetch the quotient A10. Since X0-3 and A0-1 are all in adjacent spaces, it makes it's possible to using STM to pass the argument to MMAU, and LDM to fetch the result from A10, with the base address of the instruction of "UDIVDD". For this case, base decorated address for "UDIVDD" is "0x180". The pseudo ASM code for this can be:

```
ldr R0, =UDIVDD_BASE ; Load the decorated base address for UDIVDD (MMAU_Base_Addr+0x180)

stmia R0, {R1-R4} ; store the operands to X0-X3.

                  ; When write X3 happens, it will trigger UDIVDD execution

ldmia R0, {R5, R6} ; get the result from A10, since R0 now point to A0
```

STM/LDM are interruptable-restartable instructions in ARM Cortex-M architecture. It is suggested to use STM to pass the argument to MMAU and LDM to fetch the result from A10.

14.5.2 Numeric Types in MMAU

MMAU supports arithmetic instructions on three numeric types: Unsigned Integer, Signed Integer and Fractional number in Signed Q0.31/Q0.63 format.

In MMAU Unsigned Instructions, all the numbers are treated as unsigned integer.

In MMAU Signed Instructions, all the numbers are treated as signed integer. The signed integer is represent as two's complement number with the MSB bit as sign bit.

In MMAU fractional instructions, all the number are treated as signed Q0.31 format if it is in 32-bit, or Q0.63 format if it is in 64bit.

14.5.2.1 Fractional Number in Q format

Q format is the general form of fixed point fractional number. The following description is taken from http://en.wikipedia.org/wiki/Q_(number_format).

Q is a fixed point number format where the number of fractional bits (and optionally the number of integer bits) is specified. For example, a Q15 number has 15 fractional bits; a Q1.14 number has 1 integer bit and 14 fractional bits. Q format is often used in hardware that does not have a floating-point unit and in applications that require constant resolution.

Q format numbers are (notionally) fixed point numbers (but not actually a number itself); that is, they are stored and operated upon as regular binary numbers (i.e. signed integers), thus allowing standard integer hardware/ALU to perform rational number calculations. The number of integer bits, fractional bits and the underlying word size are to be chosen by the programmer on an application-specific basis - the programmer's choices of the foregoing will depend on the range and resolution needed for the numbers. The machine itself remains oblivious to the notional fixed point representation being employed - it merely performs integer arithmetic the way it knows how. Ensuring that the computational results are valid in the Q format representation is the responsibility of the programmer.

The Q notation is written as Qm.n, where:

- *Q designates that the number is in the Q format notation - the Texas Instruments representation for signed fixed-point numbers (the "Q" being reminiscent of the standard symbol for the set of rational numbers).*
- *m is the number of bits set aside to designate the two's complement integer portion of the number, exclusive of the sign bit (therefore if m is not specified it is taken as zero).*
- *n is the number of bits used to designate the fractional portion of the number, i.e. the number of bits to the right of the binary point. (If n = 0, the Q numbers are integers - the degenerate case).*

Note that the most significant bit is always designated as the sign bit (the number is stored as a two's complement number) in order to allow standard arithmetic-logic hardware to manipulate Q numbers. Representing a signed fixed-point data type in Q format therefore always requires m+n+1 bits to account for the sign bit. Hence the smallest machine word size required to accommodate a Qm.n number is m+n+1, with the Q number left justified in the machine word.

For a given Qm.n format, using an m+n+1 bit signed integer container with n fractional bits:

- *its range is $[-2^m, 2^m - 2^{-n}]$*
- *its resolution is 2^{-n}*

MMAU's fractional arithmetic instructions support the fractional number in signed Q0.31 and Q0.63 format. So the range for them are:

For signed Q0.31:

- *its range is [-1, 1 - 2^{-31}]*
- *its resolution is 2^{-31}*

For signed Q0.63:

- *its range is [-1, 1 - 2^{-63}]*
- *its resolution is 2^{-63}*

In MMAU fractional arithmetic instructions, all the 32-bit fractional numbers are in signed Q0.31 format and all 64-bit fractional numbers are in signed Q0.63 format. The computational result is Q0.31 in 32-bit or Q0.63 in 64-bit.

For Qm.n format numbers other than Q0.31 or Q0.63, the MMAU can still be used for the calculation through the signed or unsigned arithmetic instructions. Since the divide, multiply, square root calculation on Q format numbers always cause the base of the Q notation to change, ensuring that the computational results are valid in the Q format representation is the responsibility of the programmer. Actually, with the use of Qm.n in signed/unsigned instructions, the MMAU provides a powerful extension for a wide range of fractional numeric computations with the fixed-point integer processing hardware. Please refer to Square root using Q notation for some examples on square root calculation with the general Qm.n format numbers.

14.5.2.1.1 Square root using Q notation

To improve the accuracy of unsigned integer square root, users are encouraged to use Q notation for square root calculations returning fractional values.

For the unsigned integer format used in the MMAU's square root calculation, an u(nsigned)Qm.n notation requires m+n bits (m+n = 32) for the input operand (RCND). An uQm.n format produces an uQ(m/2).(n/2) square root. As examples, consider the following tables involving the square root of 2 and square root of “pi” calculations. As expected, as the number of fractional bits (n) increases, the error between the calculated square root and the “actual” result decreases.

Table 14-3. Square Root of 2 Calculations (√2 = 1.4142135623)

RCND [Hex]	RCND Q format	Results [Hex]	RES Q Format	Decimal	% Error
0x0000_0002	uQ32.00	0x0000_0001	uQ16.00	1.0	-29.289%
0x0002_0000	uQ16.16	0x0000_016A	uQ08.08	1.4140625	-0.011%
0x0200_0000	uQ08.24	0x0000_16A0	uQ04.12	1.4140625	-0.011%

Table continues on the next page...

Table 14-3. Square Root of 2 Calculations (√2 = 1.4142135623) (continued)

RCND [Hex]	RCND Q format	Results [Hex]	RES Q Format	Decimal	% Error
0x2000_0000	uQ04.28	0x0000_5A82	uQ02.14	1.4141845703	-0.002%
0x8000_0000	uQ02.30	0x0000_B504	uQ01.15	1.4141845703	-0.002%

Table 14-4. Square Root of Pi Calculations (√Pi = 1.7724538509)

RCND [Hex]	RCND Q format	Results [Hex]	RES Q Format	Decimal	% Error
0x0000_0003	uQ32.0	0x0000_0001	uQ16.00	1.0	-43.581%
0x0003_243F	uQ16.16	0x0000_01C5	uQ08.08	1.76953125	-0.165%
0x0324_3F6A	uQ08.24	0x0000_1C5B	uQ04.12	1.772216769	-0.013%
0x3243_F6A8	uQ04.28	0x0000_716F	uQ02.14	1.7723999023	-0.003%
0xC90F_DAA0	uQ02.30	0x0000_E2DF	uQ01.15	1.7724304199	-0.001%

The application of the Q notation does not limited to integer square root calculations. It can actually be made use of by all the integer and multiply instructions with user taking care of the notation changes caused by the computation to get a valid result. In such way, the MMAU provides a powerful extension for a wide range of fractional numeric computations with the fixed-point integer processing hardware.

14.5.3 MMAU Arithmetic Computation Description

There are four types of arithmetic computation supported by MMAU, which are: divide, square root, multiply and multiply with accumulation. This section provides more details on them.

14.5.3.1 Divide Description

MMAU supports below divide instructions:

- Five unsigned integer divide instructions (UDIV, UDIVD, UDIVDA, UDIVDD,UDIVDDA)
- Five signed integer divide instructions (SDIV, SDIVD, SDIVDA, SDIVDD, SDIVDDA)
- Three fractional number divide instructions (QDIV, QDIVD, QDIVDA)

All the divide computations require input operand from one or more Xn registers. For all the divide instructions, write to X3 operand register with the decoration code (Address[11:5]) will start the divide computation in MMAU.

For 64-bit quotient result, it is always stored in A10.

For 32-bit quotient result, in unsigned integer divide, it will be stored in A0 and A1 is cleared. In signed integer divide, the quotient result will be stored in A0 and A1 is signed extended from A0. For Q0.31 result, it will be in A1, and A0 will be cleared. So for any numeric type of divide computation, even if the result is 32-bit quotient, the A10 holds the meaningful value for optional of accumulation for next instruction.

For more details on the programming model of the divide instructions and their decoration code, please refer to Table 14-2.

14.5.3.1.1 Divide Algorithm Overview

The MMAU module implements a "shift, test, and restore" radix-2 algorithm for the divide operations. When performing a signed divide calculation (for both signed integer and signed fractional number), negative input operands are converted into 2’s complement positive numbers first, then an unsigned divide is performed, and the sign of the results based on the input operand signs, namely:

- The quotient is negated if the signs of the dividend and divisor are different
- Negative Flag CSR[N] will be set in this case

The negative flag bit (CSR[N]) is set based on the raw computation result before the overflow wrap-around. For negative quotient result, when overflow happens and if the saturation is not enabled, the result in A10 is a wrap-around data which could be positive number. In this case, the CSR[N] will still be set even A10 holds a positive result. Similarly for positive overflow if the saturation is not enabled, the result in A10 is a wrap-around data which could be negative number too. In this case, the CSR[N] will not be set since the raw result should be positive. In both cases, CSR[V, VIF] will be set.

For unsigned divide operation, the result can never be overflow or negative. So in unsigned divide operation, all the computation flag (CSR[N,V,Q]) will be cleared.

The hardware implementation processes two bits per machine cycle and includes "early termination" logic where the execution time is data dependent, based on the magnitude of the positive dividend. See Table 14-5 for more execution time details.

14.5.3.1.2 Signed Integer Divide Overflow

For signed divide, there is a single "special overflow case" which happens when compute the most negative number divided by -1. The result is positive but out of range in the same bit-width of 2's complement number.

For example in 32-bit signed integer divide (SDIV), it will execute X2/X3 and get a 32-bit result and put to A0 and A1 will be signed extended when no overflow happens. For the special overflow case, if the dividend = 0x8000_0000 and the divisor is -1 (0xFFFF_FFFF), the result of this (-2^{31}/-1) operation cannot be expressed as a 32-bit 2's complement number so it is an 32-bit overflow. For this case, CSR[V] and CSR[VIF] will be set. If saturation is not enabled (address[11]=0 when start the SDIV with write access to X3), the MMAU will returns 0x8000_0000 in A0. If saturation is enabled, the most positive number will be stored in A0, which is 0x7FFF_FFFF. In this case, A1 is cleared regardless of the saturation bit. CSR[N] will not be set.

For the case in 64-bit divide (SDIVD, SDIVDA,SDIVDD, SDIVDDA), the result will be returned in 64-bit in A10. For the special overflow case, if the dividend =0x8000_0000_0000_0000 and the divisor is -1 (either in 32-bit or 64-bit), the result of this (-2^{63}/-1) operation cannot be expressed as a 64-bit 2's complement number. So if saturation is not enabled, 0x8000_0000_0000_0000 will be stored to A10 as the result or if saturation is enabled, 0x7FFF_FFFF_FFFF_FFFF will be stored into A10 as the quotient.

For any overflow condition happened in divide, both CSR[V] and CSR[VIF] will be set no matter the saturation is enabled or not. If the CSR[VIE] is set, a MMAU interrupt will be generated.

14.5.3.1.3 Q0.31 and Q0.63 Divide Overflow

For signed fractional number, the overflow happens more often then signed integer divide. Remember the range of the signed Q0.31 and Q0.63:

For signed Q0.31:

- *its range is [-1, 1 - 2^{-31}]*
- *its resolution is 2^{-31}*

For signed Q0.63:

- *its range is [-1, 1 - 2^{-63}]*
- *its resolution is 2^{-63}*

For any divide operation, if the result is out of the above range, it is an overflow. For example in the case of the abs(divisor)<=abs(dividend), or in the case of overflow is (-1/-1 = 1), the divide result will be out of the numeric range represented by the signed Q0.31 or Q0.63.

When overflow happens in fractional number divide operation, if the saturation is not enabled, the wrap-around value of the result will be returned. If the saturation is enabled, if it is positive overflow, the value 0x7FFF_FFFF (to A1) will be returned with A0 cleared. If it is negative overflow and saturation is enabled, the value of 0x8000_0000 (to A1) will be returned with A0 cleared.

For any overflow condition happened in divide, both CSR[V] & CSR[VIF] will be set no matter the saturation is enabled or not. If the CSR[VIE] is set, a MMAU interrupt will be generated.

14.5.3.1.4 Special case: Divide-by-Zero

For any divide operation, if the divisor is zero, the MMAU hardware detects this condition in early stage and terminates the calculation immediately. And both CSR[DZ, DZIF] bits will be set. The quotient result is forced to 0x0000_0000. Additionally, if CSR[DZIE] = 1, a MMAU interrupt will be generated.

14.5.3.2 Square Root Description

MMAU supports below square root instructions:

- Three unsigned integer square root instructions (USQR, USRQD, USQRDA)
- Three fractional number square root instructions (QSQR, QSQRD, QSQRDA)

For unsigned square root, the result will be in 32-bit and stored in A0. A1 will be cleared.

For Fractional Square root, the input data is signed Q0.31/Q0.63. If the input data is negative fractional number, MMAU will convert it to its 2's complement positive number and perform the square root calculation. The result will be always be a positive fractional number in Q0.31 format and stored in A1. A0 will be cleared. For square root calculation on (-1), the result will be 1 and is out of the numeric range represented by Q0.31/Q0.63, and MMAU will return (-1) in Q0.31 format for this case, and set CSR[V] and CSR[VIF], CSR[N] will be cleared. For other square root calculation, CSR[N,V,Q] flags will be cleared.

Both USQRDA and QSQRDA are taking A10 as the operand register. So there is no need for user to write to any of the Xn registers. USQRDA will return the result in A0, so read access to the A0 with its decoration code will start the calculation, the read will be pending until the calculation finished. QSQRDA will return the result in A1, so read access to the A1 with its decoration code will start the calculation, the read will be pending until the calculation finished. Specifically:

- USQRDA: read A0 with address 0x270 starts the calculation and returned A0
- QSQRDA: read A1 with address 0x2F4 starts the calculation and returned A1

For other square root calculation, the operand registers are from Xn and the computation is started when write to X3 with the corresponding decoration code (Address bit[11:5]).

For more details on the programming model of the square root instructions and their decoration code, please refer to Table 14-2.

14.5.3.2.1 Square Root Algorithm Overview

The square root algorithm begins by creating a 64-bit “one-hot” bit vector signaling the highest power of four of the contents of the Radicand register (RCND). It then iterates through an algorithm involving magnitude comparisons of the RCND register versus the working result plus bit vector summation, conditional decrementing of the radicand, a 1-bit right shift of the result, and a 2-bit right shift of the one-hot bit vector.

Processing two bits of the radicand per cycle, the result register finishes with the integer portion of the square root calculation. The module includes early termination logic so that the execution time is data dependent, based on the magnitude of the input radicand. See Table 14-6 for more execution time details. Since both algorithms share common hardware structures, the incremental cost of the square root logic is an extremely small delta to the basic divide hardware.

The square root algorithm was exhaustively compared (that is, all 2^{32} possible input values) against the standard GNU C library implementation, which converts the unsigned integer input into a double-precision floating-point number, calculates the double-precision square root and then converts it back into an unsigned integer. Each input value calculated identical square root results.

14.5.3.3 Multiply Description

MMAU supports below multiply instructions:

- Three unsigned integer multiply instructions (UMUL, UMULD, UMULDA)
- Three signed integer multiply instructions (SMUL, SMULD, SMULDA)
- Three fractional number multiply instructions (QMUL, QMULD, QMULDA)

All multiply computation product are in 64bit and stored in A10. For fractional multiply, the results are in signed Q0.63 format.

For unsigned integer multiply, the CSR[N] will be cleared.

For signed integer or fractional number multiply, negative input operands are converted into 2’s complement positive numbers first, an unsigned multiply performed, and the sign of the results based on the input operand signs, namely:

- The product is negated if the signs of the operands of the multiply operation are different
- Negative Flag CSR[N] will be set in this case

The negative flag bit (CSR[N]) is set based on the raw product result before the overflow wrap-around. For negative product result, when overflow happens and if the saturation is not enabled, the result in A10 is a wrap-around data which could be positive number. In this case, the CSR[N] will still be set even A10 hold the positive number. Similarly for positive overflow if the saturation is not enabled, the result in A10 is a wrap-around data which could be negative number. In this case, the CSR[N] will not be set since the raw result should be positive. And in both cases CSR[V, VIF] will be set.

All the multiply instructions require Xn operand registers and the write to X3 with decoration code (Address[10:5]) matching the instruction will start the calculation.

For more details on the programming model of the multiply instructions and their decoration code, please refer to Table 14-2 for detail information.

14.5.3.3.1 Multiply Overflow

For 32-bit integer multiply (MUL, SMUL), the result is 64bit and stored in A10 and overflow can never happen.

For the 32x64 signed and unsigned integer multiply, the raw product result is 94-bit. If there is an overflow in 64-bit for this raw result, the CSR[V, VIF] will be set.

For unsigned integer overflow, if saturation is enabled, A10 will be get result of 0xFFFF_FFFF_FFFF_FFFF. For signed overflow, if saturation is enabled, A10 will get the result of 0x8000_0000_0000_0000 if it is negative overflow, or 0x7FFF_FFFF_FFFF_FFFF if it is positive overflow.

For fractional number Q0.31 or Q0.63, there is only one special case for overflow, which is (-1)x(-1)=1, where 1 is out of the range in Q0.31/0.63 format. In this case, CSR[V, VIF] will be set. The A10 will get 0x7FFF_FFFF_FFFF_FFFF if saturation is enabled, or 0x8000_0000_0000_0000 if saturation is not enabled.

In all cases where multiply overflow happens, both CSR[V, VIF] will be set no matter the saturation is enabled or not. If the CSR[VIE] is set, a MMAU interrupt will be generated.

14.5.3.4 Multiply with Accumulation Description

MMAU supports below multiply with accumulation instructions:

- Three unsigned integer multiply with accumulation instructions (UMAC, UMACD, UMACDA)
- Three signed integer multiply with accumulation instructions (SMAC, SMACD, SMACDA)
- Three fractional number multiply with accumulation instructions (QMAC, QMACD, QMACDA)

The multiply with accumulation operations are performed in two steps: MMAU performs multiply in the same way as in the multiply instruction first, and then the multiply product will be added up with A10 or X21 depends on the instruction set. The final result will be stored it to A10 in 64-bit.

For multiply with accumulation, the status bits of CSR[V,VIF] are set in the same way as in multiply instructions, as described in previous section. There are two more accumulation overflow status bits in CSR: accumulation overflow flag (CSR[Q]) and accumulation overflow interrupt flag(CSR[QIF]). Both CRS[Q, QIF] will be set when there is accumulation overflow in 64bit. The negative status flag (CSR[N]) are set based on the accumulation result too.

For unsigned multiply with accumulation, since all the numbers are in positive, the multiply overflow will lead to accumulation overflow. In this case, both CSR[V,VIF] and CSR[Q,QIF] will be set.

For signed integer or signed fractional multiply with accumulation, the negative status flag is set based on the accumulation raw result before overflow wrap-around. For a negative accumulation result, when overflow happens and if the saturation is not enabled, the result in A10 is a wrap-around data which could be positive number. In this case, the CSR[N] will still be set even A10 hold the positive number. Similarly for positive overflow if the saturation is not enabled, the result in A10 is a wrap-around data which could be negative number. In this case, the CSR[N] will not be set since the raw result should be positive. And in both cases where accumulation overflows, CSR[Q, QIF] will be set.

In all cases, any multiply overflow will get both CSR[V, VIF] set and if CSR[VIE] is set to enable the multiply/divide overflow interrupt, MMAU will generate interrupt. Any accumulation overflow will get both CSR[Q, QIF] set and if CSR[QIE] is set to enable the accumulation overflow interrupt, MMAU will generate the interrupt.

For more details on the programming model of the multiply with accumulation instructions and their decoration code, please refer to Table 14-2 for detail information.

14.5.3.5 Execution times

MMAU support fast multiply hardware engine so all the multiply completes in single cycle and multiply with accumulation completes in two cycles.

For divide and square root computation, the MMAU module includes early termination logic to finish both divide and square root calculations as quickly as possible, based on the magnitude of the input operand. Accordingly, the execution time for the calculations is data dependent as defined in Table 14-5 and Table 14-6. In this context, the execution time is defined from the register write to initiate the calculation until the result register has been updated and available to read. Stated differently, it represents the time CSR[BUSY] is asserted for a given calculation. In the following two tables, "x" signals a bit with a don't care value.

Table 14-5. Divide Execution Times

ABS(Dividend[63:0])	Execution Time with CSR[BUSY] = 1 [cycles]
(01,1x)xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	33
00(01,1x)_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	32
0000_(01,1x)xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	31
0000_00(01,1x)_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	30
0000_0000_(01,1x)xx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	29
...	...
...	...
...	...
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__(01,1x)xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__00(01,1x)_xxxx_xxxx_xxxx	8
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_(01,1x)xx_xxxx_xxxx	7
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_00(01,1x)_xxxx_xxxx	6
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_(01,1x)xx_xxxx	5
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_00(01,1x)_xxxx	4
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_(01,1x)xx	3
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_00(01,1x)	2
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000	1

Table 14-6. Square Root Execution Times

RCND[63:0]	Execution Time with CSR[BUSY] = 1 [cycles]
(01,1x)xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	33

Table continues on the next page...

Table 14-6. Square Root Execution Times (continued)

RCND[63:0]	Execution Time with CSR[BUSY] = 1 [cycles]
00(01,1x)_xxxx_xxxx_xxxx_x_xxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	32
0000_(01,1x)xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	31
0000_00(01,1x)_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	30
0000_0000_(01,1x)xx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	29
...	...
...	...
...	...
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__(01,1x)xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__00(01,1x)_xxxx_xxxx_xxxx	8
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_(01,1x)xx_xxxx_xxxx	7
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_00(01,1x)_xxxx_xxxx	6
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_(01,1x)xx_xxxx	5
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_00(01,1x)_xxxx	4
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_(01,1x)xx	3
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_00(01,1x)	2
0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000__0000_0000_0000_0000	2

14.5.4 MMAU Software Interface

MMAU provide a simple register read/write interface for passing arithmetic instruction operands, fetching result. All the operation is launched by read/write access to MMAU programming visible registers with related decoration code in address[11:5]. For detail about the decorated memory-mapping to each instructions, please refer to Table 14-2 for detail information.

14.5.4.1 Operation activation and result retrieval

Except for square root with A10 as operand, all other instructions need input operand from Xn in which X3 is always needed. So the write access to X3 with the arithmetic instruction decoration code (Address[10:5]) will launch the related calculation immediately when the MMAU is in IDLE.

There are only two instructions whose executions are not started by write access to X3. They are {U|Q}SQRDA which perform square root calculation on A10 as unsigned integer or fractional number in Q0.63 format. For USQRDA, the result will be stored in

A0. So the read access to A0 with Address[10:5] matching USQRDA instruction set will trigger this operation. For QSQRDA the result is in A1, so the read access to A1 with Address[10:5] matching QSQRDA instruction set will trigger this operation.

In all cases, the computation result will be in A10 if they are in 64-bit. For fractional number result, they are in Q0.63 format.

For 32-bit signed integer result, the result will be in A0, while A1 will be A0 signed extension so that A10 is a meaningful result in 64-bit, and ready for next round of computation.

For 32-bit unsigned result, the result will be in A0, and A1 will be cleared.

For 32-bit fractional result, the result will be in A1 in Q0.31 format. And A10 will also the same value of fractional number in Q0.63 format so that the A10 is a meaningful result in 64-bit too, and get ready for next round of computation.

14.5.4.2 Context save and restore

Given that multiple memory-mapped register accesses are needed for each divide and square root calculation, interrupts may occur during the required sequence of operations. As a result, the MMAU's programming model can be saved at entry to an interrupt service routine (ISR) and then restored when redispatching to the interrupted task.

There are seven programming visible registers in MMAU X0-3, A0-1, CSR. When read/write to those registers for context saving/restoring purpose, Address[11:5] need to be all zero (to match the REG_RW op_code). It is suggested using STM/LDM to save/restore the seven registers in ISR.

When read MMAU registers for context saving, it is always suggested to read in the order X0-3, A0-1, CSR. If the MMAU is busy, reading A0 or A1 will be pended until the MMAU is IDLE. Read CSR after A0-1 can guarantee that the status flag in CSR holding the latest status from the recent execution of instruction.

14.5.4.3 DMA and Interrupt

If MMAU CSR[DRE] is set to 1, MMAU will generate dma request whenever MMAU is IDLE. MMAU is in IDLE when it is come out from power-on reset or completed the computation of any instruction. This makes it's possible to relief CPU from programing MMAU registers by using DMA to configure the arithmetic operation and fetching the computation result to system memory from MMAU.

MMAU also support three interrupt for reporting computation status on Divide by Zero, Computation Overflow.

When CSR[DZIE] is set, MMAU will generate the interrupt when CSR[DZIF] is set. CSR[DZIF] interrupt flag can only be cleared by w1c through the CSR_IF_CLR (offset 0x1C) register entry point.

When CSR[VIE] is set, MMAU will generate the interrupt when CSR[VIF] is set. CSR[VIF] interrupt flag can only be cleared by w1c through the CSR_IF_CLR (offset 0x1C) register entry point.

When CSR[QIE] is set, MMAU will generate the interrupt when CSR[QIF] is set. CSR[QIF] interrupt flag can only be cleared by w1c through the CSR_IF_CLR (offset 0x1C) register entry point.

Please refer to CSR Interrupt Flags Clearance Register (MMAU_CSR_IF_CLR) for details on MMAU status bits setting and clearance mechanism.

Chapter 15
System Mode Controller (SMC)

15.1 Introduction

The System Mode Controller (SMC) is responsible for sequencing the system into and out of all low-power Stop and Run modes.

Specifically, it monitors events to trigger transitions between power modes while controlling the power, clocks, and memories of the system to achieve the power consumption and functionality of that mode.

This chapter describes all the available low-power modes, the sequence followed to enter/exit each mode, and the functionality available while in each of the modes.

The SMC is able to function during even the deepest low power modes.

See AN4503: Power Management for Kinetis MCUs for further details on using the SMC.

15.2 Modes of operation

The ARM CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

The WFI or WFE instruction is used to invoke Sleep and Deep Sleep modes. Run, Wait, and Stop are the common terms used for the primary operating modes of Freescale microcontrollers.

The following table shows the translation between the ARM CPU modes and the Freescale MCU power modes.

ARM CPU mode	MCU mode
Sleep	Wait
Deep Sleep	Stop

Accordingly, the ARM CPU documentation refers to sleep and deep sleep, while the Freescale MCU documentation normally uses wait and stop.

In addition, Freescale MCUs also augment Stop, Wait, and Run modes in a number of ways. The power management controller (PMC) contains a run and a stop mode regulator. Run regulation is used in normal run, wait and stop modes. Stop mode regulation is used during all very low power and low leakage modes. During stop mode regulation, the bus frequencies are limited in the very low power modes.

The SMC provides the user with multiple power options. The Very Low Power Run (VLPR) mode can drastically reduce run time power when maximum bus frequency is not required to handle the application needs. From Normal Run mode, the Run Mode (RUNM) field can be modified to change the MCU into VLPR mode when limited frequency is sufficient for the application. From VLPR mode, a corresponding wait (VLPW) and stop (VLPS) mode can be entered.

Depending on the needs of the user application, a variety of stop modes are available that allow the state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. Several registers are used to configure the various modes of operation for the device.

The following table describes the power modes available for the device.

Table 15-1. Power modes

Mode	Description
RUN	The MCU can be run at full speed and the internal supply is fully regulated, that is, in run regulation. This mode is also referred to as Normal Run mode.
WAIT	The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate. Run regulation is maintained.
STOP	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
VLPR	The core, system, bus, and flash clock maximum frequencies are restricted in this mode. See the Power Management chapter for details about the maximum allowable frequencies.
VLPW	The core clock is gated off. The system, bus, and flash clocks continue to operate, although their maximum frequency is restricted. See the Power Management chapter for details on the maximum allowable frequencies.
VLPS	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
VLLS3	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low

Table continues on the next page...

Table 15-1. Power modes (continued)

Mode	Description
	leakage mode by powering down the internal logic. All system RAM contents are retained and I/O states are held. Internal logic states are not retained.
VLLS2	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic.
VLLS1	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained.
VLLS0	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained. The 1kHz LPO clock is disabled and the power on reset (POR) circuit can be optionally enabled using STOPCTRL[PORPO].

15.3 Memory map and register descriptions

Information about the registers related to the system mode controller can be found here.

Different SMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

NOTE

The SMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

SMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_E000	Power Mode Protection register (SMC_PMPROT)	8	R/W	See section	15.3.1/256

Table continues on the next page...

SMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_E001	Power Mode Control register (SMC_PMCTRL)	8	R/W	See section	15.3.2/257
4007_E002	Stop Control Register (SMC_STOPCTRL)	8	R/W	03h	15.3.3/258
4007_E003	Power Mode Status register (SMC_PMSTAT)	8	R	See section	15.3.4/260

15.3.1 Power Mode Protection register (SMC_PMPROT)

This register provides protection for entry into any low-power run or stop mode. The enabling of the low-power run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

The PMPROT register can be written only once after any system reset.

If the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the MCU is still in Normal Run mode.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 0h offset = 4007_E000h

Bit	7	6	5	4	3	2	1	0
Read	0	0	AVLP	0	0	0	AVLLS	0
Write								
Reset	0	0	*	0	0	0	0	0

* Notes:
- AVLP field: When booting in run mode, the reset value is 0. When booting in VLPR mode, the reset value is 1.

SMC_PMPROT field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 AVLP	Allow Very-Low-Power Modes

Table continues on the next page...

SMC_PMPROT field descriptions (continued)

Field	Description
	Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter any very-low-power mode (VLPR, VLPW, and VLPS). 0 VLPR, VLPW, and VLPS are not allowed. 1 VLPR, VLPW, and VLPS are allowed.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 AVLLS	Allow Very-Low-Leakage Stop Mode Provided the appropriate control bits are set up in PMCTRL, this write once bit allows the MCU to enter any very-low-leakage stop mode (VLLSx). 0 Any VLLSx mode is not allowed 1 Any VLLSx mode is allowed
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

15.3.2 Power Mode Control register (SMC_PMCTRL)

The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 1h offset = 4007_E001h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	RUNM		0	STOPA	STOPM		
Write								
Reset	0	*	*	0	0	0	0	0

* Notes:
- RUNM field: When booting in run mode, the reset value is 00. When booting in VLPR mode, the reset value is 01.

SMC_PMCTRL field descriptions

Field	Description
7 Reserved	This field is reserved. This bit is reserved for future expansion and should always be written zero.
6–5 RUNM	Run Mode Control When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. **NOTE:** RUNM may be set to VLPR only when PMSTAT=RUN. After being written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR. 00 Normal Run mode (RUN) 01 Reserved 10 Very-Low-Power Run mode (VLPR) 11 Reserved
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 STOPA	Stop Aborted When set, this read-only status bit indicates an interrupt or reset occured during the previous stop mode entry sequence, preventing the system from entering that mode. This field is cleared by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted. 0 The previous stop mode entry was successsful. 1 The previous stop mode entry was aborted.
STOPM	Stop Mode Control When written, controls entry into the selected stop mode when Sleep-Now or Sleep-On-Exit mode is entered with SLEEPDEEP=1 . Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to VLLSx, the VLLSM field in the STOPCTRL register is used to further select the particular VLLS submode which will be entered. **NOTE:** When set to STOP, the PSTOPO bits in the STOPCTRL register can be used to select a Partial Stop mode if desired. 000 Normal Stop (STOP) 001 Reserved 010 Very-Low-Power Stop (VLPS) 011 Reserved 100 Very-Low-Leakage Stop (VLLSx) 101 Reserved 110 Reseved 111 Reserved

15.3.3 Stop Control Register (SMC_STOPCTRL)

The STOPCTRL register provides various control bits allowing the user to fine tune power consumption during the stop mode selected by the STOPM field.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 2h offset = 4007_E002h

Bit	7	6	5	4	3	2	1	0
Read	PSTOPO		PORPO	0	Reserved	VLLSM		
Write								
Reset	0	0	0	0	0	0	1	1

SMC_STOPCTRL field descriptions

Field	Description
7–6 PSTOPO	Partial Stop Option These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN (or VLPR) mode, the PMC, MCG and flash remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated. 00 STOP - Normal Stop mode 01 PSTOP1 - Partial Stop with both system and bus clocks disabled 10 PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11 Reserved
5 PORPO	POR Power Option This bit controls whether the POR detect circuit is enabled in VLLS0 mode. 0 POR detect circuit is enabled in VLLS0 1 POR detect circuit is disabled in VLLS0
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This bit is reserved for future expansion and should always be written zero.
VLLSM	VLLS Mode Control This field controls which VLLS sub-mode to enter if STOPM = VLLSx. 000 VLLS0 001 VLLS1 010 VLLS2 011 VLLS3 100 Reserved 101 Reserved 110 Reserved 111 Reserved

15.3.4 Power Mode Status register (SMC_PMSTAT)

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 3h offset = 4007_E003h

Bit	7	6	5	4	3	2	1	0
Read	PMSTAT							
Write								
Reset	*	*	*	*	*	*	*	*

* Notes:
- PMSTAT field: When booting in run mode, the reset value is 0x00. When booting in VLPR mode, the reset value is 0x04.

SMC_PMSTAT field descriptions

Field	Description
PMSTAT	Power Mode Status **NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** When a PSTOP mode is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** Since the RUNM bits in the PMCTRL register are reset to VLPR on any Chip Reset not VLLS, the PMSTAT will update to VLPR shortly after the reset sequence is complete. 0000_0001 Current power mode is RUN. 0000_0010 Current power mode is STOP. 0000_0100 Current power mode is VLPR. 0000_1000 Current power mode is VLPW. 0001_0000 Current power mode is VLPS. 0010_0000 Reserved 0100_0000 Current power mode is VLLS. 1000_0000 Reserved

15.4 Functional description

15.4.1 Power mode transitions

The following figure shows the power mode state transitions available on the chip. Any reset will initially bring the MCU back to the normal RUN state. However, in order to minimize peak power consumption, the RUNM bits in the PMCTRL register can be reset to VLPR via Flash IFR settings, causing the SMC to begin transitioning the MCU into VLPR mode during the reset recovery sequence.

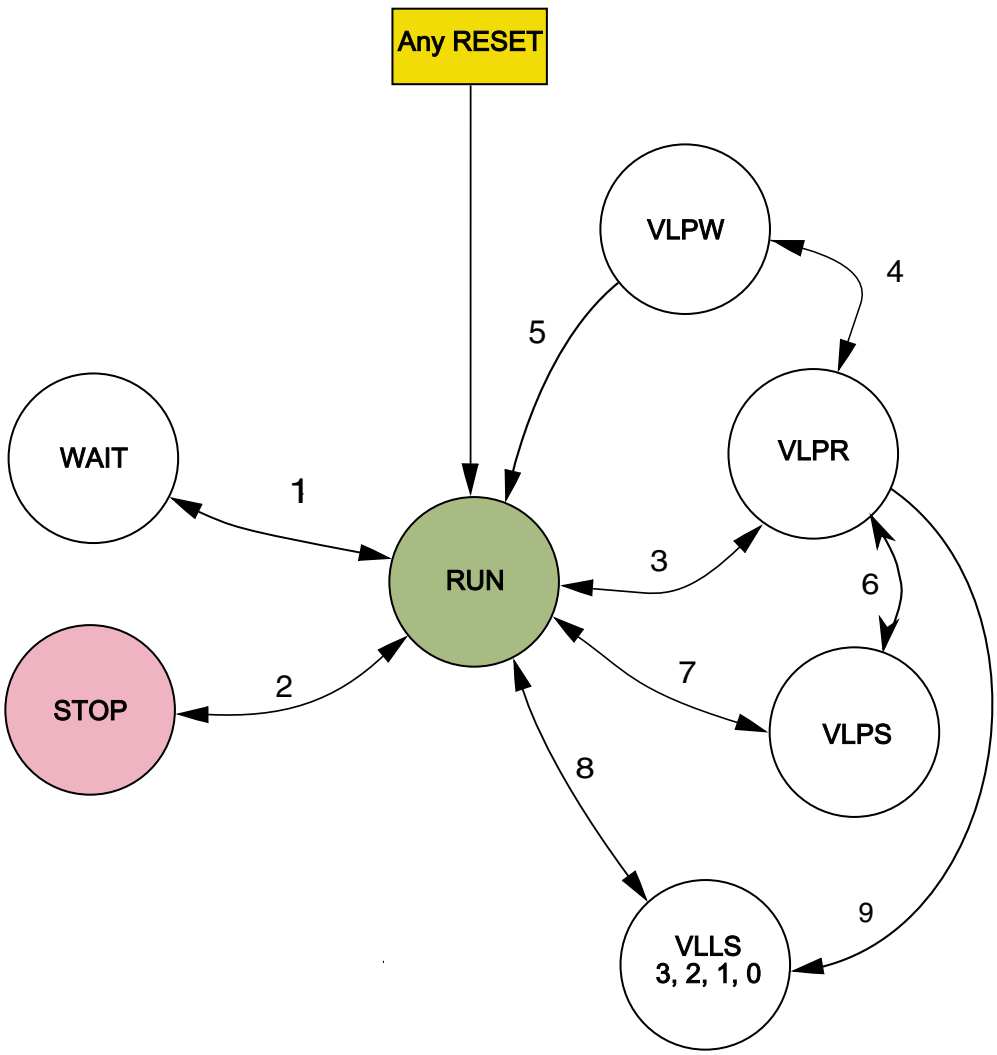


Figure 15-1. Power mode state diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 15-2. Power mode transition triggers

Transition #	From	To	Trigger conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in ARM core. See note.[1]
	WAIT	RUN	Interrupt or Reset
2	RUN	STOP	PMCTRL[RUNM]=00, PMCTRL[STOPM]=000[2] Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	STOP	RUN	Interrupt or Reset
3	RUN	VLPR	The core, system, bus and flash clock frequencies and MCG clocking mode are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies and MCG modes supported. Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10. **NOTE:** In order to limit peak current, PMPROT[AVLP] and PMCTRL[RUNM] bits can be set on any Reset via Flash IFR settings, causing the SMC to transition the MCU from RUN->VLPR during the reset recovery sequence.
	VLPR	RUN	Set PMCTRL[RUNM]=00 or Reset.
4	VLPR	VLPW	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, which is controlled in System Control Register in ARM core. See note.[1]
	VLPW	VLPR	Interrupt
5	VLPW	RUN	Reset
6	VLPR	VLPS	PMCTRL[STOPM]=000[3] or 010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	VLPS	VLPR	Interrupt **NOTE:** If VLPS was entered directly from RUN (transition #7), hardware forces exit back to RUN and does not allow a transition to VLPR.
7	RUN	VLPS	PMPROT[AVLP]=1, PMCTRL[STOPM]=010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	VLPS	RUN	Interrupt and VLPS mode was entered directly from RUN or Reset

Table continues on the next page...

Table 15-2. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
8	RUN	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100,STOPCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.
	VLLSx	RUN	Wakeup from enabled LLWU input source or RESET pin
9	VLPR	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100,STOPCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.

1. If debug is enabled, the core clock remains to support debug.
2. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of STOP
3. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=00, then VLPS mode is entered instead of STOP. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of VLPS

15.4.2 Power mode entry/exit sequencing

When entering or exiting low-power modes, the system must conform to an orderly sequence to manage transitions safely.

The SMC manages the system's entry into and exit from all power modes. This diagram illustrates the connections of the SMC with other system components in the chip that are necessary to sequence the system through all power modes.

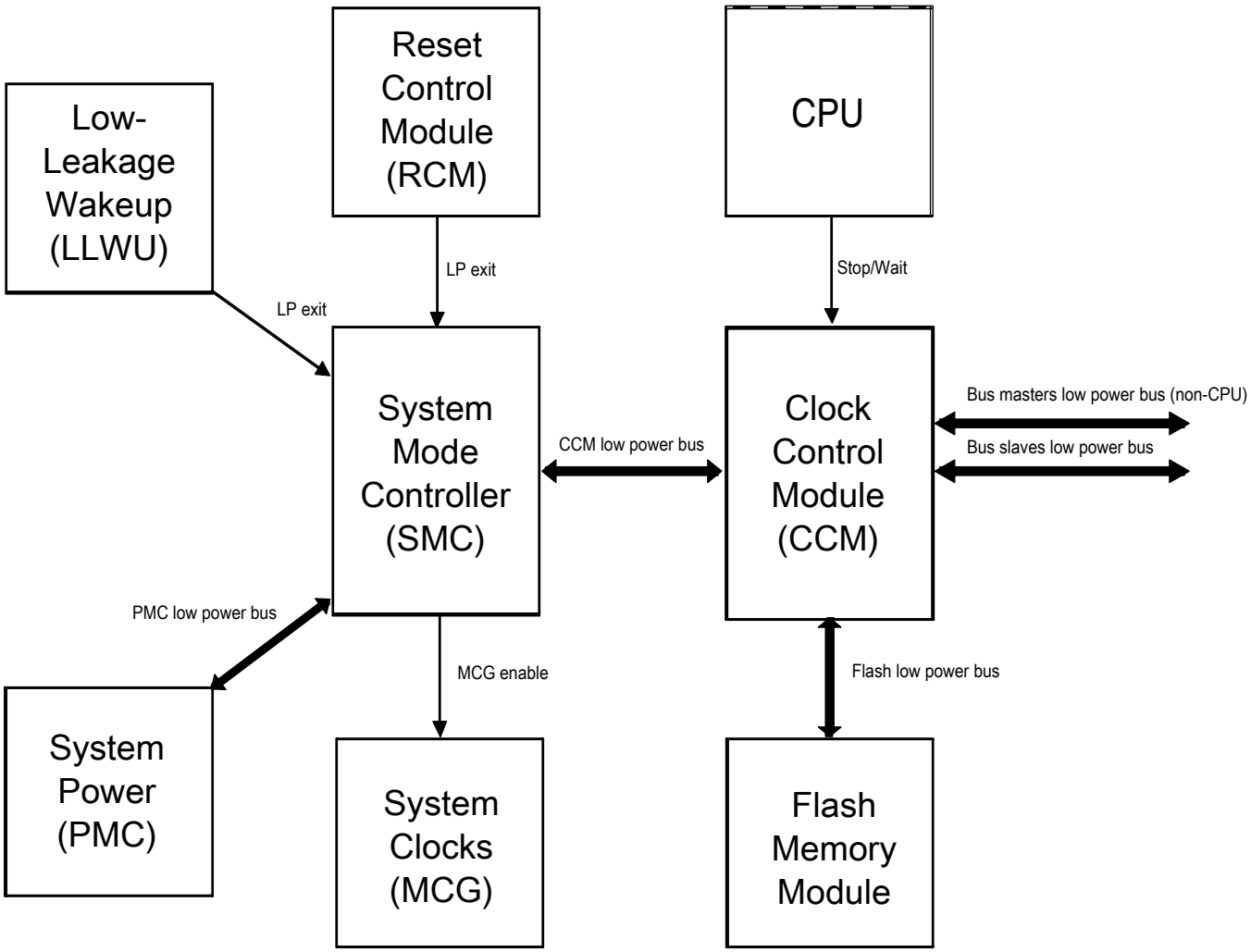


Figure 15-2. Low-power system components and connections

15.4.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS, VLLSx) is initiated by a CPU executing the WFI instruction. After the instruction is executed, the following sequence occurs:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus masters to enter Stop mode.
3. After all masters have acknowledged they are ready to enter Stop mode, requests are made to all bus slaves to enter Stop mode.
4. After all slaves have acknowledged they are ready to enter Stop mode, all system and bus clocks are gated off.
5. Clock generators are disabled in the MCG.
6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low-power mode.

15.4.2.2 Stop mode exit sequence

Exit from a low-power stop mode is initiated either by a reset or an interrupt event. The following sequence then executes to restore the system to a run mode (RUN or VLPR):

1. The on-chip regulator in the PMC and internal power switches are restored.
2. Clock generators are enabled in the MCG.
3. System and bus clocks are enabled to all masters and slaves.
4. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the low-power stop mode.

15.4.2.3 Aborted stop mode entry

If an interrupt or a reset occurs during a stop entry sequence, the SMC can abort the transition early and return to RUN mode without completely entering the stop mode. An aborted entry is possible only if the reset or interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt or reset is ignored until the PMC has completed its transition to stop mode regulation. When an aborted stop mode entry sequence occurs, SMC_PMCTRL[STOPA] is set to 1.

15.4.2.4 Transition to wait modes

For wait modes (WAIT and VLPW), the CPU clock is gated off while all other clocking continues, as in RUN and VLPR mode operation. Some modules that support stop-in-wait functionality have their clocks disabled in these configurations.

15.4.2.5 Transition from stop modes to Debug mode

The debugger module supports a transition from STOP, WAIT, VLPS, and VLPW back to a Halted state when the debugger has been enabled. As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

15.4.3 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)

15.4.3.1 RUN mode

This mode is selected after any reset. In order to reduce peak power consumption, however, the SMC will begin transitioning the MCU into VLPR mode during the reset recovery sequence. When the ARM processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF_FFFF.

To reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the SIM's registers.

15.4.3.2 Very-Low Power Run (VLPR) mode

In VLPR mode, the on-chip voltage regulator is put into a stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the SIM's registers.

Before entering this mode, the following conditions must be met:

- The MCG must be configured in a mode which is supported during VLPR. See the Power Management details for information about these MCG modes.
- All clock monitors in the MCG must be disabled.
- The maximum frequencies of the system, bus, flash, and core are restricted. See the Power Management details about which frequencies are supported.
- Mode protection must be set to allow VLP modes, that is, PMPROT[AVLP] is 1.
- PMCTRL[RUNM] must be set to 10b to enter VLPR.
- Flash programming/erasing is not allowed.

NOTE

Do not increase the clock frequency while in VLPR mode, because the regulator is slow in responding and cannot manage fast load transitions. In addition, do not modify the clock source in the MCG module or any clock divider registers. Module clock enables in the SIM can be set, however should only be cleared one at a time to limit load transitions.

To reenter Normal Run mode, clear PMCTRL[RUNM]. PMSTAT is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. When PMSTAT=RUN, the system is in run regulation and the MCU can run at full speed in any clock mode. If a higher execution frequency is desired, poll PMSTAT until it is set to RUN when returning from VLPR mode.

Any reset will initially cause the device to exit to RUN mode, however Flash IFR settings can be used to return the part to VLPR during the MCU reset flow.

15.4.4 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

15.4.4.1 WAIT mode

WAIT mode is entered when the ARM core enters the Sleep-Now or Sleep-On-Exit modes while SLEEDEEP is cleared. The ARM CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled. Clock gating to the peripheral is enabled via the SIM module.

When an interrupt request occurs, the CPU exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from WAIT mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

15.4.4.2 Very-Low-Power Wait (VLPW) mode

VLPW is entered by the entering the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the MCU is in VLPR mode.

In VLPW, the on-chip voltage regulator remains in its stop regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules by clearing the peripherals' corresponding clock gating control bits in the SIM.

VLPR mode restrictions also apply to VLPW.

When an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

A system reset will cause an exit from VLPW mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

15.4.5 Stop modes

This device contains a variety of stop modes to meet your application needs.

The stop modes range from:

- a stopped CPU, with all I/O, logic, and memory states retained, and certain asynchronous mode peripherals operating

to:

- a powered down CPU, with only I/O and a small register file retained, very few asynchronous mode peripherals operating, while the remainder of the MCU is powered down.

The choice of stop mode depends upon the user's application, and how power usage and state retention versus functional needs and recovery time may be traded off.

NOTE

All clock monitors must be disabled before entering these low-power modes: Stop, VLPS, VLPR, VLPW and VLLSx.

The various stop modes are selected by setting the appropriate fields in PMPROT and PMCTRL. The selected stop mode is entered during the sleep-now or sleep-on-exit entry with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The available stop modes are:

- Normal Stop (STOP)
- Very-Low Power Stop (VLPS)
- Very-Low-Leakage Stop (VLLSx)

15.4.5.1 STOP mode

STOP mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The MCG module can be configured to leave the reference clocks running.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

15.4.5.2 Very-Low-Power Stop (VLPS) mode

The two ways in which VLPS mode can be entered are listed here.

- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in VLPR mode and PMCTRL[STOPM] = 010 or 000.
- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in normal RUN mode and PMCTRL[STOPM] = 010. When VLPS is entered directly from RUN mode, exit to VLPR is disabled by hardware and the system will always exit back to RUN.

In VLPS, the on-chip voltage regulator remains in its stop regulation state as in VLPR.

A module capable of providing an asynchronous interrupt to the device takes the device out of VLPS and returns the device to VLPR mode.

A system reset will cause an exit from VLPS mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

15.4.5.3 Very-Low-Leakage Stop (VLLSx) modes

This device contains these very low leakage modes:

- VLLS3
- VLLS2
- VLLS1
- VLLS0

VLLSx is often used in this document to refer to all of these modes.

All VLLSx modes can be entered from normal RUN or VLPR modes.

The MCU enters the configured VLLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, the SLEEPDEEP bit is set in the System Control Register in the ARM core, and
- The device is configured as shown in Table 15-2.

In VLLS, the on-chip voltage regulator is in its stop-regulation state while most digital logic is powered off.

Before entering VLLS mode, the user should configure the Low-Leakage Wake-up (LLWU) module to enable the desired wakeup sources. The available wake-up sources in VLLS are detailed in the chip configuration details for this device.

After wakeup from VLLS, the device returns to normal RUN mode with a pending LLWU interrupt. If VLLS was entered from VLPR mode, then the MCU will begin VLPR entry shortly after wakeup. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wake-up flags to determine the source of the wake-up.

When entering VLLS, each I/O pin is latched as configured before executing VLLS. Because all digital logic in the MCU is powered off, all port and peripheral data is lost during VLLS. This information must be restored before PMC_REGSC[ACKISO] is set.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from any VLLS mode, temporarily returning the device to normal RUN mode before transitioning into VLPR during the MCU reset flow. When exiting VLLS via the $\overline{\text{RESET}}$ pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

15.4.6 Debug in low power modes

When the MCU is secure, the device disables/limits debugger operation. When the MCU is unsecure, the ARM debugger can assert two power-up request signals:

- System power up, via SYSPWR in the Debug Port Control/Stat register
- Debug power up, via CDBGPWRUPREQ in the Debug Port Control/Stat register

When asserted while in RUN, WAIT, VLPR, or VLPW, the mode controller drives a corresponding acknowledge for each signal, that is, both CDBGPWRUPACK and CSYSPWRUPACK. When both requests are asserted, the mode controller handles attempts to enter STOP and VLPS by entering an emulated stop state. In this emulated stop state:

- the regulator is in run regulation,
- the MCG-generated clock source is enabled,
- all system clocks, except the core clock, are disabled,
- the debug module has access to core registers, and
- access to the on-chip peripherals is blocked.

No debug is available while the MCU is in VLLS modes.

Entering into a VLLS mode causes all of the debug controls and settings to be powered off. To give time to the debugger to sync with the MCU, the MDM AP Control Register includes a Very-Low-Leakage Debug Request (VLLDBGREQ) bit that is set to configure the Reset Controller logic to hold the system in reset after the next recovery from a VLLS mode. This bit allows the debugger time to reinitialize the debug module before the debug session continues.

The MDM AP Control Register also includes a Very Low Leakage Debug Acknowledge (VLLDBGACK) bit that is set to release the ARM core being held in reset following a VLLS recovery. The debugger reinitializes all debug IP, and then asserts the VLLDBGACK control bit to allow the RCM to release the ARM core from reset and allow CPU operation to begin.

The VLLDBGACK bit is cleared by the debugger (or can be left set as is) or clears automatically due to the reset generated as part of the next VLLS recovery.

Chapter 16
Low-Leakage Wakeup Unit (LLWU)

16.1 Chip-specific LLWU information

16.1.1 Overview

This device includes a low leakage wakeup unit (LLWU) to allow the SoC to wakeup from VLLSx modes. This device uses the 24 external wakeup pin inputs and 6 internal modules as wakeup sources to the LLWU module.

The LLWU module is only functional in VLLSx modes.

NOTE

As there are 24 external wakeup pin inputs for this SoC, LLWU Pin Enable 7 register (LLWU_PE7), LLWU Pin Enable 8 register (LLWU_PE8) and LLWU Pin Flag 4 register (LLWU_PF4) are not available in the memory map, also the related information is not applicable in the Block diagram.

16.1.2 Wakeup Sources

The LLWU module has the following internal and external inputs. LLWU_P0-LLWU_P23 are external pin inputs, and module interrupt flags (M0IF-M7IF) are internal peripheral connections.

NOTE

- The $\overline{\text{RESET}}$ pin is also a wakeup source when the pin is enabled as $\overline{\text{RESET}}$ or GPIO via port mux control.
- Any input function muxed along with the LLWU_Px pin will cause wakeup .

Table 16-1. Wakeup sources for LLWU inputs

Input	Wakeup source
LLWU_P0	PTG6/LLWU_P0 pin
LLWU_P1	PTG2/LLWU_P1 pin
LLWU_P2	PTG1/LLWU_P2 pin
LLWU_P3	PTF6/LLWU_P3 pin
LLWU_P4	PTF0/LLWU_P4 pin
LLWU_P5	PTE6/LLWU_P5 pin
LLWU_P6	PTE5/LLWU_P6 pin
LLWU_P7	PTD7/LLWU_P7 pin
LLWU_P8	PTD6/LLWU_P8 pin
LLWU_P9	PTD4/LLWU_P9 pin
LLWU_P10	PTD2/LLWU_P10 pin
LLWU_P11	PTD0/LLWU_P11 pin
LLWU_P12	PTC5/LLWU_P12 pin
LLWU_P13	PTC3/LLWU_P13 pin
LLWU_P14	PTA6/LLWU_P14 pin
LLWU_P15	PTA4/LLWU_P15 pin
LLWU_P16	PTA0/LLWU_P16 pin
LLWU_P17	PTB1/LLWU_P17 pin
LLWU_P18	PTJ6/LLWU_P18 pin
LLWU_P19	PTK3/LLWU_P19 pin
LLWU_P20	PTF3/LLWU_P20 pin
LLWU_P21	PTI0/LLWU_P21 pin
LLWU_P22	PTI2/LLWU_P22 pin
LLWU_P23	PTL5/LLWU_P23 pin
LLWU_M0IF	LPTMR [1]
LLWU_M1IF	iRTC Alarm
LLWU_M2IF	CMP0
LLWU_M3IF	iRTC Interrupt
LLWU_M4IF	CMP1
LLWU_M5IF	CMP2
LLWU_M6IF	Reserved [2]
LLWU_M7IF	Reserved [2]

1. Requires the peripheral and the peripheral interrupt to be enabled. The LLWU's WUME bit enables the internal module flag as a wakeup input. After wakeup, the flags are cleared based on the peripheral clearing mechanism.
2. "Reserved" means the corresponding LLWU_ME[WUME*x*] and LLWU_MF5[MWUF*x*] bits are not valid on this SoC.

16.1.3 Reset due to LLWU wakeup event

Reset sequence initiated via the LLWU will follow the normal reset and boot up sequence, as mentioned in the "Reset and Boot" chapter.

16.2 Introduction

The LLWU module allows the user to select up to 32 external pins and up to 8 internal modules as interrupt wake-up sources from low-leakage power modes.

The input sources are described in the device's chip configuration details. Each of the available wake-up sources can be individually enabled.

The $\overline{\text{RESET}}$ pin is an additional source for triggering an exit from low-leakage power modes, and causes the MCU to exit VLLS through a reset flow.

The LLWU module also includes four optional digital pin filters for the external wakeup pins.

See AN4503: Power Management for Kinetis MCUs for further details on using the LLWU.

16.2.1 Features

The LLWU module features include:

- Support for up to 32 external input pins and up to 8 internal modules with individual enable bits for MCU interrupt from low leakage modes
- Input sources may be external pins or from internal peripherals capable of running in VLLS. See the chip configuration information for wakeup input sources for this device.
- External pin wake-up inputs, each of which is programmable as falling-edge, rising-edge, or any change
- Wake-up inputs that are activated after MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, the filters are disabled and bypassed.

16.2.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from VLLS, the LLWU continues to detect wake-up events until the user has acknowledged the wake-up via a write to PMC_REGSC[ACKISO].

16.2.2.1 VLLS modes

All wakeup and reset events result in VLLS exit via a reset flow.

16.2.2.2 Non-low leakage modes

The LLWU is not active in all non-low leakage modes where detection and control logic are in a static state. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When the wake-up pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within five LPO clock cycles of an active edge, the edge event will be detected by the LLWU.

16.2.2.3 Debug mode

When the chip is in Debug mode and then enters a VLLSx mode, no debug logic works in the fully-functional low-leakage mode. Upon an exit from the VLLSx mode, the LLWU becomes inactive.

16.2.3 Block diagram

The following figure is the block diagram for the LLWU module.

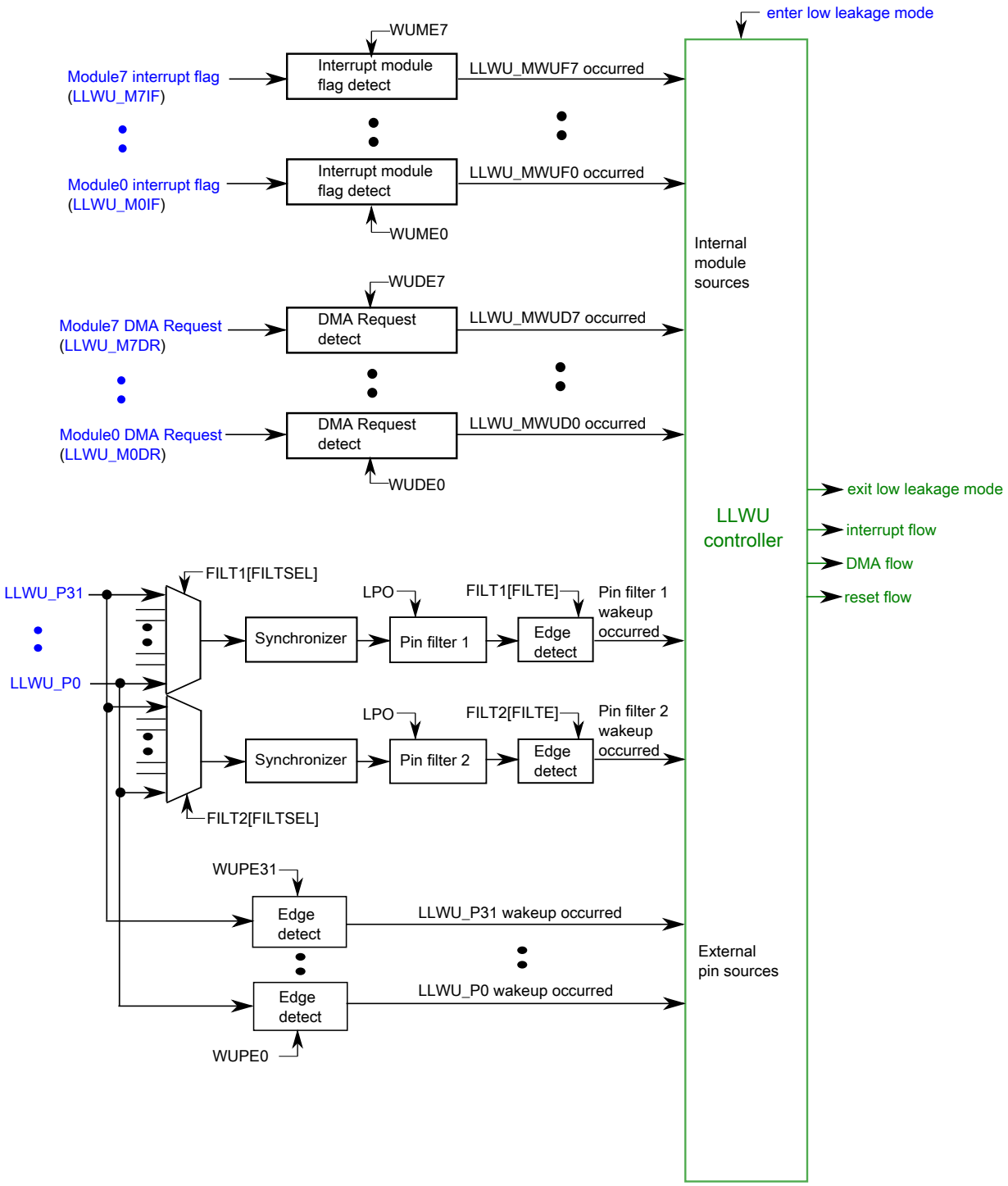


Figure 16-1. LLWU block diagram

16.3 LLWU signal descriptions

The signal properties of LLWU are shown in the table found here.

The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

Table 16-2. LLWU signal descriptions

Signal	Description	I/O
LLWU_Pn	Wakeup inputs (n = 0- 31)	I

16.4 Memory map/register definition

The LLWU includes the following registers:

- Wake-up source enable registers
  - Enable external pin input sources
  - Enable internal peripheral interrupt sources
- Wake-up flag registers
  - Indication of wakeup source that caused exit from a low-leakage power mode includes external pin or internal module interrupt
- Wake-up pin filter enable registers

NOTE

The LLWU registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

All LLWU registers are reset by Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. Each register's displayed reset value represents this subset of reset types. LLWU registers are unaffected by reset types that do not trigger Chip Reset not VLLS. For more information about the types of reset on this chip, refer to the Introduction details.

LLWU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C000	LLWU Pin Enable 1 register (LLWU_PE1)	8	R/W	00h	16.4.1/279
4007_C001	LLWU Pin Enable 2 register (LLWU_PE2)	8	R/W	00h	16.4.2/280
4007_C002	LLWU Pin Enable 3 register (LLWU_PE3)	8	R/W	00h	16.4.3/281
4007_C003	LLWU Pin Enable 4 register (LLWU_PE4)	8	R/W	00h	16.4.4/282

Table continues on the next page...

LLWU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C004	LLWU Pin Enable 5 register (LLWU_PE5)	8	R/W	00h	16.4.5/283
4007_C005	LLWU Pin Enable 6 register (LLWU_PE6)	8	R/W	00h	16.4.6/284
4007_C006	LLWU Pin Enable 7 register (LLWU_PE7)	8	R/W	00h	16.4.7/286
4007_C007	LLWU Pin Enable 8 register (LLWU_PE8)	8	R/W	00h	16.4.8/287
4007_C008	LLWU Module Enable register (LLWU_ME)	8	R/W	00h	16.4.9/288
4007_C009	LLWU Pin Flag 1 register (LLWU_PF1)	8	R/W	00h	16.4.10/289
4007_C00A	LLWU Pin Flag 2 register (LLWU_PF2)	8	R/W	00h	16.4.11/291
4007_C00B	LLWU Pin Flag 3 register (LLWU_PF3)	8	R/W	00h	16.4.12/293
4007_C00C	LLWU Pin Flag 4 register (LLWU_PF4)	8	R/W	00h	16.4.13/294
4007_C00D	LLWU Module Flag 5 register (LLWU_MF5)	8	R	00h	16.4.14/296
4007_C00E	LLWU Pin Filter 1 register (LLWU_FILT1)	8	R/W	00h	16.4.15/298
4007_C00F	LLWU Pin Filter 2 register (LLWU_FILT2)	8	R/W	00h	16.4.16/299

16.4.1 LLWU Pin Enable 1 register (LLWU_PE1)

LLWU_PE1 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P3-LLWU_P0.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 0h offset = 4007_C000h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0

LLWU_PE1 field descriptions

Field	Description
7–6 WUPE3	Wakeup Pin Enable For LLWU_P3 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

Table continues on the next page...

LLWU_PE1 field descriptions (continued)

Field	Description
5–4 WUPE2	Wakeup Pin Enable For LLWU_P2 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE1	Wakeup Pin Enable For LLWU_P1 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE0	Wakeup Pin Enable For LLWU_P0 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.2 LLWU Pin Enable 2 register (LLWU_PE2)

LLWU_PE2 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P7-LLWU_P4.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 1h offset = 4007_C001h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE7		WUPE6		WUPE5		WUPE4	
Reset	0	0	0	0	0	0	0	0

LLWU_PE2 field descriptions

Field	Description
7–6 WUPE7	Wakeup Pin Enable For LLWU_P7

Table continues on the next page...

LLWU_PE2 field descriptions (continued)

Field	Description
	Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE6	Wakeup Pin Enable For LLWU_P6 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE5	Wakeup Pin Enable For LLWU_P5 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE4	Wakeup Pin Enable For LLWU_P4 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.3 LLWU Pin Enable 3 register (LLWU_PE3)

LLWU_PE3 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P11-LLWU_P8.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 2h offset = 4007_C002h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE11		WUPE10		WUPE9		WUPE8	
Reset	0	0	0	0	0	0	0	0

LLWU_PE3 field descriptions

Field	Description
7–6 WUPE11	Wakeup Pin Enable For LLWU_P11 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE10	Wakeup Pin Enable For LLWU_P10 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE9	Wakeup Pin Enable For LLWU_P9 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE8	Wakeup Pin Enable For LLWU_P8 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.4 LLWU Pin Enable 4 register (LLWU_PE4)

LLWU_PE4 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P15-LLWU_P12.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 3h offset = 4007_C003h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE15		WUPE14		WUPE13		WUPE12	
Reset	0	0	0	0	0	0	0	0

LLWU_PE4 field descriptions

Field	Description
7–6 WUPE15	Wakeup Pin Enable For LLWU_P15 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE14	Wakeup Pin Enable For LLWU_P14 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE13	Wakeup Pin Enable For LLWU_P13 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE12	Wakeup Pin Enable For LLWU_P12 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.5 LLWU Pin Enable 5 register (LLWU_PE5)

LLWU_PE5 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P19-LLWU_P16.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset

types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 4h offset = 4007_C004h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE19		WUPE18		WUPE17		WUPE16	
Reset	0	0	0	0	0	0	0	0

LLWU_PE5 field descriptions

Field	Description
7–6 WUPE19	Wakeup Pin Enable For LLWU_P19 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE18	Wakeup Pin Enable For LLWU_P18 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE17	Wakeup Pin Enable For LLWU_P17 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE16	Wakeup Pin Enable For LLWU_P16 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.6 LLWU Pin Enable 6 register (LLWU_PE6)

LLWU_PE6 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P23-LLWU_P20.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 5h offset = 4007_C005h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE23		WUPE22		WUPE21		WUPE20	
Reset	0	0	0	0	0	0	0	0

LLWU_PE6 field descriptions

Field	Description
7–6 WUPE23	Wakeup Pin Enable For LLWU_P23 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE22	Wakeup Pin Enable For LLWU_P22 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE21	Wakeup Pin Enable For LLWU_P21 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE20	Wakeup Pin Enable For LLWU_P20 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.7 LLWU Pin Enable 7 register (LLWU_PE7)

LLWU_PE7 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P27-LLWU_P24.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 6h offset = 4007_C006h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE27		WUPE26		WUPE25		WUPE24	
Reset	0	0	0	0	0	0	0	0

LLWU_PE7 field descriptions

Field	Description
7–6 WUPE27	Wakeup Pin Enable For LLWU_P27 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE26	Wakeup Pin Enable For LLWU_P26 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE25	Wakeup Pin Enable For LLWU_P25 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE24	Wakeup Pin Enable For LLWU_P24 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection

Table continues on the next page...

LLWU_PE7 field descriptions (continued)

Field	Description
	10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.8 LLWU Pin Enable 8 register (LLWU_PE8)

LLWU_PE8 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P31-LLWU_P28.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 7h offset = 4007_C007h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE31		WUPE30		WUPE29		WUPE28	
Reset	0	0	0	0	0	0	0	0

LLWU_PE8 field descriptions

Field	Description
7–6 WUPE31	Wakeup Pin Enable For LLWU_P31 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE30	Wakeup Pin Enable For LLWU_P30 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE29	Wakeup Pin Enable For LLWU_P29 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection

Table continues on the next page...

LLWU_PE8 field descriptions (continued)

Field	Description
	10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE28	Wakeup Pin Enable For LLWU_P28 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.4.9 LLWU Module Enable register (LLWU_ME)

LLWU_ME contains the bits to enable the internal module flag as a wakeup input source for inputs MWUF7-MWUF0.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 8h offset = 4007_C008h

Bit	7	6	5	4	3	2	1	0
Read Write	WUME7	WUME6	WUME5	WUME4	WUME3	WUME2	WUME1	WUME0
Reset	0	0	0	0	0	0	0	0

LLWU_ME field descriptions

Field	Description
7 WUME7	Wakeup Module Enable For Module 7 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
6 WUME6	Wakeup Module Enable For Module 6 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
5 WUME5	Wakeup Module Enable For Module 5 Enables an internal module as a wakeup source input.

Table continues on the next page...

LLWU_ME field descriptions (continued)

Field	Description
	0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
4 WUME4	Wakeup Module Enable For Module 4 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
3 WUME3	Wakeup Module Enable For Module 3 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
2 WUME2	Wakeup Module Enable For Module 2 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
1 WUME1	Wakeup Module Enable for Module 1 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
0 WUME0	Wakeup Module Enable For Module 0 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source

16.4.10 LLWU Pin Flag 1 register (LLWU_PF1)

LLWU_PF1 contains the wakeup flags indicating which wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + 9h offset = 4007_C009h

Bit	7	6	5	4	3	2	1	0
Read	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF1 field descriptions

Field	Description
7 WUF7	Wakeup Flag For LLWU_P7 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF7. 0 LLWU_P7 input was not a wakeup source 1 LLWU_P7 input was a wakeup source
6 WUF6	Wakeup Flag For LLWU_P6 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF6. 0 LLWU_P6 input was not a wakeup source 1 LLWU_P6 input was a wakeup source
5 WUF5	Wakeup Flag For LLWU_P5 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF5. 0 LLWU_P5 input was not a wakeup source 1 LLWU_P5 input was a wakeup source
4 WUF4	Wakeup Flag For LLWU_P4 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF4. 0 LLWU_P4 input was not a wakeup source 1 LLWU_P4 input was a wakeup source
3 WUF3	Wakeup Flag For LLWU_P3 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF3. 0 LLWU_P3 input was not a wakeup source 1 LLWU_P3 input was a wakeup source
2 WUF2	Wakeup Flag For LLWU_P2 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF2. 0 LLWU_P2 input was not a wakeup source 1 LLWU_P2 input was a wakeup source
1 WUF1	Wakeup Flag For LLWU_P1

Table continues on the next page...

LLWU_PF1 field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF1. 0 LLWU_P1 input was not a wakeup source 1 LLWU_P1 input was a wakeup source
0 WUF0	Wakeup Flag For LLWU_P0 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF0. 0 LLWU_P0 input was not a wakeup source 1 LLWU_P0 input was a wakeup source

16.4.11 LLWU Pin Flag 2 register (LLWU_PF2)

LLWU_PF2 contains the wakeup flags indicating which wakeup source caused the MCU to exit or VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Ah offset = 4007_C00Ah

Bit	7	6	5	4	3	2	1	0
Read	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF2 field descriptions

Field	Description
7 WUF15	Wakeup Flag For LLWU_P15 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF15. 0 LLWU_P15 input was not a wakeup source 1 LLWU_P15 input was a wakeup source

Table continues on the next page...

LLWU_PF2 field descriptions (continued)

Field	Description
6 WUF14	Wakeup Flag For LLWU_P14 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF14. 0 LLWU_P14 input was not a wakeup source 1 LLWU_P14 input was a wakeup source
5 WUF13	Wakeup Flag For LLWU_P13 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF13. 0 LLWU_P13 input was not a wakeup source 1 LLWU_P13 input was a wakeup source
4 WUF12	Wakeup Flag For LLWU_P12 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF12. 0 LLWU_P12 input was not a wakeup source 1 LLWU_P12 input was a wakeup source
3 WUF11	Wakeup Flag For LLWU_P11 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF11. 0 LLWU_P11 input was not a wakeup source 1 LLWU_P11 input was a wakeup source
2 WUF10	Wakeup Flag For LLWU_P10 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF10. 0 LLWU_P10 input was not a wakeup source 1 LLWU_P10 input was a wakeup source
1 WUF9	Wakeup Flag For LLWU_P9 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF9. 0 LLWU_P9 input was not a wakeup source 1 LLWU_P9 input was a wakeup source
0 WUF8	Wakeup Flag For LLWU_P8 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF8. 0 LLWU_P8 input was not a wakeup source 1 LLWU_P8 input was a wakeup source

16.4.12 LLWU Pin Flag 3 register (LLWU_PF3)

LLWU_PF3 contains the wakeup flags indicating which wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Bh offset = 4007_C00Bh

Bit	7	6	5	4	3	2	1	0
Read	WUF23	WUF22	WUF21	WUF20	WUF19	WUF18	WUF17	WUF16
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF3 field descriptions

Field	Description
7 WUF23	Wakeup Flag For LLWU_P23 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF23. 0 LLWU_P23 input was not a wakeup source 1 LLWU_P23 input was a wakeup source
6 WUF22	Wakeup Flag For LLWU_P22 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF22. 0 LLWU_P22 input was not a wakeup source 1 LLWU_P22 input was a wakeup source
5 WUF21	Wakeup Flag For LLWU_P21 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF21. 0 LLWU_P21 input was not a wakeup source 1 LLWU_P21 input was a wakeup source
4 WUF20	Wakeup Flag For LLWU_P20

Table continues on the next page...

LLWU_PF3 field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF20. 0 LLWU_P20 input was not a wakeup source 1 LLWU_P20 input was a wakeup source
3 WUF19	Wakeup Flag For LLWU_P19 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF19. 0 LLWU_P19 input was not a wakeup source 1 LLWU_P19 input was a wakeup source
2 WUF18	Wakeup Flag For LLWU_P18 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF18. 0 LLWU_P18 input was not a wakeup source 1 LLWU_P18 input was a wakeup source
1 WUF17	Wakeup Flag For LLWU_P17 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF17. 0 LLWU_P17 input was not a wakeup source 1 LLWU_P17 input was a wakeup source
0 WUF16	Wakeup Flag For LLWU_P16 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF16. 0 LLWU_P16 input was not a wakeup source 1 LLWU_P16 input was a wakeup source

16.4.13 LLWU Pin Flag 4 register (LLWU_PF4)

LLWU_PF4 contains the wakeup flags indicating which wakeup source caused the MCU to exit or VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Ch offset = 4007_C00Ch

Bit	7	6	5	4	3	2	1	0
Read	WUF31	WUF30	WUF29	WUF28	WUF27	WUF26	WUF25	WUF24
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF4 field descriptions

Field	Description
7 WUF31	Wakeup Flag For LLWU_P31 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF31. 0 LLWU_P31 input was not a wakeup source 1 LLWU_P31 input was a wakeup source
6 WUF30	Wakeup Flag For LLWU_P30 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF30. 0 LLWU_P30 input was not a wakeup source 1 LLWU_P30 input was a wakeup source
5 WUF29	Wakeup Flag For LLWU_P29 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF29. 0 LLWU_P29 input was not a wakeup source 1 LLWU_P29 input was a wakeup source
4 WUF28	Wakeup Flag For LLWU_P28 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF28. 0 LLWU_P28 input was not a wakeup source 1 LLWU_P28 input was a wakeup source
3 WUF27	Wakeup Flag For LLWU_P27 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF27. 0 LLWU_P27 input was not a wakeup source 1 LLWU_P27 input was a wakeup source
2 WUF26	Wakeup Flag For LLWU_P26 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF26. 0 LLWU_P26 input was not a wakeup source 1 LLWU_P26 input was a wakeup source
1 WUF25	Wakeup Flag For LLWU_P25

Table continues on the next page...

LLWU_PF4 field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF25. 0 LLWU_P25 input was not a wakeup source 1 LLWU_P25 input was a wakeup source
0 WUF24	Wakeup Flag For LLWU_P24 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF24. 0 LLWU_P24 input was not a wakeup source 1 LLWU_P24 input was a wakeup source

16.4.14 LLWU Module Flag 5 register (LLWU_MF5)

LLWU_MF5 contains the wakeup flags indicating which internal wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

For internal peripherals that are capable of running in a low-leakage power mode, such as a real time clock module or CMP module, the flag from the associated peripheral is accessible as the MWUFx bit. The flag will need to be cleared in the peripheral instead of writing a 1 to the MWUFx bit.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Dh offset = 4007_C00Dh

Bit	7	6	5	4	3	2	1	0
Read	MWUF7	MWUF6	MWUF5	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
Write								
Reset	0	0	0	0	0	0	0	0

LLWU_MF5 field descriptions

Field	Description
7 MWUF7	Wakeup flag For module 7 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.

Table continues on the next page...

LLWU_MF5 field descriptions (continued)

Field	Description
	0 Module 7 input was not a wakeup source 1 Module 7 input was a wakeup source
6 MWUF6	Wakeup flag For module 6 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 6 input was not a wakeup source 1 Module 6 input was a wakeup source
5 MWUF5	Wakeup flag For module 5 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 5 input was not a wakeup source 1 Module 5 input was a wakeup source
4 MWUF4	Wakeup flag For module 4 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 4 input was not a wakeup source 1 Module 4 input was a wakeup source
3 MWUF3	Wakeup flag For module 3 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 3 input was not a wakeup source 1 Module 3 input was a wakeup source
2 MWUF2	Wakeup flag For module 2 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 2 input was not a wakeup source 1 Module 2 input was a wakeup source
1 MWUF1	Wakeup flag For module 1 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 1 input was not a wakeup source 1 Module 1 input was a wakeup source
0 MWUF0	Wakeup flag For module 0 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 0 input was not a wakeup source 1 Module 0 input was a wakeup source

16.4.15 LLWU Pin Filter 1 register (LLWU_FILT1)

LLWU_FILT1 is a control and status register that is used to enable/disable the digital filter 1 features for an external pin.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Eh offset = 4007_C00Eh

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE			FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

LLWU_FILT1 field descriptions

Field	Description
7 FILTF	Filter Detect Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
6–5 FILTE	Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
FILTSEL	Filter Pin Select Selects 1 of the wakeup pins to be muxed into the filter. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter

16.4.16 LLWU Pin Filter 2 register (LLWU_FILT2)

LLWU_FILT2 is a control and status register that is used to enable/disable the digital filter 2 features for an external pin.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4007_C000h base + Fh offset = 4007_C00Fh

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE			FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

LLWU_FILT2 field descriptions

Field	Description
7 FILTF	Filter Detect Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 2 was not a wakeup source 1 Pin Filter 2 was a wakeup source
6–5 FILTE	Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
FILTSEL	Filter Pin Select Selects 1 of the wakeup pins to be muxed into the filter. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter

16.5 Functional description

Thie low-leakage wakeup unit (LLWU) module allows internal peripherals and external input pins as a source of wakeup from low-leakage modes.

It is operational only in VLLSx modes.

The LLWU module contains pin enables for each external pin and internal module. For each external pin, the user can disable or select the edge type for the wakeup with the following options:

- Falling-edge
- Rising-edge
- Either-edge

When an external pin is enabled as a wakeup source, the pin must be configured as an input pin.

The LLWU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup or reset event when the filter function is enabled. Four wakeup detect filters are available for selected external pins. Glitch filtering is not provided on the internal modules.

For internal module interrupts, the WUMEx bit enables the associated module interrupt as a wakeup source.

16.5.1 VLLS modes

For any wakeup from VLLS, recovery is always via a reset flow and RCM_SRS[WAKEUP] is set indicating the low-leakage mode was active. State retention data is lost and I/O will be restored after PMC_REGSC[ACKISO] has been written.

A VLLS exit event due to $\overline{\text{RESET}}$ pin assertion causes an exit via a system reset. State retention data is lost and the I/O states immediately return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

16.5.2 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering VLLSx mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least five LPO clock cycles before entering VLLSx mode to allow the filter to initialize.

NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing PMC_REGSC[ACKISO]. In particular, pin configuration for enabled LLWU wake-up pins must be restored to avoid any LLWU flag from being falsely set when PMC_REGSC[ACKISO] is cleared.

The signal selected as a wake-up source pin must be a digital pin, as selected in the pin mux control.

Chapter 17
Power Management Controller (PMC)

17.1 Introduction

The power management controller (PMC) contains the internal voltage regulator, power on reset (POR), low voltage detect system (LVD), and high voltage detect system (HVD).

See AN4503: Power Management for Kinetis MCUs for further details on using the PMC.

17.2 Features

A list of included PMC features can be found here.

- Internal voltage regulator
- Active POR providing brown-out detect
- Low-voltage detect supporting two low-voltage trip points with four warning levels per trip point

17.3 Low-voltage detect (LVD) system

This device includes a system to guard against low-voltage conditions. This protects memory contents and controls MCU system states during supply voltage variations.

The system is comprised of a power-on reset (POR) circuit and a LVD circuit with a user-selectable trip voltage: high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by LVDSC1[LVDV]. The LVD is disabled upon entering VLPx and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The Low Voltage Detect Flag in the Low Voltage Status and Control 1 Register (LVDSC1[LVDF]) operates in a level sensitive manner. LVDSC1[LVDF] is set

when the supply voltage falls below the selected trip point (VLVD). LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC1[LVDF] remains set.

- The Low Voltage Warning Flag (LVWF) in the Low Voltage Status and Control 2 Register (LVDSC2[LVWF]) operates in a level sensitive manner. LVDSC2[LVWF] is set when the supply voltage falls below the selected monitor trip point (VLVW). LVDSC2[LVWF] is cleared by writing one to LVDSC2[LVWACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC2[LVWF] remains set.

17.3.1 LVD reset operation

By setting LVDSC1[LVDRE], the LVD generates a reset upon detection of a low-voltage condition. The low-voltage detection threshold is determined by LVDSC1[LVDV]. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD field in the SRS register of the RCM module (RCM_SRS[LVD]) is set following an LVD or power-on reset.

17.3.2 LVD interrupt operation

By configuring the LVD circuit for interrupt operation (LVDSC1[LVDIE] set and LVDSC1[LVDRE] clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK].

17.3.3 Low-voltage warning (LVW) interrupt operation

The LVD system contains a Low-Voltage Warning Flag (LVWF) in the Low Voltage Detect Status and Control 2 Register to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by setting LVDSC2[LVWIE]. If enabled, an LVW interrupt request occurs when LVDSC2[LVWF] is set. LVDSC2[LVWF] is cleared by writing 1 to LVDSC2[LVWACK].

LVDSC2[LVWV] selects one of the four trip voltages:

- Highest: V_{LVW4}

- Two mid-levels: V_{LVW3} and V_{LVW2}
- Lowest: V_{LVW1}

17.4 I/O retention

When in VLLS modes, the I/O states are held on a wake-up event (with the exception of wake-up by reset event) until the wake-up has been acknowledged via a write to REGSC[ACKISO]. In the case of VLLS exit via a RESET pin, the I/O are released and default to their reset state. In this case, no write to REGSC[ACKISO] is needed.

17.5 Memory map and register descriptions

Details about the PMC registers can be found here.

NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

The PMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

PMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_D000	Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)	8	R/W	10h	17.5.1/306
4007_D001	Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)	8	R/W	00h	17.5.2/307
4007_D002	Regulator Status And Control register (PMC_REGSC)	8	R/W	04h	17.5.3/308

17.5.1 Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)

This register contains status and control bits to support the low voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC1 settings. To protect systems that must have LVD always on, configure the Power Mode Protection (PMPROT) register of the SMC module (SMC_PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVDV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 0h offset = 4007_D000h

Bit	7	6	5	4	3	2	1	0
Read	LVDF	0	LVDIE	LVDRE	0		LVDV	
Write		LVDACK						
Reset	0	0	0	1	0	0	0	0

PMC_LVDSC1 field descriptions

Field	Description
7 LVDF	Low-Voltage Detect Flag This read-only status field indicates a low-voltage detect event. 0 Low-voltage event not detected 1 Low-voltage event detected
6 LVDACK	Low-Voltage Detect Acknowledge This write-only field is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Reads always return 0.
5 LVDIE	Low-Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1

Table continues on the next page...

PMC_LVDSC1 field descriptions (continued)

Field	Description
4 LVDRE	Low-Voltage Detect Reset Enable This write-once bit enables LVDF events to generate a hardware reset. Additional writes are ignored. 0 LVDF does not generate hardware resets 1 Force an MCU reset when LVDF = 1
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVDV	Low-Voltage Detect Voltage Select Selects the LVD trip point voltage (V_{LVD}). 00 Low trip point selected ($V_{LVD} = V_{LVDL}$) 01 High trip point selected ($V_{LVD} = V_{LVDH}$) 10 Reserved 11 Reserved

17.5.2 Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)

This register contains status and control bits to support the low voltage warning function.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC2 settings.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVW trip voltages depend on LVWV and LVDV.

NOTE

LVWV is reset solely on a POR Only event. The other fields of the register are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 1h offset = 4007_D001h

Bit	7	6	5	4	3	2	1	0
Read	LVWF	0	LVWIE	0			LVWV	
Write		LVWACK						
Reset	0	0	0	0	0	0	0	0

PMC_LVDSC2 field descriptions

Field	Description
7 LVWF	Low-Voltage Warning Flag This read-only status field indicates a low-voltage warning event. LVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. LVWF may be 1 after power-on reset, therefore, to use LVW interrupt function, before enabling LVWIE, LVWF must be cleared by writing LVWACK first. 0 Low-voltage warning event not detected 1 Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge This write-only field is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.
5 LVWIE	Low-Voltage Warning Interrupt Enable Enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF = 1
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVWV	Low-Voltage Warning Voltage Select Selects the LVW trip point voltage (V_{LVW}). The actual voltage for the warning depends on LVDSC1[LVDV]. 00 Low trip point selected ($V_{LVW} = V_{LVW1}$) 01 Mid 1 trip point selected ($V_{LVW} = V_{LVW2}$) 10 Mid 2 trip point selected ($V_{LVW} = V_{LVW3}$) 11 High trip point selected ($V_{LVW} = V_{LVW4}$)

17.5.3 Regulator Status And Control register (PMC_REGSC)

The PMC contains an internal voltage regulator. The voltage regulator design uses a bandgap reference that is also available through a buffer as input to certain internal peripherals, such as the CMP and ADC. The internal regulator provides a status bit (REGONS) indicating the regulator is in run regulation.

NOTE

This register is reset on Chip Reset Not VLLS and by reset types that trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_D000h base + 2h offset = 4007_D002h

Bit	7	6	5	4	3	2	1	0
Read	0	0	Reserved	BGEN	ACKISO	REGONS	Reserved	BGBE
Write					w1c			
Reset	0	0	0	0	0	1	0	0

PMC_REGSC field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved.
4 BGEN	Bandgap Enable In VLPx Operation BGEN controls whether the bandgap is enabled in lower power modes of operation (VLPx, and VLLSx). When on-chip peripherals require the bandgap voltage reference in low power modes of operation, set BGEN to continue to enable the bandgap operation. **NOTE:** When the bandgap voltage reference is not needed in low power modes, clear BGEN to avoid excess power consumption. 0 Bandgap voltage reference is disabled in VLPx , and VLLSx modes. 1 Bandgap voltage reference is enabled in VLPx , and VLLSx modes.
3 ACKISO	Acknowledge Isolation Reading this field indicates whether certain peripherals and the I/O pads are in a latched state as a result of having been in a VLLS mode. Writing 1 to this field when it is set releases the I/O pads and certain peripherals to their normal run mode state. **NOTE:** After recovering from a VLLS mode, user should restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins should be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared. 0 Peripherals and I/O pads are in normal run state. 1 Certain peripherals and I/O pads are in an isolated and latched state.
2 REGONS	Regulator In Run Regulation Status This read-only field provides the current status of the internal voltage regulator. 0 Regulator is in stop regulation or in transition to/from it 1 Regulator is in run regulation
1 Reserved	This field is reserved. **NOTE:** This reserved bit must remain cleared (set to 0).
0 BGBE	Bandgap Buffer Enable Enables the bandgap buffer. 0 Bandgap buffer not enabled 1 Bandgap buffer enabled

Chapter 18
Reset Control Module (RCM)

18.1 Introduction

Information found here describes the registers of the Reset Control Module (RCM). The RCM implements many of the reset functions for the chip. See the chip's reset chapter for more information.

See AN4503: Power Management for Kinetis MCUs for further details on using the RCM.

18.2 Reset memory map and register descriptions

The RCM Memory Map/Register Definition can be found here.

The Reset Control Module (RCM) registers provide reset status information and reset filter control.

NOTE

The RCM registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

RCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_B000	System Reset Status Register 0 (RCM_SRS0)	8	R	82h	18.2.1/312
4007_B001	System Reset Status Register 1 (RCM_SRS1)	8	R	00h	18.2.2/313
4007_B004	Reset Pin Filter Control register (RCM_RPFC)	8	R/W	00h	18.2.3/315
4007_B005	Reset Pin Filter Width register (RCM_RPFW)	8	R/W	00h	18.2.4/316

Table continues on the next page...

RCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_B008	Sticky System Reset Status Register 0 (RCM_SSRS0)	8	R/W	82h	18.2.5/317
4007_B009	Sticky System Reset Status Register 1 (RCM_SSRS1)	8	R/W	00h	18.2.6/318

18.2.1 System Reset Status Register 0 (RCM_SRS0)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:
- POR (including LVD) — 0x82
- LVD (without POR) — 0x02
- VLLS mode wakeup due to $\overline{\text{RESET}}$ pin assertion — 0x41
- VLLS mode wakeup due to other wakeup sources — 0x01
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_B000h base + 0h offset = 4007_B000h

Bit	7	6	5	4	3	2	1	0
Read	POR	PIN	WDOG	0	LOL	LOC	LVD	WAKEUP
Write								
Reset	1	0	0	0	0	0	1	0

RCM_SRS0 field descriptions

Field	Description
7 POR	Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 PIN	External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog

Table continues on the next page...

RCM_SRS0 field descriptions (continued)

Field	Description
	Indicates a reset has been caused by the watchdog timer timing out. This reset source can be blocked by disabling the watchdog. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LOL	Loss-of-Lock Reset Indicates a reset has been caused by a loss of lock in the MCG PLL. See the MCG description for information on the loss-of-clock event. 0 Reset not caused by a loss of lock in the PLL 1 Reset caused by a loss of lock in the PLL
2 LOC	Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The MCG clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed MCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 LVD	Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 WAKEUP	Low Leakage Wakeup Reset Indicates a reset has been caused by an enabled LLWU module wakeup source while the chip was in a low leakage mode. Any enabled wakeup source in a VLLSx mode causes a reset. This bit is cleared by any reset except WAKEUP. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source

18.2.2 System Reset Status Register 1 (RCM_SRS1)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x00
- LVD (without POR) — 0x00

- VLLS mode wakeup — 0x00
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_B000h base + 1h offset = 4007_B001h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SACKERR	0	MDM_AP	SW	LOCKUP	0
Write								
Reset	0	0	0	0	0	0	0	0

RCM_SRS1 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SACKERR	Stop Mode Acknowledge Error Reset Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 MDM_AP	MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit
2 SW	Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
1 LOCKUP	Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

18.2.3 Reset Pin Filter Control register (RCM_RPFC)

NOTE

The reset values of bits 2-0 are for Chip POR only. They are unaffected by other reset types.

NOTE

The bus clock filter is reset when disabled or when entering stop mode. The LPO filter is reset when disabled .

Address: 4007_B000h base + 4h offset = 4007_B004h

Bit	7	6	5	4	3	2	1	0
Read	0					RSTFLTSS	RSTFLTSRW	
Write								
Reset	0	0	0	0	0	0	0	0

RCM_RPFC field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RSTFLTSS	Reset Pin Filter Select in Stop Mode Selects how the reset pin filter is enabled in Stop and VLPS modes , and also during VLLS mode. On exit from VLLS mode, this bit should be reconfigured before clearing PMC_REGSC[ACKISO]. 0 All filtering disabled 1 LPO clock filter enabled
RSTFLTSRW	Reset Pin Filter Select in Run and Wait Modes Selects how the reset pin filter is enabled in run and wait modes. 00 All filtering disabled 01 Bus clock filter enabled for normal operation 10 LPO clock filter enabled for normal operation 11 Reserved

18.2.4 Reset Pin Filter Width register (RCM_RPFW)

NOTE

The reset values of the bits in the RSTFLTSEL field are for Chip POR only. They are unaffected by other reset types.

Address: 4007_B000h base + 5h offset = 4007_B005h

Bit	7	6	5	4	3	2	1	0
Read	0				RSTFLTSEL			
Write								
Reset	0	0	0	0	0	0	0	0

RCM_RPFW field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RSTFLTSEL	Reset Pin Filter Bus Clock Select Selects the reset pin bus clock filter width. 00000 Bus clock filter count is 1 00001 Bus clock filter count is 2 00010 Bus clock filter count is 3 00011 Bus clock filter count is 4 00100 Bus clock filter count is 5 00101 Bus clock filter count is 6 00110 Bus clock filter count is 7 00111 Bus clock filter count is 8 01000 Bus clock filter count is 9 01001 Bus clock filter count is 10 01010 Bus clock filter count is 11 01011 Bus clock filter count is 12 01100 Bus clock filter count is 13 01101 Bus clock filter count is 14 01110 Bus clock filter count is 15 01111 Bus clock filter count is 16 10000 Bus clock filter count is 17 10001 Bus clock filter count is 18 10010 Bus clock filter count is 19 10011 Bus clock filter count is 20 10100 Bus clock filter count is 21 10101 Bus clock filter count is 22 10110 Bus clock filter count is 23 10111 Bus clock filter count is 24 11000 Bus clock filter count is 25

Table continues on the next page...

RCM_RPFW field descriptions (continued)

Field	Description
	11001 Bus clock filter count is 26 11010 Bus clock filter count is 27 11011 Bus clock filter count is 28 11100 Bus clock filter count is 29 11101 Bus clock filter count is 30 11110 Bus clock filter count is 31 11111 Bus clock filter count is 32

18.2.5 Sticky System Reset Status Register 0 (RCM_SSRS0)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_B000h base + 8h offset = 4007_B008h

Bit	7	6	5	4	3	2	1	0
Read	SPOR	SPIN	SWDOG	0	SLOL	0	SLVD	SWAKEUP
Write	w1c	w1c	w1c		w1c		w1c	w1c
Reset	1	0	0	0	0	0	1	0

RCM_SSRS0 field descriptions

Field	Description
7 SPOR	Sticky Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 SPIN	Sticky External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 SWDOG	Sticky Watchdog Indicates a reset has been caused by the watchdog timer timing out.This reset source can be blocked by disabling the watchdog.

Table continues on the next page...

RCM_SSRS0 field descriptions (continued)

Field	Description
	0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SLOL	Sticky Loss-of-Lock Reset Indicates a reset has been caused by a loss of lock in the MCG PLL. See the MCG description for information on the loss-of-clock event. 0 Reset not caused by a loss of lock in the PLL 1 Reset caused by a loss of lock in the PLL
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 SLVD	Sticky Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 SWAKEUP	Sticky Low Leakage Wakeup Reset Indicates a reset has been caused by an enabled LLWU modulewakeup source while the chip was in a low leakage mode. Any enabled wakeup source in a VLLSx mode causes a reset. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source

18.2.6 Sticky System Reset Status Register 1 (RCM_SSRS1)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_B000h base + 9h offset = 4007_B009h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SSACKERR	0	SMDM_AP	SSW	SLOCKUP	0
Write			w1c		w1c	w1c	w1c	
Reset	0	0	0	0	0	0	0	0

RCM_SSRS1 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SSACKERR	Sticky Stop Mode Acknowledge Error Reset Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SMDM_AP	Sticky MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit
2 SSW	Sticky Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
1 SLOCKUP	Sticky Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Chapter 19
Miscellaneous Control Module (MCM)

19.1 Introduction

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

19.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration
- Crossbar master arbitration policy selection
- Flash controller speculation buffer and cache configurations
- Master attributes configurations

19.2 Memory map/register descriptions

The memory map and register descriptions found here describe the registers using byte addresses. The registers can be written only when in supervisor mode.

NOTE

For MCM registers, any of these behaviors will cause hardfault exception– access any register in user mode, access PID register by halfword or byte, write to read-only registers.

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_3008	Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)	16	R	0007h	19.2.1/322
F000_300A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	0005h	19.2.2/323
F000_300C	Platform Control Register (MCM_PLACR)	32	R/W	0000_0000h	19.2.3/323
F000_3030	Process ID register (MCM_PID)	32	R/W	0000_0000h	19.2.4/326
F000_3040	Compute Operation Control Register (MCM_CPO)	32	R/W	0000_0000h	19.2.5/327
F000_3080	Master Attribute Configuration Register (MCM_MATCR0)	32	R/W	0000_0000h	19.2.6/328

19.2.1 Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)

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

Address: F000_3000h base + 8h offset = F000_3008h

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

MCM_PLASC field descriptions

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

19.2.2 Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)

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

Address: F000_3000h base + Ah offset = F000_300Ah

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

MCM_PLAMC field descriptions

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

19.2.3 Platform Control Register (MCM_PLACR)

The PLACR register selects the arbitration policy for the crossbar masters and configures the flash memory controller.

The speculation buffer and cache in the flash memory controller is configurable via PLACR[15:10].

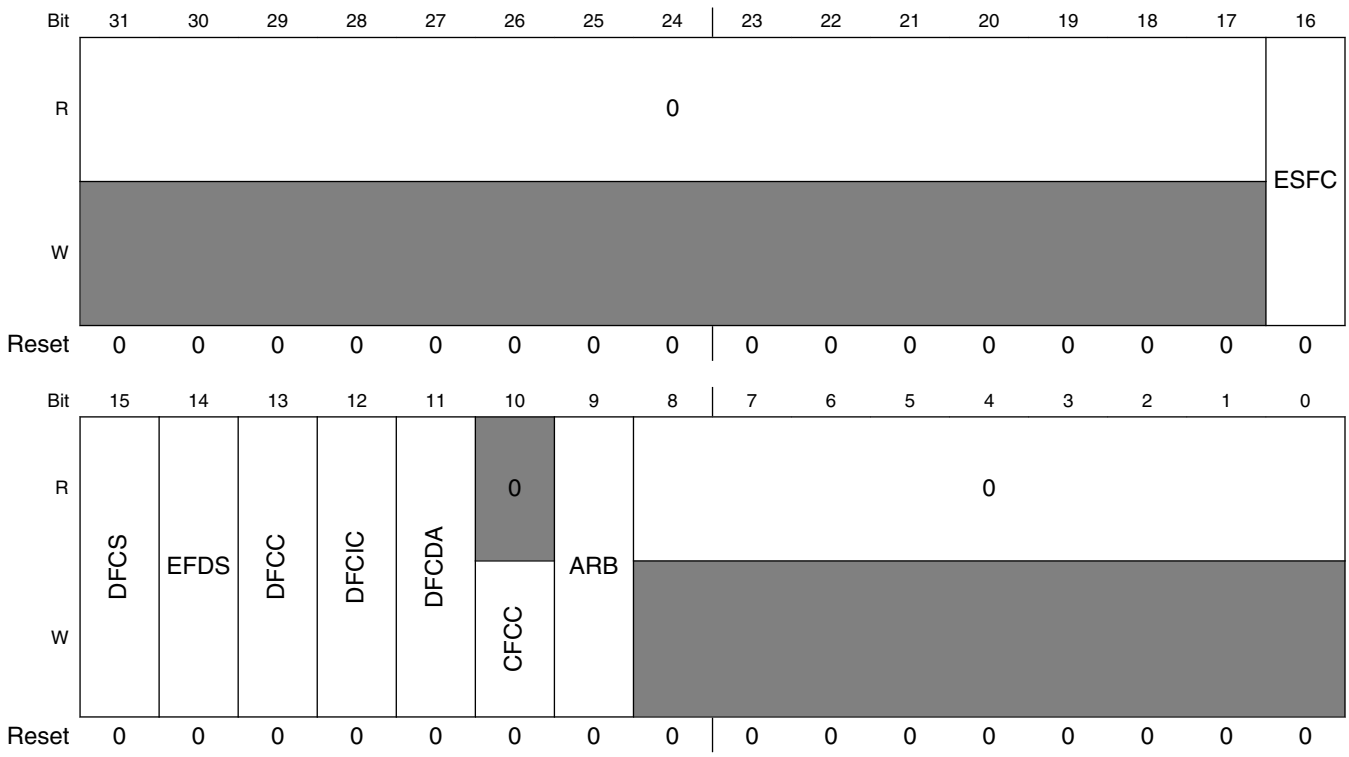
The speculation buffer is enabled only for instructions after reset. It is possible to have these states for the speculation buffer:

DFCS	EFDS	Description
0	0	Speculation buffer is on for instruction and off for data.
0	1	Speculation buffer is on for instruction and on for data.
1	X	Speculation buffer is off.

The cache in flash controller is enabled and caching both instruction and data type fetches after reset. It is possible to have these states for the cache:

DFCC	DFCIC	DFCDA	Description
0	0	0	Cache is on for both instruction and data.
0	0	1	Cache is on for instruction and off for data.
0	1	0	Cache is off for instruction and on for data.
0	1	1	Cache is off for both instruction and data.
1	X	X	Cache is off.

Address: F000_3000h base + Ch offset = F000_300Ch



MCM_PLACR field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ESFC	Enable Stalling Flash Controller Enables stalling flash controller when flash is busy.

Table continues on the next page...

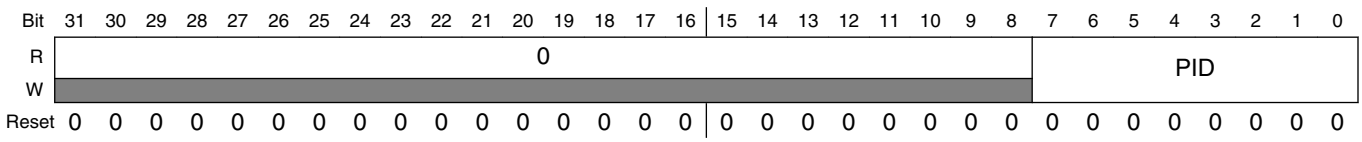
MCM_PLACR field descriptions (continued)

Field	Description
	When software needs to access the flash memory while a flash memory resource is being manipulated by a flash command, software can enable a stall mechanism to avoid a read collision. The stall mechanism allows software to execute code from the same block on which flash operations are being performed. However, software must ensure the sector the flash operations are being performed on is not the same sector from which the code is executing. ESFC enables the stall mechanism. This bit must be set only just before the flash operation is executed and must be cleared when the operation completes. 0 Disable stalling flash controller when flash is busy. 1 Enable stalling flash controller when flash is busy.
15 DFCS	Disable Flash Controller Speculation Disables flash controller speculation. 0 Enable flash controller speculation. 1 Disable flash controller speculation.
14 EFDS	Enable Flash Data Speculation Enables flash data speculation. 0 Disable flash data speculation. 1 Enable flash data speculation.
13 DFCC	Disable Flash Controller Cache Disables flash controller cache. 0 Enable flash controller cache. 1 Disable flash controller cache.
12 DFCIC	Disable Flash Controller Instruction Caching Disables flash controller instruction caching. 0 Enable flash controller instruction caching. 1 Disable flash controller instruction caching.
11 DFCDA	Disable Flash Controller Data Caching Disables flash controller data caching. 0 Enable flash controller data caching 1 Disable flash controller data caching.
10 CFCC	Clear Flash Controller Cache Writing a 1 to this field clears the cache. Writing a 0 to this field is ignored. This field always reads as 0.
9 ARB	Arbitration select 0 Fixed-priority arbitration for the crossbar masters 1 Round-robin arbitration for the crossbar masters
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

19.2.4 Process ID register (MCM_PID)

This register drives the M0_PID value in the Memory Protection Unit (MPU). System software loads this register before passing control to a given user mode process. If the PID of the process does not match the value in this register, a bus error occurs. See the MPU chapter for more details.

Address: F000_3000h base + 30h offset = F000_3030h



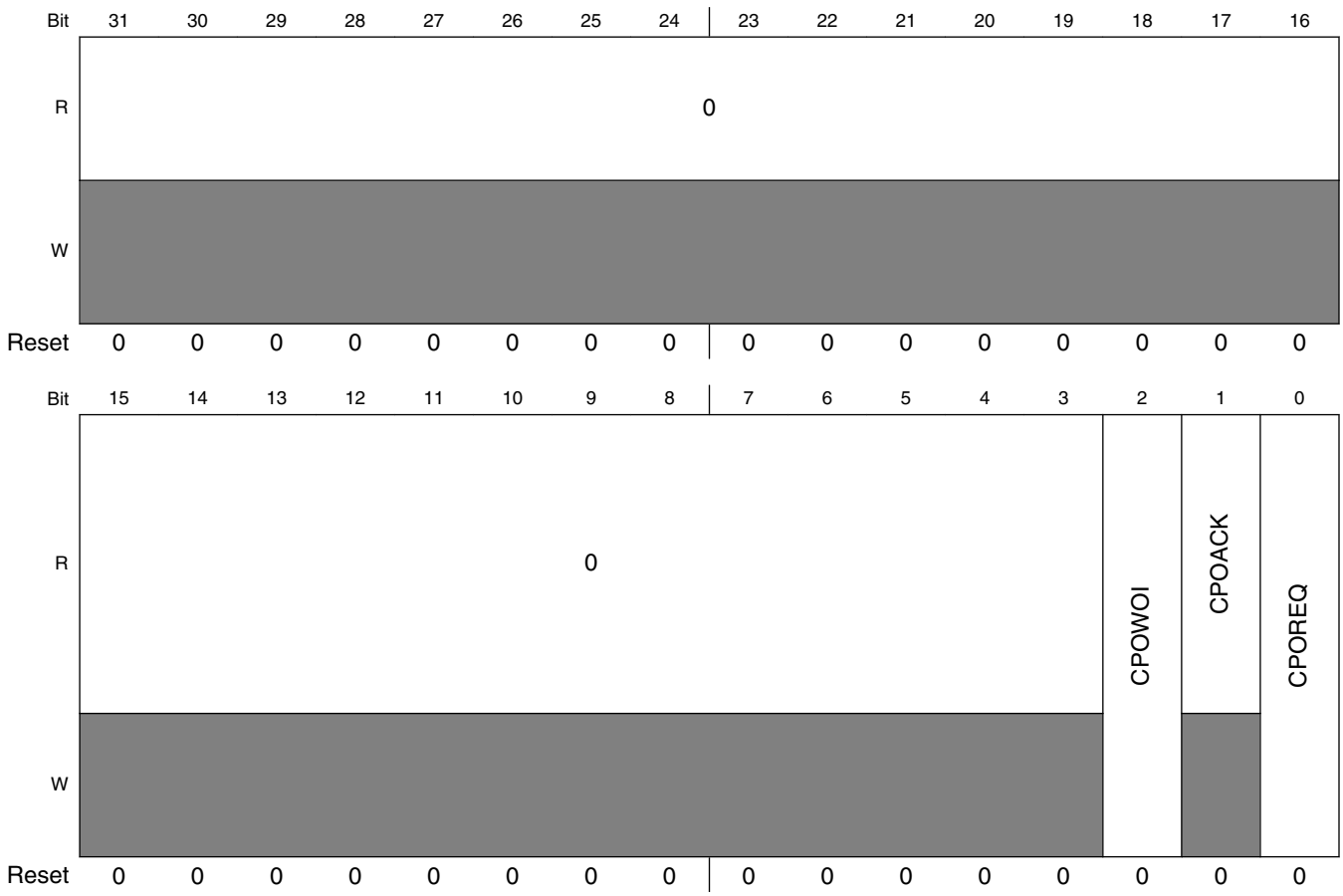
MCM_PID field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PID	M0_PID For MPU Drives the M0_PID value in the MPU. 00h Reserved for privileged secure tasks 01h - 7Fh Allocated for user secure tasks 80h - FFh Allocated for user nonsecure tasks

19.2.5 Compute Operation Control Register (MCM_CPO)

This register controls the Compute Operation.

Address: F000_3000h base + 40h offset = F000_3040h



MCM_CPO field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CPOWOI	Compute Operation Wake-up on Interrupt 0 No effect. 1 When set, the CPOREQ is cleared on any interrupt or exception vector fetch.
1 CPOACK	Compute Operation Acknowledge 0 Compute operation entry has not completed or compute operation exit has completed. 1 Compute operation entry has completed or compute operation exit has not completed.
0 CPOREQ	Compute Operation Request This bit is auto-cleared by vector fetching if CPOWOI = 1.

Table continues on the next page...

MCM_CPO field descriptions (continued)

Field	Description
	0 Request is cleared. 1 Request Compute Operation.

19.2.6 Master Attribute Configuration Register (MCM_MATCR*n*)

This 32-bit register consists of four 8-bit attribute configuration register fields for each system bus master. Each MATCRn register field allows the corresponding bus master's secure/nonsecure and privileged/user attributes to be forced to a static state, or simply enable the attributes generated by the master module. This information is then used on a per bus transaction basis to control access. See the MPU chapter for more details.

For all ATCn values, if the state indicates privileged (either statically or via the master module's attribute), then the other state indicator is forced to secure.

NOTE

Writes to ATC0 configure the processor's master attributes, and software must take special care with this value. Typically, the processor would use an ATC0 value of {0b11- or 0b00-} because the other values can create a configuration where access to system level resources is not allowed.

Each MATCRn register field also contains a lock bit (RO) that may be set to disable writes to the register, preserving the configured state until the next system reset.

Address: F000_3000h base + 80h offset + (1d × i), where i=0d to 0d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								RO2	0				ATC2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RO0	0				ATC0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCM_MATCR*n* field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 RO2	Read-Only Master n

Table continues on the next page...

MCM_MATCR*n* field descriptions (continued)

Field	Description
	This register bit provides a mechanism to lock the configuration state defined by ATCn. Once asserted, this bit remains set and attempted writes to the ATCn are ignored until the next system reset clears the flag. 0 Writes to the ATCn are allowed. 1 Writes to the ATCn are ignored.
22–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 ATC2	Attribute Configuration Master n This 3-bit field defines the privileged/user and secure/nonsecure attribute configurations. 00x Master attributes are statically forced to {privileged, secure}. 010 Master attributes are statically forced to {user, secure}. 011 Master attributes are statically forced to {user, nonsecure}. 100 Enable master attribute {privileged or user} and statically force {secure}. 101 Enable master attribute {privileged or user} and statically force {nonsecure}. 11x Enable master attribute {privileged or user, secure or nonsecure}
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 RO0	Read-Only Master n This register bit provides a mechanism to lock the configuration state defined by ATCn. Once asserted, this bit remains set and attempted writes to the ATCn register are ignored until the next system reset clears the flag. 0 Writes to the ATCn are allowed. 1 Writes to the ATCn are ignored.
6–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ATC0	Attribute Configuration Master n This 3-bit field defines the privileged/user and secure/nonsecure attribute configurations. 00x Master attributes are statically forced to {privileged, secure}. 010 Master attributes are statically forced to {user, secure}. 011 Master attributes are statically forced to {user, nonsecure}. 100 Enable master attribute {privileged or user} and statically force {secure}. 101 Enable master attribute {privileged or user} and statically force {nonsecure}. 11x Enable master attribute {privileged or user, secure or nonsecure}

Chapter 20
Bit Manipulation Engine (BME)

20.1 Introduction

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space in Cortex-M0+ based microcontrollers.

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

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

BME decorated references are only available on system bus transactions generated by the processor core and targeted at the standard 512 KB peripheral address space based at 0x4000_0000[1]. The decoration semantic is embedded into address bits[28:19], creating a 448 MB space at addresses 0x4400_0000–0x5FFF_FFFF for AIPS; these bits are stripped out of the actual address sent to the peripheral bus controller and used by the BME to define and control its operation.

1. To be perfectly accurate, the peripheral address space occupies a 516 KB region: 512 KB based at 0x4000_0000 plus a 4 KB space based at 0x400F_F000 for GPIO accesses. This organization provides compatibility with the Kinetis K Family. Attempted accesses to the memory space located between 0x4008_0000 - 0x400F_EFFF are error terminated due to an illegal address.

20.1.1 Overview

The following figure is a generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers.

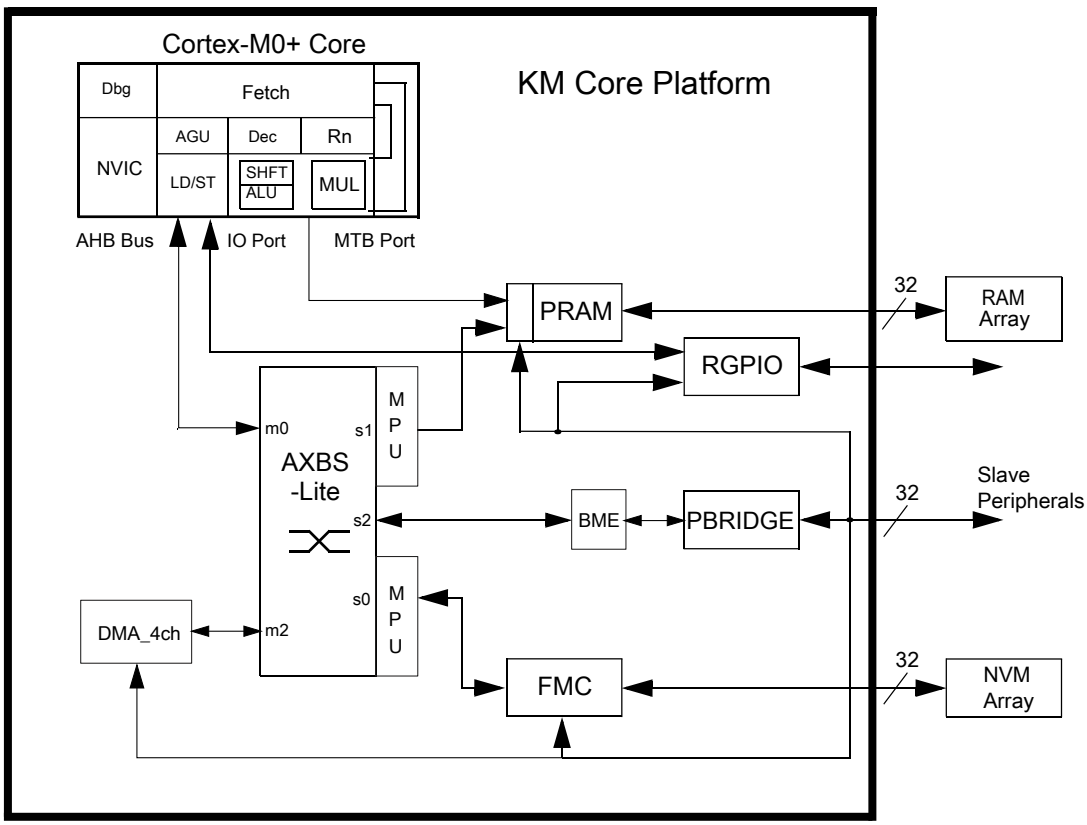


Figure 20-1. Cortex-M0+ core platform block diagram

As shown in the block diagram, the BME module interfaces to a switch AHB slave port as its primary input and sources an AHB bus output to the Peripheral Bridge (PBRIDGE) controller. The BME hardware microarchitecture is a 2-stage pipeline design matching the protocol of the AMBA-AHB system bus interfaces. The PBRIDGE module converts the AHB system bus protocol into the IPS/APB protocol used by the attached slave peripherals.

20.1.2 Features

The key features of the BME include:

- Lightweight implementation of decorated storage for selected address spaces

- Additional access semantics encoded into the reference address
- Resides between a switch slave port and a peripheral bridge bus controller
- Two-stage pipeline design matching the AHB system bus protocol
- Combinationally passes non-decorated accesses to peripheral bridge bus controller
- Conversion of decorated loads and stores from processor core into atomic read-modify-writes
- Decorated loads support unsigned bit field extracts, load-and-{set,clear} 1-bit operations
- Decorated stores support bit field inserts, logical AND, OR, and XOR operations
- Support for byte, halfword and word-sized decorated operations
- Supports minimum signal toggling on AHB output bus to reduce power dissipation

20.1.3 Modes of operation

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

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

20.2 Memory map and register definition

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

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

The peripheral address space occupies a 516 KB region: 512 KB based at 0x4000_0000 plus a 4 KB space based at 0x400F_F000 for GPIO accesses; the decorated address space is mapped to the 448 MB region located at 0x4400_0000–0x5FFF_FFFF.

20.3 Functional description

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

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

Consider decorated store operations first, then decorated loads.

20.3.1 BME decorated stores

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

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

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

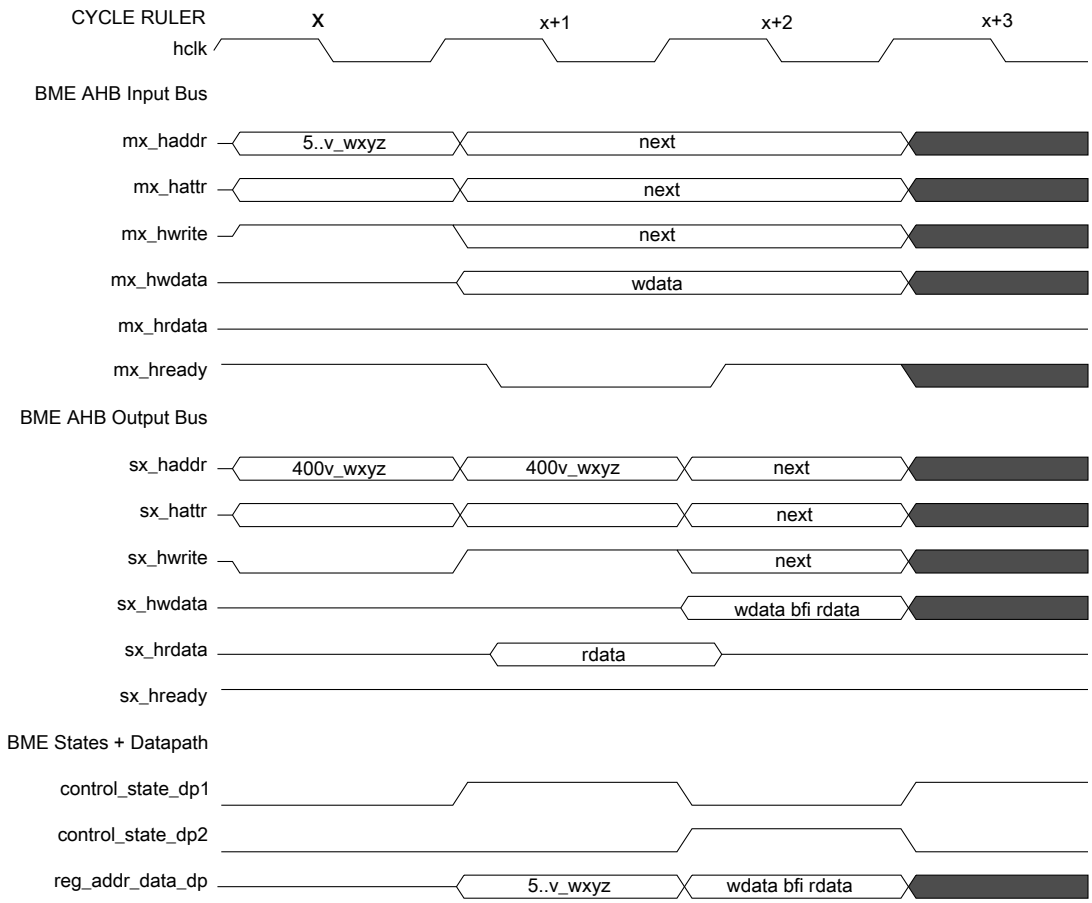


Figure 20-2. Decorated store: bit field insert timing diagram

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

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

NOTE

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

20.3.1.1 Decorated store logical AND (AND)

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

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

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

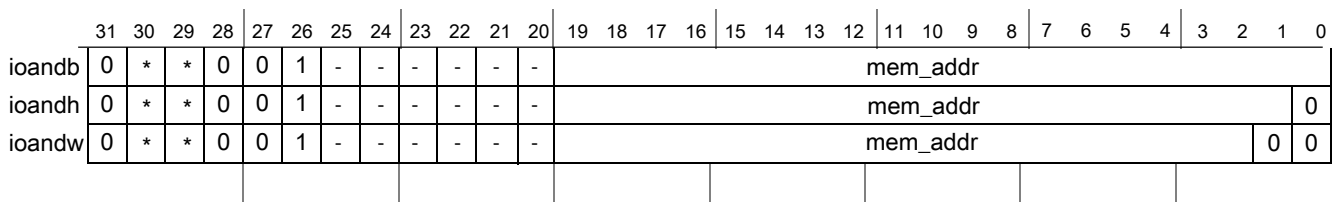


Figure 20-3. Decorated store address: logical AND

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

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

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

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

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

Table 20-1. Cycle definitions of decorated store: logical AND

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert	Recirculate captured addr + attr to memory as slave_wt	<next>

Table continues on the next page...

Table 20-1. Cycle definitions of decorated store: logical AND (continued)

Pipeline stage	Cycle		
	x	x+1	x+2
	master_wt to slave_rd; Capture address, attributes		
BME AHB_dp	<previous>	Perform memory read; Form (rdata & wdata) and capture destination data in register	Perform write sending registered data to memory

20.3.1.2 Decorated store logical OR (OR)

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

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

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

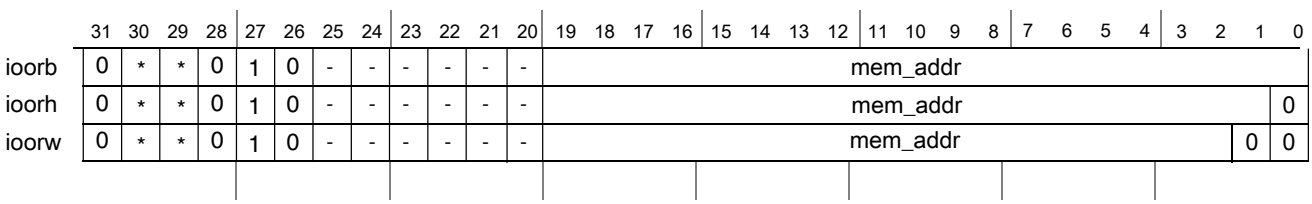


Figure 20-4. Decorated address store: logical OR

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

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

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

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

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

Table 20-2. Cycle definitions of decorated store: logical OR

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

20.3.1.3 Decorated store logical XOR (XOR)

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

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

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

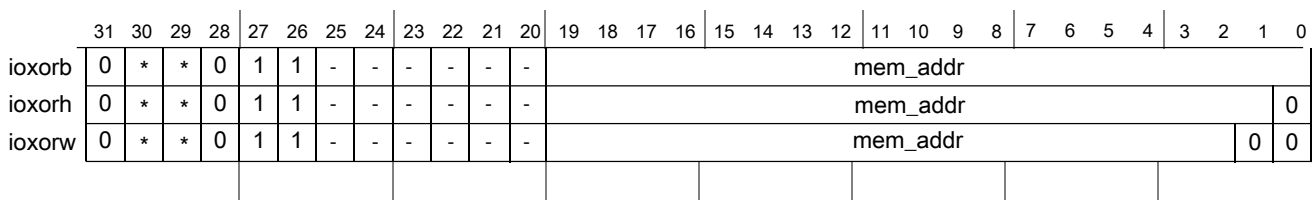


Figure 20-5. Decorated address store: logical XOR

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

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

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

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

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

Table 20-3. Cycle definitions of decorated store: logical XOR

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

20.3.1.4 Decorated store bit field insert (BFI)

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

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

NOTE

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

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

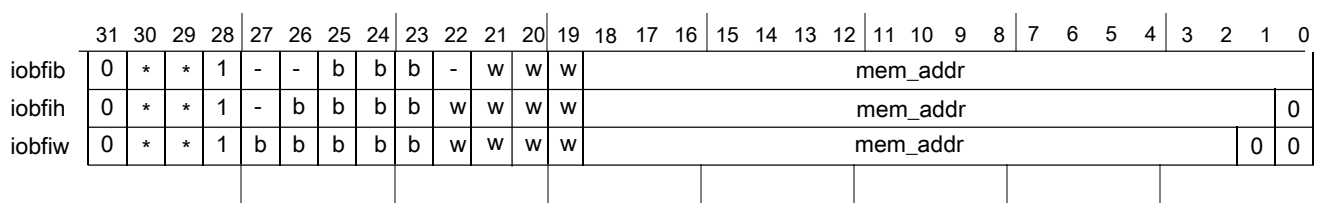


Figure 20-6. Decorated address store: bit field insert

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

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

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

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

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

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

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

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

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

Table 20-4. Cycle definitions of decorated store: bit field insert

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

20.3.2 BME decorated loads

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

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

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

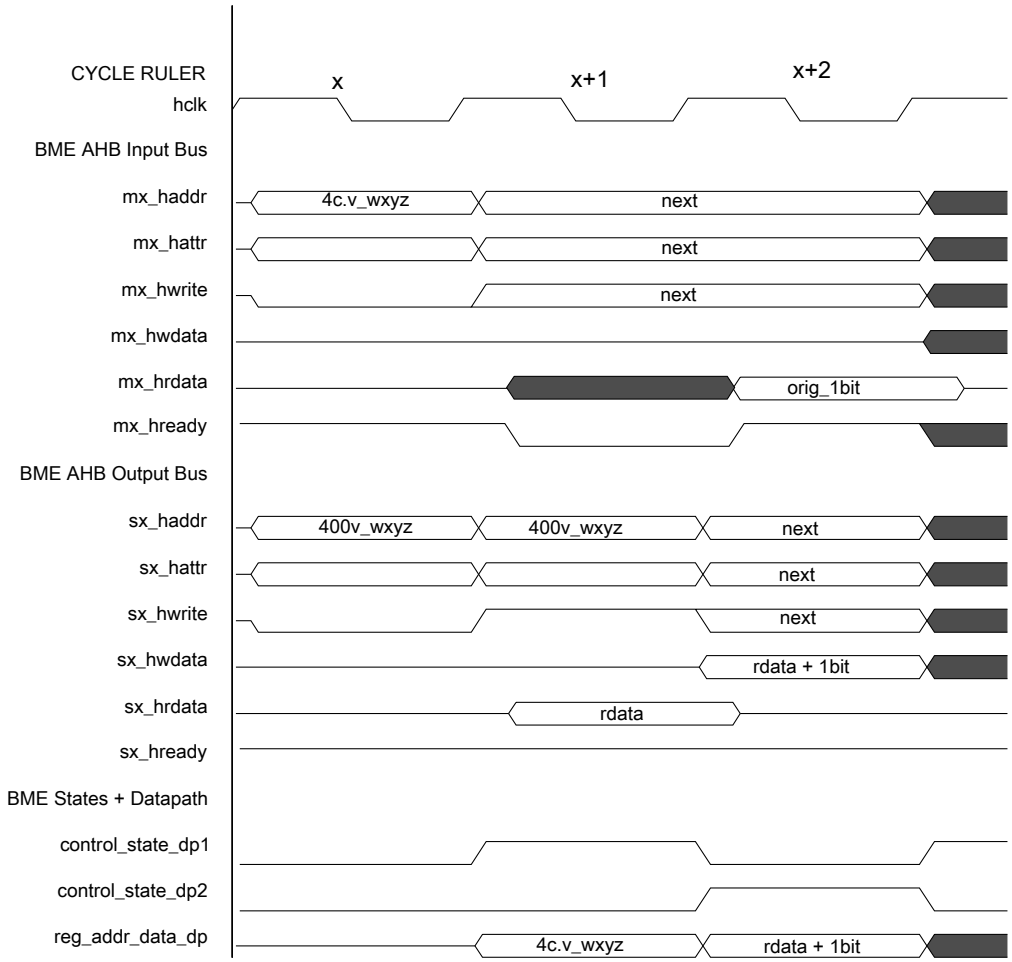


Figure 20-7. Decorated load: load-and-set 1-bit field insert timing diagram

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

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

NOTE

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

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

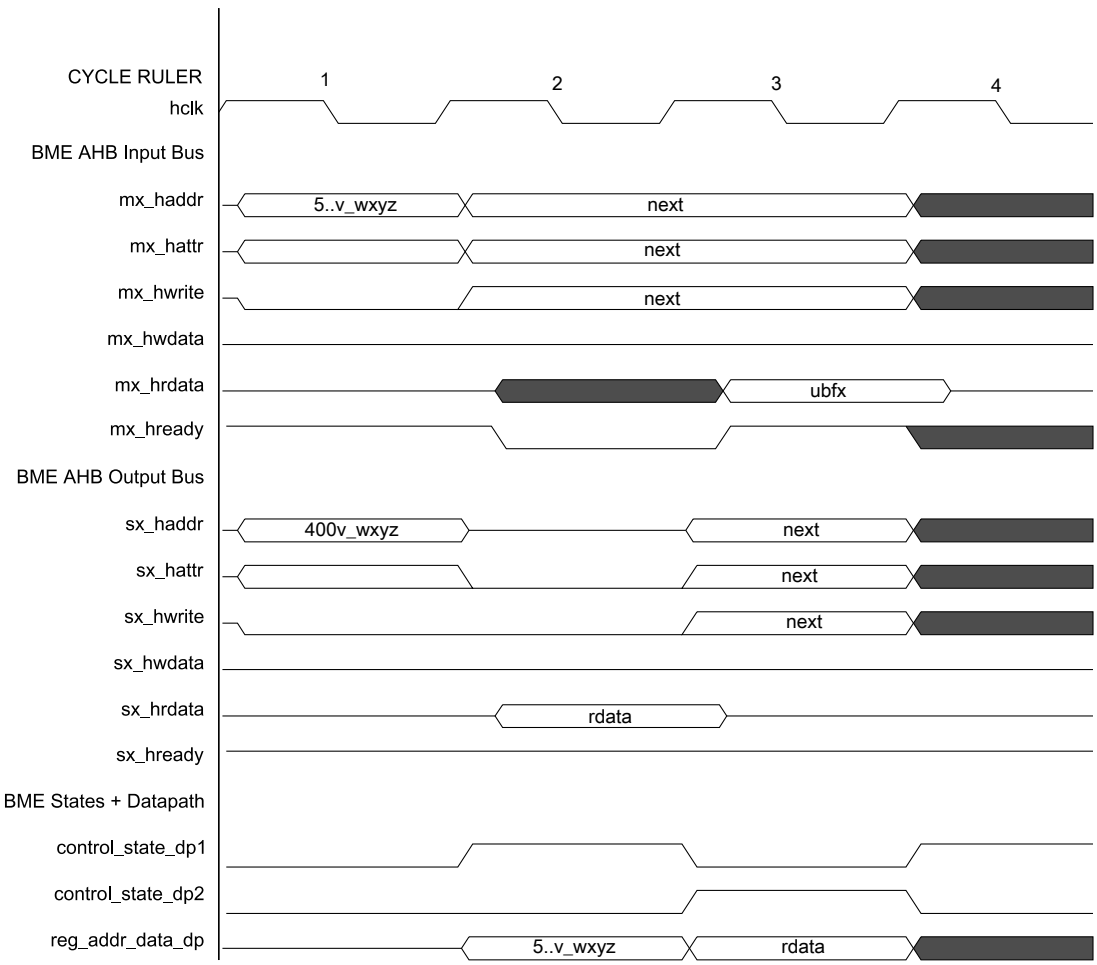


Figure 20-8. Decorated load: unsigned bit field insert timing diagram

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

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

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

NOTE

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

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

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

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

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

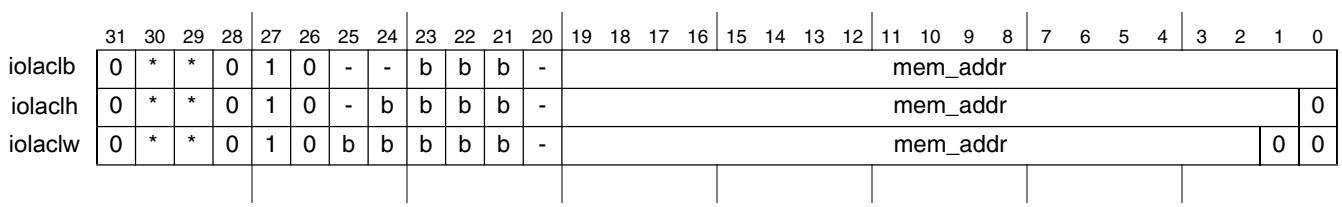


Figure 20-9. Decorated load address: load-and-clear 1 bit

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

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

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

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

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

Table 20-5. Cycle definitions of decorated load: load-and-clear 1 bit

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

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

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

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

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

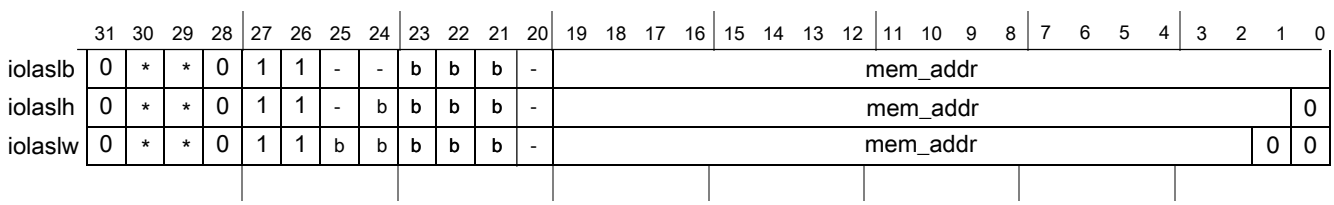


Figure 20-10. Decorated load address: load-and-set 1 bit

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

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

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

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

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

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

Table 20-6. Cycle definitions of decorated load: load-and-set 1-bit

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

20.3.2.3 Decorated load unsigned bit field extract (UBFX)

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

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

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

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

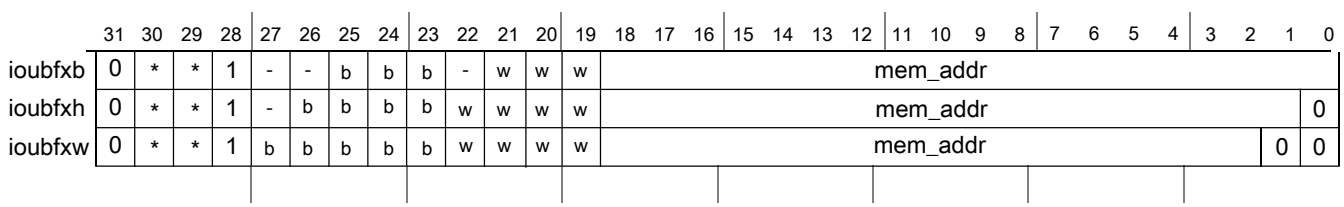


Figure 20-11. Decorated load address: unsigned bit field extract

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

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

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

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

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

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

Table 20-7. Cycle definitions of decorated load: unsigned bit field extract

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

20.3.3 Additional details on decorated addresses and GPIO accesses

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

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

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

Table 20-8. Decorated peripheral and GPIO address details

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

20.4 Application information

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

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

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

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

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

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

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

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

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

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

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

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

Chapter 21
Micro Trace Buffer (MTB)

21.1 Introduction

Microcontrollers using the Cortex-M0+ processor core include support for a CoreSight Micro Trace Buffer to provide program trace capabilities.

The proper name for this function is the CoreSight Micro Trace Buffer for the Cortex-M0+ Processor; in this document, it is simply abbreviated as the MTB.

The simple program trace function creates instruction address change-of-flow data packets in a user-defined region of the system RAM. Accordingly, the system RAM controller manages requests from two sources:

- AMBA-AHB reads and writes from the system bus
- program trace packet writes from the processor

As part of the MTB functionality, there is a DWT (Data Watchpoint and Trace) module that allows the user to define watchpoint addresses, or optionally, an address and data value, that when triggered, can be used to start or stop the program trace recording.

This document details the functionality of both the MTB_RAM and MTB_DWT capabilities.

21.1.1 Overview

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown as follows:

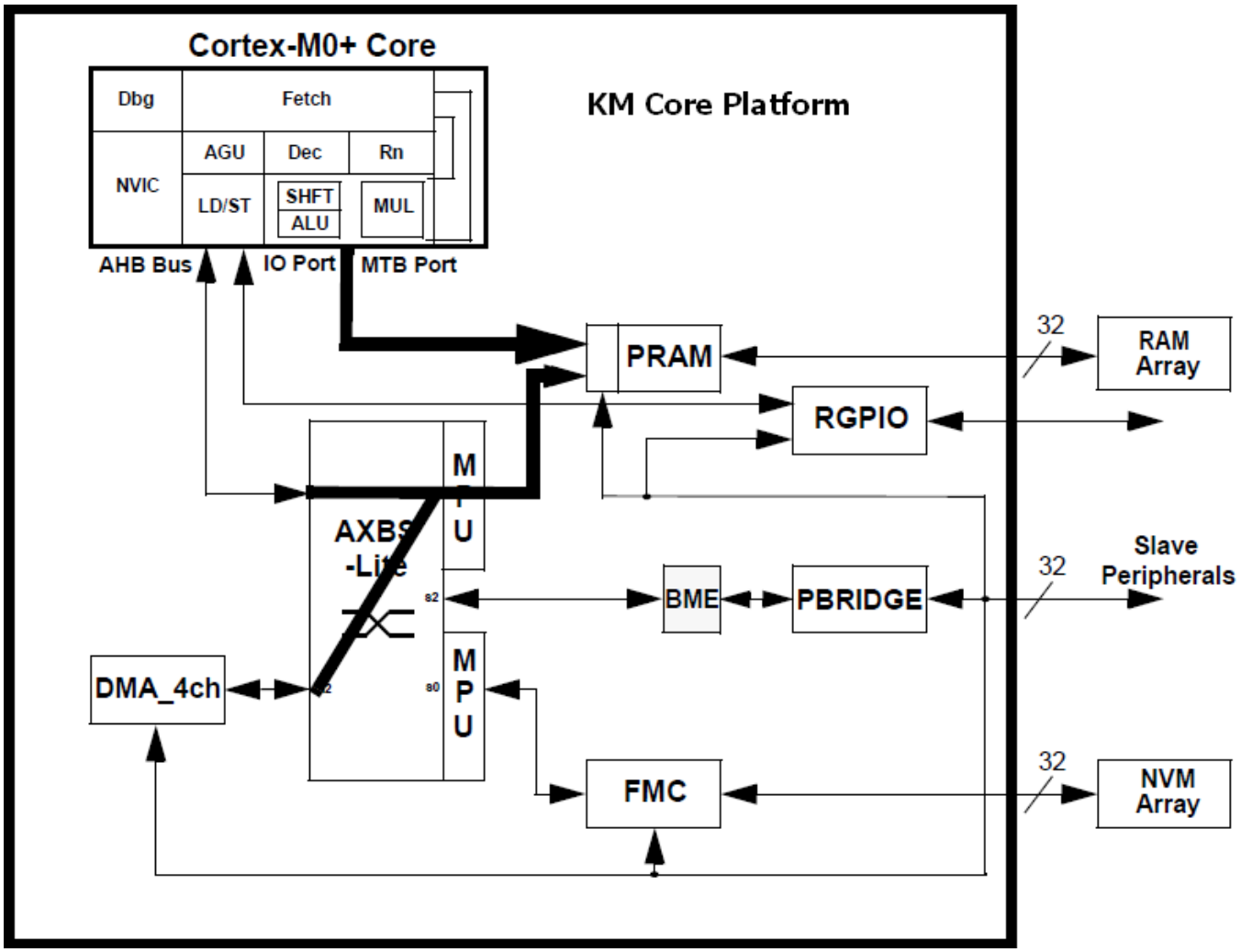


Figure 21-1. Core platform block diagram

As shown in the block diagram, the platform RAM (PRAM) controller connects to two input buses:

- the crossbar slave port for system bus accesses
- a "private execution MTB port" from the core

The logical paths from the crossbar master input ports to the PRAM controller are highlighted along with the private execution trace port from the processor core. The private MTB port signals the instruction address information needed for the 64-bit program trace packets written into the system RAM. The PRAM controller output interfaces to the attached RAM array. In this document, the PRAM controller is the MTB_RAM controller.

The following information is taken from the ARM CoreSight Micro Trace Buffer documentation.

"The execution trace packet consists of a pair of 32-bit words that the MTB generates when it detects the processor PC value changes non-sequentially. A non-sequential PC change can occur during branch instructions or during exception entry.

The processor can cause a trace packet to be generated for any instruction.

The following figure shows how the execution trace information is stored in memory as a sequence of packets.

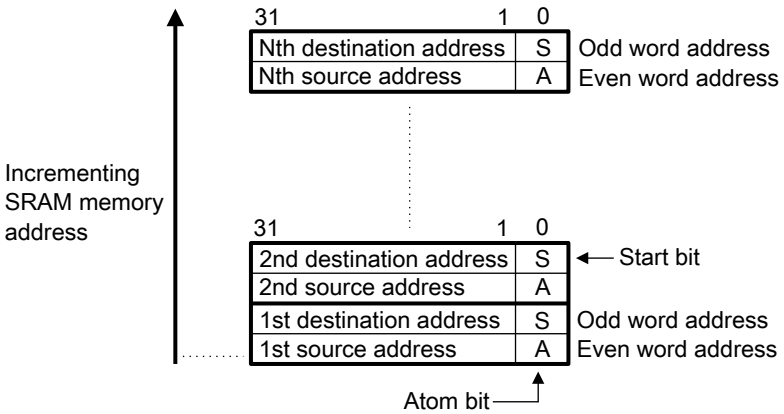


Figure 21-2. MTB execution trace storage format

The first, lower addressed, word contains the source of the branch, the address it branched from. The value stored only records bits[31:1] of the source address, because Thumb instructions are at least halfword aligned. The least significant bit of the value is the A-bit. The A-bit indicates the atomic state of the processor at the time of the branch, and can differentiate whether the branch originated from an instruction in a program, an exception, or a PC update in Debug state. When it is zero the branch originated from an instruction, when it is one the branch originated from an exception or PC update in Debug state. This word is always stored at an even word location.

The second, higher addressed word contains the destination of the branch, the address it branched to. The value stored only records bits[31:1] of the branch address. The least significant bit of the value is the S-bit. The S-bit indicates where the trace started. An S-bit value of 1 indicates where the first packet after the trace started and a value of 0 is used for other packets. Because it is possible to start and stop tracing multiple times in a trace session, the memory might contain several packets with the S-bit set to 1. This word is always stored in the next higher word in memory, an odd word address.

When the A-bit is set to 1, the source address field contains the architecturally-preferred return address for the exception. For example, if an exception was caused by an SVC instruction, then the source address field contains the address of the following instruction. This is different from the case where the A-bit is set to 0. In this case, the source address contains the address of the branch instruction.

For an exception return operation, two packets are generated:

- The first packet has the:
  - Source address field set to the address of the instruction that causes the exception return, BX or POP.
  - Destination address field set to bits[31:1] of the EXC_RETURN value. See the ARM v6-M Architecture Reference Manual.
  - The A-bit set to 0.
- The second packet has the:
  - Source address field set to bits[31:1] of the EXC_RETURN value.
  - Destination address field set to the address of the instruction where execution commences.
  - A-bit set to 1."

Given the recorded change-of-flow trace packets in system RAM and the memory image of the application, a debugger can read out the data and create an instruction-by-instruction program trace. In keeping with the low area and power implementation cost design targets, the MTB trace format is less efficient than other CoreSight trace modules, for example, the ETM (Embedded Trace Macrocell). Since each branch packet is 8 bytes in size, a 1 KB block of system RAM can contain 128 branches. Using the Dhrystone 2.1 benchmark's dynamic runtime as an example, this corresponds to about 875 instructions per KB of trace RAM, or with a zero wait state memory, this corresponds to approximately 1600 processor cycles per KB. This metric is obviously very sensitive to the runtime characteristics of the user code.

The MTB_DWT function (not shown in the core platform block diagram) monitors the processor address and data buses so that configurable watchpoints can be detected to trigger the appropriate response in the MTB recording.

21.1.2 Features

The key features of the MTB_RAM and MTB_DWT include:

- Memory controller for system RAM and Micro Trace Buffer for program trace packets
- Read/write capabilities for system RAM accesses, write-only for program trace packets
- Supports zero wait state response to system bus accesses when no trace data is being written
- Can buffer two AHB address phases and one data write for system RAM accesses
- Supports 64-bit program trace packets including source and destination instruction addresses

- Program trace information in RAM available to MCU's application code or external debugger
- Program trace watchpoint configuration accessible by MCU's application code or debugger
- Location and size of RAM trace buffer is configured by software
- Two DWT comparators (addresses or address + data) provide programmable start/stop recording
- CoreSight compliant debug functionality

21.1.3 Modes of operation

The MTB_RAM and MTB_DWT functions do not support any special modes of operation. The MTB_RAM controller, as a memory-mapped device located on the platform's slave AHB system bus, responds strictly on the basis of memory addresses for accesses to its attached RAM array. The MTB private execution bus provides program trace packet write information to the RAM controller. Both the MTB_RAM and MTB_DWT modules are memory-mapped, so their programming models can be accessed.

All functionality associated with the MTB_RAM and MTB_DWT modules resides in the core platform's clock domain; this includes its connections with the RAM array.

21.2 External signal description

The MTB_RAM and MTB_DWT modules do not directly support any external interfaces.

The internal interface includes a standard AHB bus with a 32-bit datapath width from the appropriate crossbar slave port plus the private execution trace bus from the processor core. The signals in the private execution trace bus are detailed in the following table taken from the ARM CoreSight Micro Trace Buffer documentation. The signal direction is defined as viewed by the MTB_RAM controller.

Table 21-1. Private execution trace port from the core to MTB_RAM

Signal	Direction	Description
LOCKUP	Input	Indicates the processor is in the Lockup state. This signal is driven LOW for cycles when the processor is executing normally and driven HIGH for every cycle the processor is waiting in the Lockup state. This signal is valid on every cycle.
IAESEQ	Input	Indicates the next instruction address in execute, IAEX, is sequential, that is non-branching.

Table continues on the next page...

Table 21-1. Private execution trace port from the core to MTB_RAM (continued)

Signal	Direction	Description
IAEXEN	Input	IAEX register enable.
IAEX[30:0]	Input	Registered address of the instruction in the execution stage, shifted right by one bit, that is, PC >> 1.
ATOMIC	Input	Indicates the processor is performing non-instruction related activities.
EDBGRQ	Output	Request for the processor to enter the Debug state, if enabled, and halt.

In addition, there are two signals formed by the MTB_DWT module and driven to the MTB_RAM controller: TSTART (trace start) and TSTOP (trace stop). These signals can be configured using the trace watchpoints to define programmable addresses and data values to affect the program trace recording state.

21.3 Memory map and register definition

The MTB_RAM and MTB_DWT modules each support a sparsely-populated 4 KB address space for their programming models. For each address space, there are a variety of control and configurable registers near the base address, followed by a large unused address space and finally a set of CoreSight registers to support dynamic determination of the debug configuration for the device.

Accesses to the programming model follow standard ARM conventions. Taken from the ARM CoreSight Micro Trace Buffer documentation, these are:

- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in UNPREDICTABLE behavior.
- The behavior of the MTB is UNPREDICTABLE if the registers with UNKNOWN reset values are not programmed prior to enabling trace.
- Unless otherwise stated in the accompanying text:
  - Do not modify reserved register bits
  - Ignore reserved register bits on reads
  - All register bits are reset to a logic 0 by a system or power-on reset
  - Use only word size, 32-bit, transactions to access all registers

21.3.1 MTB_RAM Memory Map

MTB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_0000	MTB Position Register (MTB_POSITION)	32	R/W	Undefined	21.3.1.1/ 358
F000_0004	MTB Master Register (MTB_MASTER)	32	R/W	See section	21.3.1.2/ 360
F000_0008	MTB Flow Register (MTB_FLOW)	32	R/W	Undefined	21.3.1.3/ 361
F000_000C	MTB Base Register (MTB_BASE)	32	R	Undefined	21.3.1.4/ 363
F000_0F00	Integration Mode Control Register (MTB_MODECTRL)	32	R	0000_0000h	21.3.1.5/ 364
F000_0FA0	Claim TAG Set Register (MTB_TAGSET)	32	R	0000_0000h	21.3.1.6/ 364
F000_0FA4	Claim TAG Clear Register (MTB_TAGCLEAR)	32	R	0000_0000h	21.3.1.7/ 365
F000_0FB0	Lock Access Register (MTB_LOCKACCESS)	32	R	0000_0000h	21.3.1.8/ 365
F000_0FB4	Lock Status Register (MTB_LOCKSTAT)	32	R	0000_0000h	21.3.1.9/ 366
F000_0FB8	Authentication Status Register (MTB_AUTHSTAT)	32	R	0000_0000h	21.3.1.10/ 366
F000_0FBC	Device Architecture Register (MTB_DEVICEARCH)	32	R	4770_0A31h	21.3.1.11/ 367
F000_0FC8	Device Configuration Register (MTB_DEVICECFG)	32	R	0000_0000h	21.3.1.12/ 367
F000_0FCC	Device Type Identifier Register (MTB_DEVICETYPID)	32	R	0000_0031h	21.3.1.13/ 368
F000_0FD0	Peripheral ID Register (MTB_PERIPHID4)	32	R	See section	21.3.1.14/ 368
F000_0FD4	Peripheral ID Register (MTB_PERIPHID5)	32	R	See section	21.3.1.14/ 368
F000_0FD8	Peripheral ID Register (MTB_PERIPHID6)	32	R	See section	21.3.1.14/ 368
F000_0FDC	Peripheral ID Register (MTB_PERIPHID7)	32	R	See section	21.3.1.14/ 368
F000_0FE0	Peripheral ID Register (MTB_PERIPHID0)	32	R	See section	21.3.1.14/ 368
F000_0FE4	Peripheral ID Register (MTB_PERIPHID1)	32	R	See section	21.3.1.14/ 368
F000_0FE8	Peripheral ID Register (MTB_PERIPHID2)	32	R	See section	21.3.1.14/ 368
F000_0FEC	Peripheral ID Register (MTB_PERIPHID3)	32	R	See section	21.3.1.14/ 368

Table continues on the next page...

MTB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_0FF0	Component ID Register (MTB_COMPID0)	32	R	See section	21.3.1.15/ 369
F000_0FF4	Component ID Register (MTB_COMPID1)	32	R	See section	21.3.1.15/ 369
F000_0FF8	Component ID Register (MTB_COMPID2)	32	R	See section	21.3.1.15/ 369
F000_0FFC	Component ID Register (MTB_COMPID3)	32	R	See section	21.3.1.15/ 369

21.3.1.1 MTB Position Register (MTB_POSITION)

The MTB_POSITION register contains the Trace Write Address Pointer and Wrap fields. This register can be modified by the explicit programming model writes. It is also automatically updated by the MTB hardware when trace packets are being recorded.

The base address of the system RAM in the memory map dictates special consideration for the placement of the MTB. Consider the following guidelines:

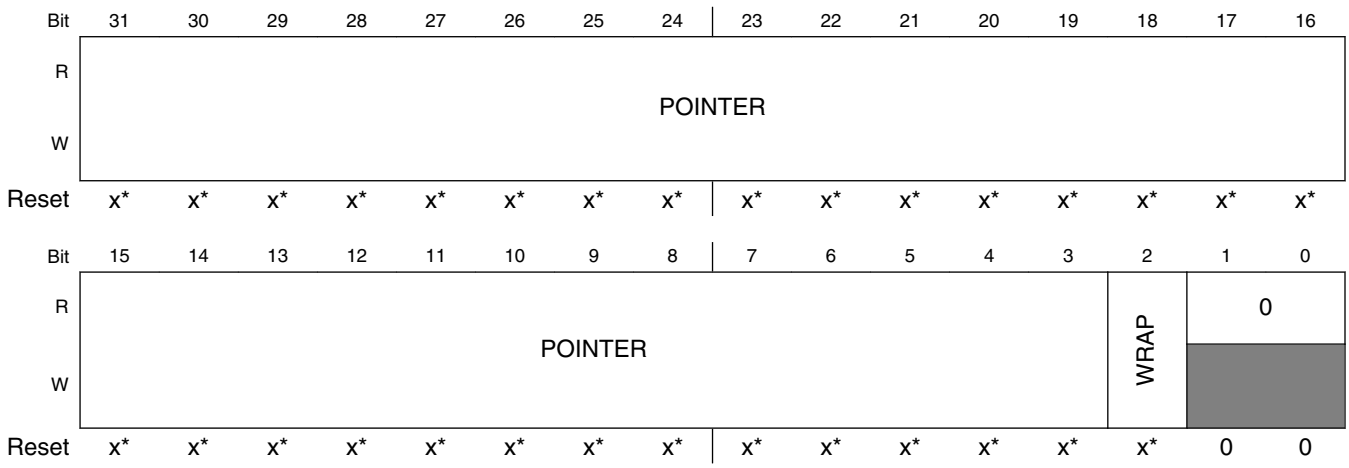
For the standard configuration where the size of the MTB is ≤ 25% of the total RAM capacity, it is recommended the MTB be based at the address defined by the MTB_BASE register. The read-only MTB_BASE register is defined by the expression (0x2000_0000 - (RAM_Size/4)). For this configuration, the MTB_POSITION register is initialized to MTB_BASE & 0x0000_7FF8.

If the size of the MTB is more than 25% but less than or equal to 50% of the total RAM capacity, it is recommended the MTB be based at address 0x2000_0000. In this configuration, the MTB_POSITION register is initialized to (0x2000_0000 & 0x0000_7FF8) = 0x0000_00000.

Following these two suggested placements provides a full-featured circular memory buffer containing program trace packets.

In the unlikely event an even larger trace buffer is required, a write-once capacity of 75% of the total RAM capacity can be based at address 0x2000_0000. The MTB_POSITION register is initialized to (0x2000_0000 & 0x0000_7FF8) = 0x0000_0000. However, this configuration cannot support operation as a circular queue and instead requires the use of the MTB_FLOW[WATERMARK] capability to automatically disable tracing or halting the processor as the number of packet writes approach the buffer capacity. See the MTB_FLOW register description for more details.

Address: F000_0000h base + 0h offset = F000_0000h



* Notes:
- x = Undefined at reset.

MTB_POSITION field descriptions

Field	Description

31–3
POINTER
	Trace Packet Address Pointer[28:0]
	
	Because a packet consists of two words, the POINTER field is the address of the first word of a packet. This field contains bits[31:3] of the RAM address where the next trace packet is written. Therefore, it points to an unused location and is automatically incremented.
	
	A debug agent can calculate the system memory map address for the current location in the MTB using the following "generic" equation:
	
	Given mtb_size = 1 << (MTB_MASTER[MASK] + 4),
	
	systemAddress = MTB_BASE + (((MTB_POSITION & 0xFFFF_FFF8) + (mtb_size - (MTB_BASE & (mtb_size-1)))) & 0x0000_7FF8);
	
	For this device, a simpler expression also applies. See the following pseudo-code:
	
	if ((MTB_POSITION >> 13) == 0x3) systemAddress = (0x1FFF << 16) + (0x1 << 15) + (MTB_POSITION & 0x7FF8); else systemAddress = (0x2000 << 16) + (0x0 << 15) + (MTB_POSITION & 0x7FF8);
	
	NOTE: The size of the RAM is parameterized and the most significant bits of the POINTER field are RAZ/WI.
	
	For these devices, POSITION[31:15] == POSITION[POINTER[28:12]] are RAZ/WI. Therefore, the active bits in this field are POSITION[14:3] == POSITION[POINTER[11:0]].

2 WRAP	WRAP This field is set to 1 automatically when the POINTER value wraps as determined by the MTB_MASTER[MASK] field in the MASTER Trace Control Register. A debug agent might use the WRAP field to determine whether the trace information above and below the pointer address is valid.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.3.1.2 MTB Master Register (MTB_MASTER)

The MTB_MASTER register contains the main program trace enable plus other trace controls. This register can be modified by the explicit programming model writes. MTB_MASTER[EN] and MTB_MASTER[HALTREQ] fields are also automatically updated by the MTB hardware.

Before MTB_MASTER[EN] or MTB_MASTER[TSTARTEN] are set to 1, the software must initialize the MTB_POSITION and MTB_FLOW registers.

If MTB_FLOW[WATERMARK] is used to stop tracing or to halt the processor, MTB_MASTER[MASK] must still be set to a value that prevents MTB_POSITION[POINTER] from wrapping before it reaches the MTB_FLOW[WATERMARK] value.

NOTE

The format of this mask field is different than MTBDWT_MASKn[MASK].

Address: F000_0000h base + 4h offset = F000_0004h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						HALTREQ	RAMPRIV	SFRWPRIV	TSTOPEN	TSTARTEN	MASK				
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MTB_MASTER field descriptions

Field	Description
31 EN	Main Trace Enable When this field is 1, trace data is written into the RAM memory location addressed by MTB_POSITION[POINTER]. The MTB_POSITION[POINTER] value auto increments after the trace data packet is written. EN can be automatically set to 0 using the MTB_FLOW[WATERMARK] field and the MTB_FLOW[AUTOSTOP] bit.

Table continues on the next page...

MTB_MASTER field descriptions (continued)

Field	Description
	EN is automatically set to 1 if TSTARTEN is 1 and the TSTART signal is HIGH. EN is automatically set to 0 if TSTOPEN is 1 and the TSTOP signal is HIGH. **NOTE:** If EN is set to 0 because MTB_FLOW[WATERMARK] is set, then it is not automatically set to 1 if TSTARTEN is 1 and the TSTART input is HIGH. In this case, tracing can only be restarted if MTB_FLOW[WATERMARK] or MTB_POSITION[POINTER] value is changed by software.
30–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 HALTREQ	Halt Request This field is connected to the halt request signal of the trace logic, EDBGRQ. When HALTREQ is set to 1, the EDBFGRQ is asserted if DBGEN (invasive debug enable, one of the debug authentication interface signals) is also HIGH. HALTREQ can be automatically set to 1 using MTB_FLOW[WATERMARK].
8 RAMPRIV	RAM Privilege If this field is 0, then user or privileged AHB read and write accesses to the RAM are permitted. If this field is 1, then only privileged AHB read and write accesses to the RAM are permitted and user accesses are RAZ/WI. The HPROT[1] signal determines if an access is a user or privileged mode reference.
7 SFRWPRIV	Special Function Register Write Privilege If this field is 0, then user or privileged AHB read and write accesses to the MTB_RAM Special Function Registers (programming model) are permitted. If this field is 1, then only privileged write accesses are permitted; user write accesses are ignored. The HPROT[1] signal determines if an access is user or privileged. Note MTB_RAM SFR read access are not controlled by this bit and are always permitted.
6 TSTOPEN	Trace Stop Input Enable If this field is 1 and the TSTOP signal is HIGH, then EN is set to 0. If a trace packet is being written to memory, the write is completed before tracing is stopped.
5 TSTARTEN	Trace Start Input Enable If this field is 1 and the TSTART signal is HIGH, then EN is set to 1. Tracing continues until a stop condition occurs.
MASK	Mask This value determines the maximum size of the trace buffer in RAM. It specifies the most-significant bit of the MTB_POSITION[POINTER] field that can be updated by automatic increment. If the trace tries to advance past this power of 2, the MTB_POSITION[WRAP] bit is set to 1, the MTB_POSITION[MASK+3:3] == MTB_POSITION[POINTER[MASK:0]] bits are set to 0, and the MTB_POSITION[14:MASK+3] == MTB_POSITION[POINTER[11:MASK+1]] bits remain unchanged. This field causes the trace packet information to be stored in a circular buffer of size 2^[MASK+4] bytes, that can be positioned in memory at multiples of this size. As detailed in the MTB_POSITION description, typical "upper limits" for the MTB size are RAM_Size/4 or RAM_Size/2. Values greater than the maximum have the same effect as the maximum.

21.3.1.3 MTB Flow Register (MTB_FLOW)

The MTB_FLOW register contains the watermark address and the autostop/autohalt control bits.

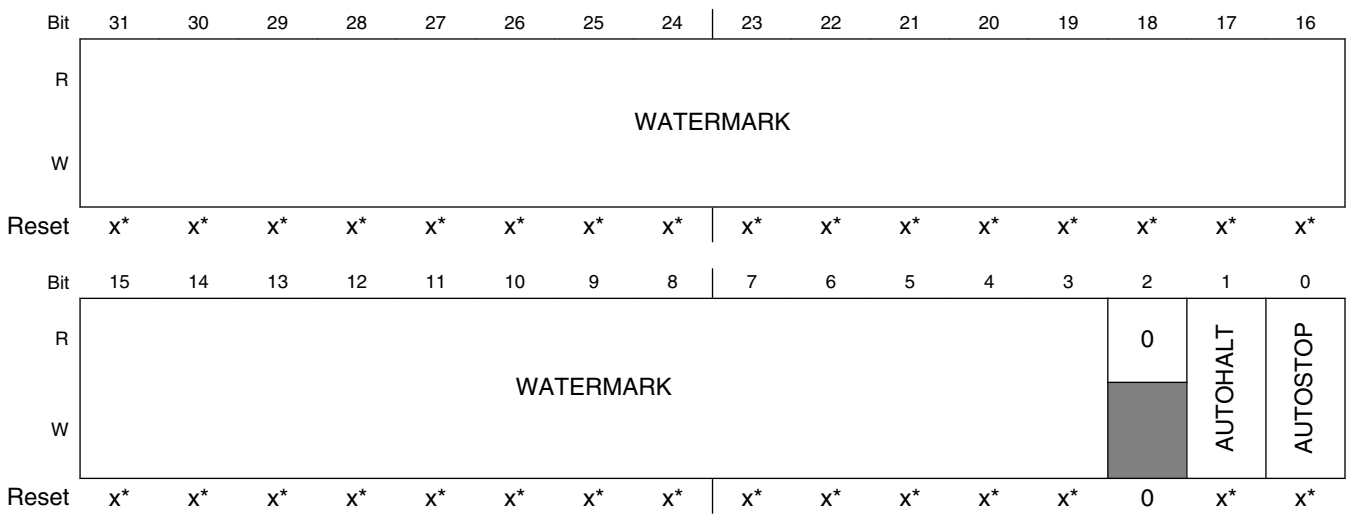
If tracing is stopped using the watermark autostop feature, it cannot be restarted until software clears the watermark autostop. This can be achieved in one of the following ways:

- Changing the MTB_POSITION[POINTER] field value to point to the beginning of the trace buffer, or
- Setting MTB_FLOW[AUTOSTOP] = 0.

A debug agent can use MTB_FLOW[AUTOSTOP] to fill the trace buffer once only without halting the processor.

A debug agent can use MTB_FLOW[AUTOHALT] to fill the trace buffer once before causing the Cortex-M0+ processor to enter the Debug state. To enter Debug state, the Cortex-M0+ processor might have to perform additional branch type operations. Therefore, the MTB_FLOW[WATERMARK] field must be set below the final entry in the trace buffer region.

Address: F000_0000h base + 8h offset = F000_0008h



* Notes:
- x = Undefined at reset.

MTB_FLOW field descriptions

Field	Description
31–3 WATERMARK	WATERMARK[28:0] This field contains an address in the same format as the MTB_POSITION[POINTER] field. When MTB_POSITION[POINTER] matches the WATERMARK field value, actions defined by the AUTOHALT and AUTOSTOP bits are performed.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MTB_FLOW field descriptions (continued)

Field	Description
1 AUTOHALT	AUTOHALT If this field is 1 and WATERMARK is equal to MTB_POSITION[POINTER], then MTB_MASTER[HALTREQ] is automatically set to 1. If the DBGEN signal is HIGH, the MTB asserts this halt request to the Cortex-M0+ processor by asserting the EDBGRQ signal.
0 AUTOSTOP	AUTOSTOP If this field is 1 and WATERMARK is equal to MTB_POSITION[POINTER], then MTB_MASTER[EN] is automatically set to 0. This stops tracing.

21.3.1.4 MTB Base Register (MTB_BASE)

The read-only MTB_BASE Register indicates where the RAM is located in the system memory map. This register is provided to enable auto discovery of the MTB RAM location, by a debug agent and is defined by a hardware design parameter. For this device, the base address is defined by the expression: MTB_BASE[BASEADDR] = 0x2000_0000 - (RAM_Size/4)

Address: F000_0000h base + Ch offset = F000_000Ch

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

* Notes:
- x = Undefined at reset.

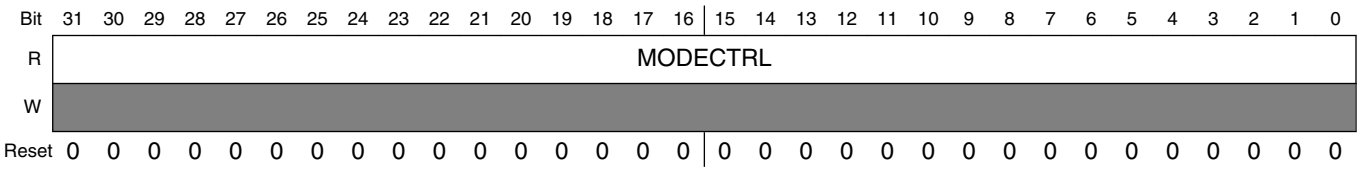
MTB_BASE field descriptions

Field	Description
BASEADDR	BASEADDR This value is defined with a hardwired signal and the expression: 0x2000_0000 - (RAM_Size/4). For example, if the total RAM capacity is 16 KB, this field is 0x1FFF_F000.

21.3.1.5 Integration Mode Control Register (MTB_MODECTRL)

This register enables the device to switch from a functional mode, or default behavior, into integration mode. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + F00h offset = F000_0F00h



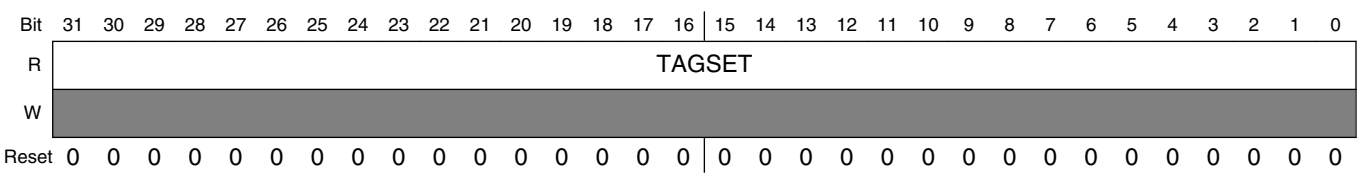
MTB_MODECTRL field descriptions

Field	Description
MODECTRL	MODECTRL Hardwired to 0x0000_0000

21.3.1.6 Claim TAG Set Register (MTB_TAGSET)

The Claim Tag Set Register returns the number of bits that can be set on a read, and enables individual bits to be set on a write. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FA0h offset = F000_0FA0h



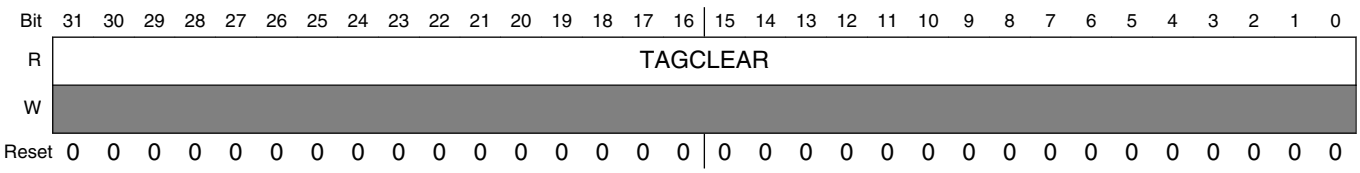
MTB_TAGSET field descriptions

Field	Description
TAGSET	TAGSET Hardwired to 0x0000_0000

21.3.1.7 Claim TAG Clear Register (MTB_TAGCLEAR)

The read/write Claim Tag Clear Register is used to read the claim status on debug resources. A read indicates the claim tag status. Writing 1 to a specific bit clears the corresponding claim tag to 0. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FA4h offset = F000_0FA4h



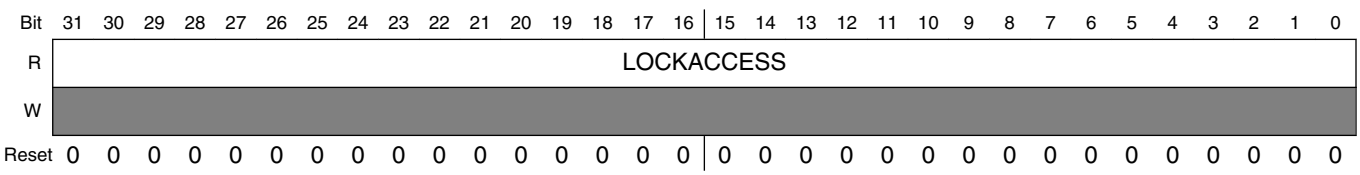
MTB_TAGCLEAR field descriptions

Field	Description
TAGCLEAR	TAGCLEAR Hardwired to 0x0000_0000

21.3.1.8 Lock Access Register (MTB_LOCKACCESS)

The Lock Access Register enables a write access to component registers. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FB0h offset = F000_0FB0h



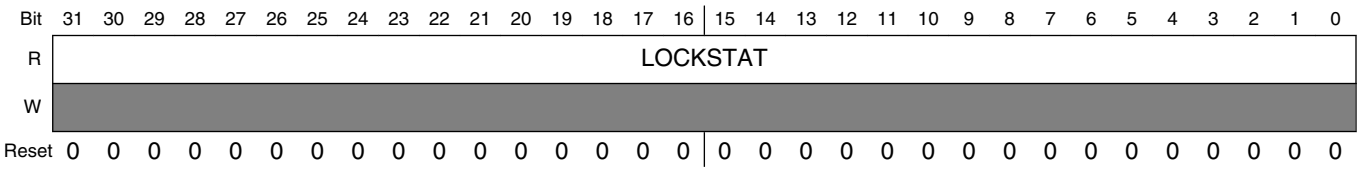
MTB_LOCKACCESS field descriptions

Field	Description
LOCKACCESS	Hardwired to 0x0000_0000

21.3.1.9 Lock Status Register (MTB_LOCKSTAT)

The Lock Status Register indicates the status of the lock control mechanism. This register is used in conjunction with the Lock Access Register. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FB4h offset = F000_0FB4h



MTB_LOCKSTAT field descriptions

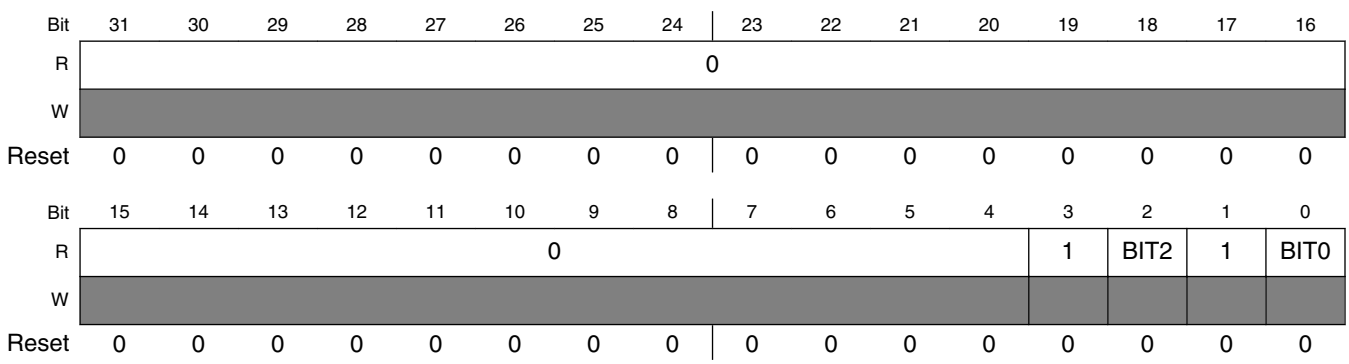
Field	Description
LOCKSTAT	LOCKSTAT Hardwired to 0x0000_0000

21.3.1.10 Authentication Status Register (MTB_AUTHSTAT)

The Authentication Status Register reports the required security level and current status of the security enable bit pairs. Where functionality changes on a given security level, this change must be reported in this register. It is connected to specific signals used during the auto-discovery process by an external debug agent.

MTB_AUTHSTAT[3:2] indicates if nonsecure, noninvasive debug is enabled or disabled, while MTB_AUTHSTAT[1:0] indicates the enabled/disabled state of nonsecure, invasive debug. For both 2-bit fields, 0b10 indicates the functionality is disabled and 0b11 indicates it is enabled.

Address: F000_0000h base + FB8h offset = F000_0FB8h



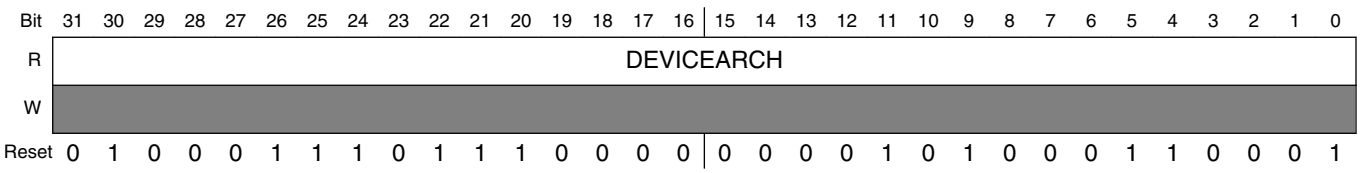
MTB_AUTHSTAT field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	BIT3 This read-only field is reserved and always has the value 1.
2 BIT2	BIT2 Connected to NIDEN or DBGEN signal.
1 Reserved	BIT1 This read-only field is reserved and always has the value 1.
0 BIT0	Connected to DBGEN.

21.3.1.11 Device Architecture Register (MTB_DEVICEARCH)

This register indicates the device architecture. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FBCh offset = F000_0FBCh



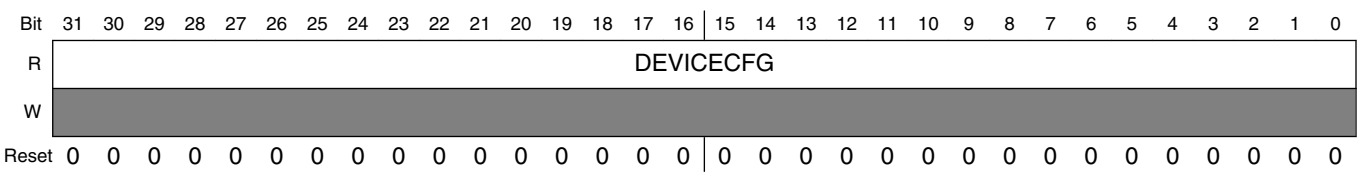
MTB_DEVICEARCH field descriptions

Field	Description
DEVICEARCH	DEVICEARCH Hardwired to 0x4770_0A31.

21.3.1.12 Device Configuration Register (MTB_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FC8h offset = F000_0FC8h



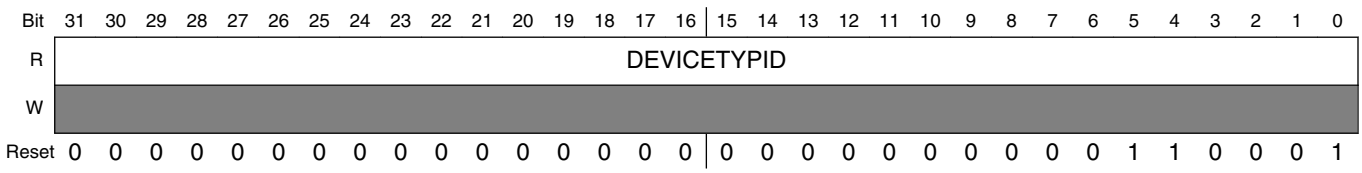
MTB_DEVICECFG field descriptions

Field	Description
DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

21.3.1.13 Device Type Identifier Register (MTB_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FCCh offset = F000_0FCCh



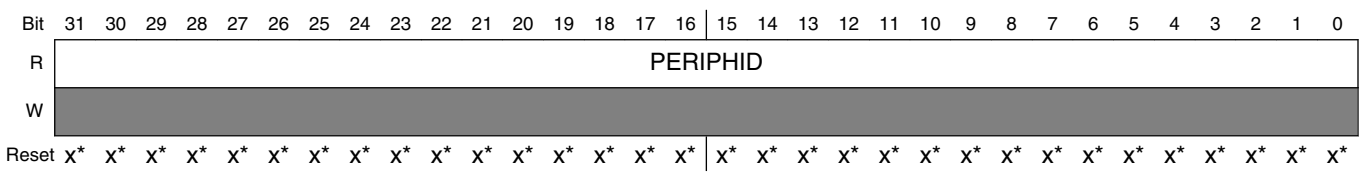
MTB_DEVICETYPID field descriptions

Field	Description
DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0031.

21.3.1.14 Peripheral ID Register (MTB_PERIPHID*n*)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FD0h offset + (4d × i), where i=0d to 7d



MTB_PERIPHID*n* field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID4 is hardwired to 0x0000_0004; ID0 to 0x0000_0032; ID1 to 0x0000_00B9; ID2 to 0x0000_001B; and all the others to 0x0000_0000.

21.3.1.15 Component ID Register (MTB_COMPID*n*)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FF0h offset + (4d × i), where i=0d to 3d

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

MTB_COMPID*n* field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

21.3.2 MTB_DWT Memory Map

The MTB_DWT programming model supports a very simplified subset of the v7M debug architecture and follows the standard ARM DWT definition.

MTBDWT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_1000	MTB DWT Control Register (MTBDWT_CTRL)	32	R	2F00_0000h	21.3.2.1/ 370
F000_1020	MTB_DWT Comparator Register (MTBDWT_COMP0)	32	R/W	0000_0000h	21.3.2.2/ 371
F000_1024	MTB_DWT Comparator Mask Register (MTBDWT_MASK0)	32	R/W	0000_0000h	21.3.2.3/ 372
F000_1028	MTB_DWT Comparator Function Register 0 (MTBDWT_FCT0)	32	R/W	0000_0000h	21.3.2.4/ 373
F000_1030	MTB_DWT Comparator Register (MTBDWT_COMP1)	32	R/W	0000_0000h	21.3.2.2/ 371
F000_1034	MTB_DWT Comparator Mask Register (MTBDWT_MASK1)	32	R/W	0000_0000h	21.3.2.3/ 372
F000_1038	MTB_DWT Comparator Function Register 1 (MTBDWT_FCT1)	32	R/W	0000_0000h	21.3.2.5/ 375
F000_1200	MTB_DWT Trace Buffer Control Register (MTBDWT_TBCTRL)	32	R/W	2000_0000h	21.3.2.6/ 376

Table continues on the next page...

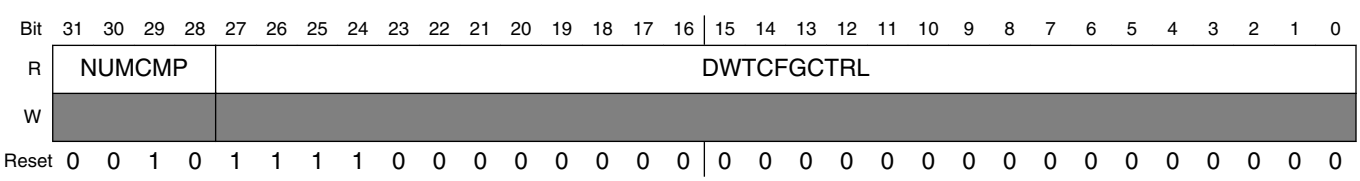
MTBDWT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_1FC8	Device Configuration Register (MTBDWT_DEVICECFG)	32	R	0000_0000h	21.3.2.7/ 378
F000_1FCC	Device Type Identifier Register (MTBDWT_DEVICETYPID)	32	R	0000_0004h	21.3.2.8/ 378
F000_1FD0	Peripheral ID Register (MTBDWT_PERIPHID4)	32	R	See section	21.3.2.9/ 379
F000_1FD4	Peripheral ID Register (MTBDWT_PERIPHID5)	32	R	See section	21.3.2.9/ 379
F000_1FD8	Peripheral ID Register (MTBDWT_PERIPHID6)	32	R	See section	21.3.2.9/ 379
F000_1FDC	Peripheral ID Register (MTBDWT_PERIPHID7)	32	R	See section	21.3.2.9/ 379
F000_1FE0	Peripheral ID Register (MTBDWT_PERIPHID0)	32	R	See section	21.3.2.9/ 379
F000_1FE4	Peripheral ID Register (MTBDWT_PERIPHID1)	32	R	See section	21.3.2.9/ 379
F000_1FE8	Peripheral ID Register (MTBDWT_PERIPHID2)	32	R	See section	21.3.2.9/ 379
F000_1FEC	Peripheral ID Register (MTBDWT_PERIPHID3)	32	R	See section	21.3.2.9/ 379
F000_1FF0	Component ID Register (MTBDWT_COMPID0)	32	R	See section	21.3.2.10/ 379
F000_1FF4	Component ID Register (MTBDWT_COMPID1)	32	R	See section	21.3.2.10/ 379
F000_1FF8	Component ID Register (MTBDWT_COMPID2)	32	R	See section	21.3.2.10/ 379
F000_1FFC	Component ID Register (MTBDWT_COMPID3)	32	R	See section	21.3.2.10/ 379

21.3.2.1 MTB DWT Control Register (MTBDWT_CTRL)

The MTBDWT_CTRL register provides read-only information on the watchpoint configuration for the MTB_DWT.

Address: F000_1000h base + 0h offset = F000_1000h



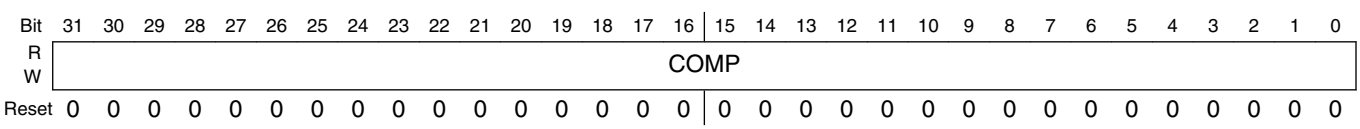
MTBDWT_CTRL field descriptions

Field	Description
31–28 NUMCMP	Number of comparators The MTB_DWT implements two comparators.
DWTCFGCTRL	DWT configuration controls This field is hardwired to 0xF00_0000, disabling all the remaining DWT functionality. The specific fields and their state are: MTBDWT_CTRL[27] = NOTRCPKT = 1, trace sample and exception trace is not supported MTBDWT_CTRL[26] = NOEXTTRIG = 1, external match signals are not supported MTBDWT_CTRL[25] = NOCYCCNT = 1, cycle counter is not supported MTBDWT_CTRL[24] = NOPRFCNT = 1, profiling counters are not supported MTBDWT_CTRL[22] = CYCEBTENA = 0, no POSTCNT underflow packets generated MTBDWT_CTRL[21] = FOLDEVTENA = 0, no folded instruction counter overflow events MTBDWT_CTRL[20] = LSUEVTENA = 0, no LSU counter overflow events MTBDWT_CTRL[19] = SLEEPEVTENA = 0, no sleep counter overflow events MTBDWT_CTRL[18] = EXCEVTENA = 0, no exception overhead counter events MTBDWT_CTRL[17] = CPIEVTENA = 0, no CPI counter overflow events MTBDWT_CTRL[16] = EXCTRCENA = 0, generation of exception trace disabled MTBDWT_CTRL[12] = PCSAMPLENA = 0, no periodic PC sample packets generated MTBDWT_CTRL[11:10] = SYNCTAP = 0, no synchronization packets MTBDWT_CTRL[9] = CYCTAP = 0, cycle counter is not supported MTBDWT_CTRL[8:5] = POSTINIT = 0, cycle counter is not supported MTBDWT_CTRL[4:1] = POSTPRESET = 0, cycle counter is not supported MTBDWT_CTRL[0] = CYCCNTENA = 0, cycle counter is not supported

21.3.2.2 MTB_DWT Comparator Register (MTBDWT_COMP*n*)

The MTBDWT_COMPn registers provide the reference value for comparator n.

Address: F000_1000h base + 20h offset + (16d × i), where i=0d to 1d



MTBDWT_COMP*n* field descriptions

Field	Description
COMP	Reference value for comparison If MTBDWT_COMP0 is used for a data value comparator and the access size is byte or halfword, the data value must be replicated across all appropriate byte lanes of this register. For example, if the data is a

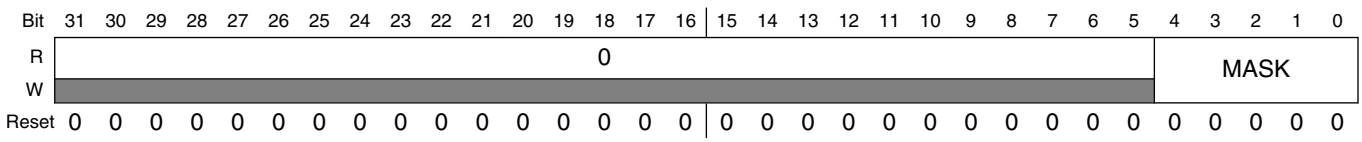
MTBDWT_COMP*n* field descriptions (continued)

Field	Description
	byte-sized "x" value, then COMP[31:24] = COMP[23:16] = COMP[15:8] = COMP[7:0] = "x". Likewise, if the data is a halfword-size "y" value, then COMP[31:16] = COMP[15:0] = "y".

21.3.2.3 MTB_DWT Comparator Mask Register (MTBDWT_MASK*n*)

The MTBDWT_MASKn registers define the size of the ignore mask applied to the reference address for address range matching by comparator n. Note the format of this mask field is different than the MTB_MASTER[MASK].

Address: F000_1000h base + 24h offset + (16d × i), where i=0d to 1d



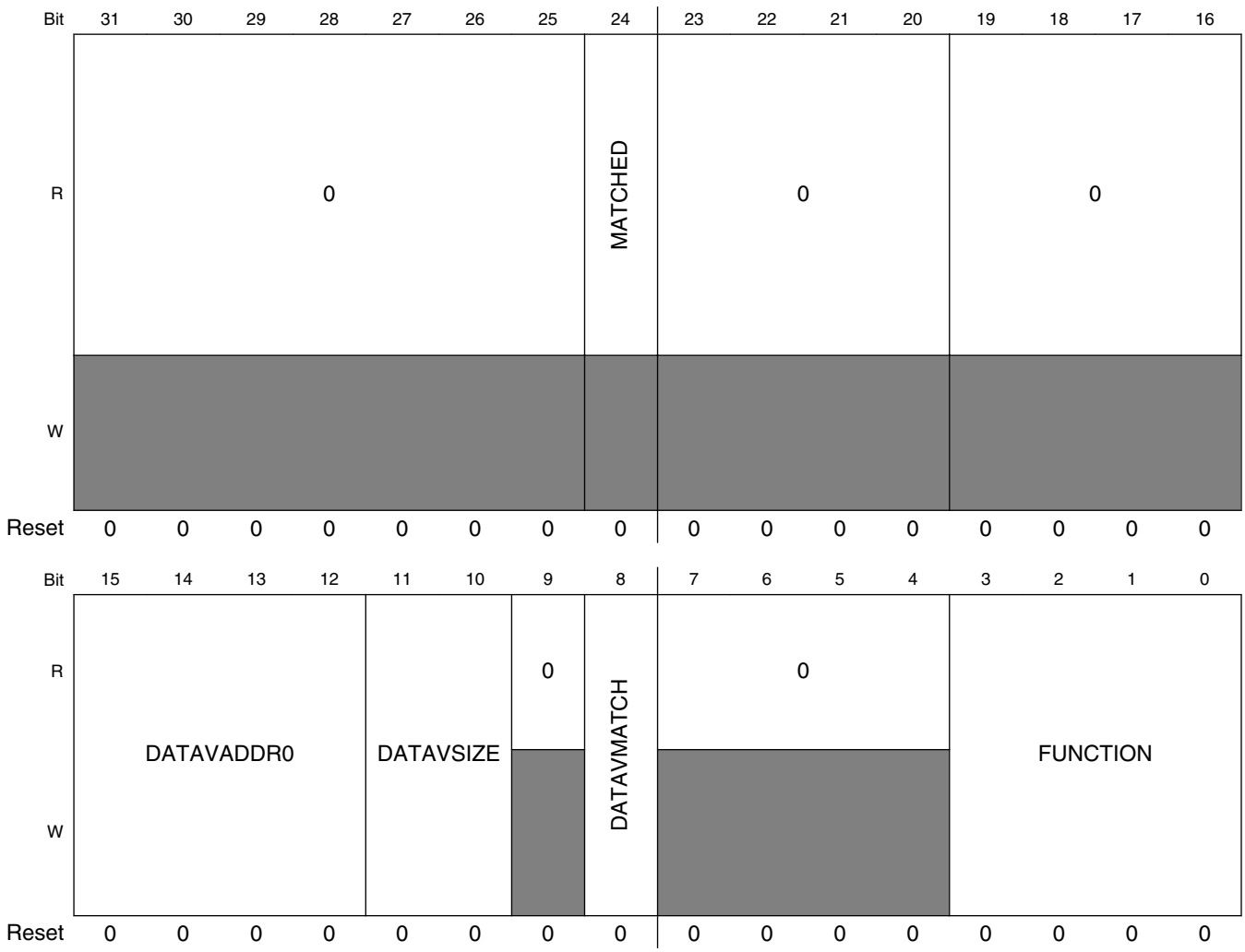
MTBDWT_MASK*n* field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MASK	MASK The value of the ignore mask, 0-31 bits, is applied to address range matching. MASK = 0 is used to include all bits of the address in the comparison, except if MASK = 0 and the comparator is configured to watch instruction fetch addresses, address bit [0] is ignored by the hardware since all fetches must be at least halfword aligned. For MASK != 0 and regardless of watch type, address bits [x-1:0] are ignored in the address comparison. Using a mask means the comparator matches on a range of addresses, defined by the unmasked most significant bits of the address, bits [31:x]. The maximum MASK value is 24, producing a 16 Mbyte mask. An attempted write of a MASK value > 24 is limited by the MTBDWT hardware to 24. If MTBDWT_COMP0 is used as a data value comparator, then MTBDWT_MASK0 should be programmed to zero.

21.3.2.4 MTB_DWT Comparator Function Register 0 (MTBDWT_FCT0)

The MTBDWT_FCTn registers control the operation of comparator n.

Address: F000_1000h base + 28h offset = F000_1028h



MTBDWT_FCT0 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 MATCHED	Comparator match If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit. 0 No match. 1 Match occurred.

Table continues on the next page...

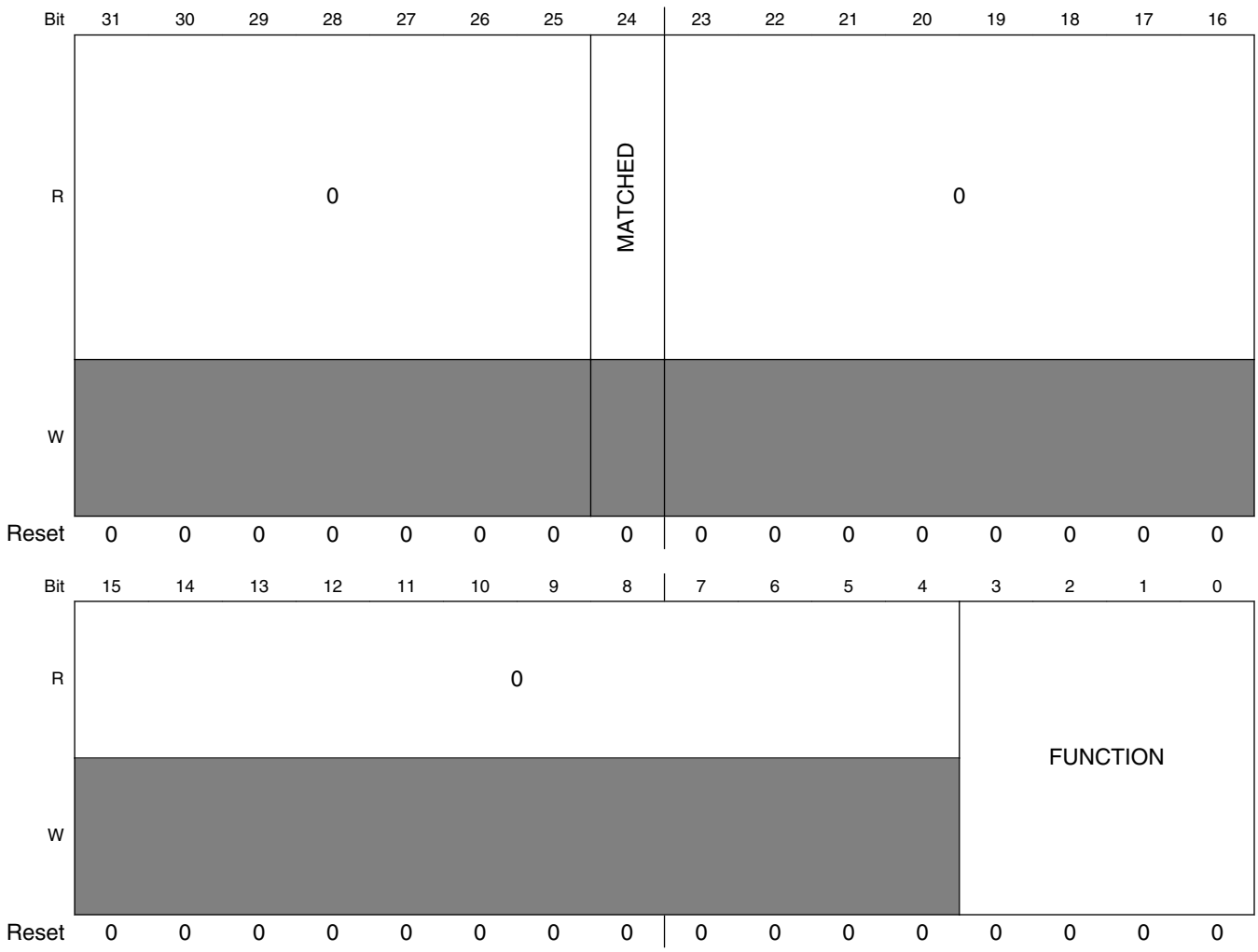
MTBDWT_FCT0 field descriptions (continued)

Field	Description
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 DATAVADDR0	Data Value Address 0 Since the MTB_DWT implements two comparators, the DATAVADDR0 field is restricted to values {0,1}. When the DATAVMATCH bit is asserted, this field defines the comparator number to use for linked address comparison. If MTBDWT_COMP0 is used as a data watchpoint and MTBDWT_COMP1 as an address watchpoint, DATAVADDR0 must be set.
11–10 DATAVSIZE	Data Value Size For data value matching, this field defines the size of the required data comparison. 00 Byte. 01 Halfword. 10 Word. 11 Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 DATAVMATCH	Data Value Match When this field is 1, it enables data value comparison. For this implementation, MTBDWT_COMP0 supports address or data value comparisons; MTBDWT_COMP1 only supports address comparisons. 0 Perform address comparison. 1 Perform data value comparison.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FUNCTION	Function Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

21.3.2.5 MTB_DWT Comparator Function Register 1 (MTBDWT_FCT1)

The MTBDWT_FCTn registers control the operation of comparator n. Since the MTB_DWT only supports data value comparisons on comparator 0, there are several fields in the MTBDWT_FCT1 register that are RAZ/WI (bits 12, 11:10, 8).

Address: F000_1000h base + 38h offset = F000_1038h



MTBDWT_FCT1 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 MATCHED	Comparator match If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit.

Table continues on the next page...

MTBDWT_FCT1 field descriptions (continued)

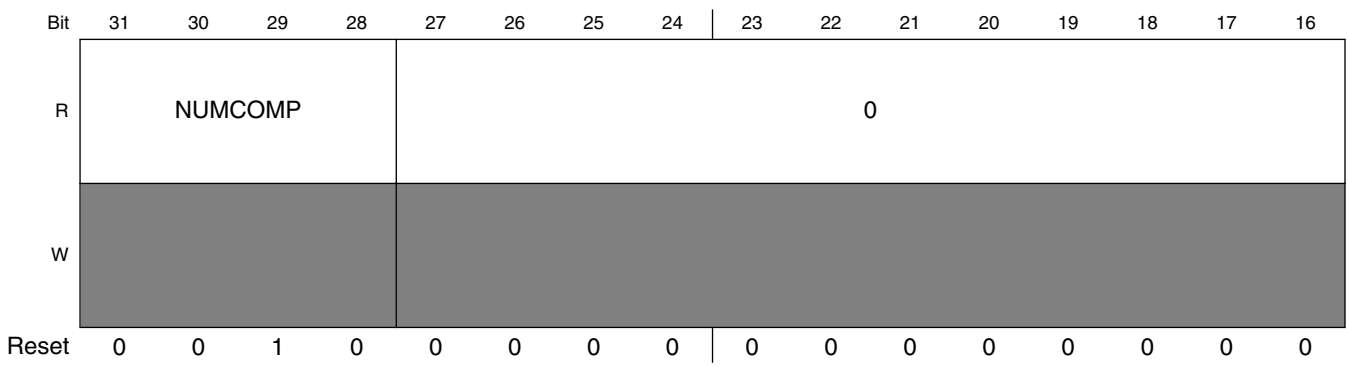
Field	Description
	0 No match. 1 Match occurred.
23–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FUNCTION	Function Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

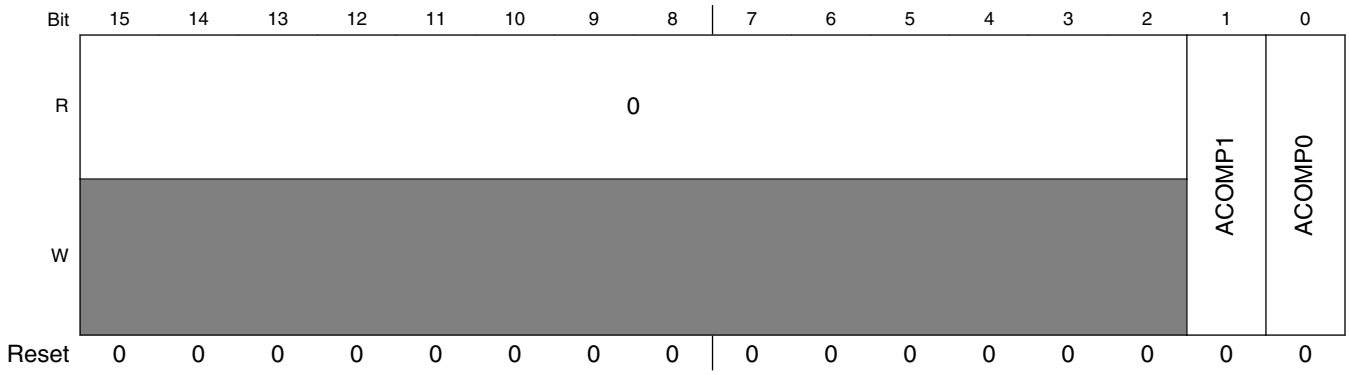
21.3.2.6 MTB_DWT Trace Buffer Control Register (MTBDWT_TBCTRL)

The MTBDWT_TBCTRL register defines how the watchpoint comparisons control the actual trace buffer operation.

Recall the MTB supports starting and stopping the program trace based on the watchpoint comparisons signaled via TSTART and TSTOP. The watchpoint comparison signals are enabled in the MTB's control logic by setting the appropriate enable bits, MTB_MASTER[TSTARTEN, TSTOPEN]. In the event of simultaneous assertion of both TSTART and TSTOP, TSTART takes priority.

Address: F000_1000h base + 200h offset = F000_1200h





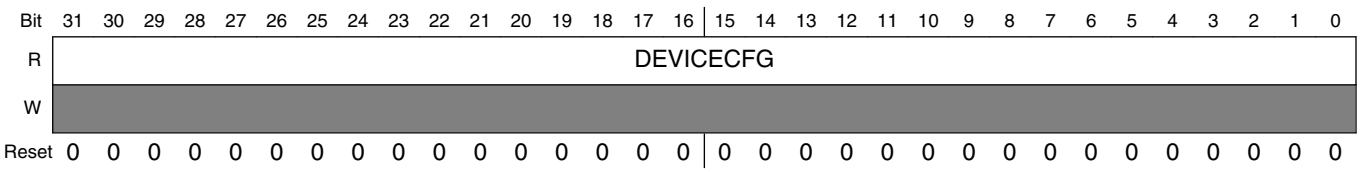
MTBDWT_TBCTRL field descriptions

Field	Description
31–28 NUMCOMP	Number of Comparators This read-only field specifies the number of comparators in the MTB_DWT. This implementation includes two registers.
27–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 ACOMP1	Action based on Comparator 1 match When the MTBDWT_FCT1[MATCHED] is set, it indicates MTBDWT_COMP1 address compare has triggered and the trace buffer's recording state is changed. 0 Trigger TSTOP based on the assertion of MTBDWT_FCT1[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWT_FCT1[MATCHED].
0 ACOMP0	Action based on Comparator 0 match When the MTBDWT_FCT0[MATCHED] is set, it indicates MTBDWT_COMP0 address compare has triggered and the trace buffer's recording state is changed. The assertion of MTBDWT_FCT0[MATCHED] is caused by the following conditions: • Address match in MTBDWT_COMP0 when MTBDWT_FCT0[DATAVMATCH] = 0 • Data match in MTBDWT_COMP0 when MTBDWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,0} • Data match in MTBDWT_COMP0 and address match in MTBDWT_COMP1 when MTBDWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,1} 0 Trigger TSTOP based on the assertion of MTBDWT_FCT0[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWT_FCT0[MATCHED].

21.3.2.7 Device Configuration Register (MTBDWT_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FC8h offset = F000_1FC8h



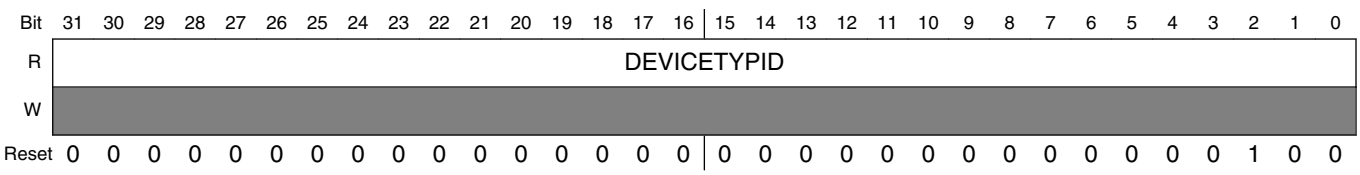
MTBDWT_DEVICECFG field descriptions

Field	Description
DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

21.3.2.8 Device Type Identifier Register (MTBDWT_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FCCh offset = F000_1FCCh



MTBDWT_DEVICETYPID field descriptions

Field	Description
DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0004.

21.3.2.9 Peripheral ID Register (MTBDWT_PERIPHID*n*)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FD0h offset + (4d × i), where i=0d to 7d

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

MTBDWT_PERIPHID*n* field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

21.3.2.10 Component ID Register (MTBDWT_COMPID*n*)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FF0h offset + (4d × i), where i=0d to 3d

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

MTBDWT_COMPID*n* field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

21.3.3 System ROM Memory Map

The System ROM Table registers are also mapped into a sparsely-populated 4 KB address space.

For core configurations like that supported by Cortex-M0+, ARM recommends that a debugger identifies and connects to the debug components using the CoreSight debug infrastructure.

ARM recommends that a debugger follows the flow as shown in the following figure to discover the components in the CoreSight debug infrastructure. In this case, a debugger reads the peripheral and component ID registers for each CoreSight component in the CoreSight system.

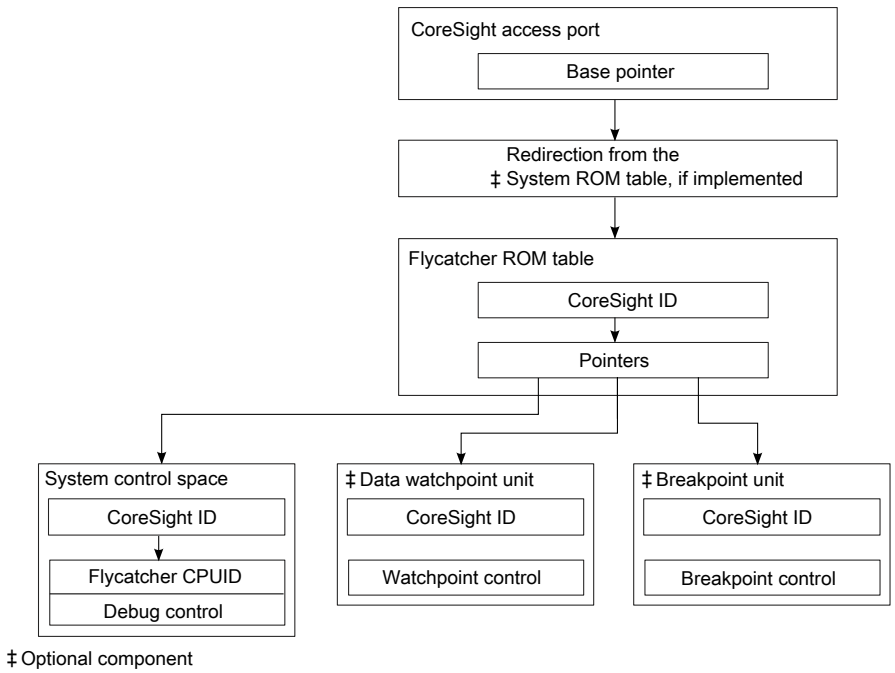


Figure 21-3. CoreSight discovery process

ROM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_2000	Entry (ROM_ENTRY0)	32	R	See section	21.3.3.1/ 381
F000_2004	Entry (ROM_ENTRY1)	32	R	See section	21.3.3.1/ 381
F000_2008	Entry (ROM_ENTRY2)	32	R	See section	21.3.3.1/ 381
F000_200C	End of Table Marker Register (ROM_TABLEMARK)	32	R	0000_0000h	21.3.3.2/ 382
F000_2FCC	System Access Register (ROM_SYSACCESS)	32	R	0000_0001h	21.3.3.3/ 382
F000_2FD0	Peripheral ID Register (ROM_PERIPHID4)	32	R	See section	21.3.3.4/ 383

Table continues on the next page...

ROM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_2FD4	Peripheral ID Register (ROM_PERIPHID5)	32	R	See section	21.3.3.4/ 383
F000_2FD8	Peripheral ID Register (ROM_PERIPHID6)	32	R	See section	21.3.3.4/ 383
F000_2FDC	Peripheral ID Register (ROM_PERIPHID7)	32	R	See section	21.3.3.4/ 383
F000_2FE0	Peripheral ID Register (ROM_PERIPHID0)	32	R	See section	21.3.3.4/ 383
F000_2FE4	Peripheral ID Register (ROM_PERIPHID1)	32	R	See section	21.3.3.4/ 383
F000_2FE8	Peripheral ID Register (ROM_PERIPHID2)	32	R	See section	21.3.3.4/ 383
F000_2FEC	Peripheral ID Register (ROM_PERIPHID3)	32	R	See section	21.3.3.4/ 383
F000_2FF0	Component ID Register (ROM_COMPID0)	32	R	See section	21.3.3.5/ 383
F000_2FF4	Component ID Register (ROM_COMPID1)	32	R	See section	21.3.3.5/ 383
F000_2FF8	Component ID Register (ROM_COMPID2)	32	R	See section	21.3.3.5/ 383
F000_2FFC	Component ID Register (ROM_COMPID3)	32	R	See section	21.3.3.5/ 383

21.3.3.1 Entry (ROM_ENTRY*n*)

The System ROM Table begins with "n" relative 32-bit addresses, one for each debug component present in the device and terminating with an all-zero value signaling the end of the table at the "n+1"-th value.

It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + 0h offset + (4d × i), where i=0d to 2d

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

ROM_ENTRY*n* field descriptions

Field	Description
ENTRY	ENTRY

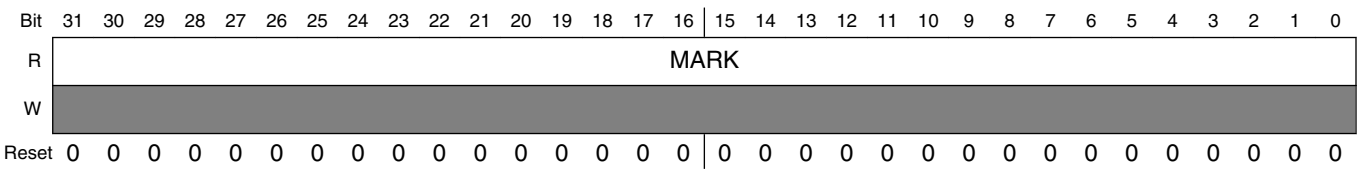
ROM_ENTRY*n* field descriptions (continued)

Field	Description
	Entry 0 (MTB) is hardwired to 0xFFFF_E003; Entry 1 (MTBDWT) to 0xFFFF_F003; Entry 2 (CM0+ ROM Table) to 0xF00F_D003.

21.3.3.2 End of Table Marker Register (ROM_TABLEMARK)

This register indicates end of table marker. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + Ch offset = F000_200Ch



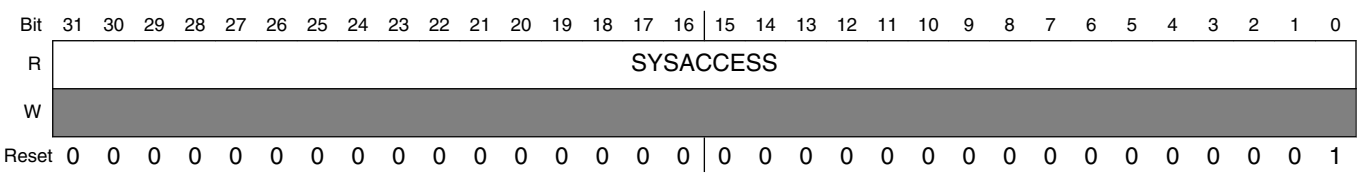
ROM_TABLEMARK field descriptions

Field	Description
MARK	MARK Hardwired to 0x0000_0000

21.3.3.3 System Access Register (ROM_SYSACCESS)

This register indicates system access. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FCCh offset = F000_2FCCh



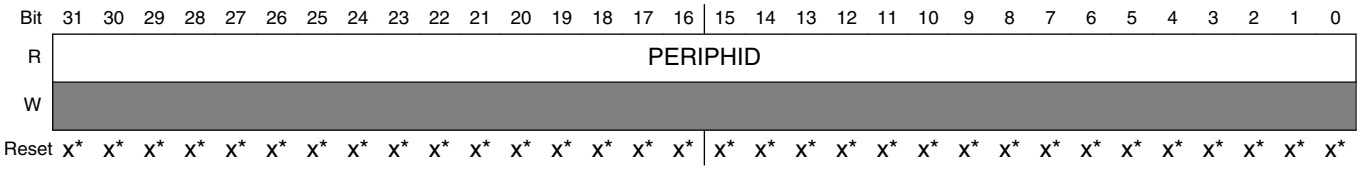
ROM_SYSACCESS field descriptions

Field	Description
SYSACCESS	SYSACCESS Hardwired to 0x0000_0001

21.3.3.4 Peripheral ID Register (ROM_PERIPHID*n*)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FD0h offset + (4d × i), where i=0d to 7d



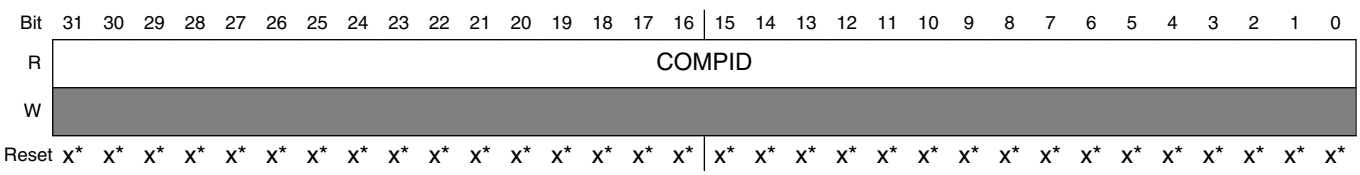
ROM_PERIPHID*n* field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

21.3.3.5 Component ID Register (ROM_COMPID*n*)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FF0h offset + (4d × i), where i=0d to 3d



ROM_COMPID*n* field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0010; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

Chapter 22
Crossbar Switch Lite (AXBS-Lite)

22.1 Chip-specific AXBS-Lite information

22.1.1 Overview

The AHB Crossbar Switch (or AXBS) allows connection between multiple masters with different slaves and performs arbitration between them. On this device the AXBS connects two masters with three slaves. The following figure shows the AXBS configuration.

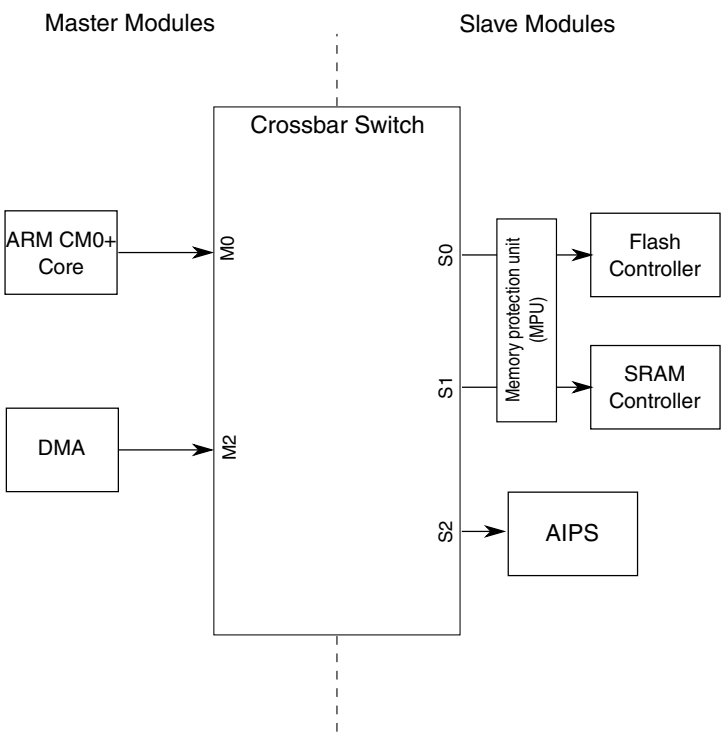


Figure 22-1. AXBS Configuration

22.1.2 Crossbar Switch Master Assignments

The master assignment is as follows:

Table 22-1. AXBS Master Assignments

Master Module	Master Port Number
ARM®Cortex®-M0+ Core	0
Reserved	1
DMA	2

22.1.3 Crossbar Switch Slave Assignments

This device contains 3 slaves connected to the crossbar switch. Flash memory and SRAM controller modules are protected by MPU. AIPS peripheral bridge is not protected by MPU as it has its own protection built-in. The slave assignment is as follows:

Table 22-2. AXBS Slave Assignments

Slave Module	Slave Port Number
Flash Memory Controller	0
SRAM Controller	1
AIPS (Peripheral Bridge)	2

22.2 Introduction

The information found here provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows up to four bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave.

22.2.1 Features

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves
- 32-bit data bus
- Operation at a 1-to-1 clock frequency with the bus masters
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).

22.3 Memory Map / Register Definition

This crossbar switch is designed for minimal gate count. It, therefore, has no memory-mapped configuration registers.

Please see the chip-specific information for information on whether the arbitration method in the crossbar switch is programmable, and by which module.

22.4 Functional Description

22.4.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it simply waits.

A master is given control of the targeted slave port only after a previous access to a different slave port completes, regardless of its priority on the newly targeted slave port. This prevents deadlock from occurring when:

- A higher priority master has:

    - An outstanding request to one slave port that has a long response time and
    - A pending access to a different slave port, and
- A lower priority master is also making a request to the same slave port as the pending access of the higher priority master.

After the master has control of the slave port it is targeting, the master remains in control of the slave port until it relinquishes the slave port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it remains parked with the last master to use the slave port. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port.

22.4.2 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round-robin

The selection of the global slave port arbitration is controlled by the MCM module. For fixed priority, set MCM_PLACR[ARB] or MCM_CPCR[CBRR] to 0. For round robin, set MCM_PLACR[ARB] or MCM_CPCR[CBRR] to 1. This arbitration setting applies to all slave ports.

22.4.2.1 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (for example, in a system with 5 masters, master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the proper master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port:

Table 22-3. How the Crossbar Switch grants control of a slave port to a master

When	Then the Crossbar Switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running an undefined length burst transfer. • The requesting master's priority level is higher than that of the current master.	At the next arbitration point for the undefined length burst transfer **NOTE:** Arbitration points for an undefined length burst are defined in the MGPCR for each master.
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

22.4.2.2 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and master 0, 4, and 5 make simultaneous requests, they are serviced in the order: 4 then 5 then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) as the fixed master priority does not affect the master selection.

22.5 Initialization/application information

No initialization is required for the crossbar switch.

See the AXBS section of the configuration chapter for the reset state of the arbitration scheme.

Chapter 23
Peripheral Bridge (AIPS-Lite)

23.1 Introduction

The peripheral bridge converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

The peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB. (It might be possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

23.1.1 Features

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Supports a pair of 32-bit transactions for selected 64-bit memory accesses
- Programming model provides memory protection functionality

23.1.2 General operation

The slave devices connected to the peripheral bridge are modules which contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge. The peripheral bridge performs a bus protocol conversion of the master transactions and generates the following as inputs to the peripherals:

- Module enables
- Module addresses
- Transfer attributes

- Byte enables
- Write data

The peripheral bridge selects and captures read data from the peripheral interface and returns it to the crossbar switch.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map.

The AIPS-Lite module uses the data width of accessed peripheral to perform proper data byte lane routing; bus decomposition (bus sizing) is performed when the access size is larger than the peripheral's data width.

23.2 Memory map/register definition

The 32-bit peripheral bridge registers can be accessed only in secure, privileged mode. Additionally, these registers must be read from or written to only by a 32-bit aligned access. The peripheral bridge registers are mapped into the Peripheral Access Control Register A PACRA[PACR0] address space.

AIPS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_0020	Peripheral Access Control Register (AIPS_PACRA)	32	R/W	See section	23.2.1/393
4000_0024	Peripheral Access Control Register (AIPS_PACRB)	32	R/W	See section	23.2.1/393
4000_0040	Peripheral Access Control Register (AIPS_PACRE)	32	R/W	See section	23.2.2/396
4000_0044	Peripheral Access Control Register (AIPS_PACRF)	32	R/W	See section	23.2.2/396
4000_0048	Peripheral Access Control Register (AIPS_PACRG)	32	R/W	See section	23.2.2/396
4000_004C	Peripheral Access Control Register (AIPS_PACRH)	32	R/W	See section	23.2.2/396
4000_0050	Peripheral Access Control Register (AIPS_PACRI)	32	R/W	See section	23.2.2/396
4000_0054	Peripheral Access Control Register (AIPS_PACRJ)	32	R/W	See section	23.2.2/396
4000_0058	Peripheral Access Control Register (AIPS_PACRK)	32	R/W	See section	23.2.2/396
4000_005C	Peripheral Access Control Register (AIPS_PACRL)	32	R/W	See section	23.2.2/396
4000_0060	Peripheral Access Control Register (AIPS_PACRM)	32	R/W	See section	23.2.2/396
4000_0064	Peripheral Access Control Register (AIPS_PACRN)	32	R/W	See section	23.2.2/396
4000_0068	Peripheral Access Control Register (AIPS_PACRO)	32	R/W	See section	23.2.2/396
4000_006C	Peripheral Access Control Register (AIPS_PACRP)	32	R/W	See section	23.2.2/396

23.2.1 Peripheral Access Control Register (AIPS_PACR*n*)

Each peripheral on this chip has a 4-bit PACR field which includes a read-only "lock" bit plus a 3-bit attribute check field which defines the access rights (privileged/user, secure/nonsecure) needed to reference the slave peripheral's address space. Access to these registers is allowed using only 32-bit privileged, secure reads and writes.

Each 32-bit register controls eight peripherals.

A peripheral's assignment to a PACR field is defined by the memory map slot to which the peripheral is assigned. See this chip's peripheral bridge memory map information for the assignment of a particular peripheral.

The following table shows the top-level structure of PACRs.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x20	PACRA	PACR0	PACR1	PACR2	PACR3	PACR4	PACR5	PACR6	PACR7
0x24	PACRB	PACR8	PACR9	PACR10	PACR11	PACR12	PACR13	PACR14	PACR15

The following table defines the attribute check values for each AC field.

NOTE

For access to peripherals that have their own access attribute checks, such as GPIO, refer also to their attributes checks definitions.

Table 23-1. Attribute check (AC) values

AC	User nonsecure	User secure	Privileged secure
0b000	Read/Write	Read/Write	Read/Write
0b001	Read	Read/Write	Read/Write
0b010	None	Read/Write	Read/Write
0b011	Read	Read	Read/Write
0b100	None	Read	Read/Write
0b101	None	None	Read/Write
0b110	None	None	Read
0b111	None	None	None

Address: 4000_0000h base + 20h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RO0	AC0			RO1	AC1			RO2	AC2			RO3	AC3		
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RO4	AC4			RO5	AC5			RO6	AC6			RO7	AC7		
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset value is chip-dependent and can be found in the AIPS chip-specific information.

AIPS_PACR*n* field descriptions

Field	Description
31 RO0	Read Only Indicates the lock status of this register's AC0 field. After this field is set to 1, it cannot be changed and attempted writes to AC0 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
30–28 AC0	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
27 RO1	Read Only Indicates the lock status of this register's AC1 field. After this field is set to 1, it cannot be changed and attempted writes to AC1 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
26–24 AC1	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
23 RO2	Read Only Indicates the lock status of this register's AC2 field. After this field is set to 1, it cannot be changed and attempted writes to AC2 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
22–20 AC2	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
19 RO3	Read Only Indicates the lock status of this register's AC3 field. After this field is set to 1, it cannot be changed and attempted writes to AC3 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
18–16 AC3	Attribute Check

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
15 RO4	Read Only Indicates the lock status of this register's AC4 field. After this field is set to 1, it cannot be changed and attempted writes to AC4 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
14–12 AC4	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
11 RO5	Read Only Indicates the lock status of this register's AC5 field. After this field is set to 1, it cannot be changed and attempted writes to AC5 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
10–8 AC5	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
7 RO6	Read Only Indicates the lock status of this register's AC6 field. After this field is set to 1, it cannot be changed and attempted writes to AC6 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
6–4 AC6	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
3 RO7	Read Only Indicates the lock status of this register's AC7 field. After this field is set to 1, it cannot be changed and attempted writes to AC7 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
AC7	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.

23.2.2 Peripheral Access Control Register (AIPS_PACR*n*)

Each peripheral on this chip has a 4-bit PACR field which includes a read-only “lock” bit plus a 3-bit attribute check field which defines the access rights (privileged/user, secure/nonsecure) needed to reference the slave peripheral’s address space. Access to these registers is allowed using only 32-bit privileged, secure reads and writes.

Each 32-bit register controls eight peripherals. PACRE-PACR P control access to the 96 off-platform peripherals.

A peripheral's assignment to a PACR field is defined by the memory map slot to which the peripheral is assigned. See this device's peripheral bridge memory map information for the assignment of a particular peripheral.

The following table shows the top-level structure of PACRs.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x40	PACRE	PACR32	PACR33	PACR34	PACR35	PACR36	PACR37	PACR38	PACR39
0x44	PACRF	PACR40	PACR41	PACR42	PACR43	PACR44	PACR45	PACR46	PACR47
0x48	PACRG	PACR48	PACR49	PACR50	PACR51	PACR52	PACR53	PACR54	PACR55
0x4C	PACRH	PACR56	PACR57	PACR58	PACR59	PACR60	PACR61	PACR62	PACR63
0x50	PACRI	PACR64	PACR65	PACR66	PACR67	PACR68	PACR69	PACR70	PACR71
0x54	PACRJ	PACR72	PACR73	PACR74	PACR75	PACR76	PACR77	PACR78	PACR79
0x58	PACRK	PACR80	PACR81	PACR82	PACR83	PACR84	PACR85	PACR86	PACR87
0x5C	PACRL	PACR88	PACR89	PACR90	PACR91	PACR92	PACR93	PACR94	PACR95
0x60	PACRM	PACR96	PACR97	PACR98	PACR99	PACR100	PACR101	PACR102	PACR103
0x64	PACRN	PACR104	PACR105	PACR106	PACR107	PACR108	PACR109	PACR110	PACR111
0x68	PACRO	PACR112	PACR113	PACR114	PACR115	PACR116	PACR117	PACR118	PACR119
0x6C	PACRP	PACR120	PACR121	PACR122	PACR123	PACR124	PACR125	PACR126	PACR127

Address: 4000_0000h base + 40h offset + (4d × i), where i=0d to 11d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RO0	AC0			RO1	AC1			RO2	AC2			RO3	AC3		
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RO4	AC4			RO5	AC5			RO6	AC6			RO7	AC7		
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset value is chip-dependent and can be found in the AIPS chip-specific information.

AIPS_PACR*n* field descriptions

Field	Description
31 RO0	Read Only Indicates the lock status of this register's AC0 field. After this field is set to 1, it cannot be changed and attempted writes to AC0 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
30–28 AC0	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
27 RO1	Read Only Indicates the lock status of this register's AC1 field. After this field is set to 1, it cannot be changed and attempted writes to AC1 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
26–24 AC1	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
23 RO2	Read Only Indicates the lock status of this register's AC2 field. After this field is set to 1, it cannot be changed and attempted writes to AC2 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
22–20 AC2	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
19 RO3	Read Only Indicates the lock status of this register's AC3 field. After this field is set to 1, it cannot be changed and attempted writes to AC3 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
18–16 AC3	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
15 RO4	Read Only Indicates the lock status of this register's AC4 field. After this field is set to 1, it cannot be changed and attempted writes to AC4 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
14–12 AC4	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
11 RO5	Read Only Indicates the lock status of this register's AC5 field. After this field is set to 1, it cannot be changed and attempted writes to AC5 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
10–8 AC5	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
7 RO6	Read Only Indicates the lock status of this register's AC6 field. After this field is set to 1, it cannot be changed and attempted writes to AC6 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
6–4 AC6	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.
3 RO7	Read Only Indicates the lock status of this register's AC7 field. After this field is set to 1, it cannot be changed and attempted writes to AC7 are ignored until the next system reset clears this field. 0 Writes to corresponding AC field are allowed. 1 Writes to corresponding AC field are ignored.
AC7	Attribute Check Defines the attributes required to access the corresponding slave peripheral's memory space. See Table 23-1, "Attribute check (AC) values" for details.

23.3 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

Support is provided for generating a pair of 32-bit slave accesses when performing certain 64-bit peripheral accesses.

The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

23.3.1 Access support

All combinations of access size and peripheral data port width are supported. An access that is larger than the target peripheral's data width will be decomposed to multiple, smaller accesses. Bus decomposition is terminated by a transfer error caused by an access to an empty register area.

Chapter 24
DMA Controller Module

24.1 Introduction

Information found here describes the direct memory access (DMA) controller module. It provides an overview of the module and describes in detail its signals and programming model.

The latter sections of this chapter describe operations, features, and supported data transfer modes in detail.

An example of using several features of the DMA module is described in AN4631: Using the Asynchronous DMA features of the Kinetis L Series.

Note

The designation *n* is used throughout this section to refer to registers or signals associated with one of the four identical DMA channels: DMA0, DMA1, DMA2, or DMA3.

24.1.1 Overview

The DMA controller module enables fast transfers of data, providing an efficient way to move blocks of data with minimal processor interaction. The DMA module, shown in the following figure, has four channels that allow 8-bit, 16-bit, or 32-bit data transfers. Each channel has a dedicated Source Address register (SAR*n*), Destination Address register (DAR*n*), Status register (DSR*n*), Byte Count register (BCR*n*), and Control register (DCR*n*). Collectively, the combined program-visible registers associated with each channel define a transfer control descriptor (TCD). All transfers are dual address, moving data from a source memory location to a destination memory location with the module operating as a 32-bit bus master connected to the system bus. The programming model is accessed through a 32-bit connection with the slave peripheral bus. DMA data transfers may be explicitly initiated by software or by peripheral hardware requests.

The following figure is a simplified block diagram of the 4-channel DMA controller.

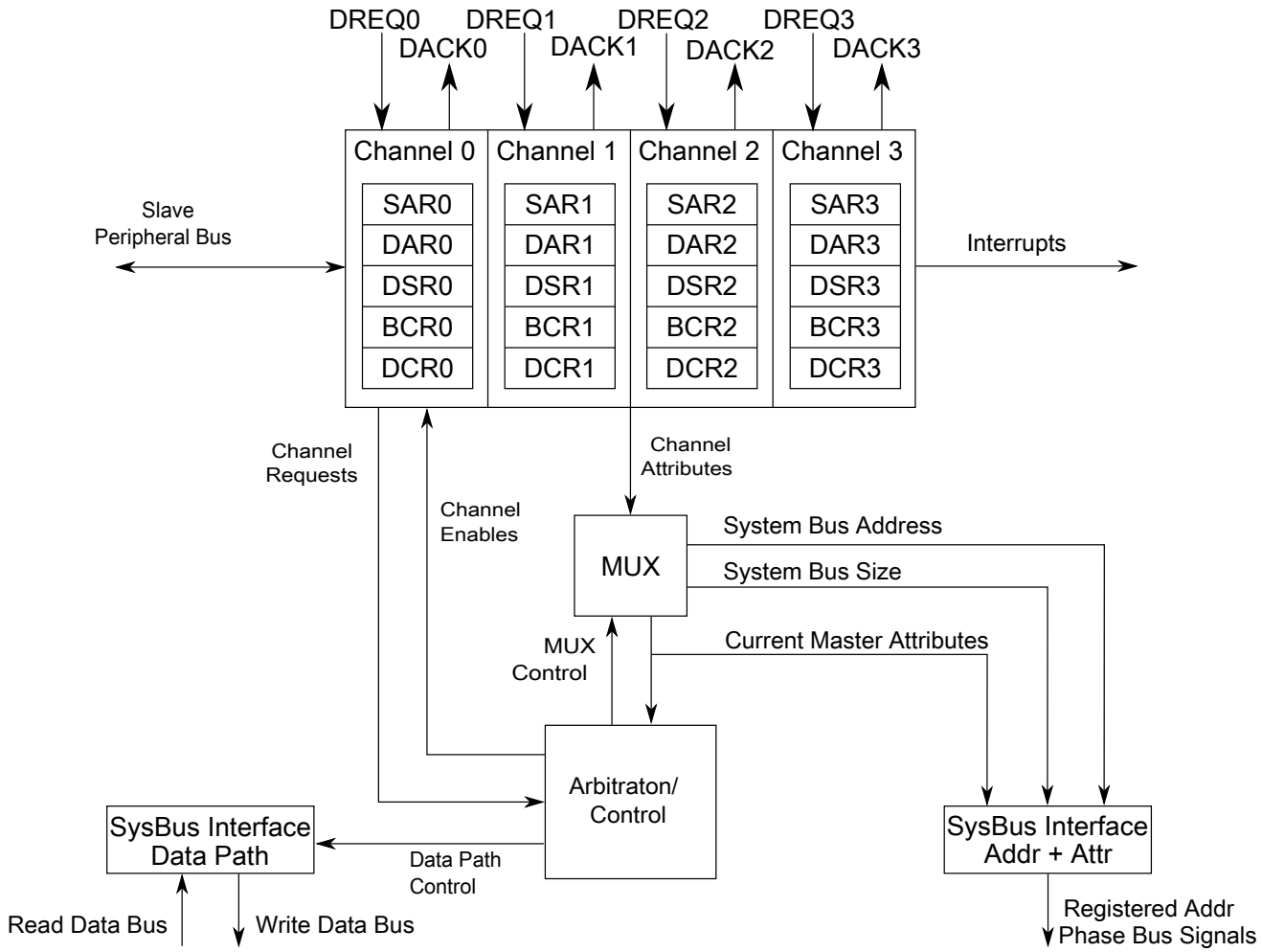


Figure 24-1. 4-Channel DMA Block Diagram

The terms *peripheral request* and *DREQ* refer to a DMA request from one of the on-chip peripherals or package pins. The DMA provides hardware handshake signals: either a DMA acknowledge (DACK) or a done indicator back to the peripheral.

24.1.2 Features

The DMA controller module features:

- Four independently programmable DMA controller channels
- Dual-address transfers via 32-bit master connection to the system bus
- Data transfers in 8-, 16-, or 32-bit blocks
- Continuous-mode or cycle-steal transfers from software or peripheral initiation

- Automatic hardware acknowledge/done indicator from each channel
- Independent source and destination address registers
- Optional modulo addressing and automatic updates of source and destination addresses
- Independent transfer sizes for source and destination
- Optional auto-alignment feature for source or destination accesses
- Optional automatic single or double channel linking
- Programming model accessed via 32-bit slave peripheral bus
- Channel arbitration on transfer boundaries using fixed priority scheme
- Programmable support for channel access control and attribute generation

24.2 DMA Transfer Overview

The DMA module can move data within system memory (including memory and peripheral devices) with minimal processor intervention, greatly improving overall system performance.

The DMA module consists of four independent, functionally equivalent channels, so references to DMA in this chapter apply to any of the channels. It is not possible to address all four channels at once.

As soon as a channel has been initialized, it may be started by setting DCR*n*[START] or a properly-selected peripheral DMA request, depending on the status of DCR*n*[ERQ].

The DMA controller supports dual-address transfers using its bus master connection to the system bus. The DMA channels support transfers up to 32 data bits in size and have the same memory map addressibility as the processor.

- Dual-address transfers—A dual-address transfer consists of a read followed by a write and is initiated by a request using the DCRn[START] bit or by a peripheral DMA request. The read data is temporarily held in the DMA channel hardware until the write operation. Two types of single transfers occur: a read from a source address followed by a write to a destination address. See the following figure.

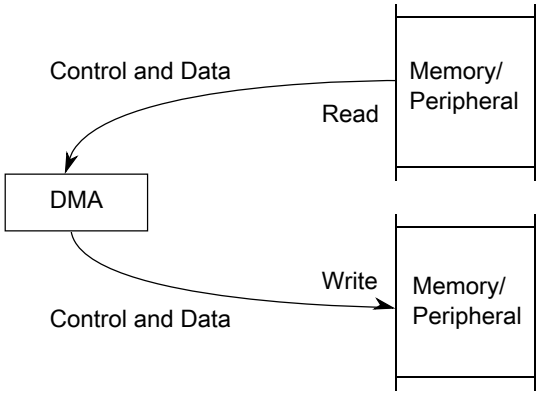


Figure 24-2. Dual-Address Transfer

Any operation involving a DMA channel follows the same three steps:

1. Channel initialization—The transfer control descriptor, contained in the channel registers, is loaded with address pointers, a byte-transfer count, and control information using accesses from the slave peripheral bus.
2. Data transfer—The DMA accepts requests for data transfers. Upon receipt of a request, it provides address and bus control for the transfers via its master connection to the system bus and temporary storage for the read data. The channel performs one or more source read and destination write data transfers.
3. Channel termination—Occurs after the operation is finished successfully or due to an error. The channel indicates the operation status in the channel's DSR, described in the definitions of the DMA Status Registers (DSRn) and Byte Count Registers (BCRn).

24.3 Memory Map/Register Definition

Information about the registers related to the DMA controller module can be found here.

Descriptions of each register and its bit assignments follow. Modifying DMA control registers during a transfer can result in undefined operation. The following table shows the mapping of DMA controller registers. The DMA programming model is accessed via the slave peripheral bus. The concatenation of the source and destination address registers, the status and byte count register, and the control register create a 128-bit transfer control descriptor (TCD) that defines the operation of each DMA channel.

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_8100	Source Address Register (DMA_SAR0)	32	R/W	0000_0000h	24.3.1/405
4000_8104	Destination Address Register (DMA_DAR0)	32	R/W	0000_0000h	24.3.2/406
4000_8108	DMA Status Register / Byte Count Register (DMA_DSR_BCR0)	32	R/W	0000_0000h	24.3.3/407
4000_810C	DMA Control Register (DMA_DCR0)	32	R/W	0000_0000h	24.3.4/409
4000_8110	Source Address Register (DMA_SAR1)	32	R/W	0000_0000h	24.3.1/405
4000_8114	Destination Address Register (DMA_DAR1)	32	R/W	0000_0000h	24.3.2/406
4000_8118	DMA Status Register / Byte Count Register (DMA_DSR_BCR1)	32	R/W	0000_0000h	24.3.3/407
4000_811C	DMA Control Register (DMA_DCR1)	32	R/W	0000_0000h	24.3.4/409
4000_8120	Source Address Register (DMA_SAR2)	32	R/W	0000_0000h	24.3.1/405
4000_8124	Destination Address Register (DMA_DAR2)	32	R/W	0000_0000h	24.3.2/406
4000_8128	DMA Status Register / Byte Count Register (DMA_DSR_BCR2)	32	R/W	0000_0000h	24.3.3/407
4000_812C	DMA Control Register (DMA_DCR2)	32	R/W	0000_0000h	24.3.4/409
4000_8130	Source Address Register (DMA_SAR3)	32	R/W	0000_0000h	24.3.1/405
4000_8134	Destination Address Register (DMA_DAR3)	32	R/W	0000_0000h	24.3.2/406
4000_8138	DMA Status Register / Byte Count Register (DMA_DSR_BCR3)	32	R/W	0000_0000h	24.3.3/407
4000_813C	DMA Control Register (DMA_DCR3)	32	R/W	0000_0000h	24.3.4/409

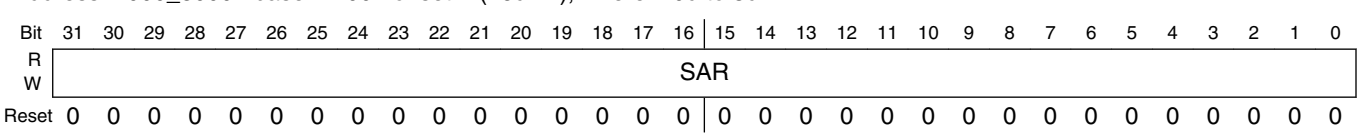
24.3.1 Source Address Register (DMA_SAR*n*)

Restriction

For this register:

- Only 32-bit writes are allowed. 16-bit and 8-bit writes result in a bus error.
- Only several values are allowed to be written to bits 31-20 of this register, see the value list in the field description. A write of any other value to these bits causes a configuration error when the channel starts to execute. For more information about the configuration error, see the description of the CE field of DSR.

Address: 4000_8000h base + 100h offset + (16d × i), where i=0d to 3d



DMA_SAR*n* field descriptions

Field	Description
SAR	SAR Each SAR contains the byte address used by the DMA controller to read data. The SARn is typically aligned on a 0-modulo-ssize boundary—that is, on the natural alignment of the source data. **Restriction:** Bits 31-20 of this register must be written with one of only several allowed values. Each of these allowed values corresponds to a valid region of the device's memory map. The allowed values are: • 0x000x_xxxx • 0x1FFx_xxxx • 0x200x_xxxx • 0x400x_xxxx • 0xF00x_xxxx After being written with one of the allowed values, bits 31-20 read back as the written value. After being written with any other value, bits 31-20 read back as an indeterminate value.

24.3.2 Destination Address Register (DMA_DAR*n*)

Restriction

For this register:

- Only 32-bit writes are allowed. 16-bit and 8-bit writes result in a bus error.
- Only several values are allowed to be written to bits 31-20 of this register, see the value list in the field description. A write of any other value to these bits causes a configuration error when the channel starts to execute. For more information about the configuration error, see the description of the CE field of DSR.

Address: 4000_8000h base + 104h offset + (16d × i), where i=0d to 3d

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

DMA_DAR*n* field descriptions

Field	Description
DAR	DAR Each DAR contains the byte address used by the DMA controller to write data. The DARn is typically aligned on a 0-modulo-dsize boundary—that is, on the natural alignment of the destination data.

DMA_DAR*n* field descriptions (continued)

Field	Description
	Restriction: Bits 31-20 of this register must be written with one of only several allowed values. Each of these allowed values corresponds to a valid region of the device's memory map. The allowed values are: • 0x000x_xxxx • 0x1FFx_xxxx • 0x200x_xxxx • 0x400x_xxxx • 0xF00x_xxxx After being written with one of the allowed values, bits 31-20 read back as the written value. After being written with any other value, bits 31-20 read back as an indeterminate value.

24.3.3 DMA Status Register / Byte Count Register (DMA_DSR_BCR*n*)

DSR and BCR are two logical registers that occupy one 32-bit address. DSRn occupies bits 31–24, and BCRn occupies bits 23–0. DSRn contains flags indicating the channel status, and BCRn contains the number of bytes yet to be transferred for a given block.

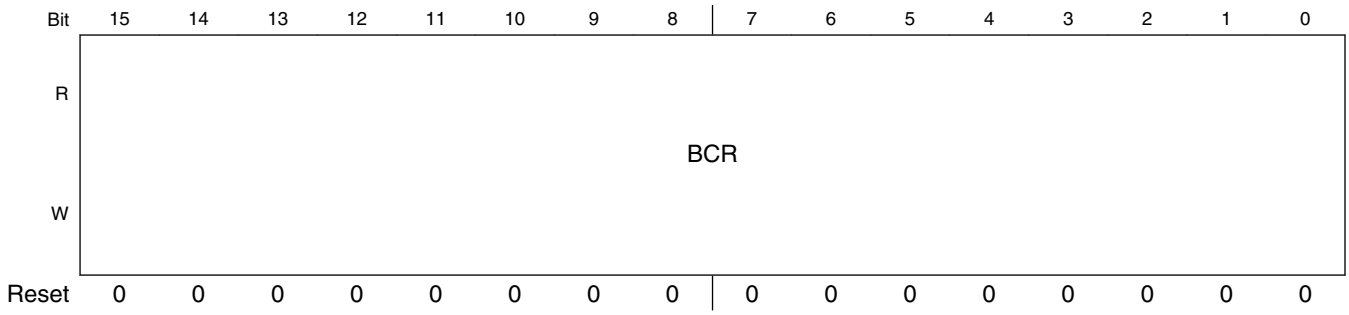
On the successful completion of the write transfer, BCRn decrements by 1, 2, or 4 for 8-bit, 16-bit, or 32-bit accesses, respectively. BCRn is cleared if a 1 is written to DSR[DONE].

In response to an event, the DMA controller writes to the appropriate DSRn bit. Only a write to DSRn[DONE] results in action. DSRn[DONE] is set when the block transfer is complete.

When a transfer sequence is initiated and BCRn[BCR] is not a multiple of 4 or 2 when the DMA is configured for 32-bit or 16-bit transfers, respectively, DSRn[CE] is set and no transfer occurs.

Address: 4000_8000h base + 108h offset + (16d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CE	BES	BED	0	REQ	BSY	DONE	BCR							
W								w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



DMA_DSR_BCR*n* field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 CE	Configuration Error Any of the following conditions causes a configuration error: • BCR, SAR, or DAR does not match the requested transfer size. • A value greater than 0F_FFFFh is written to BCR. • Bits 31-20 of SAR or DAR are written with a value other than one of the allowed values. See SAR and DAR. • SSIZE or DSIZE is set to an unsupported value. • BCR equals 0 when the DMA receives a start condition. CE is cleared at hardware reset or by writing a 1 to DONE. 0 No configuration error exists. 1 A configuration error has occurred.
29 BES	Bus Error on Source BES is cleared at hardware reset or by writing a 1 to DONE. 0 No bus error occurred. 1 The DMA channel terminated with a bus error during the read portion of a transfer.
28 BED	Bus Error on Destination BED is cleared at hardware reset or by writing a 1 to DONE. 0 No bus error occurred. 1 The DMA channel terminated with a bus error during the write portion of a transfer.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 REQ	Request 0 No request is pending or the channel is currently active. Cleared when the channel is selected. 1 The DMA channel has a transfer remaining and the channel is not selected.
25 BSY	Busy 0 DMA channel is inactive. Cleared when the DMA has finished the last transaction. 1 BSY is set the first time the channel is enabled after a transfer is initiated.
24 DONE	Transactions Done

Table continues on the next page...

DMA_DSR_BCR*n* field descriptions (continued)

Field	Description
	Set when all DMA controller transactions complete as determined by transfer count, or based on error conditions. When BCR reaches 0, DONE is set when the final transfer completes successfully. DONE can also be used to abort a transfer by resetting the status bits. When a transfer completes, software must clear DONE before reprogramming the DMA. 0 DMA transfer is not yet complete. Writing a 0 has no effect. 1 DMA transfer completed. Writing a 1 to this bit clears all DMA status bits and should be used in an interrupt service routine to clear the DMA interrupt and error bits.
BCR	BCR This field contains the number of bytes yet to be transferred for a given block. **Restriction:** BCR must be written with a value equal to or less than 0F_FFFFh. After being written with a value in this range, bits 23-20 of BCR read back as 0000b. A write to BCR of a value greater than 0F_FFFFh causes a configuration error when the channel starts to execute. After being written with a value in this range, bits 23-20 of BCR read back as 0001b.

24.3.4 DMA Control Register (DMA_DCR*n*)

Address: 4000_8000h base + 10Ch offset + (16d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EINT	ERQ	CS	AA	CHACR		UMNSM		EADREQ	SINC	SSIZE		DINC	DSIZE		0
W																START
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SMOD				DMOD				D_REQ	0	LINKCC		LCH1		LCH2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_DCR*n* field descriptions

Field	Description
31 EINT	Enable Interrupt on Completion of Transfer

Table continues on the next page...

DMA_DCR*n* field descriptions (continued)

Field	Description
	Determines whether an interrupt is generated by completing a transfer or by the occurrence of an error condition. 0 No interrupt is generated. 1 Interrupt signal is enabled.
30 ERQ	Enable Peripheral Request **CAUTION:** Be careful: a collision can occur between START and D_REQ when ERQ is 1. 0 Peripheral request is ignored. 1 Enables peripheral request to initiate transfer. A software-initiated request (setting START) is always enabled.
29 CS	Cycle Steal 0 DMA continuously makes read/write transfers until the BCR decrements to 0. 1 Forces a single read/write transfer per request.
28 AA	Auto-align AA and SIZE bits determine whether the source or destination is auto-aligned; that is, transfers are optimized based on the address and size. 0 Auto-align disabled 1 If SSIZE indicates a transfer no smaller than DSIZE, source accesses are auto-aligned; otherwise, destination accesses are auto-aligned. Source alignment takes precedence over destination alignment. If auto-alignment is enabled, the appropriate address register increments, regardless of DINC or SINC.
27–26 CHACR	Channel Access Control This 2-bit field defines the access control mode needed to reference the channel's transfer channel descriptor (TCDn). When a read or write access occurs to any fields in the TCDn, if the system mode has sufficient privileges as defined by CHACR, the access completes. Otherwise, the access aborts and terminates in an error. **Restriction:** The CHACR field can be changed only during operation in privileged secure mode. In any other mode, attempted changes to this field are not performed, and the access terminates with an error. The following table defines the access of each mode for each value of CHACR.

CHACR field	Access allowed to each processor mode		
	Privileged, Secure	User, Secure	User, Nonsecure
00	Read + Write	Read + Write	Read + Write
01	Read + Write	Read + Write	— (none)
1x	Read + Write	— (none)	— (none)

25–24 UMNSM	User Mode, Nonsecure Mode This 2-bit field defines the privileged/user and secure/nonsecure attributes for the DMA channel as it executes. The contents of the write data operand selects how the value is to be loaded into this field.

Table continues on the next page...

DMA_DCR*n* field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
When a given channel is activated and executes, the DMA generates the appropriate privileged/user and secure/nonsecure attributes for all source reads and destination writes using UMNSM.

The following table defines the association between the write data operand, the initial system mode, and the resulting "next state" of UMNSM.
<table>
<tr><th rowspan="2">Write to UMNSM field</th><th colspan="3">Initial processor mode and result of UMNSM write</th></tr>
<tr><th>Privileged, Secure</th><th>User, Secure</th><th>User, Nonsecure</th></tr>
<tr><td>00</td><td>00 = {Privileged, Secure}</td><td>10 = {User, Secure}</td><td>11 = {User, Nonsecure}</td></tr>
<tr><td>01</td><td>00 = {Privileged, Secure}</td><td>No change; error terminated</td><td>No change; error terminated</td></tr>
<tr><td>10</td><td>10 = {User, Secure}</td><td>10 = {User, Secure}</td><td>No change; error terminated</td></tr>
<tr><td>11</td><td>11 = {User, Nonsecure}</td><td>11 = {User, Nonsecure}</td><td>11 = {User, Nonsecure}</td></tr>
</table>
00 Channel attributes are set to the current mode.
01 If the current mode is privileged and secure, then attributes are set to {privileged, secure}. Otherwise, writing this value terminates in an error.
10 If the current mode is privileged and secure or if the current mode is user and secure, then attributes are set to {user, secure}. Otherwise, writing this value terminates in an error.
11 If the current mode is privileged and secure, user and secure, or user and nonsecure, then attributes are set to {user, nonsecure}.
</td></tr>
<tr><td>23
EADREQ</td><td>
Enable asynchronous DMA requests

Enables the channel to support asynchronous DREQs while the MCU is in Stop mode.

0 Disabled
1 Enabled
</td></tr>
<tr><td>22
SINC</td><td>
Source Increment

Controls whether the source address increments after each successful transfer.

0 No change to SAR after a successful transfer.
1 The SAR increments by 1, 2, 4 as determined by the transfer size.
</td></tr>
<tr><td>21–20
SSIZE</td><td>
Source Size

Determines the data size of the source bus cycle for the DMA controller.

00 32-bit
01 8-bit
10 16-bit
11 Reserved (generates a configuration error (DSRn[CE]) if incorrectly specified at time of channel activation)
</td></tr>
<tr><td>19
DINC</td><td>
Destination Increment

Controls whether the destination address increments after each successful transfer.

0 No change to the DAR after a successful transfer.
1 The DAR increments by 1, 2, 4 depending upon the size of the transfer.
</td></tr>
</table>

Table continues on the next page...

DMA_DCR*n* field descriptions (continued)

Field	Description
18–17 DSIZE	Destination Size Determines the data size of the destination bus cycle for the DMA controller. 00 32-bit 01 8-bit 10 16-bit 11 Reserved (generates a configuration error (DSRn[CE]) if incorrectly specified at time of channel activation)
16 START	Start Transfer 0 DMA inactive 1 The DMA begins the transfer in accordance to the values in the TCDn. START is cleared automatically after one module clock and always reads as logic 0.
15–12 SMOD	Source Address Modulo Defines the size of the source data circular buffer used by the DMA Controller. If enabled (SMOD is non-zero), the buffer base address is located on a boundary of the buffer size. The value of this boundary is based upon the initial source address (SAR). The base address should be aligned to a 0-modulo-(circular buffer size) boundary. Misaligned buffers are not possible. The boundary is forced to the value determined by the upper address bits in the field selection. 0000 Buffer disabled 0001 Circular buffer size is 16 bytes. 0010 Circular buffer size is 32 bytes. 0011 Circular buffer size is 64 bytes. 0100 Circular buffer size is 128 bytes. 0101 Circular buffer size is 256 bytes. 0110 Circular buffer size is 512 bytes. 0111 Circular buffer size is 1 KB. 1000 Circular buffer size is 2 KB. 1001 Circular buffer size is 4 KB. 1010 Circular buffer size is 8 KB. 1011 Circular buffer size is 16 KB. 1100 Circular buffer size is 32 KB. 1101 Circular buffer size is 64 KB. 1110 Circular buffer size is 128 KB. 1111 Circular buffer size is 256 KB.
11–8 DMOD	Destination Address Modulo Defines the size of the destination data circular buffer used by the DMA Controller. If enabled (DMOD value is non-zero), the buffer base address is located on a boundary of the buffer size. The value of this boundary depends on the initial destination address (DAR). The base address should be aligned to a 0-modulo-(circular buffer size) boundary. Misaligned buffers are not possible. The boundary is forced to the value determined by the upper address bits in the field selection. 0000 Buffer disabled 0001 Circular buffer size is 16 bytes 0010 Circular buffer size is 32 bytes 0011 Circular buffer size is 64 bytes 0100 Circular buffer size is 128 bytes

Table continues on the next page...

DMA_DCR*n* field descriptions (continued)

Field	Description
	0101 Circular buffer size is 256 bytes 0110 Circular buffer size is 512 bytes 0111 Circular buffer size is 1 KB 1000 Circular buffer size is 2 KB 1001 Circular buffer size is 4 KB 1010 Circular buffer size is 8 KB 1011 Circular buffer size is 16 KB 1100 Circular buffer size is 32 KB 1101 Circular buffer size is 64 KB 1110 Circular buffer size is 128 KB 1111 Circular buffer size is 256 KB
7 D_REQ	Disable Request DMA hardware automatically clears the corresponding DCRn[ERQ] bit when the byte count register reaches 0. 0 ERQ bit is not affected. 1 ERQ bit is cleared when the BCR is exhausted.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 LINKCC	Link Channel Control Allows DMA channels to have their transfers linked. The current DMA channel triggers a DMA request to the linked channels (LCH1 or LCH2) depending on the condition described by the LINKCC bits. If not in cycle steal mode (DCRn[CS]=0) and LINKCC equals 01 or 10, no link to LCH1 occurs. If LINKCC equals 01, a link to LCH1 is created after each cycle-steal transfer performed by the current DMA channel is completed. As the last cycle-steal is performed and the BCR reaches zero, then the link to LCH1 is closed and a link to LCH2 is created. 00 No channel-to-channel linking 01 Perform a link to channel LCH1 after each cycle-steal transfer followed by a link to LCH2 after the BCR decrements to 0. 10 Perform a link to channel LCH1 after each cycle-steal transfer 11 Perform a link to channel LCH1 after the BCR decrements to 0.
3–2 LCH1	Link Channel 1 Indicates the DMA channel assigned as link channel 1. The link channel number cannot be the same as the currently executing channel, and generates a configuration error if this is attempted (DSRn[CE] is set). 00 DMA Channel 0 01 DMA Channel 1 10 DMA Channel 2 11 DMA Channel 3
LCH2	Link Channel 2 Indicates the DMA channel assigned as link channel 2. The link channel number cannot be the same as the currently executing channel, and generates a configuration error if this is attempted (DSRn[CE] is set). 00 DMA Channel 0 01 DMA Channel 1

Table continues on the next page...

DMA_DCR*n* field descriptions (continued)

Field	Description
	10 DMA Channel 2 11 DMA Channel 3

24.4 Functional Description

In the following discussion, the term DMA request implies that DCR*n*[START] is set, or DCRn[ERQ] is set and then followed by assertion of the properly selected DMA peripheral request. DCR*n*[START] is cleared when the channel is activated.

Before initiating a dual-address access, the DMA module verifies that DCR*n*[SSIZE] and DCR*n*[DSIZE] are consistent with the source and destination addresses. If they are not consistent, the configuration error bit, DSR*n*[CE], is set. If misalignment is detected, no transfer occurs, DSR*n*[CE] is set, and, depending on the DCR configuration, an interrupt event may be issued. If the auto-align bit, DCR*n*[AA], is set, error checking is performed on the appropriate registers.

A read/write transfer sequence reads data from the source address and writes it to the destination address. The number of bytes transferred is the largest of the sizes specified by DCR*n*[SSIZE] and DCR*n*[DSIZE] in the DMA Control Registers (DCRn).

Source and destination address registers (SAR*n* and DAR*n*) can be programmed in the DCR*n* to increment at the completion of a successful transfer.

24.4.1 Transfer requests (Cycle-Steal and Continuous modes)

The DMA channel supports software-initiated or peripheral-initiated requests. A request is issued by setting DCR*n*[START] or when the selected peripheral request asserts and DCRn[ERQ] is set. Setting DCR*n*[ERQ] enables recognition of the peripheral DMA requests. Selecting between cycle-steal and continuous modes minimizes bus usage for either type of request.

- Cycle-steal mode (DCR*n*[CS] = 1)—Only one complete transfer from source to destination occurs for each request. If DCRn[ERQ] is set, the request is peripheral initiated. A software-initiated request is enabled by setting DCR*n*[START].
- Continuous mode (DCR*n*[CS] = 0)—After a software-initiated or peripheral request, the DMA continuously transfers data until BCRn reaches 0. The DMA performs the specified number of transfers, then retires the channel.

In either mode, the crossbar switch performs independent arbitration on each slave port after each transaction.

24.4.2 Channel initialization and startup

Before a data transfer starts, the channel's transfer control descriptor must be initialized with information describing configuration, request-generation method, and pointers to the data to be moved.

24.4.2.1 Channel prioritization

The four DMA channels are prioritized based on number, with channel 0 having highest priority and channel 3 having the lowest, that is, channel 0 > channel 1 > channel 2 > channel 3.

Simultaneous peripheral requests activate the channels based on this priority order. Once activated, a channel runs to completion as defined by DCR*n*[CS] and BCR*n*.

24.4.2.2 DMA Access Control

For the transfer channel descriptor (TCD*n*) of each channel, the DMA Controller can specify the required access control state needed to read or write contents.

The system architecture defines functionality that extends the traditional concept of user and privileged access modes by adding an access attribute indicating a *secure/nonsecure* state. A total of three access states are supported, where the relative “priority” is defined as:

- Privileged Secure > User Secure > User Nonsecure

In this model, the privileged state explicitly implies the secure state.

The DMA Controller can manage privileged/user and secure/nonsecure attributes on an accessby- access basis. Each channel’s DMA Control Register (DCR*n*) contains two fields for controlling the attributes: CHACR and UMNSM.

The value of DCR*n*[CHACR] specifies the access rights for the corresponding TCD*n*. This mechanism allows system software to "allocate" a given DMA channel to a specific operating mode. DCR*n*[CHACR] can be written only when the system is in privileged secure mode.

DCR*n*[UMNSM] defines the privileged/user and secure/nonsecure attributes for the DMA channel as it executes. For an allowed write, an update to DCR*n*[UMNSM] results either in preserving the attributes of the current system mode or in changing the attributes to a lower-level mode.

After the CHACR and UMNSM fields are configured and the TCD*n* is initialized: When a given channel is activated and executes, the DMA Controller uses the UMNSM values and generates the appropriate privileged/user and secure/nonsecure attributes for all source reads and destination writes.

24.4.2.3 Programming the DMA Controller Module

CAUTION

During a channel's execution, writes to programming model registers can corrupt the data transfer. The DMA module itself does not have a mechanism to prevent writes to registers during a channel's execution.

General guidelines for programming the DMA are:

- TCD*n* is initialized.
  - SAR*n* is loaded with the source (read) address. If the transfer is from a peripheral device to memory or to another peripheral, the source address is the location of the peripheral data register. If the transfer is from memory to a peripheral device or to memory, the source address is the starting address of the data block. This can be any appropriately aligned address.
  - DAR*n* is initialized with the destination (write) address. If the transfer is from a peripheral device to memory, or from memory to memory, DAR*n* is loaded with the starting address of the data block to be written. If the transfer is from memory to a peripheral device, or from a peripheral device to a peripheral device, DAR*n* is loaded with the address of the peripheral data register. This address can be any appropriately aligned address.

- SAR*n* and DAR*n* change after each data transfer depending on DCR*n*[SSIZE, DSIZE, SINC, DINC, SMOD, DMOD] and the starting addresses. Increment values can be 1, 2, or 4 for 8-bit, 16-bit, or 32-bit transfers, respectively. If the address register is programmed to remain unchanged, the register is not incremented after the data transfer.
- BCR*n*[BCR] must be loaded with the total number of bytes to be transferred. It is decremented by 1, 2, or 4 at the end of each transfer, depending on the transfer size. DSR*n*[DONE] must be cleared for channel startup.
- DCR*n*[CHACR, UMNSM] can be programmed as the TCD*n* is initialized to allocate the channel to an access control state and define its access attributes while executing.

- After the channel has been initialized, it may be started by setting DCR*n*[START] or a properly selected peripheral DMA request, depending on the status of DCR*n*[ERQ]. For a software-initiated transfer, the channel can be started by setting DCR*n*[START] as part of a single 32-bit write to the last 32 bits of the TCD*n*; that is, it is not required to write the DCR*n* with START cleared and then perform a second write to explicitly set START.
- Programming the channel for a software-initiated request causes the channel to request the system bus and start transferring data immediately. If the channel is programmed for peripheral-initiated request, a properly selected peripheral DMA request must be asserted before the channel begins the system bus transfers.
- The hardware can automatically clear DCR*n*[ERQ], disabling the peripheral request, when BCR*n* reaches zero by setting DCR*n*[D_REQ].
- Changes to DCR*n* are effective immediately while the channel is active. To avoid problems with changing a DMA channel setup, write a one to DSR*n*[DONE] to stop the DMA channel.

24.4.3 Dual-Address Data Transfer Mode

Each channel supports dual-address transfers. Dual-address transfers consist of a source data read and a destination data write. The DMA controller module begins a dual-address transfer sequence after a DMA request. If no error condition exists, DSR*n*[REQ] is set.

- Dual-address read—The DMA controller drives the SAR*n* value onto the system address bus. If DCR*n*[SINC] is set, the SAR*n* increments by the appropriate number of bytes upon a successful read cycle. When the appropriate number of read cycles complete (multiple reads if the destination size is larger than the source), the DMA initiates the write portion of the transfer.

If a termination error occurs, DSR*n*[BES, DONE] are set and DMA transactions stop.

- Dual-address write—The DMA controller drives the DAR*n* value onto the system address bus. When the appropriate number of write cycles complete (multiple writes if the source size is larger than the destination), DAR*n* increments by the appropriate number of bytes if DCRn[DINC] is set. BCRn decrements by the appropriate number of bytes. DSR*n*[DONE] is set when BCR*n* reaches zero. If the BCR*n* is greater than zero, another read/write transfer is initiated if continuous mode is enabled (DCRn[CS] = 0).

  If a termination error occurs, DSR*n*[BED, DONE] are set and DMA transactions stop.

24.4.4 Advanced Data Transfer Controls: Auto-Alignment

Typically, auto-alignment for DMA transfers applies for transfers of large blocks of data. As a result, it does not apply for peripheral-initiated cycle-steal transfers.

Auto-alignment allows block transfers to occur at the optimal size based on the address, byte count, and programmed size. To use this feature, DCR*n*[AA] must be set. The source is auto-aligned if DCR*n*[SSIZE] indicates a transfer size larger than DCR*n*[DSIZE]. Source alignment takes precedence over the destination when the source and destination sizes are equal. Otherwise, the destination is auto-aligned. The address register chosen for alignment increments regardless of the increment value. Configuration error checking is performed on registers not chosen for alignment.

If BCR*n* is greater than 16, the address determines transfer size. Transfers of 8 bits, 16 bits, or 32 bits are transferred until the address is aligned to the programmed size boundary, at which time accesses begin using the programmed size. If BCR*n* is less than 16 at the start of a transfer, the number of bytes remaining dictates transfer size.

Consider this example:

- AA equals 1.
- SAR*n* equals 0x2000_0001.
- BCR*n* equals 0x00_00F0.

- SSIZE equals 00 (32 bits).
- DSIZE equals 01 (8 bits).

Because SSIZE > DSIZE, the source is auto-aligned. Error checking is performed on destination registers. The access sequence is as follows:

1. Read 1 byte from 0x2000_0001, increment SAR*n*, write 1 byte (using DAR*n*).
2. Read 2 bytes from 0x2000_0002, increment SAR*n*, write 2 bytes.
3. Read 4 bytes from 0x2000_0004, increment SAR*n*, write 4 bytes.
4. Repeat 4-byte operations until SAR*n* equals 0x2000_00F0.
5. Read byte from 0x2000_00F0, increment SAR*n*, write byte.

If DSIZE is another size, data writes are optimized to write the largest size allowed based on the address, but not exceeding the configured size.

24.4.5 Termination

An unsuccessful transfer can terminate for one of the following reasons:

- Error conditions—When the DMA encounters a read or write cycle that terminates with an error condition, DSR*n*[BES] is set for a read and DSR*n*[BED] is set for a write before the transfer is halted. If the error occurred in a write cycle, data in the internal holding registers is lost.
- Interrupts—If DCR*n*[EINT] is set, the DMA drives the appropriate interrupt request signal. The processor can read DSR*n* to determine whether the transfer terminated successfully or with an error. DSR*n*[DONE] is then written with a 1 to clear the interrupt, DSR*n*[DONE], and error status bits.

Chapter 25
Direct Memory Access Multiplexer (DMAMUX)

25.1 Chip-specific DMAMUX information

25.1.1 DMA Request Sources

The DMA Request mux allows up to 64 DMA request signals to be mapped to the 4 channels of the DMA Controller.

This device includes 4 DMA request MUXes, which allows up to 64 DMA request signals to be mapped to any of the 4 DMA channels. There is 1 DMA mux module for each channel of DMA.

Because of the mux, there is no hard correlation between any of the DMA request sources and a specific DMA channel.

The DMA request sources for this device are MUXed as described in the table below.

NOTE

User should NOT configure the DMAMUX when the correspondant DMA channel is active.

Table 25-1. DMA request sources - MUX

Source number	Source module	Source description	Async DMA capable
0	—	Channel disabled[1]	
1	Reserved	Not used	
2	UART0	Receive	
3	UART0	Transmit	
4	UART1	Receive	
5	UART1	Transmit	
6	UART2	Receive	

Table continues on the next page...

Table 25-1. DMA request sources - MUX (continued)

Source number	Source module	Source description	Async DMA capable
7	UART2	Transmit	
8	UART3	Receive	
9	UART3	Transmit	
10			
11			
12			
13			
14			
15			
16	SPI0	Receive	
17	SPI0	Transmit	
18	SPI1	Receive	
19	SPI1	Transmit	
20			
21			
22	I^2C0	—	
23	I^2C1	—	
24	QTMR	CH0 OFLAG	
25	QTMR	CH1 OFLAG	
26	QTMR	CH2 OFLAG	
27	QTMR	CH3 OFLAG	
28	XBAR	DMA request 0	Yes
29	XBAR	DMA request 1	Yes
30	XBAR	DMA request 2	Yes
31	XBAR	DMA request 3	Yes
32	AFE_CH0		Yes
33	AFE_CH1		Yes
34	AFE_CH2		Yes
35	AFE_CH3		Yes
36	Port control module	Port J	Yes
37	Port control module	Port K	Yes
38	Port control module	Port L	Yes
39	Port control module	Port M	Yes
40	SAR ADC	—	Yes
41		—	
42	CMP0	—	Yes
43	CMP1	—	Yes
44	CMP2	—	Yes
45		—	

Table continues on the next page...

Table 25-1. DMA request sources - MUX (continued)

Source number	Source module	Source description	Async DMA capable
46		—	
47	MMAU		Yes
48	PDB	—	
49	Port control module	Port A	Yes
50	Port control module	Port B	Yes
51	Port control module	Port C	Yes
52	Port control module	Port D	Yes
53	Port control module	Port E	Yes
54	Port control module	Port F	Yes
55	Port control module	Port G	Yes
56	Port control module	Port H	Yes
57	Port control module	Port I	Yes
58	LPUART0	Receive	Yes
59	LPUART0	Transmit	Yes
60	DMA MUX	Always enabled	
61	DMA MUX	Always enabled	
62	DMA MUX	Always enabled	
63	DMA MUX	Always enabled	

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

25.2 Introduction

25.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the four DMA channels. This process is illustrated in the following figure.

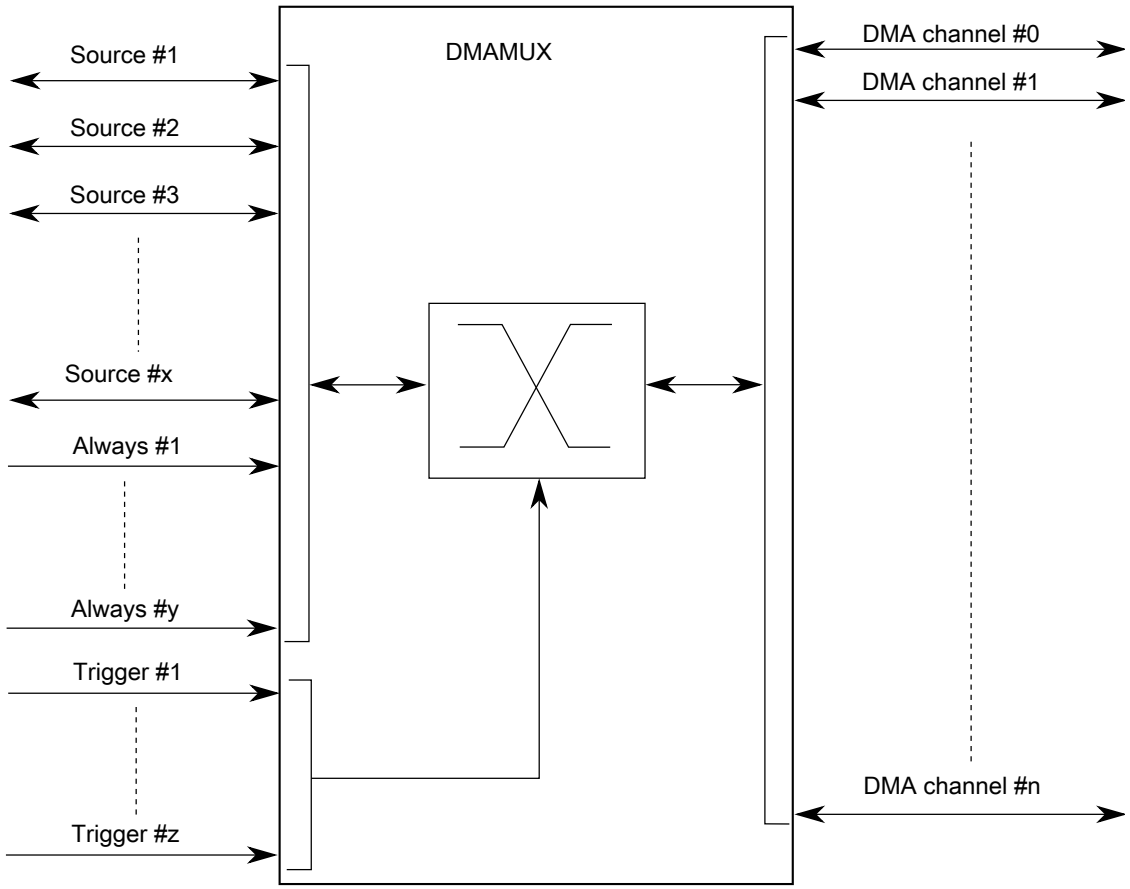


Figure 25-1. DMAMUX block diagram

25.2.2 Features

The DMAMUX module provides these features:

- Up to 59 peripheral slots and up to four always-on slots can be routed to four channels.
- four independently selectable DMA channel routers.
  - The first four channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the possible peripheral DMA slots or to one of the always-on slots.

25.2.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode

  In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

- Periodic Trigger mode

  In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically. Configuration of the period is done in the registers of the periodic interrupt timer (PIT). This mode is available only for channels 0–3.

25.3 External signal description

The DMAMUX has no external pins.

25.4 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

25.4.1 Endianness

This module's memory map uses big-endian ordering. This means:

- For 8-bit registers, the lower address byte is read as the most significant byte.
- For 16-bit registers, the lower address word is read as the most significant word.

The following figure provides examples of this.

Example 1: 8-bit register structure

Address	Register Data
00h	AAh
01h	BBh
02h	CCh
03h	DDh

For this structure, an 8-bit read of address 00h will yield DDh.

Example 2: 16-bit register structure

Address	Register Data
00h	AABBh
02h	CCDDh

For this structure, a 16-bit read of address 00h will yield CCDDh.

Figure 25-2. Examples of big-endian register access results

DMAMUX memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_1000	Channel Configuration register (DMAMUX_CHCFG0)	8	R/W	00h	25.4.2/426
4002_1001	Channel Configuration register (DMAMUX_CHCFG1)	8	R/W	00h	25.4.2/426
4002_1002	Channel Configuration register (DMAMUX_CHCFG2)	8	R/W	00h	25.4.2/426
4002_1003	Channel Configuration register (DMAMUX_CHCFG3)	8	R/W	00h	25.4.2/426

25.4.2 Channel Configuration register (DMAMUX_CHCFG*n*)

Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the trigger or source settings, a DMA channel must be disabled via CHCFGn[ENBL].

Address: 4002_1000h base + 0h offset + (1d × i), where i=0d to 3d

Bit	7	6	5	4	3	2	1	0
Read Write	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

DMAMUX_CHCFG*n* field descriptions

Field	Description
7 ENBL	DMA Channel Enable

Table continues on the next page...

DMAMUX_CHCFG*n* field descriptions (continued)

Field	Description
	Enables the DMA channel. 0 DMA channel is disabled. This mode is primarily used during configuration of the DMAMux. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 1 DMA channel is enabled
6 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel. 0 Triggering is disabled. If triggering is disabled and ENBL is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode) 1 Triggering is enabled. If triggering is enabled and ENBL is set, the DMAMUX is in Periodic Trigger mode.
SOURCE	DMA Channel Source (Slot) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about the peripherals and their slot numbers.

25.5 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels.

As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in Enabling and configuring sources is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

- Channels that implement the normal routing functionality plus periodic triggering capability
- Channels that implement only the normal routing functionality

25.5.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first 4 channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention. The trigger is generated by the periodic interrupt timer (PIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the PIT. See the section on periodic interrupt timer for more information on this topic.

Note

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

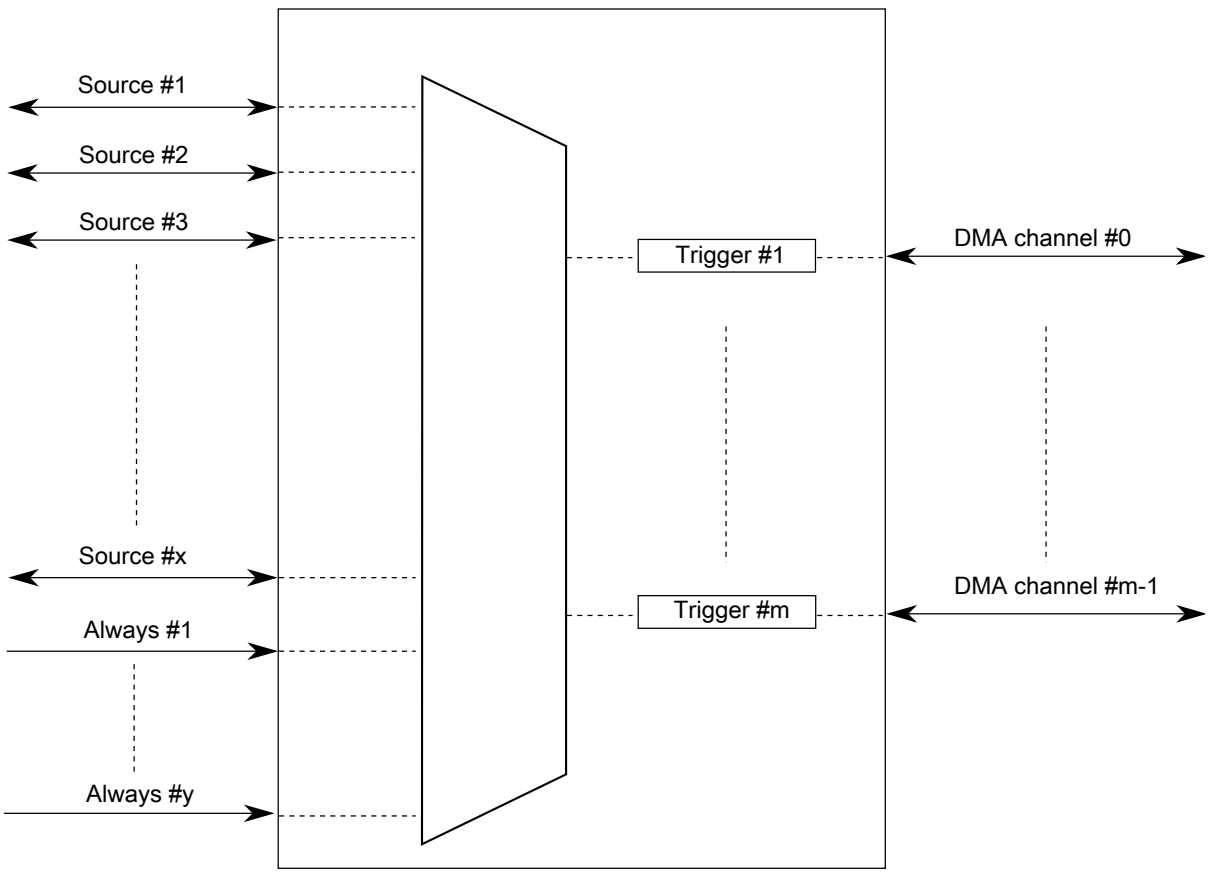


Figure 25-3. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

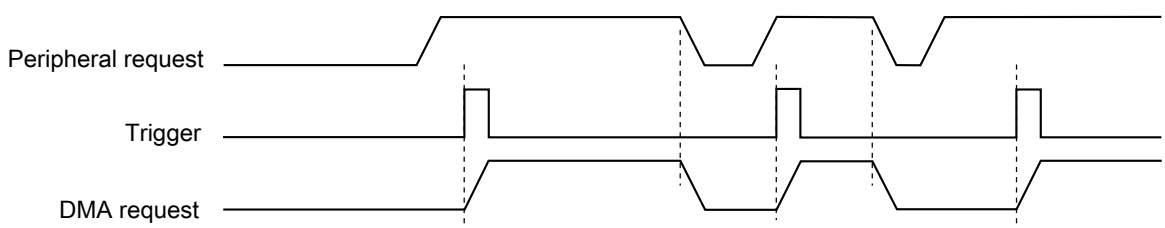


Figure 25-4. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

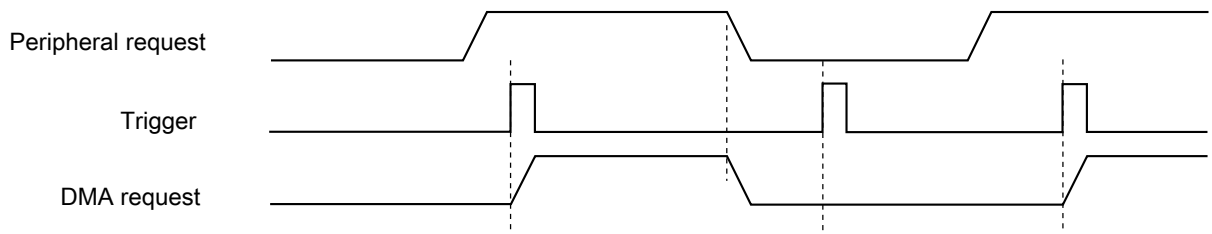


Figure 25-5. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

  As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

  By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

25.5.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in Modes of operation.

25.5.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are four additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

25.6 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

25.6.1 Reset

The reset state of each individual bit is shown in Memory map/register definition. In summary, after reset, all channels are disabled and must be explicitly enabled before use.

25.6.2 Enabling and configuring sources

To enable a source with periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.

4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0x00 to CHCFG1 (base address + 0x01).
2. Configure channel 1 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC5 to CHCFG1 (base address + 0x01).

The following code example illustrates steps 1 and 4 above:

```
void DMAMUX_Init(uint8_t DMA_CH, uint8_t DMAMUX_SOURCE)
{
    DMAMUX_0.CHCFG[DMA_CH].B.SOURCE = DMAMUX_SOURCE;
    DMAMUX_0.CHCFG[DMA_CH].B.ENBL   = 1;
    DMAMUX_0.CHCFG[DMA_CH].B.TRIG   = 1;
}
```

To enable a source, without periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is set while CHCFG[TRIG] is cleared.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0x00 to CHCFG1 (base address + 0x01).
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1 (base address + 0x01).

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;
*CHCFG1 = 0x85;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8 (base address + 0x08).
3. Write 0x87 to CHCFG8 (base address + 0x08). (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;
*CHCFG8 = 0x87;
```

Chapter 26
Memory Protection Unit (MPU)

26.1 Introduction

The memory protection unit (MPU) provides hardware access control for all memory references generated in the device.

26.2 Overview

The MPU concurrently monitors all system bus transactions and evaluates their appropriateness using pre-programmed region descriptors that define memory spaces and their access rights. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with a protection error response.

26.2.1 Block diagram

A simplified block diagram of the MPU module is shown in the following figure.

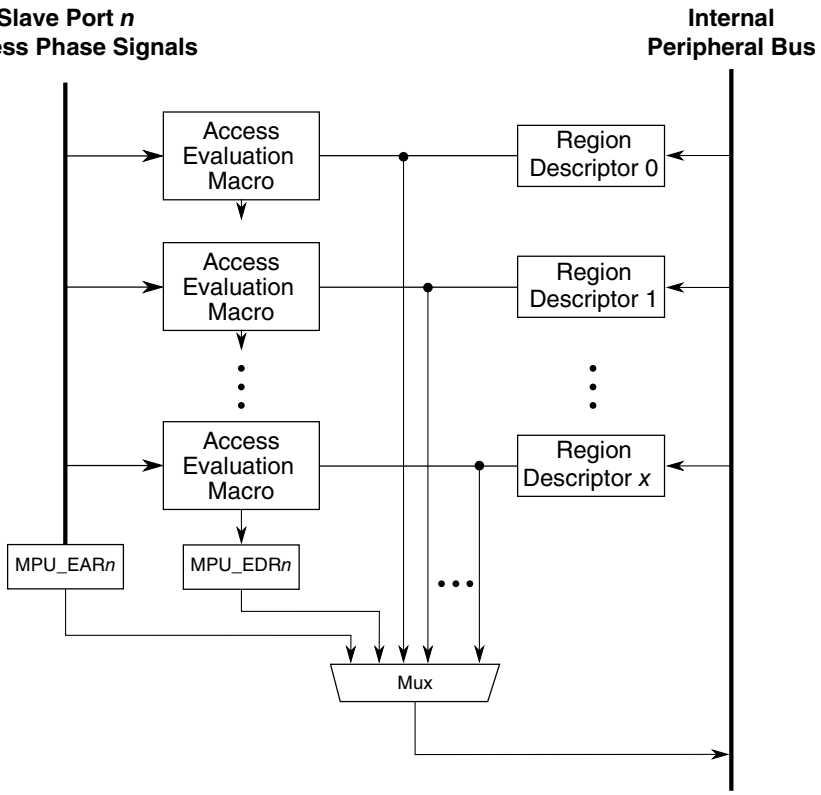


Figure 26-1. MPU block diagram

The hardware's two-dimensional connection matrix is clearly visible with the basic access evaluation macro shown as the replicated submodule block. The crossbar switch slave ports are shown on the left, the region descriptor registers in the middle, and the peripheral bus interface on the right side. The evaluation macro contains two magnitude comparators connected to the start and end address registers from each region descriptor as well as the combinational logic blocks to determine the region hit and the access protection error. For details of the access evaluation macro, see Access evaluation macro.

26.2.2 Features

The MPU implements a two-dimensional hardware array of memory region descriptors and the crossbar slave ports to continuously monitor the legality of every memory reference generated by each bus master in the system.

The feature set includes:

- 8 program-visible 128-bit region descriptors, accessible by four 32-bit words each
  - Each region descriptor defines a modulo-32 byte space, aligned anywhere in memory

    - Region sizes can vary from 32 bytes to 4 Gbytes
  - Two access control permissions defined in a single descriptor word
    - Masters 0–3: read, write, and execute attributes for supervisor and user accesses
    - Masters 4–7: read and write attributes
  - Hardware-assisted maintenance of the descriptor valid bit minimizes coherency issues
  - Alternate programming model view of the access control permissions word
  - Priority given to granting permission over denying access for overlapping region descriptors
- Detects access protection errors if a memory reference does not hit in any memory region, or if the reference is illegal in all hit memory regions. If an access error occurs, the reference is terminated with an error response, and the MPU inhibits the bus cycle being sent to the targeted slave device.
- Error registers, per slave port, capture the last faulting address, attributes, and other information
- Global MPU enable/disable control bit

26.3 Memory map/register definition

The programming model is partitioned into three groups:

- Control/status registers
- The data structure containing the region descriptors
- The alternate view of the region descriptor access control values

The programming model can only be referenced using 32-bit accesses. Attempted references using different access sizes, to undefined, that is, reserved, addresses, or with a non-supported access type, such as a write to a read-only register, or a read of a write-only register, generate an error termination.

The programming model can be accessed only in supervisor mode.

NOTE

See the chip configuration details for any chip-specific register information in this module.

MPU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_A000	Control/Error Status Register (MPU_CESR)	32	R/W	0081_2001h	26.3.1/439
4000_A010	Error Address Register, slave port n (MPU_EAR0)	32	R	0000_0000h	26.3.2/440
4000_A014	Error Detail Register, slave port n (MPU_EDR0)	32	R	0000_0000h	26.3.3/441
4000_A018	Error Address Register, slave port n (MPU_EAR1)	32	R	0000_0000h	26.3.2/440
4000_A01C	Error Detail Register, slave port n (MPU_EDR1)	32	R	0000_0000h	26.3.3/441
4000_A400	Region Descriptor n, Word 0 (MPU_RGD0_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A404	Region Descriptor n, Word 1 (MPU_RGD0_WORD1)	32	R/W	See section	26.3.5/443
4000_A408	Region Descriptor n, Word 2 (MPU_RGD0_WORD2)	32	R/W	See section	26.3.6/443
4000_A40C	Region Descriptor n, Word 3 (MPU_RGD0_WORD3)	32	R/W	See section	26.3.7/446
4000_A410	Region Descriptor n, Word 0 (MPU_RGD1_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A414	Region Descriptor n, Word 1 (MPU_RGD1_WORD1)	32	R/W	See section	26.3.5/443
4000_A418	Region Descriptor n, Word 2 (MPU_RGD1_WORD2)	32	R/W	See section	26.3.6/443
4000_A41C	Region Descriptor n, Word 3 (MPU_RGD1_WORD3)	32	R/W	See section	26.3.7/446
4000_A420	Region Descriptor n, Word 0 (MPU_RGD2_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A424	Region Descriptor n, Word 1 (MPU_RGD2_WORD1)	32	R/W	See section	26.3.5/443
4000_A428	Region Descriptor n, Word 2 (MPU_RGD2_WORD2)	32	R/W	See section	26.3.6/443
4000_A42C	Region Descriptor n, Word 3 (MPU_RGD2_WORD3)	32	R/W	See section	26.3.7/446
4000_A430	Region Descriptor n, Word 0 (MPU_RGD3_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A434	Region Descriptor n, Word 1 (MPU_RGD3_WORD1)	32	R/W	See section	26.3.5/443
4000_A438	Region Descriptor n, Word 2 (MPU_RGD3_WORD2)	32	R/W	See section	26.3.6/443
4000_A43C	Region Descriptor n, Word 3 (MPU_RGD3_WORD3)	32	R/W	See section	26.3.7/446
4000_A440	Region Descriptor n, Word 0 (MPU_RGD4_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A444	Region Descriptor n, Word 1 (MPU_RGD4_WORD1)	32	R/W	See section	26.3.5/443
4000_A448	Region Descriptor n, Word 2 (MPU_RGD4_WORD2)	32	R/W	See section	26.3.6/443
4000_A44C	Region Descriptor n, Word 3 (MPU_RGD4_WORD3)	32	R/W	See section	26.3.7/446
4000_A450	Region Descriptor n, Word 0 (MPU_RGD5_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A454	Region Descriptor n, Word 1 (MPU_RGD5_WORD1)	32	R/W	See section	26.3.5/443
4000_A458	Region Descriptor n, Word 2 (MPU_RGD5_WORD2)	32	R/W	See section	26.3.6/443
4000_A45C	Region Descriptor n, Word 3 (MPU_RGD5_WORD3)	32	R/W	See section	26.3.7/446
4000_A460	Region Descriptor n, Word 0 (MPU_RGD6_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A464	Region Descriptor n, Word 1 (MPU_RGD6_WORD1)	32	R/W	See section	26.3.5/443
4000_A468	Region Descriptor n, Word 2 (MPU_RGD6_WORD2)	32	R/W	See section	26.3.6/443
4000_A46C	Region Descriptor n, Word 3 (MPU_RGD6_WORD3)	32	R/W	See section	26.3.7/446
4000_A470	Region Descriptor n, Word 0 (MPU_RGD7_WORD0)	32	R/W	0000_0000h	26.3.4/442
4000_A474	Region Descriptor n, Word 1 (MPU_RGD7_WORD1)	32	R/W	See section	26.3.5/443
4000_A478	Region Descriptor n, Word 2 (MPU_RGD7_WORD2)	32	R/W	See section	26.3.6/443
4000_A47C	Region Descriptor n, Word 3 (MPU_RGD7_WORD3)	32	R/W	See section	26.3.7/446

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_A800	Region Descriptor Alternate Access Control n (MPU_RGDAAC0)	32	R/W	See section	26.3.8/447
4000_A804	Region Descriptor Alternate Access Control n (MPU_RGDAAC1)	32	R/W	See section	26.3.8/447
4000_A808	Region Descriptor Alternate Access Control n (MPU_RGDAAC2)	32	R/W	See section	26.3.8/447
4000_A80C	Region Descriptor Alternate Access Control n (MPU_RGDAAC3)	32	R/W	See section	26.3.8/447
4000_A810	Region Descriptor Alternate Access Control n (MPU_RGDAAC4)	32	R/W	See section	26.3.8/447
4000_A814	Region Descriptor Alternate Access Control n (MPU_RGDAAC5)	32	R/W	See section	26.3.8/447
4000_A818	Region Descriptor Alternate Access Control n (MPU_RGDAAC6)	32	R/W	See section	26.3.8/447
4000_A81C	Region Descriptor Alternate Access Control n (MPU_RGDAAC7)	32	R/W	See section	26.3.8/447

26.3.1 Control/Error Status Register (MPU_CESR)

Address: 4000_A000h base + 0h offset = 4000_A000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SPERR		0						1	0			HRL			
W	w1c															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NSP				NRGD				0							VLD
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1

MPU_CESR field descriptions

Field	Description
31–30 SPERR	Slave Port n Error Indicates a captured error in EARn and EDRn. This bit is set when the hardware detects an error and records the faulting address and attributes. It is cleared by writing one to it. If another error is captured at the exact same cycle as the write, the flag remains set. A find-first-one instruction or equivalent can detect the presence of a captured error. The following shows the correspondence between the bit number and slave port number: • Bit 31 corresponds to slave port 0. • Bit 30 corresponds to slave port 1.

Table continues on the next page...

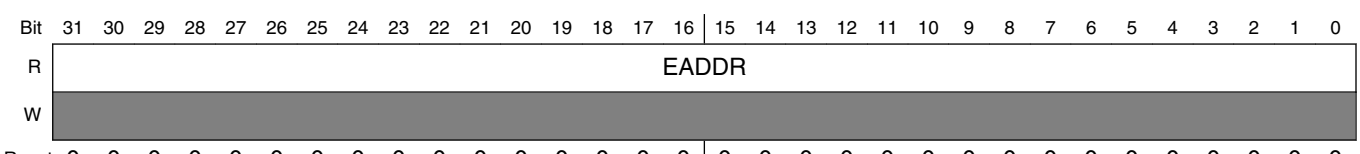
MPU_CESR field descriptions (continued)

Field	Description
	0 No error has occurred for slave port n. 1 An error has occurred for slave port n.
29–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
22–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 HRL	Hardware Revision Level Specifies the MPU's hardware and definition revision level. It can be read by software to determine the functional definition of the module.
15–12 NSP	Number Of Slave Ports Specifies the number of slave ports connected to the MPU.
11–8 NRGD	Number Of Region Descriptors Indicates the number of region descriptors implemented in the MPU. 0000 8 region descriptors 0001 12 region descriptors 0010 16 region descriptors
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 VLD	Valid Global enable/disable for the MPU. 0 MPU is disabled. All accesses from all bus masters are allowed. 1 MPU is enabled

26.3.2 Error Address Register, slave port n (MPU_EAR*n*)

When the MPU detects an access error on slave port n, the 32-bit reference address is captured in this read-only register and the corresponding bit in CESR[SPERR] set. Additional information about the faulting access is captured in the corresponding EDRn at the same time. This register and the corresponding EDRn contain the most recent access error; there are no hardware interlocks with CESR[SPERR], as the error registers are always loaded upon the occurrence of each protection violation.

Address: 4000_A000h base + 10h offset + (8d × i), where i=0d to 1d



MPU_EAR*n* field descriptions

Field	Description
EADDR	Error Address Indicates the reference address from slave port n that generated the access error

26.3.3 Error Detail Register, slave port n (MPU_EDR*n*)

When the MPU detects an access error on slave port n, 32 bits of error detail are captured in this read-only register and the corresponding bit in CESR[SPERR] is set. Information on the faulting address is captured in the corresponding EARn register at the same time. This register and the corresponding EARn register contain the most recent access error; there are no hardware interlocks with CESR[SPERR] as the error registers are always loaded upon the occurrence of each protection violation.

Address: 4000_A000h base + 14h offset + (8d × i), where i=0d to 1d

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EPID								EMN				EATTR			ERW
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MPU_EDR*n* field descriptions

Field	Description
31–16 EACD	Error Access Control Detail Indicates the region descriptor with the access error. • If EDRn contains a captured error and EACD is cleared, an access did not hit in any region descriptor. • If only a single EACD bit is set, the protection error was caused by a single non-overlapping region descriptor. • If two or more EACD bits are set, the protection error was caused by an overlapping set of region descriptors.
15–8 EPID	Error Process Identification Records the process identifier of the faulting reference. The process identifier is typically driven only by processor cores; for other bus masters, this field is cleared.
7–4 EMN	Error Master Number Indicates the bus master that generated the access error.

Table continues on the next page...

MPU_EDR*n* field descriptions (continued)

Field	Description
3–1 EATTR	Error Attributes Indicates attribute information about the faulting reference. **NOTE:** All other encodings are reserved. 000 User mode, instruction access 001 User mode, data access 010 Supervisor mode, instruction access 011 Supervisor mode, data access
0 ERW	Error Read/Write Indicates the access type of the faulting reference. 0 Read 1 Write

26.3.4 Region Descriptor n, Word 0 (MPU_RGD*n*_WORD0)

The first word of the region descriptor defines the 0-modulo-32 byte start address of the memory region. Writes to this register clear the region descriptor's valid bit (RGDn_WORD3[VLD]).

NOTE

RGD0_WORD0 register is read-only. Write access to RGD0_WORD0 generates error.

Address: 4000_A000h base + 400h offset + (16d × i), where i=0d to 7d

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

MPU_RGD*n*_WORD0 field descriptions

Field	Description
31–5 SRTADDR	Start Address Defines the most significant bits of the 0-modulo-32 byte start address of the memory region.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

26.3.5 Region Descriptor n, Word 1 (MPU_RGD*n*_WORD1)

The second word of the region descriptor defines the 31-modulo-32 byte end address of the memory region. Writes to this register clear the region descriptor's valid bit (RGDn_WORD3[VLD]).

NOTE

RGD0_WORD1 register is read-only. Write access to RGD0_WORD1 generates error.

Address: 4000_A000h base + 404h offset + (16d × i), where i=0d to 7d

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

* Notes:
- Reset value of RGD0_WORD1 is FFFF_FFFFh

  Reset value of RGD[1:7]_WORD1 is 0000_001Fh

MPU_RGD*n*_WORD1 field descriptions

Field	Description
31–5 ENDADDR	End Address Defines the most significant bits of the 31-modulo-32 byte end address of the memory region. **NOTE:** The MPU does not verify that ENDADDR ≥ SRTADDR.
Reserved	This field is reserved.

26.3.6 Region Descriptor n, Word 2 (MPU_RGD*n*_WORD2)

The third word of the region descriptor defines the access control rights of the memory region. The access control privileges depend on two broad classifications of bus masters:

- Bus masters 0–3 have a 5-bit field defining separate privilege rights for user and supervisor mode accesses, as well as the optional inclusion of a process identification field within the definition.
- Bus masters 4–7 are limited to separate read and write permissions.

For the privilege rights of bus masters 0–3, there are three flags associated with this function:

- Read (r) refers to accessing the referenced memory address using an operand (data) fetch

- Write (w) refers to updating the referenced memory address using a store (data) instruction
- Execute (x) refers to reading the referenced memory address using an instruction fetch

Writes to RGDn_WORD2 clear the region descriptor's valid bit (RGDn_WORD3[VLD]). If only updating the access controls, write to RGDAACn instead because stores to these locations do not affect the descriptor's valid bit.

Address: 4000_A000h base + 408h offset + (16d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M7RE	M7WE	M6RE	M6WE	M5RE	M5WE	M4RE	M4WE	M3PE	M3SM		M3UM			M2PE	M2SM
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M2SM	M2UM			M1PE	M1SM		M1UM			M0PE	M0SM		M0UM		
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- Reset value of RGD0_WORD2 is 0001_F01Fh

  Reset value of RGD[1:7]_WORD2 is 0000_0000h

MPU_RGD*n*_WORD2 field descriptions

Field	Description
31 M7RE	Bus Master 7 Read Enable 0 Bus master 7 reads terminate with an access error and the read is not performed 1 Bus master 7 reads allowed
30 M7WE	Bus Master 7 Write Enable 0 Bus master 7 writes terminate with an access error and the write is not performed 1 Bus master 7 writes allowed
29 M6RE	Bus Master 6 Read Enable 0 Bus master 6 reads terminate with an access error and the read is not performed 1 Bus master 6 reads allowed
28 M6WE	Bus Master 6 Write Enable 0 Bus master 6 writes terminate with an access error and the write is not performed 1 Bus master 6 writes allowed

Table continues on the next page...

MPU_RGD*n*_WORD2 field descriptions (continued)

Field	Description
27 M5RE	Bus Master 5 Read Enable 0 Bus master 5 reads terminate with an access error and the read is not performed 1 Bus master 5 reads allowed
26 M5WE	Bus Master 5 Write Enable 0 Bus master 5 writes terminate with an access error and the write is not performed 1 Bus master 5 writes allowed
25 M4RE	Bus Master 4 Read Enable 0 Bus master 4 reads terminate with an access error and the read is not performed 1 Bus master 4 reads allowed
24 M4WE	Bus Master 4 Write Enable 0 Bus master 4 writes terminate with an access error and the write is not performed 1 Bus master 4 writes allowed
23 M3PE	Bus Master 3 Process Identifier Enable 0 Do not include the process identifier in the evaluation 1 Include the process identifier and mask (RGDn_WORD3) in the region hit evaluation
22–21 M3SM	Bus Master 3 Supervisor Mode Access Control Defines the access controls for bus master 3 in Supervisor mode. 00 r/w/x; read, write and execute allowed 01 r/x; read and execute allowed, but no write 10 r/w; read and write allowed, but no execute 11 Same as User mode defined in M3UM
20–18 M3UM	Bus Master 3 User Mode Access Control Defines the access controls for bus master 3 in User mode. M3UM consists of three independent bits, enabling read (r), write (w), and execute (x) permissions. 0 An attempted access of that mode may be terminated with an access error (if not allowed by another descriptor) and the access not performed. 1 Allows the given access type to occur
17 M2PE	Bus Master 2 Process Identifier Enable See M3PE description.
16–15 M2SM	Bus Master 2 Supervisor Mode Access Control See M3SM description.
14–12 M2UM	Bus Master 2 User Mode Access control See M3UM description.
11 M1PE	Bus Master 1 Process Identifier enable See M3PE description.
10–9 M1SM	Bus Master 1 Supervisor Mode Access Control

Table continues on the next page...

MPU_RGD*n*_WORD2 field descriptions (continued)

Field	Description
	See M3SM description.
8–6 M1UM	Bus Master 1 User Mode Access Control See M3UM description.
5 M0PE	Bus Master 0 Process Identifier enable See M0PE description.
4–3 M0SM	Bus Master 0 Supervisor Mode Access Control See M3SM description.
M0UM	Bus Master 0 User Mode Access Control See M3UM description.

26.3.7 Region Descriptor n, Word 3 (MPU_RGD*n*_WORD3)

The fourth word of the region descriptor contains the optional process identifier and mask, plus the region descriptor's valid bit.

NOTE

RGD0_WORD3 register is read-only. Write access to RGD0_WORD3 generates error.

Address: 4000_A000h base + 40Ch offset + (16d × i), where i=0d to 7d

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

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

* Notes:

- Reset value of RGD0_WORD3 is 0000_0001h

  Reset value of RGD[1:7]_WORD3 is 0000_0000h

MPU_RGD*n*_WORD3 field descriptions

Field	Description
31–24 PID	Process Identifier Specifies the process identifier that is included in the region hit determination if RGDn_WORD2[MxPE] is set. PIDMASK can mask individual bits in this field.

Table continues on the next page...

MPU_RGD*n*_WORD3 field descriptions (continued)

Field	Description
23–16 PIDMASK	Process Identifier Mask Provides a masking capability so that multiple process identifiers can be included as part of the region hit determination. If a bit in PIDMASK is set, then the corresponding PID bit is ignored in the comparison. This field and PID are included in the region hit determination if RGDn_WORD2[MxPE] is set. For more information on the handling of the PID and PIDMASK, see “Access Evaluation - Hit Determination.”
15–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 VLD	Valid Signals the region descriptor is valid. Any write to RGDn_WORD0–2 clears this bit. 0 Region descriptor is invalid 1 Region descriptor is valid

26.3.8 Region Descriptor Alternate Access Control n (MPU_RGDAAC*n*)

Because software may adjust only the access controls within a region descriptor (RGDn_WORD2) as different tasks execute, an alternate programming view of this 32-bit entity is available. Writing to this register does not affect the descriptor’s valid bit.

Address: 4000_A000h base + 800h offset + (4d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M7RE	M7WE	M6RE	M6WE	M5RE	M5WE	M4RE	M4WE	M3PE	M3SM		M3UM			M2PE	M2SM
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M2SM	M2UM			M1PE	M1SM			M1UM			M0PE	M0SM		M0UM	
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- Reset value of RGDAAC0 is 0001_F01Fh

  Reset value of RGDAAC[1:7] is 0000_0000h

MPU_RGDAAC*n* field descriptions

Field	Description
31 M7RE	Bus Master 7 Read Enable 0 Bus master 7 reads terminate with an access error and the read is not performed 1 Bus master 7 reads allowed
30 M7WE	Bus Master 7 Write Enable 0 Bus master 7 writes terminate with an access error and the write is not performed 1 Bus master 7 writes allowed
29 M6RE	Bus Master 6 Read Enable 0 Bus master 6 reads terminate with an access error and the read is not performed 1 Bus master 6 reads allowed
28 M6WE	Bus Master 6 Write Enable 0 Bus master 6 writes terminate with an access error and the write is not performed 1 Bus master 6 writes allowed
27 M5RE	Bus Master 5 Read Enable 0 Bus master 5 reads terminate with an access error and the read is not performed 1 Bus master 5 reads allowed
26 M5WE	Bus Master 5 Write Enable 0 Bus master 5 writes terminate with an access error and the write is not performed 1 Bus master 5 writes allowed
25 M4RE	Bus Master 4 Read Enable 0 Bus master 4 reads terminate with an access error and the read is not performed 1 Bus master 4 reads allowed
24 M4WE	Bus Master 4 Write Enable 0 Bus master 4 writes terminate with an access error and the write is not performed 1 Bus master 4 writes allowed
23 M3PE	Bus Master 3 Process Identifier Enable 0 Do not include the process identifier in the evaluation 1 Include the process identifier and mask (RGDn.RGDAAC) in the region hit evaluation
22–21 M3SM	Bus Master 3 Supervisor Mode Access Control Defines the access controls for bus master 3 in Supervisor mode. 00 r/w/x; read, write and execute allowed 01 r/x; read and execute allowed, but no write 10 r/w; read and write allowed, but no execute 11 Same as User mode defined in M3UM
20–18 M3UM	Bus Master 3 User Mode Access Control Defines the access controls for bus master 3 in user mode. M3UM consists of three independent bits, enabling read (r), write (w), and execute (x) permissions. The bit assignment sequence is as M3UM[2:0] -> rwx.

Table continues on the next page...

MPU_RGDAAC*n* field descriptions (continued)

Field	Description
	0 An attempted access of that mode may be terminated with an access error (if not allowed by another descriptor) and the access not performed. 1 Allows the given access type to occur
17 M2PE	Bus Master 2 Process Identifier Enable See M3PE description.
16–15 M2SM	Bus Master 2 Supervisor Mode Access Control See M3SM description.
14–12 M2UM	Bus Master 2 User Mode Access Control See M3UM description.
11 M1PE	Bus Master 1 Process Identifier Enable See M3PE description.
10–9 M1SM	Bus Master 1 Supervisor Mode Access Control See M3SM description.
8–6 M1UM	Bus Master 1 User Mode Access Control See M3UM description.
5 M0PE	Bus Master 0 Process Identifier Enable See M3PE description.
4–3 M0SM	Bus Master 0 Supervisor Mode Access Control See M3SM description.
M0UM	Bus Master 0 User Mode Access Control See M3UM description.

26.4 Functional description

In this section, the functional operation of the MPU is detailed, including the operation of the access evaluation macro and the handling of error-terminated bus cycles.

26.4.1 Access evaluation macro

The basic operation of the MPU is performed in the access evaluation macro, a hardware structure replicated in the two-dimensional connection matrix. As shown in the following figure, the access evaluation macro inputs the crossbar bus address phase signals and the contents of a region descriptor (RGD*n*) and performs two major functions:

- Region hit determination
- Detection of an access protection violation

The following figure shows a functional block diagram.

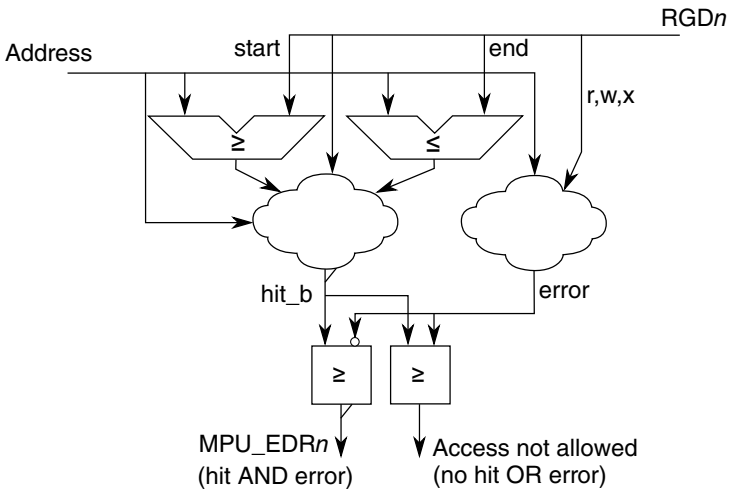


Figure 26-2. MPU access evaluation macro

26.4.1.1 Hit determination

To determine whether the current reference hits in the given region, two magnitude comparators are used with the region's start and end addresses. The boolean equation for this portion of the hit determination is:

```
region_hit = ((addr[31:5] >= RGDn_Word0[SRTADDR]) & (addr[31:5] <= RGDn_Word1[ENDADDR])) &
RGDn_Word3[VLD]
```

where addr is the current reference address, RGD*n*_Word0[SRTADDR] and RGD*n*_Word1[ENDADDR] are the start and end addresses, and RGD*n*_Word3[VLD] is the valid bit.

NOTE

The MPU does not verify that ENDADDR ≥ SRTADDR.

In addition to the comparison of the reference address versus the region descriptor's start and end addresses, the optional process identifier is examined against the region descriptor's PID and PIDMASK fields. A process identifier hit term is formed as follows:

```
pid_hit = ~RGDn_Word2[MxPE] | current_access_is_privileged |
          ((current_pid |
          RGDn_Word3[PIDMASK]) == (RGDn_Word3[PID] | RGDn_Word3[PIDMASK]))
```

where the current_pid is the selected process identifier from the current bus master, and RGD*n*_Word3[PID] and RGD*n*_Word3[PIDMASK] are the process identifier fields from region descriptor *n*. For bus masters that do not output a process identifier, the MPU forces the pid_hit term to assert.

NOTE

In the supervisor mode pid_hit term is always asserted.

26.4.1.2 Privilege violation determination

While the access evaluation macro is determining region hit, the logic is also evaluating if the current access is allowed by the permissions defined in the region descriptor. Using the master and supervisor/user mode signals, a set of effective permissions is generated from the appropriate fields in the region descriptor. The protection violation logic then evaluates the access against the effective permissions using the specification shown below.

Table 26-1. Protection violation definition

Description	MxUM			Protection violation?
	r	w	x	
Instruction fetch read	—	—	0	Yes, no execute permission
	—	—	1	No, access is allowed
Data read	0	—	—	Yes, no read permission
	1	—	—	No, access is allowed
Data write	—	0	—	Yes, no write permission
	—	1	—	No, access is allowed

26.4.2 Putting it all together and error terminations

For each slave port monitored, the MPU performs a reduction-AND of all the individual terms from each access evaluation macro. This expression then terminates the bus cycle with an error and reports a protection error for three conditions:

- If the access does not hit in any region descriptor, a protection error is reported.
- If the access hits in a single region descriptor and that region signals a protection violation, a protection error is reported.
- If the access hits in multiple (overlapping) regions and all regions signal protection violations, a protection error is reported.

As shown in the third condition, granting permission is a higher priority than denying access for overlapping regions. This approach is more flexible to system software in region descriptor assignments. For an example of the use of overlapping region descriptors, see Application information.

26.4.3 Power management

Disabling the MPU by clearing CESR[VLD] minimizes power dissipation. To minimize the power dissipation of an enabled MPU, invalidate unused region descriptors by clearing the associated RGDn_Word3[VLD] bits.

26.5 Initialization information

At system startup, load the appropriate number of region descriptors, including setting RGD*n*_Word3[VLD]. Setting CESR[VLD] enables the module.

If the system requires that all the loaded region descriptors be enabled simultaneously, first ensure that the entire MPU is disabled (CESR[VLD]=0).

Note

A region descriptor must be set to allow access to the MPU registers if further changes are needed.

26.6 Application information

In an operational system, interfacing with the MPU is generally classified into the following activities:

- Creating a new memory region—Load the appropriate region descriptor into an available RGD*n*, using four sequential 32-bit writes. The hardware assists in the maintenance of the valid bit, so if this approach is followed, there are no coherency issues with the multi-cycle descriptor writes. (Clearing RGD*n*_Word3[VLD] deletes/removes an existing memory region.)
- Altering only access privileges—To not affect the valid bit, write to the alternate version of the access control word (RGDAAC*n*), so there are no coherency issues involved with the update. When the write completes, the memory region's access rights switch instantaneously to the new value.

- Changing a region's start and end addresses—Write a minimum of three words to the region descriptor (RGD*n*_Word{0,1,3}). Word 0 and 1 redefine the start and end addresses, respectively. Word 3 re-enables the region descriptor valid bit. In most situations, all four words of the region descriptor are rewritten.
- Accessing the MPU—Allocate a region descriptor to restrict MPU access to supervisor mode from a specific master.
- Detecting an access error—The current bus cycle is terminated with an error response and EAR*n* and EDR*n* capture information on the faulting reference. The error-terminated bus cycle typically initiates an error response in the originating bus master. For example, a processor core may respond with a bus error exception, while a data movement bus master may respond with an error interrupt. The processor can retrieve the captured error address and detail information simply by reading E{A,D}R*n*. CESR[SPERR] signals which error registers contain captured fault data.
- Overlapping region descriptors—Applying overlapping regions often reduces the number of descriptors required for a given set of access controls. In the overlapping memory space, the protection rights of the corresponding region descriptors are logically summed together (the boolean OR operator).

  The following dual-core system example contains four bus masters:

  - The two processors: CP0, CP1
  - Two DMA engines: DMA1, a traditional data movement engine transferring data between RAM and peripherals and DMA2, a second engine transferring data to/from the RAM only

  Consider the following region descriptor assignments:

Table 26-2. Overlapping region descriptor example

Region description	RGDn		CP0	CP1	DMA1	DMA2	
CP0 code	0		rwx	r--	—	—	Flash
CP1 code	1		r--	rwx	—	—	
CP0 data & stack	2		rw-	—	—	—	RAM
CP0 → CP1 shared data	2	3	r--	r--	—	—	
CP1 → CP0 shared data	4						
CP1 data & stack	4		—	rw-	—	—	
Shared DMA data	5		rw-	rw-	rw	rw	
MPU	6		rw-	rw-	—	—	Peripheral space
Peripherals	7		rw-	rw-	rw	—	

In this example, there are eight descriptors used to span nine regions in the three main spaces of the system memory map: flash, RAM, and peripheral space. Each region indicates the specific permissions for each of the four bus masters and this definition provides an appropriate set of shared, private and executable memory spaces.

Of particular interest are the two overlapping spaces: region descriptors 2 & 3 and 3 & 4.

The space defined by RGD2 with no overlap is a private data and stack area that provides read/write access to CP0 only. The overlapping space between RGD2 and RGD3 defines a shared data space for passing data from CP0 to CP1 and the access controls are defined by the logical OR of the two region descriptors. Thus, CP0 has (rw- | r--) = (rw-) permissions, while CP1 has (--- | r--) = (r--) permission in this space. Both DMA engines are excluded from this shared processor data region. The overlapping spaces between RGD3 and RGD4 defines another shared data space, this one for passing data from CP1 to CP0. For this overlapping space, CP0 has (r-- | ---) = (r--) permission, while CP1 has (rw- | r--) = (rw-) permission. The non-overlapped space of RGD4 defines a private data and stack area for CP1 only.

The space defined by RGD5 is a shared data region, accessible by all four bus masters. Finally, the slave peripheral space mapped onto the IPS bus is partitioned into two regions:

- One containing the MPU's programming model accessible only to the two processor cores
- The remaining peripheral region accessible to both processors and the traditional DMA1 master

This example shows one possible application of the capabilities of the MPU in a typical system.

Chapter 27
Flash Memory Module (FTFA)

27.1 Introduction

The flash memory module includes the following accessible memory regions:

- Program flash memory for vector space and code store

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The flash memory module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

27.1.1 Features

The flash memory module includes the following features.

NOTE

See the device's Chip Configuration details for the exact amount of flash memory available on your device.

27.1.1.1 Program Flash Memory Features

- Sector size of 2 KB
- Program flash protection scheme prevents accidental program or erase of stored data
- Automated, built-in, program and erase algorithms with verify
- Read access to one program flash block is possible while programming or erasing data in the other program flash block

27.1.1.2 Other Flash Memory Module Features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

27.1.2 Block Diagram

The block diagram of the flash memory module is shown in the following figure.

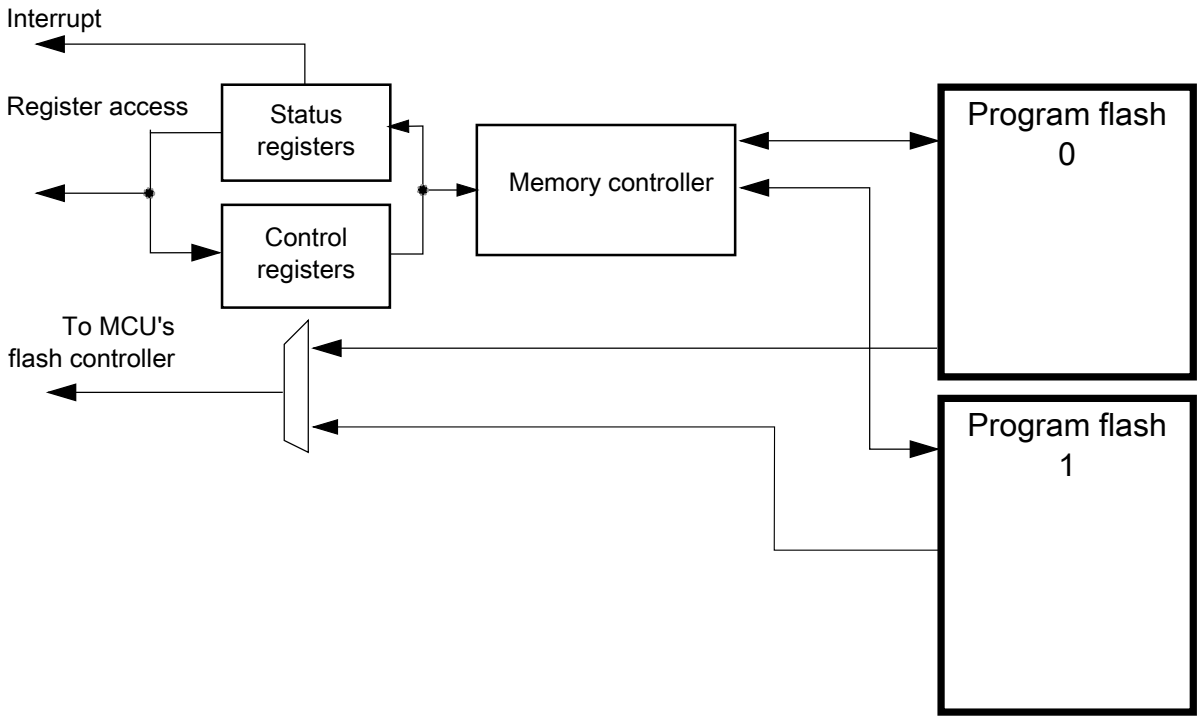


Figure 27-1. Flash Block Diagram

27.1.3 Glossary

Command write sequence — A series of MCU writes to the flash FCCOB register group that initiates and controls the execution of flash algorithms that are built into the flash memory module.

Endurance — The number of times that a flash memory location can be erased and reprogrammed.

FCCOB (Flash Common Command Object) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the flash memory module.

Flash block — A macro within the flash memory module which provides the nonvolatile memory storage.

Flash Memory Module — All flash blocks plus a flash management unit providing high-level control and an interface to MCU buses.

HSRUN — An MCU power mode enabling high-speed access to the memory resources in the flash module. The user has no access to the flash command set when the MCU is in HSRUN mode.

IFR — Nonvolatile information register found in each flash block, separate from the main memory array.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

NVM — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

NVM Normal Mode — An NVM mode that provides basic user access to flash memory module resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the flash memory module.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Program flash — The program flash memory provides nonvolatile storage for vectors and code store.

Program flash Sector — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

Retention — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

RWW— Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

Secure — An MCU state conveyed to the flash memory module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

27.2 External Signal Description

The flash memory module contains no signals that connect off-chip.

27.3 Memory Map and Registers

This section describes the memory map and registers for the flash memory module.

Data read from unimplemented memory space in the flash memory module is undefined. Writes to unimplemented or reserved memory space (registers) in the flash memory module are ignored.

27.3.1 Flash Configuration Field Description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the flash memory module.

Flash Configuration Field Byte Address	Size (Bytes)	Field Description
0x0_0400–0x0_0407	8	Backdoor Comparison Key. Refer to Verify Backdoor Access Key Command and Unsecuring the Chip Using Backdoor Key Access.
0x0_0408–0x0_040B	4	Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).
0x0_040F	1	Reserved
0x0_040E	1	Reserved
0x0_040D	1	Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).
0x0_040C	1	Flash security byte. Refer to the description of the Flash Security Register (FSEC).

27.3.2 Program Flash IFR Map

The program flash IFR is nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see the Read Once, Program Once, and Read Resource commands in Read Once Command, Program Once Command and Read Resource Command).

The contents of the program flash IFR are summarized in the table found here and further described in the subsequent paragraphs.

The program flash IFR is located within the program flash 0 memory block .

Address Range	Size (Bytes)	Field Description
0x00 – 0xBF	192	Reserved
0xC0 – 0xFF	64	Program Once Field

27.3.2.1 Program Once Field

The Program Once Field in the program flash IFR provides 64 bytes of user data storage separate from the program flash main array. The user can program the Program Once Field one time only as there is no program flash IFR erase mechanism available to the user. The Program Once Field can be read any number of times. This section of the program flash IFR is accessed in 4-byte records using the Read Once and Program Once commands (see Read Once Command and Program Once Command).

27.3.3 Register Descriptions

The flash memory module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

FTFA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0000	Flash Status Register (FTFA_FSTAT)	8	R/W	00h	27.3.3.1/ 461
4002_0001	Flash Configuration Register (FTFA_FCNFG)	8	R/W	00h	27.3.3.2/ 463
4002_0002	Flash Security Register (FTFA_FSEC)	8	R	Undefined	27.3.3.3/ 464
4002_0003	Flash Option Register (FTFA_FOPT)	8	R	Undefined	27.3.3.4/ 465
4002_0004	Flash Common Command Object Registers (FTFA_FCCOB3)	8	R/W	00h	27.3.3.5/ 466
4002_0005	Flash Common Command Object Registers (FTFA_FCCOB2)	8	R/W	00h	27.3.3.5/ 466
4002_0006	Flash Common Command Object Registers (FTFA_FCCOB1)	8	R/W	00h	27.3.3.5/ 466

Table continues on the next page...

FTFA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0007	Flash Common Command Object Registers (FTFA_FCCOB0)	8	R/W	00h	27.3.3.5/ 466
4002_0008	Flash Common Command Object Registers (FTFA_FCCOB7)	8	R/W	00h	27.3.3.5/ 466
4002_0009	Flash Common Command Object Registers (FTFA_FCCOB6)	8	R/W	00h	27.3.3.5/ 466
4002_000A	Flash Common Command Object Registers (FTFA_FCCOB5)	8	R/W	00h	27.3.3.5/ 466
4002_000B	Flash Common Command Object Registers (FTFA_FCCOB4)	8	R/W	00h	27.3.3.5/ 466
4002_000C	Flash Common Command Object Registers (FTFA_FCCOBB)	8	R/W	00h	27.3.3.5/ 466
4002_000D	Flash Common Command Object Registers (FTFA_FCCOBA)	8	R/W	00h	27.3.3.5/ 466
4002_000E	Flash Common Command Object Registers (FTFA_FCCOB9)	8	R/W	00h	27.3.3.5/ 466
4002_000F	Flash Common Command Object Registers (FTFA_FCCOB8)	8	R/W	00h	27.3.3.5/ 466
4002_0010	Program Flash Protection Registers (FTFA_FPROT3)	8	R/W	Undefined	27.3.3.6/ 467
4002_0011	Program Flash Protection Registers (FTFA_FPROT2)	8	R/W	Undefined	27.3.3.6/ 467
4002_0012	Program Flash Protection Registers (FTFA_FPROT1)	8	R/W	Undefined	27.3.3.6/ 467
4002_0013	Program Flash Protection Registers (FTFA_FPROT0)	8	R/W	Undefined	27.3.3.6/ 467

27.3.3.1 Flash Status Register (FTFA_FSTAT)

The FSTAT register reports the operational status of the flash memory module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

NOTE

When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands until the flag is cleared (by writing a one to it).

Address: 4002_0000h base + 0h offset = 4002_0000h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
Write	w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0

FTFA_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag Indicates that a flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command, and CCIF stays low until command completion or command violation. CCIF is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value. 0 Flash command in progress 1 Flash command has completed
6 RDCOLERR	Flash Read Collision Error Flag Indicates that the MCU attempted a read from a flash memory resource that was being manipulated by a flash command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect. 0 No collision error detected 1 Collision error detected
5 ACCERR	Flash Access Error Flag Indicates an illegal access has occurred to a flash memory resource caused by a violation of the command write sequence or issuing an illegal flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR while CCIF is set. Writing a 0 to the ACCERR bit has no effect. 0 No access error detected 1 Access error detected
4 FPVIOL	Flash Protection Violation Flag Indicates an attempt was made to program or erase an address in a protected area of program flash memory during a command write sequence . While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to FPVIOL while CCIF is set. Writing a 0 to the FPVIOL bit has no effect. 0 No protection violation detected 1 Protection violation detected
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 MGSTAT0	Memory Controller Command Completion Status Flag The MGSTAT0 status flag is set if an error is detected during execution of a flash command or during the flash reset sequence. As a status flag, this field cannot (and need not) be cleared by the user like the other error flags in this register.

Table continues on the next page...

FTFA_FSTAT field descriptions (continued)

Field	Description
	The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.

27.3.3.2 Flash Configuration Register (FTFA_FCNFG)

This register provides information on the current functional state of the flash memory module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. The unassigned bits read as noted and are not writable.

Address: 4002_0000h base + 1h offset = 4002_0001h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	0	0	0	0
Write								
Reset	0	0	0	0	0	0	0	0

FTFA_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable Controls interrupt generation when a flash command completes. 0 Command complete interrupt disabled 1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable Controls interrupt generation when a flash memory read collision error occurs. 0 Read collision error interrupt disabled 1 Read collision error interrupt enabled. An interrupt request is generated whenever a flash memory read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request Issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command. ERSAREQ sets when an erase all request is triggered external to the flash memory module and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the flash memory module when the operation completes. 0 No request or request complete 1 Request to: 1. run the Erase All Blocks command,

Table continues on the next page...

FTFA_FCNFG field descriptions (continued)

Field	Description
	2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state.
4 ERSSUSP	Erase Suspend Allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0 No suspend requested 1 Suspend the current Erase Flash Sector command execution.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

27.3.3.3 Flash Security Register (FTFA_FSEC)

This read-only register holds all bits associated with the security of the MCU and flash memory module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 2h offset = 4002_0002h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		MEEN		FSLACC		SEC	
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable Enables or disables backdoor key access to the flash memory module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access)

Table continues on the next page...

FTFA_FSEC field descriptions (continued)

Field	Description
	10 Backdoor key access enabled 11 Backdoor key access disabled
5–4 MEEN	Mass Erase Enable Enables and disables mass erase capability of the flash memory module. When SEC is set to unsecure, the MEEN setting does not matter. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled
3–2 FSLACC	Freescale Failure Analysis Access Code Enables or disables access to the flash memory contents during returned part failure analysis at Freescale. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by Freescale factory test must begin with a full erase to unsecure the part. When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), Freescale factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when SEC is set to secure. When SEC is set to unsecure, the FSLACC setting does not matter. 00 Freescale factory access granted 01 Freescale factory access denied 10 Freescale factory access denied 11 Freescale factory access granted
SEC	Flash Security Defines the security state of the MCU. In the secure state, the MCU limits access to flash memory module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the flash memory module is unsecured using backdoor key access, SEC is forced to 10b. 00 MCU security status is secure. 01 MCU security status is secure. 10 MCU security status is unsecure. (The standard shipping condition of the flash memory module is unsecure.) 11 MCU security status is secure.

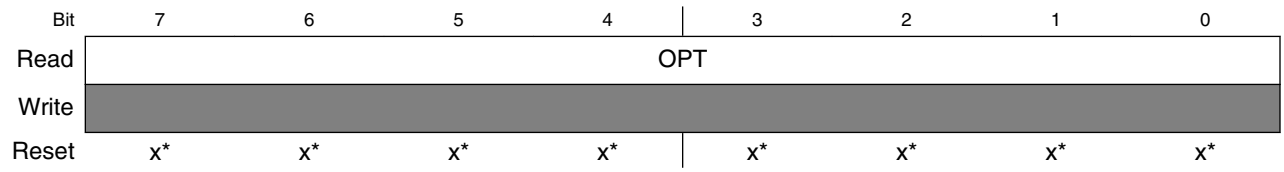
27.3.3.4 Flash Option Register (FTFA_FOPT)

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only .

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value. However, the register is written to 0xFF if the contents of the flash nonvolatile option byte are 0x00.

Address: 4002_0000h base + 3h offset = 4002_0003h



* Notes:
- x = Undefined at reset.

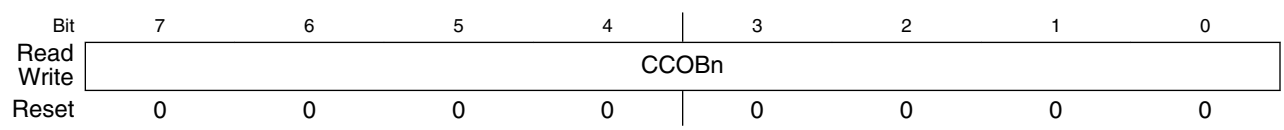
FTFA_FOPT field descriptions

Field	Description
OPT	Nonvolatile Option These bits are loaded from flash to this register at reset. Refer to the device's Chip Configuration details for the definition and use of these bits.

27.3.3.5 Flash Common Command Object Registers (FTFA_FCCOB*n*)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Address: 4002_0000h base + 4h offset + (1d × i), where i=0d to 11d



FTFA_FCCOB*n* field descriptions

Field	Description
CCOBn	The FCCOB register provides a command code and relevant parameters to the memory controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command's execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes. Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1 by the memory controller.

FTFA_FCCOB*n* field descriptions (continued)

Field	Description
	The following table shows a generic flash command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific flash command, typically an address and/or data values. **NOTE:** The command parameter table is written in terms of FCCOB Number (which is equivalent to the byte number). This number is a reference to the FCCOB register name and is not the register address.

FCCOB Number	Typical Command Parameter Contents [7:0]
0	FCMD (a code that defines the flash command)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0]
4	Data Byte 0
5	Data Byte 1
6	Data Byte 2
7	Data Byte 3
8	Data Byte 4
9	Data Byte 5
A	Data Byte 6
B	Data Byte 7

FCCOB Endianness :

The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number.

27.3.3.6 Program Flash Protection Registers (FTFA_FPROT*n*)

The FPROT registers define which program flash regions are protected from program and erase operations. Protected flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any flash command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions. Each bit protects a 1/32 region of the program flash memory except for memory configurations with less than 64 KB of program flash where each assigned bit protects 2 KB . For configurations with 48 KB of program flash memory or less, FPROT0 is not used. For configurations with 32 KB of program flash memory or less, FPROT1 is not used. For configurations with 16 KB of program flash memory, FPROT2 is not used. The bitfields are defined in each register as follows:

Program flash protection register	Program flash protection bits
FPROT0	PROT[31:24]
FPROT1	PROT[23:16]
FPROT2	PROT[15:8]
FPROT3	PROT[7:0]

During the reset sequence, the FPROT registers are loaded with the contents of the program flash protection bytes in the Flash Configuration Field as indicated in the following table.

Program flash protection register	Flash Configuration Field offset address
FPROT0	0x000B
FPROT1	0x000A
FPROT2	0x0009
FPROT3	0x0008

To change the program flash protection that is loaded during the reset sequence, unprotect the sector of program flash memory that contains the Flash Configuration Field. Then, reprogram the program flash protection byte.

Address: 4002_0000h base + 10h offset + (1d × i), where i=0d to 3d

Bit	7	6	5	4	3	2	1	0
Read Write	PROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FPROT*n* field descriptions

Field	Description
PROT	Program Flash Region Protect Each program flash region can be protected from program and erase operations by setting the associated PROT bit. The protection can only be increased, meaning that currently unprotected memory can be protected, but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **Restriction:** The user must never write to any FPROT register while a command is running (CCIF=0). Trying to alter data in any protected area in the program flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A full block erase of a program flash block is not possible if it contains any protected region. Each bit in the 32-bit protection register represents 1/32 of the total program flash except for memory configurations with less than 64 KB of program flash where each assigned bit protects 2 KB .

FTFA_FPROT*n* field descriptions (continued)

Field	Description
	0 Program flash region is protected. 1 Program flash region is not protected

27.4 Functional Description

The information found here describes functional details of the flash memory module.

27.4.1 Flash Protection

Individual regions within the flash memory can be protected from program and erase operations.

Protection is controlled by the following registers:

- FPROT*n* —
  - For 2^n program flash sizes, four registers typically protect 32 regions of the program flash memory as shown in the following figure

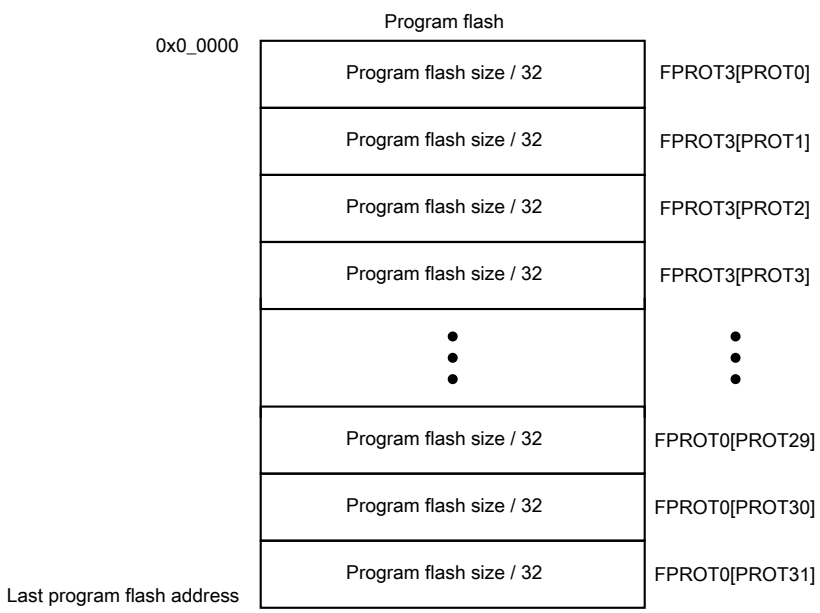


Figure 27-2. Program flash protection

NOTE

Flash protection features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Not

all features described in the application note are available on this device.

27.4.2 Interrupts

The flash memory module can generate interrupt requests to the MCU upon the occurrence of various flash events.

These interrupt events and their associated status and control bits are shown in the following table.

Table 27-1. Flash Interrupt Sources

Flash Event	Readable Status Bit	Interrupt Enable Bit
Flash Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
Flash Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]

Note

Vector addresses and their relative interrupt priority are determined at the MCU level.

Some devices also generate a bus error response as a result of a Read Collision Error event. See the chip configuration information to determine if a bus error response is also supported.

27.4.3 Flash Operation in Low-Power Modes

27.4.3.1 Wait Mode

When the MCU enters wait mode, the flash memory module is not affected. The flash memory module can recover the MCU from wait via the command complete interrupt (see Interrupts).

27.4.3.2 Stop Mode

When the MCU requests stop mode, if a flash command is active (CCIF = 0) the command execution completes before the MCU is allowed to enter stop mode.

CAUTION

The MCU should never enter stop mode while any flash command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the flash memory module does not accept flash commands.

27.4.4 Flash Reads and Ignored Writes

The flash memory module requires only the flash address to execute a flash memory read.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

27.4.5 Read While Write (RWW)

The following simultaneous accesses are allowed:

- The user may read from one logical program flash memory space while flash commands are active in the other logical program flash memory space.

Simultaneous operations are further discussed in Allowed Simultaneous Flash Operations.

27.4.6 Flash Program and Erase

All flash functions except read require the user to setup and launch a flash command through a series of peripheral bus writes.

The user cannot initiate any further flash commands until notified that the current command has completed. The flash command structure and operation are detailed in Flash Command Operations.

27.4.7 Flash Command Operations

Flash command operations are typically used to modify flash memory contents.

The next sections describe:

- The command write sequence used to set flash command parameters and launch execution
- A description of all flash commands available

27.4.7.1 Command Write Sequence

Flash commands are specified using a command write sequence illustrated in Figure 27-3. The flash memory module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

Attempts to launch a flash command in VLP mode will be ignored. Attempts to launch a flash command in HSRUN mode will be trapped with the ACCERR flag being set.

27.4.7.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired flash command. The individual registers that make up the FCCOB data set can be written in any order.

27.4.7.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing FSTAT[CCIF] by writing a '1' to it. FSTAT[CCIF] remains 0 until the flash command completes.

The FSTAT register contains a blocking mechanism that prevents a new command from launching (can't clear FSTAT[CCIF]) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

27.4.7.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The flash memory module reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

   If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. FSTAT[ACCERR] reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

   Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, FSTAT[FPVIOL] (protection error) flag is set.

   Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting FSTAT[CCIF].

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in FSTAT[MGSTAT0]. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicable, are reported back to the user via the FCCOB and FSTAT registers.
4. The flash memory module sets FSTAT[CCIF] signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

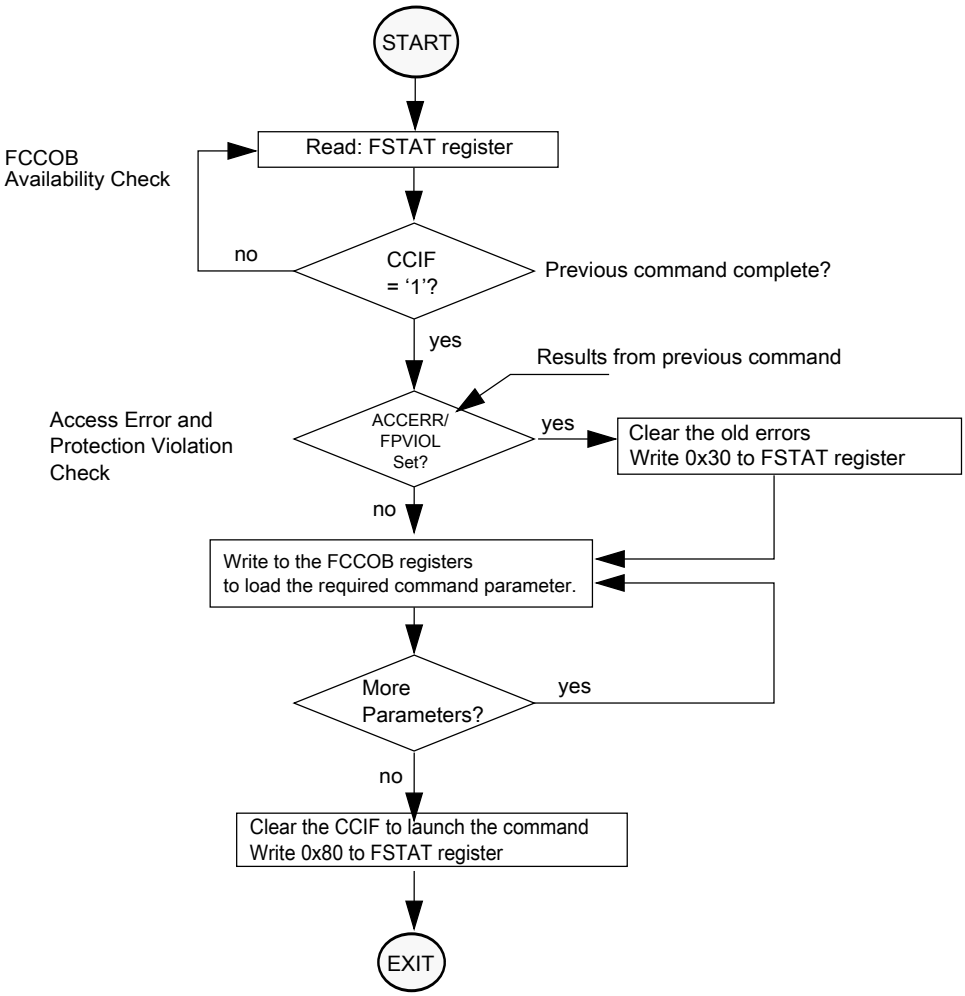


Figure 27-3. Generic flash command write sequence flowchart

27.4.7.2 Flash Commands

The following table summarizes the function of all flash commands.

FCMD	Command	Program flash 0	Program flash 1	Function
0x00	Read 1s Block	×	×	Verify that a program flash block is erased.
0x01	Read 1s Section	×	×	Verify that a given number of program flash locations from a starting address are erased.
0x02	Program Check	×	×	Tests previously-programmed locations at margin read levels.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Function
0x03	Read Resource	IFR, ID	IFR	Read 4 bytes from program flash IFR or version ID.
0x06	Program Longword	×	×	Program 4 bytes in a program flash block.
0x08	Erase Flash Block	×	×	Erase a program flash block. An erase of any flash block is only possible when unprotected.
0x09	Erase Flash Sector	×	×	Erase all bytes in a program flash sector.
0x40	Read 1s All Blocks	×	×	Verify that all program flash blocks are erased then release MCU security.
0x41	Read Once	IFR		Read 4 bytes of a dedicated 64 byte field in the program flash 0 IFR.
0x43	Program Once	IFR		One-time program of 4 bytes of a dedicated 64-byte field in the program flash 0 IFR.
0x44	Erase All Blocks	×	×	Erase all program flash blocks. Then, verify-erase and release MCU security. **NOTE:** An erase is only possible when all memory locations are unprotected.
0x45	Verify Backdoor Access Key	×	×	Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.

27.4.7.3 Allowed Simultaneous Flash Operations

Only the operations marked 'OK' in the following table are permitted to run simultaneously on the program flash memories. Some operations cannot be executed simultaneously because certain hardware resources are shared by the memories.

Table 27-2. Allowed Simultaneous Memory Operations

		Program Flash 0			Program Flash 1		
		Read	Program	Sector Erase	Read	Program	Sector Erase
Program flash 0	Read	—				OK	OK
	Program		—		OK		
	Sector Erase			—	OK		
Program flash 1	Read		OK	OK	—		
	Program	OK				—	
	Sector Erase	OK					—

27.4.8 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Block, Read 1s Section) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. Basic flash array reads use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

27.4.9 Flash Command Description

This section describes all flash commands that can be launched by a command write sequence.

The flash memory module sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that FSTAT[ACCERR] and FSTAT[FPVIOL] are cleared prior to starting the command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the flash memory module is running a command (FSTAT[CCIF] = 0) on that same block. The flash memory module may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

27.4.9.1 Read 1s Block Command

The Read 1s Block command checks to see if an entire program flash block has been erased to the specified margin level. The FCCOB flash address bits determine which flash block is erase-verified.

Table 27-3. Read 1s Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x00 (RD1BLK)
1	Flash address [23:16] in the flash block to be verified
2	Flash address [15:8] in the flash block to be verified
3	Flash address [7:0][1] in the flash block to be verified
4	Read-1 Margin Choice

1. Must be longword aligned (Flash address [1:0] = 00).

After clearing CCIF to launch the Read 1s Block command, the flash memory module sets the read margin for 1s according to Table 27-4 and then reads all locations within the selected program flash block.

Table 27-4. Margin Level Choices for Read 1s Block

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 27-5. Read 1s Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

27.4.9.2 Read 1s Section Command

The Read 1s Section command checks if a section of program flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of phrases to be verified.

Table 27-6. Read 1s Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x01 (RD1SEC)
1	Flash address [23:16] of the first phrase to be verified
2	Flash address [15:8] of the first phrase to be verified
3	Flash address [7:0][1] of the first phrase to be verified
4	Number of phrases to be verified [15:8]
5	Number of phrases to be verified [7:0]
6	Read-1 Margin Choice

1. Must be phrase aligned (Flash address [2:0] = 000).

Upon clearing CCIF to launch the Read 1s Section command, the flash memory module sets the read margin for 1s according to Table 27-7 and then reads all locations within the specified section of flash memory. If the flash memory module fails to read all 1s (that is, the flash section is not erased), FSTAT[MGSTAT0] is set. FSTAT[CCIF] sets after the Read 1s Section operation completes.

Table 27-7. Margin Level Choices for Read 1s Section

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 27-8. Read 1s Section Command Error Handling

Error condition	Error bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin code is supplied.	FSTAT[ACCERR]
An invalid flash address is supplied.	FSTAT[ACCERR]
Flash address is not phrase aligned.	FSTAT[ACCERR]
The requested section crosses a Flash block boundary.	FSTAT[ACCERR]
The requested number of phrases is 0.	FSTAT[ACCERR]
Read-1s fails.	FSTAT[MGSTAT0]

27.4.9.3 Program Check Command

The Program Check command tests a previously programmed program flash longword to see if it reads correctly at the specified margin level.

Table 27-9. Program Check Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x02 (PGMCHK)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Margin Choice
8	Byte 0 expected data
9	Byte 1 expected data
A	Byte 2 expected data
B	Byte 3 expected data

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the flash memory module sets the read margin for 1s according to Table 27-10, reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, FSTAT[MGSTAT0] is set.

The flash memory module then sets the read margin for 0s, re-reads, and compares again. If the comparison at margin-0 fails, FSTAT[MGSTAT0] is set. FSTAT[CCIF] is set after the Program Check operation completes.

The supplied address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10,
- Byte 0 data is programmed to byte address start+0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 27-10. Margin Level Choices for Program Check

Read Margin Choice	Margin Level Description
0x01	Read at 'User' margin-1 and 'User' margin-0
0x02	Read at 'Factory' margin-1 and 'Factory' margin-0

Table 27-11. Program Check Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]

Table continues on the next page...

Table 27-11. Program Check Command Error Handling (continued)

Error Condition	Error Bit
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

27.4.9.4 Read Resource Command

The Read Resource command allows the user to read data from special-purpose memory resources located within the flash memory module. The special-purpose memory resources available include program flash IFR space and the Version ID field. Each resource is assigned a select code as shown in Table 27-13.

Table 27-12. Read Resource Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
Returned Values	
4	Read Data [31:24]
5	Read Data [23:16]
6	Read Data [15:8]
7	Read Data [7:0]
User-provided values	
8	Resource Select Code (see Table 27-13)

1. Must be longword aligned (Flash address [1:0] = 00).

Table 27-13. Read Resource Select Codes

Resource Select Code	Description	Resource Size	Local Address Range
0x00	Program Flash 0 IFR	256 Bytes	0x00_0000–0x00_00FF
0x01[1]	Version ID	8 Bytes	0x00_0000–0x00_0007

1. Located in program flash 0 reserved space.

After clearing CCIF to launch the Read Resource command, four consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag sets after the Read Resource operation completes. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 27-14. Read Resource Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid resource code is entered	FSTAT[ACCERR]
Flash address is out-of-range for the targeted resource.	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]

27.4.9.5 Program Longword Command

The Program Longword command programs four previously-erased bytes in the program flash memory using an embedded algorithm.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 27-15. Program Longword Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x06 (PGM4)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Byte 0 program value
5	Byte 1 program value
6	Byte 2 program value
7	Byte 3 program value

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Longword command, the flash memory module programs the data bytes into the flash using the supplied address. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Longword operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in FSTAT[MGSTAT0]. The CCIF flag is set after the Program Longword operation completes.

The supplied address must be longword aligned (flash address [1:0] = 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10, and
- Byte 0 data is programmed to byte address start+0b11.

Table 27-16. Program Longword Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

27.4.9.6 Erase Flash Block Command

The Erase Flash Block operation erases all addresses in a single program flash.

Table 27-17. Erase Flash Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x08 (ERSBLK)
1	Flash address [23:16] in the flash block to be erased
2	Flash address [15:8] in the flash block to be erased
3	Flash address [7:0][1] in the flash block to be erased

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Erase Flash Block command, the flash memory module erases the main array of the selected flash block and verifies that it is erased. The Erase Flash Block command aborts and sets the FSTAT[FPVIOL] bit if any region

within the block is protected (see the description of the FPROT registers). If the erase verify fails, FSTAT[MGSTAT0] is set. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 27-18. Erase Flash Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Any area of the selected flash block is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s Block command to verify all bits are erased.

27.4.9.7 Erase Flash Sector Command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 27-19. Erase Flash Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x09 (ERSSCR)
1	Flash address [23:16] in the flash sector to be erased
2	Flash address [15:8] in the flash sector to be erased
3	Flash address [7:0][1] in the flash sector to be erased

1. Must be phrase aligned (flash address [2:0] = 000).

After clearing CCIF to launch the Erase Flash Sector command, the flash memory module erases the selected program flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description of the FPROT registers). If the erase-verify fails the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase Flash Sector operation completes. The Erase Flash Sector command is suspendable (see the FCNFG[ERSSUSP] bit and Figure 27-4).

Table 27-20. Erase Flash Sector Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid Flash address is supplied	FSTAT[ACCERR]
Flash address is not phrase aligned	FSTAT[ACCERR]
The selected program flash sector is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s Section command to verify all bits are erased.

27.4.9.7.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit when CCIF is clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see Erase Flash Sector Command), the flash memory module samples the state of the ERSSUSP bit at convenient points. If the flash memory module detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the flash memory module sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the flash memory module detects that a suspend request has been made, the flash memory module clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the flash memory module sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the flash memory module has acknowledged it.

27.4.9.7.2 Resuming a Suspended Erase Flash Sector Operation

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The flash memory module acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit of 4.3 msec between the request to resume the Erase Flash Sector operation (CCIF is cleared) and the request to suspend the operation again (ERSSUSP is set). This minimum time period is required to ensure that the Erase Flash Sector operation will eventually complete. If the minimum period is continually violated, i.e. the suspend requests come repeatedly and too quickly, no forward progress is made by the Erase Flash Sector algorithm. The resume/suspend sequence runs indefinitely without completing the erase.

27.4.9.7.3 Aborting a Suspended Erase Flash Sector Operation

The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the flash memory module starts the new command using the new FCCOB contents.

Note

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

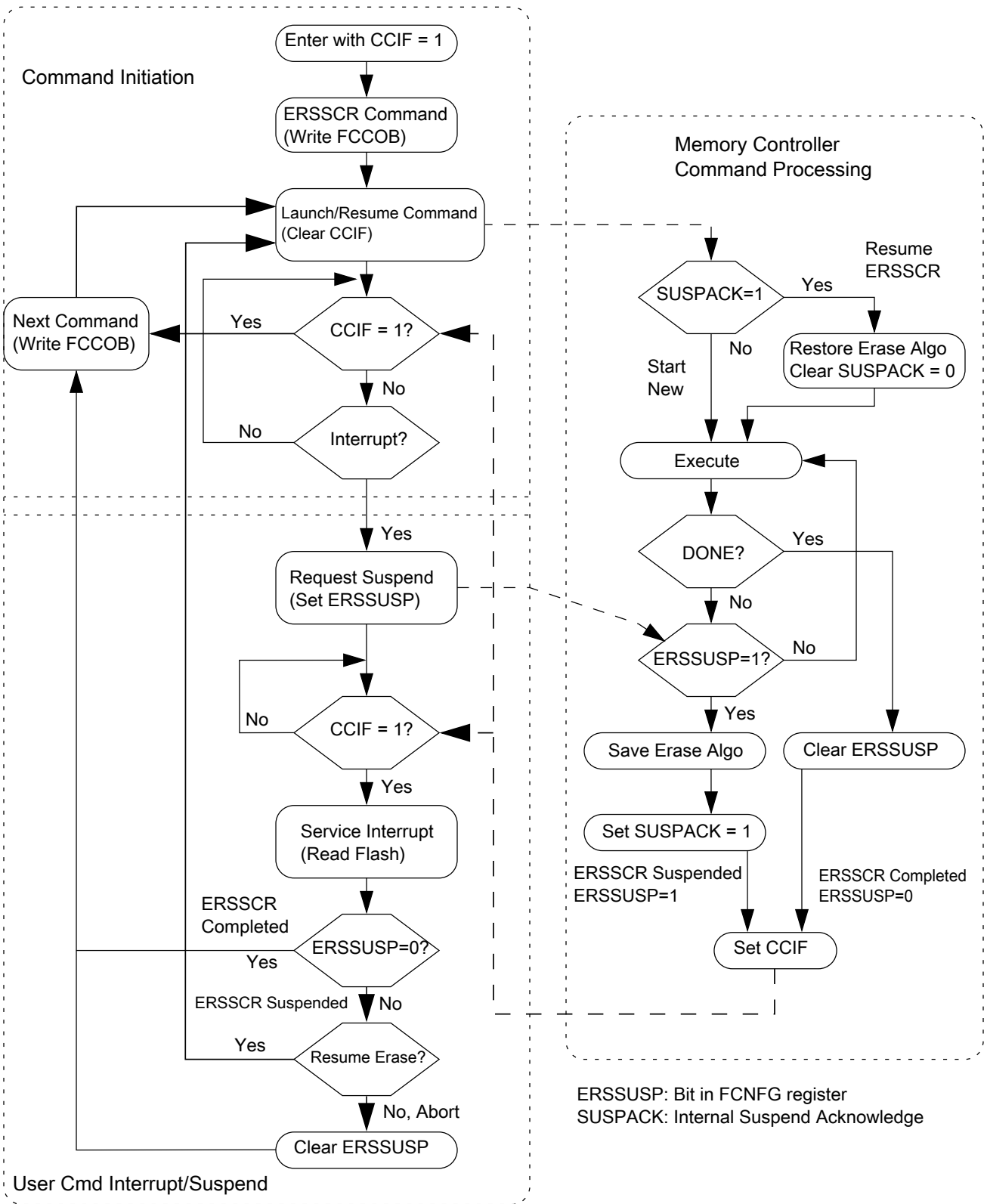


Figure 27-4. Suspend and Resume of Erase Flash Sector Operation

27.4.9.8 Read 1s All Blocks Command

The Read 1s All Blocks command checks if the program flash blocks have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

Table 27-21. Read 1s All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x40 (RD1ALL)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Blocks command, the flash memory module :

- sets the read margin for 1s according to Table 27-22,
- checks the contents of the program flash are in the erased state.

If the flash memory module confirms that these memory resources are erased, security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see Flash Configuration Field Description) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 27-22. Margin Level Choices for Read 1s All Blocks

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 27-23. Read 1s All Blocks Command Error Handling

Error Condition	Error Bit
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

27.4.9.9 Read Once Command

The Read Once command provides read access to special 64-byte fields located in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. These fields are programmed using the Program Once command described in Program Once Command.

Table 27-24. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Program Once record index (0x00 - 0x0F)
2	Not used
3	Not used
Returned Values	
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value

After clearing CCIF to launch the Read Once command, a 4-byte Program Once record is read and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. Valid record index values for the Read Once command range from 0x00 - 0x0F. During execution of the Read Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data. The Read Once command can be executed any number of times.

Table 27-25. Read Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]

27.4.9.10 Program Once Command

The Program Once command enables programming to special 64-byte fields in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. These records can be read using the Read Once command (see Read Once Command) or using the Read Resource command (see Read Resource Command). These records can be programmed only once since the program flash 0 IFR cannot be erased.

Table 27-26. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x0F)
2	Not Used
3	Not Used
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value

After clearing CCIF to launch the Program Once command, the flash memory module first verifies that the selected record is erased. If erased, then the selected record is programmed using the values provided. The Program Once command also verifies that the programmed values read back correctly. The CCIF flag is set after the Program Once operation has completed.

Any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. Valid record index values for the Program Once command range from 0x00 - 0x0F. During execution of the Program Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data.

Table 27-27. Program Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]
The requested record has already been programmed to a non-FFFF value[1]	FSTAT[ACCERR]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

1. If a Program Once record is initially programmed to 0xFFFF_FFFF, the Program Once command is allowed to execute again on that same record.

27.4.9.11 Erase All Blocks Command

The Erase All Blocks operation erases all flash memory, verifies all memory contents, and releases MCU security.

Table 27-28. Erase All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x44 (ERSALL)

After clearing CCIF to launch the Erase All Blocks command, the flash memory module erases all program flash memory, then verifies that all are erased.

If the flash memory module verifies that all flash memories were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state. The Erase All Blocks command aborts if any flash region is protected. The security byte and all other contents of the flash configuration field (see Flash Configuration Field Description) are erased by the Erase All Blocks command. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks operation completes.

Table 27-29. Erase All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any region of the program flash memory is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s All Blocks command to verify all bits are erased.

27.4.9.11.1 Triggering an Erase All External to the Flash Memory Module

The functionality of the Erase All Blocks command is also available in an uncommanded fashion outside of the flash memory. Refer to the device's Chip Configuration details for information on this functionality.

Before invoking the external erase all function, the FSTAT[ACCERR and PVIOL] flags must be cleared and the FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory regardless of the protection settings. If the post-erase verify passes, the routine then releases security by setting the FSEC[SEC] field register to the unsecure state. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting is available as described in Erase All Blocks Command.

27.4.9.12 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command only executes if the mode and security conditions are satisfied (see Flash Commands by Mode). Execution of the Verify Backdoor Access Key command is further qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash

Configuration Field (see Flash Configuration Field Description). The column labelled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 27-30. Verify Backdoor Access Key Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]	Flash Configuration Field Offset Address
0	0x45 (VFYKEY)	
1-3	Not Used	
4	Key Byte 0	0x0_0003
5	Key Byte 1	0x0_0002
6	Key Byte 2	0x0_0001
7	Key Byte 3	0x0_0000
8	Key Byte 4	0x0_0007
9	Key Byte 5	0x0_0006
A	Key Byte 6	0x0_0005
B	Key Byte 7	0x0_0004

After clearing CCIF to launch the Verify Backdoor Access Key command, the flash memory module checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the flash memory module sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the flash memory module compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the flash memory module module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

Table 27-31. Verify Backdoor Access Key Command Error Handling

Error Condition	Error Bit
The supplied key is all-0s or all-Fs	FSTAT[ACCERR]
An incorrect backdoor key is supplied	FSTAT[ACCERR]
Backdoor key access has not been enabled (see the description of the FSEC register)	FSTAT[ACCERR]
This command is launched and the backdoor key has mismatched since the last power down reset	FSTAT[ACCERR]

27.4.10 Security

The flash memory module provides security information to the MCU based on contents of the FSEC security register.

The MCU then limits access to flash memory resources as defined in the device's Chip Configuration details. During reset, the flash memory module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see Flash Configuration Field Description).

The following fields are available in the FSEC register. The settings are described in the Flash Security Register (FTFA_FSEC) details.

Flash security features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Note that not all features described in the application note are available on this device.

Table 27-32. FSEC register fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Freescale Factory Access
SEC	MCU security

27.4.10.1 Changing the Security State

The security state out of reset can be permanently changed by programming the security byte of the flash configuration field. This assumes that you are starting from a mode where the necessary program flash erase and program commands are available and that the region of the program flash containing the flash configuration field is unprotected. If the flash security byte is successfully programmed, its new value takes affect after the next chip reset.

27.4.10.1.1 Unsecuring the Chip Using Backdoor Key Access

The chip can be unsecured by using the backdoor key access feature, which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see Flash Configuration Field Description). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see Verify Backdoor Access Key Command) can be run; it allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the chip. The entire 8-byte key cannot be all 0s or all 1s; that is,

0000_0000_0000_0000h and FFFF_FFFF_FFFF_FFFFh are not accepted by the Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN bits are in the enabled state, the chip can be unsecured by the following backdoor key access sequence:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key Command
2. If the Verify Backdoor Access Key command is successful, the chip is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits further use of the Verify Backdoor Access Key command. A reset of the chip is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the chip is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field (Flash Configuration Field Description). After the next reset of the chip, the security state of the flash memory module reverts back to the flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured chip has full control of the contents of the Flash Configuration Field. The chip may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

27.4.11 Reset Sequence

On each system reset the flash memory module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FOPT, and FSEC registers.

FSTAT[CCIF] is cleared throughout the reset sequence. The flash memory module holds off CPU access during the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any flash command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

Chapter 28
Flash Memory Controller (FMC)

28.1 Introduction

The Flash Memory Controller (FMC) is a memory acceleration unit. A list of features provided by the FMC can be found here.

- an interface between bus masters and the 64-bit program flash memory.
- a buffer and a cache that can accelerate program flash memory data transfers.

28.1.1 Overview

The Flash Memory Controller manages the interface between bus masters and the 64-bit program flash memory. The FMC receives status information detailing the configuration of the flash memory and uses this information to ensure a proper interface. The FMC supports 8-bit, 16-bit, and 32-bit read operations from the program flash memory. A write operation to program flash memory results in a bus error.

In addition, the FMC provides two separate mechanisms for accelerating the interface between bus masters and program flash memory. A 64-bit speculation buffer can prefetch the next 64-bit flash memory location, and a 4-way, 4-set program flash memory cache can store previously accessed program flash memory data for quick access times.

28.1.2 Features

The features of FMC module include:

- Interface between bus masters and the 64-bit program flash memory:
  - 8-bit, 16-bit, and 32-bit read operations to nonvolatile flash memory.
- Acceleration of data transfer from the program flash memory to the device:

- 64-bit prefetch speculation buffer for program flash accesses with controls for instruction/data access
- 4-way, 4-set, 64-bit line size program flash memory cache for a total of sixteen 64-bit entries with invalidation control

28.2 Modes of operation

The FMC operates only when a bus master accesses the program flash memory.

In terms of chip power modes:

- The FMC operates only in Run and Wait modes, including VLPR and VLPW modes.
- For any power mode where the program flash memory cannot be accessed, the FMC is disabled.

28.3 External signal description

The FMC has no external (off-chip) signals.

28.4 Memory map and register descriptions

The MCM's programming model provides control and configuration of the FMC's features.

For details, see the description of the MCM's Platform Control Register (PLACR).

28.5 Functional description

The FMC is a flash acceleration unit with flexible buffers for user configuration.

Besides managing the interface between bus masters and the program flash memory, the FMC can be used to customize the program flash memory cache and buffer to provide single-cycle system clock data access times. Whenever a hit occurs for the prefetch speculation buffer or the cache (when enabled), the requested data is transferred within a single system clock.

Upon system reset, the FMC is configured as follows:

- Flash cache is enabled.
- Instruction speculation and caching are enabled.

- Data speculation is disabled.
- Data caching is enabled.

Though the default configuration provides flash acceleration, advanced users may desire to customize the FMC buffer configurations to maximize throughput for their use cases. For example, the user may adjust the controls to enable buffering per access type (data or instruction).

NOTE

When reconfiguring the FMC, do not program the control and configuration inputs to the FMC while the program flash memory is being accessed. Instead, change them with a routine executing from RAM in supervisor mode.

Chapter 29
Watchdog Timer (WDOG)

29.1 Chip-specific WDOG information

29.1.1 Overview

This device includes the Independent Watchdog (WDOG) to provide fault tolerant operation for metering and general purpose use cases. With the introduction of the IEC 60730 standards, it is required that even automatic electronic controls in household appliances ensure safe and reliable operation of their components.

The WDOG module provides several clocking and robust refresh mechanisms, which make this module an important system integrity and reliability component on this device.

29.1.2 Clock Connections

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

Table 29-1. WDOG clock connections

Module clock	Chip clock
LPO Oscillator	1 kHz LPO Clock
Alt Clock	32 kHz Clock[1]
Fast Test Clock	Bus Clock
System Bus Clock	Bus Clock

1. The clock which is selected by SIM_OPT1[OSC32KSEL]

29.2 Introduction

The Watchdog Timer (WDOG) keeps a watch on the system functioning and resets it in case of its failure. Reasons for failure include run-away software code and the stoppage of the system clock that in a safety critical system can lead to serious consequences. In such cases, the watchdog brings the system into a safe state of operation. The watchdog monitors the operation of the system by expecting periodic communication from the software, generally known as servicing or refreshing the watchdog. If this periodic refreshing does not occur, the watchdog resets the system.

29.3 Features

The features of the Watchdog Timer (WDOG) include:

- Clock source input independent from CPU/bus clock. Choice between two clock sources:
  - Low-power oscillator (LPO)
  - External system clock
- Unlock sequence for allowing updates to write-once WDOG control/configuration bits.
- All WDOG control/configuration bits are writable once only within 256 bus clock cycles of being unlocked.
  - You need to always update these bits after unlocking within 256 bus clock cycles. Failure to update these bits resets the system.
- Programmable time-out period specified in terms of number of WDOG clock cycles.
- Ability to test WDOG timer and reset with a flag indicating watchdog test.
  - Quick test—Small time-out value programmed for quick test.
  - Byte test—Individual bytes of timer tested one at a time.
  - Read-only access to the WDOG timer—Allows dynamic check that WDOG timer is operational.

NOTE

Reading the watchdog timer counter while running the watchdog on the bus clock might not give the accurate counter value.

- Windowed refresh option
  - Provides robust check that program flow is faster than expected.
  - Programmable window.
  - Refresh outside window leads to reset.
- Robust refresh mechanism
  - Write values of 0xA602 and 0xB480 to WDOG Refresh Register within 20 bus clock cycles.
- Count of WDOG resets as they occur.
- Configurable interrupt on time-out to provide debug breadcrumbs. This is followed by a reset after 256 bus clock cycles.

29.4 Functional overview

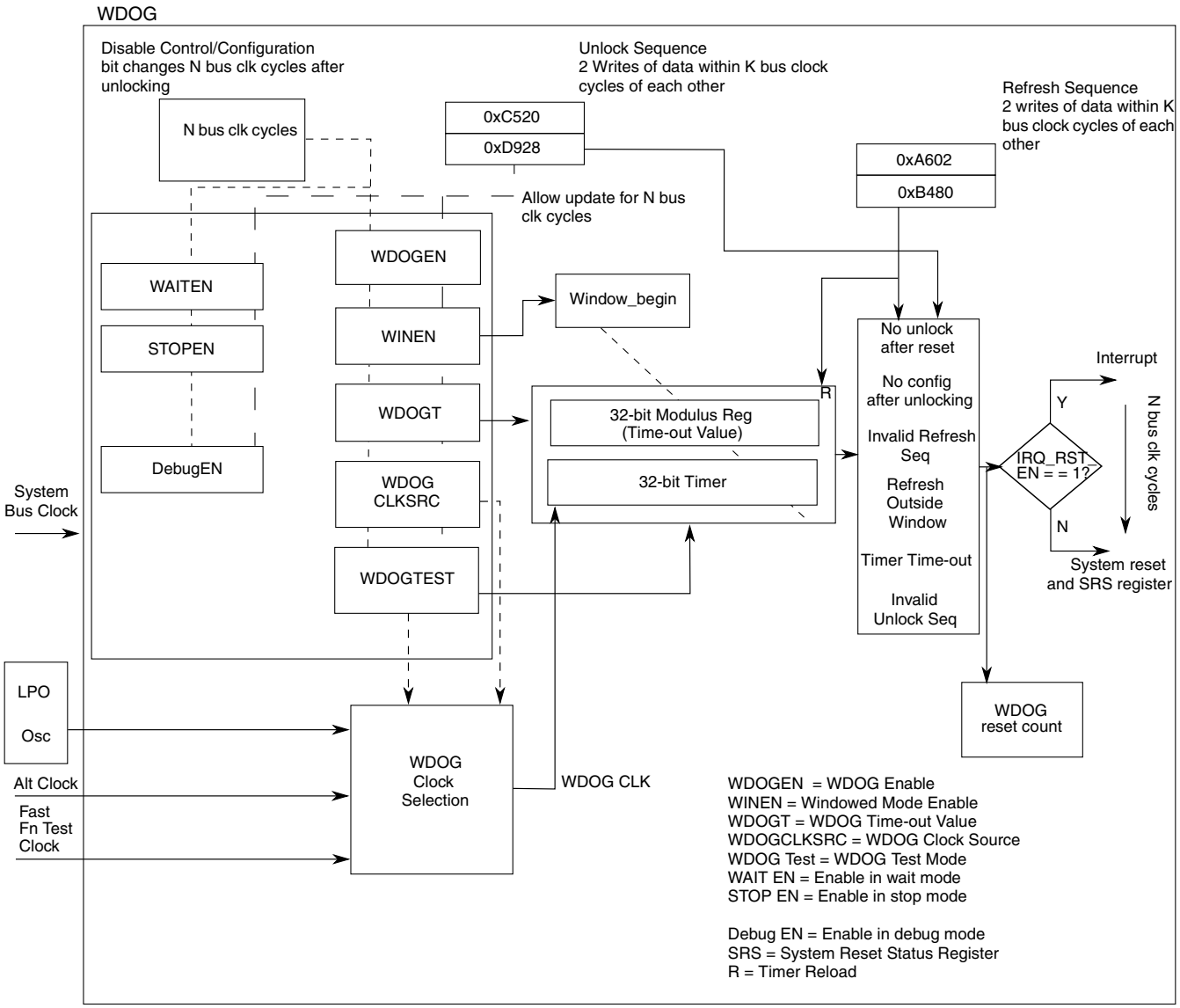


Figure 29-1. WDOG operation

The preceding figure shows the operation of the watchdog. The values for N and K are:

- N = 256
- K = 20

The watchdog is a fail safe mechanism that brings the system into a known initial state in case of its failure due to CPU clock stopping or a run-away condition in code execution. In its simplest form, the watchdog timer runs continuously off a clock source and expects to be serviced periodically, failing which it resets the system. This ensures that the software is executing correctly and has not run away in an unintended direction. Software can adjust the period of servicing or the time-out value for the watchdog timer to meet the needs of the application.

You can select a windowed mode of operation that expects the servicing to be done only in a particular window of the time-out period. An attempted servicing of the watchdog outside this window results in a reset. By operating in this mode, you can get an indication of whether the code is running faster than expected. The window length is also user programmable.

If a system fails to update/refresh the watchdog due to an unknown and persistent cause, it will be caught in an endless cycle of resets from the watchdog. To analyze the cause of such conditions, you can program the watchdog to first issue an interrupt, followed by a reset. In the interrupt service routine, the software can analyze the system stack to aid debugging.

To enhance the independence of watchdog from the system, it runs off an independent LPO oscillator clock. You can also switch over to an alternate clock source if required, through a control register bit.

29.4.1 Unlocking and updating the watchdog

As long as ALLOW_UPDATE in the watchdog control register is set, you can unlock and modify the write-once-only control and configuration registers:

1. Write 0xC520 followed by 0xD928 within 20 bus clock cycles to a specific unlock register (WDOG_UNLOCK).
2. Wait one bus clock cycle. You cannot update registers on the bus clock cycle immediately following the write of the unlock sequence.
3. An update window equal in length to the watchdog configuration time (WCT) opens. Within this window, you can update the configuration and control register bits.

These register bits can be modified only once after unlocking.

If none of the configuration and control registers is updated within the update window, the watchdog issues a reset, that is, interrupt-then-reset, to the system. Trying to unlock the watchdog within the WCT after an initial unlock has no effect. During the update operation, the watchdog timer is not paused and continues running in the background. After the update window closes, the watchdog timer restarts and the watchdog functions according to the new configuration.

The update feature is useful for applications that have an initial, non-safety critical part, where the watchdog is kept disabled or with a conveniently long time-out period. This means the application coder does not have to frequently service the watchdog. After the critical part of the application begins, the watchdog can be reconfigured as needed.

The watchdog issues a reset, that is, interrupt-then-reset if enabled, to the system for any of these invalid unlock sequences:

- Write any value other than 0xC520 or 0xD928 to the unlock register.
- ALLOW_UPDATE is set and a gap of more than 20 bus clock cycles is inserted between the writing of the unlock sequence values.

An attempted refresh operation between the two writes of the unlock sequence and in the WCT time following a successful unlock, goes undetected. Also, see Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the unlock register.

Note

A context switch during unlocking and refreshing may lead to a watchdog reset.

29.4.2 Watchdog configuration time (WCT)

To prevent unintended modification of the watchdog's control and configuration register bits, you are allowed to update them only within a period of 256 bus clock cycles after unlocking. This period is known as the watchdog configuration time (WCT). In addition, these register bits can be modified only once after unlocking them for editing, even after reset.

You must unlock the registers within WCT after system reset, failing which the WDOG issues a reset to the system. In other words, you must write at least the first word of the unlocking sequence within the WCT after reset. After this is done, you have a further 20 bus clock cycles, the maximum allowed gap between the words of the unlock sequence, to complete the unlocking operation. Thereafter, to make sure that you do not forget to configure the watchdog, the watchdog issues a reset if none of the WDOG control and configuration registers is updated in the WCT after unlock. After the close of this window or after the first write, these register bits are locked out from any further changes.

The watchdog timer keeps running according to its default configuration through unlocking and update operations that can extend up to a maximum total of 2xWCT + 20 bus clock cycles. Therefore, it must be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.

Updates in the write-once registers take effect only after the WCT window closes with the following exceptions for which changes take effect immediately:

- Stop and Debug mode enable
- IRQ_RST_EN

The operations of refreshing the watchdog goes undetected during the WCT.

29.4.3 Refreshing the watchdog

A robust refreshing mechanism has been chosen for the watchdog. A valid refresh is a write of 0xA602 followed by 0xB480 within 20 bus clock cycles to watchdog refresh register. If these two values are written more than 20 bus cycles apart or if something other than these two values is written to the register, a watchdog reset, or interrupt-then-reset if enabled, is issued to the system. A valid refresh makes the watchdog timer restart on the next bus clock. Also, an attempted unlock operation in between the two writes of the refresh sequence goes undetected. See Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the refresh register.

29.4.4 Windowed mode of operation

In this mode of operation, a restriction is placed on the point in time within the time-out period at which the watchdog can be refreshed. The refresh is considered valid only when the watchdog timer increments beyond a certain count as specified by the watchdog window register. This is known as refreshing the watchdog within a window of the total time-out period. If a refresh is attempted before the timer reaches the window value, the watchdog generates a reset, or interrupt-then-reset if enabled. If there is no refresh at all, the watchdog times out and generates a reset or interrupt-then-reset if enabled.

29.4.5 Watchdog disabled mode of operation

When the watchdog is disabled through the WDOG_EN bit in the watchdog status and control register, the watchdog timer is reset to zero and is disabled from counting until you enable it or it is enabled again by the system reset. In this mode, the watchdog timer cannot be refreshed–there is no requirement to do so while the timer is disabled. However, the watchdog still generates a reset, or interrupt-then-reset if enabled, on a non-time-out exception. See Generated Resets and Interrupts. You need to unlock the watchdog before enabling it. A system reset brings the watchdog out of the disabled mode.

29.4.6 Low-power modes of operation

The low-power modes of operation of the watchdog are described in the following table:

Table 29-2. Low-power modes of operation

Mode	Behavior
Wait	If the WDOG is enabled (WAIT_EN = 1), it can run on bus clock or low-power oscillator clock (CLK_SRC = x) to generate interrupt (IRQ_RST_EN=1) followed by a reset on time-out. After reset the WDOG reset counter increments by one.
Stop	Where the bus clock is gated, the WDOG can run only on low-power oscillator clock (CLK_SRC=0) if it is enabled in stop (STOP_EN=1). In this case, the WDOG runs to time-out twice, and then generates a reset from its backup circuitry. Therefore, if you program the watchdog to time-out after 100 ms and then enter such a stop mode, the reset will occur after 200 ms. Also, in this case, no interrupt will be generated irrespective of the value of IRQ_RST_EN bit. After WDOG reset, the WDOG reset counter will also not increment.
Power-Down	The watchdog is • powered off in VLLSx mode

29.4.7 Low-power and Debug modes of operation

You can program the watchdog to disable in system stop, wait, and debug modes through STOP_EN, WAIT_EN, DBG_EN in the watchdog control register. This results in the watchdog timer pausing for the duration of the mode. Register read/writes are still allowed, which means that operations like refresh, unlock, and so on are allowed. Upon exit from the mode, the timer resumes its operation from the point of pausing.

The entry of the system into the debugabove mentioned modes does not excuse it from compulsorily configuring the watchdog in the WCT time after unlock, unless the system bus clock is gated off, in which case the internal state machine pauses too. Failing to do so still results in a reset, or interrupt-then-reset, if enabled, to the system. Also, all of the exception conditions that result in a reset to the system, as described in Generated Resets and Interrupts, are still valid in this these modes. So, if an exception condition occurs and the system bus clock is on, a reset occurs, or interrupt-then-reset, if enabled.

The entry into Debug mode within WCT after reset is treated differently. The WDOG timer is kept reset to zero and there is no need to unlock and configure it within WCT. You must not try to refresh or unlock the WDOG in this state or unknown behavior may result. Upon exit from this these modes, the WDOG timer restarts and the WDOG has to be unlocked and configured within WCT.

29.5 Testing the watchdog

For IEC 60730 and other safety standards, the expectation is that anything that monitors a safety function must be tested, and this test is required to be fault tolerant. To test the watchdog, its main timer and its associated compare and reset logic must be tested. To

this end, two tests are implemented for the watchdog, as described in Quick Test and Byte Test. A control bit is provided to put the watchdog into functional test mode. There is also an overriding test-disable control bit which allows the functional test mode to be disabled permanently. After it is set, this test-disable bit can only be cleared by a reset.

These two tests achieve the overall aim of testing the counter functioning and the compare and reset logic.

Note

Do not enable the watchdog interrupt during these tests. If required, you must ensure that the effective time-out value is greater than WCT time. See Generated Resets and Interrupts for more details.

To run a particular test:

1. Select either quick test or byte test..
2. Set a certain test mode bit to put the watchdog in the functional test mode. Setting this bit automatically switches the watchdog timer to a fast clock source. The switching of the clock source is done to achieve a faster time-out and hence a faster test.

In a successful test, the timer times out after reaching the programmed time-out value and generates a system reset.

Note

After emerging from a reset due to a watchdog test, unlock and configure the watchdog. The refresh and unlock operations and interrupt are not automatically disabled in the test mode.

29.5.1 Quick test

In this test, the time-out value of watchdog timer is programmed to a very low value to achieve quick time-out. The only difference between the quick test and the normal mode of the watchdog is that TESTWDOG is set for the quick test. This allows for a faster test of the watchdog reset mechanism.

29.5.2 Byte test

The byte test is a more thorough a test of the watchdog timer. In this test, the timer is split up into its constituent byte-wide stages that are run independently and tested for time-out against the corresponding byte of the time-out value register. The following figure explains the splitting concept:

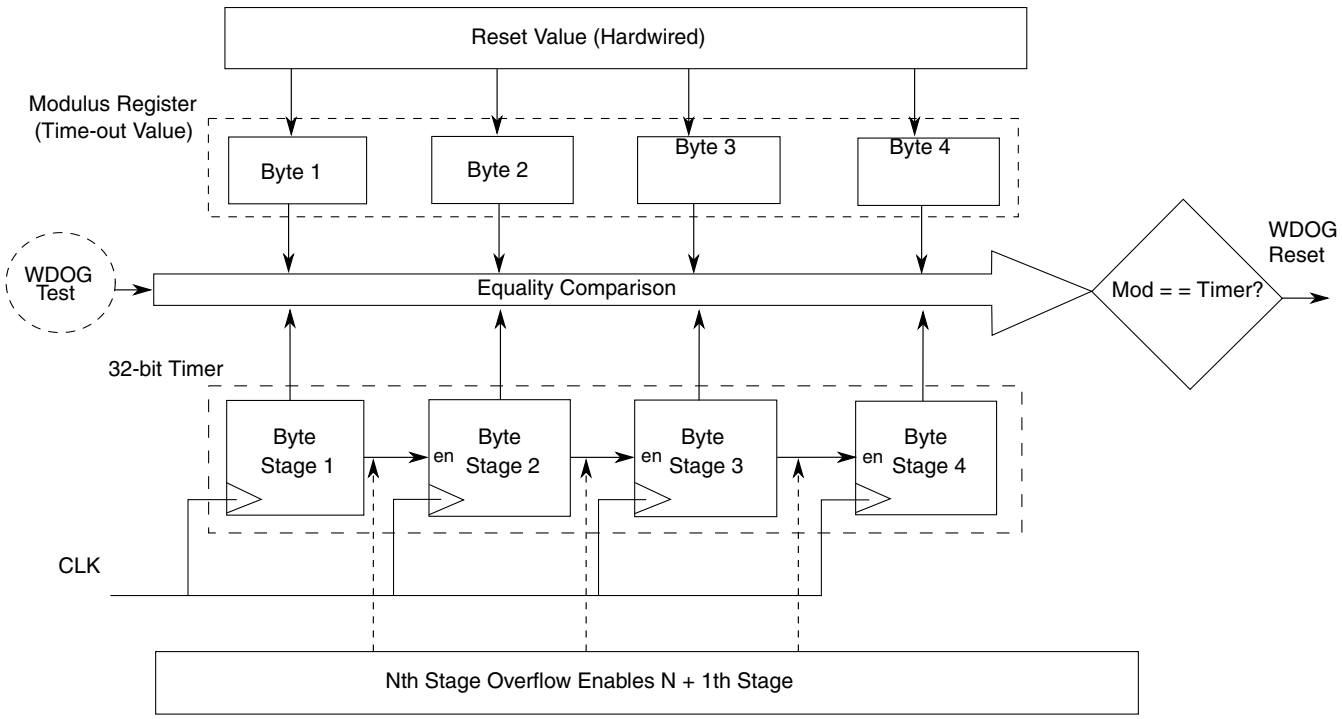


Figure 29-2. Watchdog timer byte splitting

Each stage is an 8-bit synchronous counter followed by combinational logic that generates an overflow signal. The overflow signal acts as an enable to the N + 1th stage.

In the test mode, when an individual byte, N, is tested, byte N – 1 is loaded forcefully with 0xFF, and both these bytes are allowed to run off the clock source. By doing so, the overflow signal from stage N – 1 is generated immediately, enabling counter stage N. The Nth stage runs and compares with the Nth byte of the time-out value register. In this way, the byte N is also tested along with the link between it and the preceding stage. No other stages, N – 2, N – 3... and N + 1, N + 2... are enabled for the test on byte N. These disabled stages, except the most significant stage of the counter, are loaded with a value of 0xFF.

29.6 Backup reset generator

The backup reset generator generates the final reset which goes out to the system. It has a backup mechanism which ensures that in case the bus clock stops and prevents the main state machine from generating a reset exception/interrupt, the watchdog timer's time-out is separately routed out as a reset to the system. Two successive timer time-outs without an intervening system reset result in the backup reset generator routing out the time-out signal as a reset to the system.

29.7 Generated resets and interrupts

The watchdog generates a reset in the following events, also referred to as exceptions:

- A watchdog time-out
- Failure to unlock the watchdog within WCT time after system reset deassertion
- No update of the control and configuration registers within the WCT window after unlocking. At least one of the following registers must be written to within the WCT window to avoid reset:
  - WDOG_ST_CTRL_H, WDOG_ST_CTRL_L
  - WDOG_TO_VAL_H, WDOG_TO_VAL_L
  - WDOG_WIN_H, WDOG_WIN_L
  - WDOG_PRESCALER
- A value other than the unlock sequence or the refresh sequence is written to the unlock and/or refresh registers, respectively.
- A gap of more than 20 bus cycles exists between the writes of two values of the unlock sequence.
- A gap of more than 20 bus cycles exists between the writes of two values of the refresh sequence.

The watchdog can also generate an interrupt. If IRQ_RST_EN is set, then on the above mentioned events WDOG_ST_CTRL_L[INT_FLG] is set, generating an interrupt. A watchdog reset is also generated WCT time later to ensure the watchdog is fault tolerant. The interrupt can be cleared by writing 1 to INT_FLG.

The gap of WCT between interrupt and reset means that the WDOG time-out value must be greater than WCT. Otherwise, if the interrupt was generated due to a time-out, a second consecutive time-out will occur in that WCT gap. This will trigger the backup reset generator to generate a reset to the system, prematurely ending the interrupt service routine execution. Also, jobs such as counting the number of watchdog resets would not be done.

29.8 Memory map and register definition

This section consists of the memory map and register descriptions.

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_3000	Watchdog Status and Control Register High (WDOG_STCTRLH)	16	R/W	See section	29.8.1/513
4005_3002	Watchdog Status and Control Register Low (WDOG_STCTRLL)	16	R/W	0001h	29.8.2/514
4005_3004	Watchdog Time-out Value Register High (WDOG_TOVALH)	16	R/W	004Ch	29.8.3/515
4005_3006	Watchdog Time-out Value Register Low (WDOG_TOVALL)	16	R/W	4B40h	29.8.4/515
4005_3008	Watchdog Window Register High (WDOG_WINH)	16	R/W	0000h	29.8.5/516
4005_300A	Watchdog Window Register Low (WDOG_WINL)	16	R/W	0010h	29.8.6/516
4005_300C	Watchdog Refresh register (WDOG_REFRESH)	16	R/W	B480h	29.8.7/517
4005_300E	Watchdog Unlock register (WDOG_UNLOCK)	16	R/W	D928h	29.8.8/517
4005_3010	Watchdog Timer Output Register High (WDOG_TMROUTH)	16	R/W	0000h	29.8.9/517
4005_3012	Watchdog Timer Output Register Low (WDOG_TMROUTL)	16	R/W	0000h	29.8.10/ 518
4005_3014	Watchdog Reset Count register (WDOG_RSTCNT)	16	R/W	0000h	29.8.11/ 518
4005_3016	Watchdog Prescaler register (WDOG_PRESC)	16	R/W	0400h	29.8.12/ 518

29.8.1 Watchdog Status and Control Register High (WDOG_STCTRLH)

Address: 4005_3000h base + 0h offset = 4005_3000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	DISTESTWDOG	BYTESEL[1:0]		TESTSEL	TESTWDOG	0	Reserved	0	STOPEN	DBGEN	ALLOWUPDATE	WINEN	IRQRSTEN	CLKSRC	WDOGEN
Write																
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1

WDOG_STCTRLH field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DISTESTWDOG	Allows the WDOG's functional test mode to be disabled permanently. After it is set, it can only be cleared by a reset. It cannot be unlocked for editing after it is set. 0 WDOG functional test mode is not disabled. 1 WDOG functional test mode is disabled permanently until reset.
13–12 BYTESEL[1:0]	This 2-bit field selects the byte to be tested when the watchdog is in the byte test mode. 00 Byte 0 selected 01 Byte 1 selected 10 Byte 2 selected 11 Byte 3 selected
11 TESTSEL	Effective only if TESTWDOG is set. Selects the test to be run on the watchdog timer. 0 Quick test. The timer runs in normal operation. You can load a small time-out value to do a quick test. 1 Byte test. Puts the timer in the byte test mode where individual bytes of the timer are enabled for operation and are compared for time-out against the corresponding byte of the programmed time-out value. Select the byte through BYTESEL[1:0] for testing.
10 TESTWDOG	Puts the watchdog in the functional test mode. In this mode, the watchdog timer and the associated compare and reset generation logic is tested for correct operation. The clock for the timer is switched from the main watchdog clock to the fast clock input for watchdog functional test. The TESTSEL bit selects the test to be run.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 Reserved	This field is reserved.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 STOPEN	Enables or disables WDOG in Stop mode. 0 WDOG is disabled in CPU Stop mode. 1 WDOG is enabled in CPU Stop mode.

Table continues on the next page...

WDOG_STCTRLH field descriptions (continued)

Field	Description
5 DBGEN	Enables or disables WDOG in Debug mode. 0 WDOG is disabled in CPU Debug mode. 1 WDOG is enabled in CPU Debug mode.
4 ALLOWUPDATE	Enables updates to watchdog write-once registers, after the reset-triggered initial configuration window (WCT) closes, through unlock sequence. 0 No further updates allowed to WDOG write-once registers. 1 WDOG write-once registers can be unlocked for updating.
3 WINEN	Enables Windowing mode. 0 Windowing mode is disabled. 1 Windowing mode is enabled.
2 IRQRSTEN	Used to enable the debug breadcrumbs feature. A change in this bit is updated immediately, as opposed to updating after WCT. 0 WDOG time-out generates reset only. 1 WDOG time-out initially generates an interrupt. After WCT, it generates a reset.
1 CLKSRC	Selects clock source for the WDOG timer and other internal timing operations. 0 WDOG clock sourced from LPO . 1 WDOG clock sourced from alternate clock source.
0 WDOGEN	Enables or disables the WDOG's operation. In the disabled state, the watchdog timer is kept in the reset state, but the other exception conditions can still trigger a reset/interrupt. A change in the value of this bit must be held for more than one WDOG_CLK cycle for the WDOG to be enabled or disabled. 0 WDOG is disabled. 1 WDOG is enabled.

29.8.2 Watchdog Status and Control Register Low (WDOG_STCTRLL)

Address: 4005_3000h base + 2h offset = 4005_3002h

Bit	15	14	13	12	11	10	9	8
Read Write	INTFLG	Reserved						
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved							
Reset	0	0	0	0	0	0	0	1

WDOG_STCTRLL field descriptions

Field	Description
15 INTFLG	Interrupt flag. It is set when an exception occurs. IRQRSTEN = 1 is a precondition to set this flag. INTFLG = 1 results in an interrupt being issued followed by a reset, WCT later. The interrupt can be cleared by writing 1 to this bit. It also gets cleared on a system reset.
Reserved	This field is reserved. **NOTE:** Do not modify this field value.

29.8.3 Watchdog Time-out Value Register High (WDOG_TOVALH)

Address: 4005_3000h base + 4h offset = 4005_3004h

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

WDOG_TOVALH field descriptions

Field	Description
TOVALHIGH	Defines the upper 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

29.8.4 Watchdog Time-out Value Register Low (WDOG_TOVALL)

The time-out value of the watchdog must be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.

Address: 4005_3000h base + 6h offset = 4005_3006h

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

WDOG_TOVALL field descriptions

Field	Description
TOVALLOW	Defines the lower 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

29.8.5 Watchdog Window Register High (WDOG_WINH)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: 4005_3000h base + 8h offset = 4005_3008h

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

WDOG_WINH field descriptions

Field	Description
WINHIGH	Defines the upper 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the watchdog clock. In this mode, the watchdog can be refreshed only when the timer has reached a value greater than or equal to this window length. A refresh outside this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

29.8.6 Watchdog Window Register Low (WDOG_WINL)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: 4005_3000h base + Ah offset = 4005_300Ah

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

WDOG_WINL field descriptions

Field	Description
WINLOW	Defines the lower 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the pre-scaled watchdog clock. In this mode, the watchdog can be refreshed only when the timer reaches a value greater than or equal to this window length value. A refresh outside of this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

29.8.7 Watchdog Refresh register (WDOG_REFRESH)

Address: 4005_3000h base + Ch offset = 4005_300Ch

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

WDOG_REFRESH field descriptions

Field	Description
WDOGREFRESH	Watchdog refresh register. A sequence of 0xA602 followed by 0xB480 within 20 bus clock cycles written to this register refreshes the WDOG and prevents it from resetting the system. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system, or if IRQRSTEN is set, it interrupts and then resets the system.

29.8.8 Watchdog Unlock register (WDOG_UNLOCK)

Address: 4005_3000h base + Eh offset = 4005_300Eh

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

WDOG_UNLOCK field descriptions

Field	Description
WDOGUNLOCK	Writing the unlock sequence values to this register to makes the watchdog write-once registers writable again. The required unlock sequence is 0xC520 followed by 0xD928 within 20 bus clock cycles. A valid unlock sequence opens a window equal in length to the WCT within which you can update the registers. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system or if IRQRSTEN is set, it interrupts and then resets the system. The unlock sequence is effective only if ALLOWUPDATE is set.

29.8.9 Watchdog Timer Output Register High (WDOG_TMROUTH)

Address: 4005_3000h base + 10h offset = 4005_3010h

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

WDOG_TMROUTH field descriptions

Field	Description
TIMEROUTHIGH	Shows the value of the upper 16 bits of the watchdog timer.

29.8.10 Watchdog Timer Output Register Low (WDOG_TMROUTL)

During Stop mode, the WDOG_TIMER_OUT will be caught at the pre-stop value of the watchdog timer. After exiting Stop mode, a maximum delay of 1 WDOG_CLK cycle + 3 bus clock cycles will occur before the WDOG_TIMER_OUT starts following the watchdog timer.

Address: 4005_3000h base + 12h offset = 4005_3012h

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

WDOG_TMROUTL field descriptions

Field	Description
TIMEROUTLOW	Shows the value of the lower 16 bits of the watchdog timer.

29.8.11 Watchdog Reset Count register (WDOG_RSTCNT)

Address: 4005_3000h base + 14h offset = 4005_3014h

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

WDOG_RSTCNT field descriptions

Field	Description
RSTCNT	Counts the number of times the watchdog resets the system. This register is reset only on a POR. Writing 1 to the bit to be cleared enables you to clear the contents of this register.

29.8.12 Watchdog Prescaler register (WDOG_PRESC)

Address: 4005_3000h base + 16h offset = 4005_3016h

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

WDOG_PRESC field descriptions

Field	Description
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 PRESCVAL	3-bit prescaler for the watchdog clock source. A value of zero indicates no division of the input WDOG clock. The watchdog clock is divided by (PRESCVAL + 1) to provide the prescaled WDOG_CLK.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

29.9 Watchdog operation with 8-bit access

29.9.1 General guideline

When performing 8-bit accesses to the watchdog's 16-bit registers where the intention is to access both the bytes of a register, place the two 8-bit accesses one after the other in your code.

29.9.2 Refresh and unlock operations with 8-bit access

One exception condition that generates a reset to the system is the write of any value other than those required for a legal refresh/update sequence to the respective refresh and unlock registers.

For an 8-bit access to these registers, writing a correct value requires at least two bus clock cycles, resulting in an invalid value in the registers for one cycle. Therefore, the system is reset even if the intention is to write a correct value to the refresh/unlock register. Keeping this in mind, the exception condition for 8-bit accesses is slightly modified.

Whereas the match for a correct value for a refresh/unlock sequence is as according to the original definition, the match for an incorrect value is done byte-wise on the refresh/unlock rather than for the whole 16-bit value. This means that if the high byte of the refresh/unlock register contains any value other than high bytes of the two values that make up the sequence, it is treated as an exception condition, leading to a reset or interrupt-then-reset. The same holds true for the lower byte of the refresh or unlock register. Take the refresh operation that expects a write of 0xA602 followed by 0xB480 to the refresh register, as an example.

Table 29-3. Refresh for 8-bit access

	WDOG_REFRESH[15:8]	WDOG_REFRESH[7:0]	Sequence value1 or value2 match	Mismatch exception
Current Value	0xB4	0x80	Value2 match	No
Write 1	0xB4	0x02	No match	No
Write 2	0xA6	0x02	Value1 match	No
Write 3	0xB4	0x02	No match	No
Write 4	0xB4	0x80	Value2 match. Sequence complete.	No
Write 5	0x02	0x80	No match	Yes

As shown in the preceding table, the refresh register holds its reset value initially. Thereafter, two 8-bit accesses are performed on the register to write the first value of the refresh sequence. No mismatch exception is registered on the intermediate write, Write1. The sequence is completed by performing two more 8-bit accesses, writing in the second value of the sequence for a successful refresh. It must be noted that the match of value2 takes place only when the complete 16-bit value is correctly written, write4. Hence, the requirement of writing value2 of the sequence within 20 bus clock cycles of value1 is checked by measuring the gap between write2 and write4.

It is reiterated that the condition for matching values 1 and 2 of the refresh or unlock sequence remains unchanged. The difference for 8-bit accesses is that the criterion for detecting a mismatch is less strict. Any 16-bit access still needs to adhere to the original guidelines, mentioned in the sections Refreshing the Watchdog.

29.10 Restrictions on watchdog operation

This section mentions some exceptions to the watchdog operation that may not be apparent to you.

- Restriction on unlock/refresh operations—In the period between the closure of the WCT window after unlock and the actual reload of the watchdog timer, unlock and refresh operations need not be attempted.
- The update and reload of the watchdog timer happens two to three watchdog clocks after WCT window closes, following a successful configuration on unlock.
- Clock Switching Delay—The watchdog uses glitch-free multiplexers at two places – one to choose between the LPO oscillator input and alternate clock input, and the other to choose between the watchdog functional clock and fast clock input for

watchdog functional test. A maximum time period of ~2 clock A cycles plus ~2 clock B cycles elapses from the time a switch is requested to the occurrence of the actual clock switch, where clock A and B are the two input clocks to the clock mux.

- For the windowed mode, there is a two to three bus clock latency between the watchdog counter going past the window value and the same registering in the bus clock domain.
- For proper operation of the watchdog, the watchdog clock must be at least five times slower than the system bus clock at all times. An exception is when the watchdog clock is synchronous to the bus clock wherein the watchdog clock can be as fast as the bus clock.
- WCT must be equivalent to at least three watchdog clock cycles. If not ensured, this means that even after the close of the WCT window, you have to wait for the synchronized system reset to deassert in the watchdog clock domain, before expecting the configuration updates to take effect.
- The time-out value of the watchdog should be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.
- You must take care not only to refresh the watchdog within the watchdog timer's actual time-out period, but also provide enough allowance for the time it takes for the refresh sequence to be detected by the watchdog timer, on the watchdog clock.
- Updates cannot be made in the bus clock cycle immediately following the write of the unlock sequence, but one bus clock cycle later.
- It should be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.
- An attempted refresh operation, in between the two writes of the unlock sequence and in the WCT time following a successful unlock, will go undetected.
- Trying to unlock the watchdog within the WCT time after an initial unlock has no effect.
- The refresh and unlock operations and interrupt are not automatically disabled in the watchdog functional test mode.
- After emerging from a reset due to a watchdog functional test, you are still expected to go through the mandatory steps of unlocking and configuring the watchdog. The watchdog continues to be in its functional test mode and therefore you should pull the watchdog out of the functional test mode within WCT time of reset.

- After emerging from a reset due to a watchdog functional test, you still need to go through the mandatory steps of unlocking and configuring the watchdog.
- You must ensure that both the clock inputs to the glitchless clock multiplexers are alive during the switching of clocks. Failure to do so results in a loss of clock at their outputs.
- There is a gap of two to three watchdog clock cycles from the point that stop mode is entered to the watchdog timer actually pausing, due to synchronization. The same holds true for an exit from the stop mode, this time resulting in a two to three watchdog clock cycle delay in the timer restarting. In case the duration of the stop mode is less than one watchdog clock cycle, the watchdog timer is not guaranteed to pause.
- Consider the case when the first refresh value is written, following which the system enters stop mode with system bus clk still on. If the second refresh value is not written within 20 bus cycles of the first value, the system is reset, or interrupt-then-reset if enabled.

Chapter 30
External Watchdog Monitor (EWM)

30.1 Chip-specific EWM information

30.1.1 Overview

This device includes an additional watchdog: the External Watchdog Monitor (EWM) apart from the main Watchdog (WDOG), to monitor external circuits as well as the MCU software flow.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM if allowed to time-out, provides an independent output pin ($\overline{\text{EWM\_OUT}}$) that, when asserted, resets or places an external circuit into a safe mode.

30.1.2 Clock Connections

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

Table 30-1. EWM clock connections

Module clock	Chip clock
Low Power Clock	1 kHz LPO Clock
Alternate Clock	32 kHz Oscillator Clock

30.1.3 Low Power Modes

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

Table 30-2. EWM low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS
Power Down	VLLS3, VLLS2, VLLS1, VLLS0

30.2 Introduction

The watchdog is generally used to monitor the flow and execution of embedded software within an MCU. The watchdog consists of a counter that if allowed to overflow, forces an internal reset (asynchronous) to all on-chip peripherals and optionally assert the $\overline{\text{RESET}}$ pin to reset external devices/circuits. The overflow of the watchdog counter must not occur if the software code works well and services the watchdog to re-start the actual counter.

For safety, a redundant watchdog system, External Watchdog Monitor (EWM), is designed to monitor external circuits, as well as the MCU software flow. This provides a back-up mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM if allowed to time-out, provides an independent $\overline{\text{EWM\_out}}$ pin that when asserted resets or places an external circuit into a safe mode. The CPU resets the EWM counter that is logically ANDed with an external digital input pin. This pin allows an external circuit to influence the reset_out signal.

30.2.1 Features

Features of EWM module include:

- Independent LPO clock source
- Programmable time-out period specified in terms of number of EWM LPO clock cycles.
- Windowed refresh option
  - Provides robust check that program flow is faster than expected.
  - Programmable window.
  - Refresh outside window leads to assertion of $\overline{\text{EWM\_out}}$.
- Robust refresh mechanism

- Write values of 0xB4 and 0x2C to EWM Refresh Register within 15 (*EWM_service_time*) peripheral bus clock cycles.

- One output port, $\overline{\text{EWM\_out}}$, when asserted is used to reset or place the external circuit into safe mode.
- One Input port, EWM_in, allows an external circuit to control the $\overline{\text{EWM\_out}}$ signal.

30.2.2 Modes of Operation

This section describes the module's operating modes.

30.2.2.1 Stop Mode

When the EWM is in stop mode, the CPU services to the EWM cannot occur. On entry to stop mode, the EWM's counter freezes.

There are two possible ways to exit from Stop mode:

- On exit from stop mode through a reset, the EWM remains disabled.
- On exit from stop mode by an interrupt, the EWM is re-enabled, and the counter continues to be clocked from the same value prior to entry to stop mode.

Note the following if the EWM enters the stop mode during CPU service mechanism: At the exit from stop mode by an interrupt, refresh mechanism state machine starts from the previous state which means, if first service command is written correctly and EWM enters the stop mode immediately, the next command has to be written within the next 15 (*EWM_service_time*) peripheral bus clocks after exiting from stop mode. User must mask all interrupts prior to executing EWM service instructions.

30.2.2.2 Wait Mode

The EWM module treats the stop and wait modes as the same. EWM functionality remains the same in both of these modes.

30.2.2.3 Debug Mode

Entry to debug mode has no effect on the EWM.

- If the EWM is enabled prior to entry of debug mode, it remains enabled.
- If the EWM is disabled prior to entry of debug mode, it remains disabled.

30.2.3 Block Diagram

This figure shows the EWM block diagram.

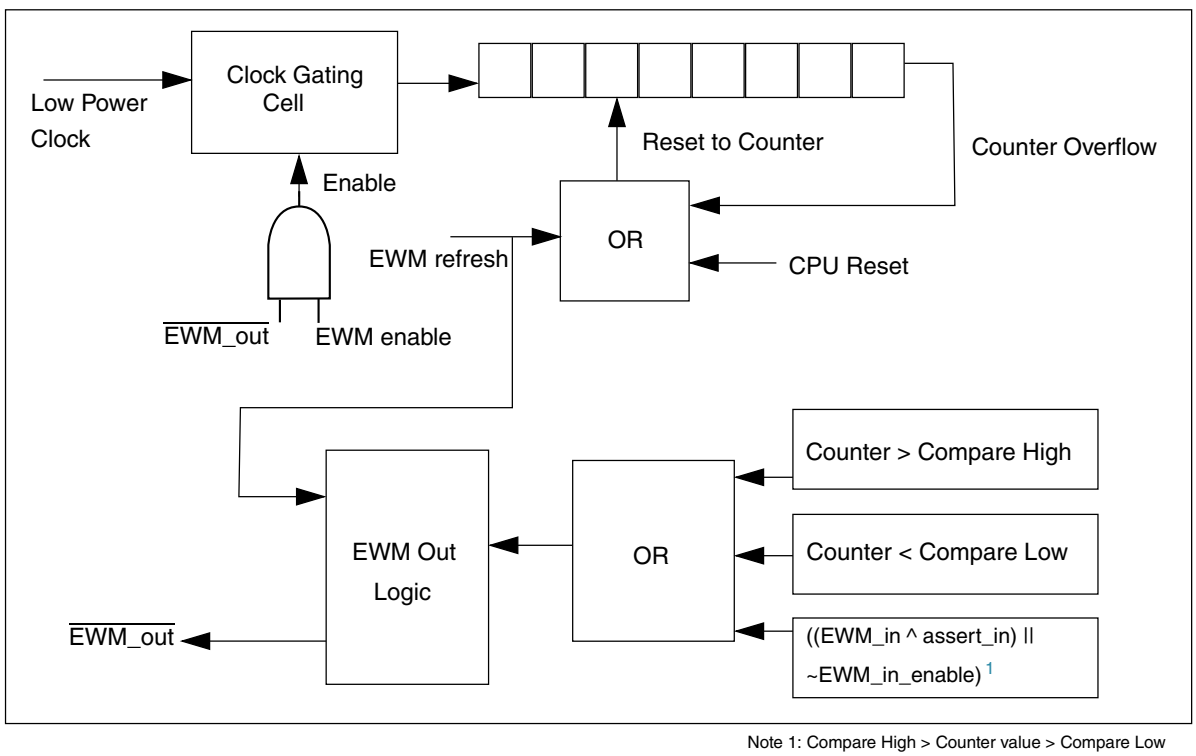


Figure 30-1. EWM Block Diagram

30.3 EWM Signal Descriptions

The EWM has two external signals and internal options for the counter clock sources, as shown in the following table.

Table 30-3. EWM Signal Descriptions

Signal	Description	I/O
EWM_in	EWM input for safety status of external safety circuits. The polarity of EWM_in is programmable using the EWM_CTRL[ASSIN] bit. The default polarity is active-low.	I
$\overline{\text{EWM\_out}}$	EWM reset out signal	O
lpo_clk[3:0]	Low power clock sources for running counter	I

30.4 Memory Map/Register Definition

This section contains the module memory map and registers.

EWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_1000	Control Register (EWM_CTRL)	8	R/W	00h	30.4.1/527
4006_1001	Service Register (EWM_SERV)	8	W (always reads 0)	00h	30.4.2/528
4006_1002	Compare Low Register (EWM_CMPL)	8	R/W	00h	30.4.3/528
4006_1003	Compare High Register (EWM_CMPH)	8	R/W	FFh	30.4.4/529
4006_1004	Clock Control Register (EWM_CLKCTRL)	8	R/W	00h	30.4.5/529
4006_1005	Clock Prescaler Register (EWM_CLKPRESCALER)	8	R/W	00h	30.4.6/530

30.4.1 Control Register (EWM_CTRL)

The CTRL register is cleared by any reset.

NOTE

INEN, ASSIN and EWMEN bits can be written once after a CPU reset. Modifying these bits more than once, generates a bus transfer error.

Address: 4006_1000h base + 0h offset = 4006_1000h

Bit	7	6	5	4	3	2	1	0
Read	0				INTEN	INEN	ASSIN	EWMEN
Write								
Reset	0	0	0	0	0	0	0	0

EWM_CTRL field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 INTEN	Interrupt Enable. This bit when set and $\overline{\text{EWM\_out}}$ is asserted, an interrupt request is generated. To de-assert interrupt request, user should clear this bit by writing 0.
2 INEN	Input Enable. This bit when set, enables the EWM_in port.

Table continues on the next page...

EWM_CTRL field descriptions (continued)

Field	Description
1 ASSIN	EWM_in's Assertion State Select. Default assert state of the EWM_in signal is logic zero. Setting ASSIN bit inverts the assert state to a logic one.
0 EWMEN	EWM enable. This bit when set, enables the EWM module. This resets the EWM counter to zero and deasserts the $\overline{\text{EWM\_out}}$ signal. Clearing EWMEN bit disables the EWM, and therefore it cannot be enabled until a reset occurs, due to the write-once nature of this bit.

30.4.2 Service Register (EWM_SERV)

The SERV register provides the interface from the CPU to the EWM module. It is write-only and reads of this register return zero.

Address: 4006_1000h base + 1h offset = 4006_1001h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write	SERVICE							
Reset	0	0	0	0	0	0	0	0

EWM_SERV field descriptions

Field	Description
SERVICE	The EWM service mechanism requires the CPU to write two values to the SERV register: a first data byte of 0xB4, followed by a second data byte of 0x2C. The EWM service is illegal if either of the following conditions is true. • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte. This fixed number of cycles is called *EWM_service_time*.

30.4.3 Compare Low Register (EWM_CMPL)

The CMPL register is reset to zero after a CPU reset. This provides no minimum time for the CPU to service the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

Address: 4006_1000h base + 2h offset = 4006_1002h

Bit	7	6	5	4	3	2	1	0
Read Write	COMPAREL							
Reset	0	0	0	0	0	0	0	0

EWM_CMPL field descriptions

Field	Description
COMPAREL	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) minimum service time is required.

30.4.4 Compare High Register (EWM_CMPH)

The CMPH register is reset to 0xFF after a CPU reset. This provides a maximum of 256 clocks time, for the CPU to service the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

The valid values for CMPH are up to 0xFE because the EWM counter never expires when CMPH = 0xFF. The expiration happens only if EWM counter is greater than CMPH.

Address: 4006_1000h base + 3h offset = 4006_1003h

Bit	7	6	5	4	3	2	1	0
Read Write	COMPAREH							
Reset	1	1	1	1	1	1	1	1

EWM_CMPH field descriptions

Field	Description
COMPAREH	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) maximum service time is required.

30.4.5 Clock Control Register (EWM_CLKCTRL)

This CLKCTRL register is reset to 0x00 after a CPU reset.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

User should select the required low power clock before enabling the EWM.

Address: 4006_1000h base + 4h offset = 4006_1004h

Bit	7	6	5	4	3	2	1	0
Read	0						CLKSEL	
Write								
Reset	0	0	0	0	0	0	0	0

EWM_CLKCTRL field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKSEL	EWM has 4 possible low power clock sources for running EWM counter. One of the clock source can be selected by writing into this field. • 00 - lpo_clk[0] will be selected for running EWM counter. • 01 - lpo_clk[1] will be selected for running EWM counter. • 10 - lpo_clk[2] will be selected for running EWM counter. • 11 - lpo_clk[3] will be selected for running EWM counter.

30.4.6 Clock Prescaler Register (EWM_CLKPRESCALER)

This CLKPRESCALER register is reset to 0x00 after a CPU reset.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

Write the required prescaler value before enabling the EWM.

NOTE

The implementation of this register is chip-specific. See the Chip Configuration details.

Address: 4006_1000h base + 5h offset = 4006_1005h

Bit	7	6	5	4	3	2	1	0
Read Write	CLK_DIV							
Reset	0	0	0	0	0	0	0	0

EWM_CLKPRESCALER field descriptions

Field	Description
CLK_DIV	Selected low power clock source for running the EWM counter can be prescaled as below. • Prescaled clock frequency = low power clock source frequency / (1 + CLK_DIV)

30.5 Functional Description

The following sections describe functional details of the EWM module.

30.5.1 The $\overline{\text{EWM\_out}}$ Signal

The $\overline{\text{EWM\_out}}$ is a digital output signal used to gate an external circuit (application specific) that controls critical safety functions. For example, the $\overline{\text{EWM\_out}}$ could be connected to the high voltage transistors circuits that control an AC motor in a large appliance.

The $\overline{\text{EWM\_out}}$ signal remains deasserted when the EWM is being regularly serviced by the CPU within the programmable service window, indicating that the application code is executed as expected.

The $\overline{\text{EWM\_out}}$ signal is asserted in any of the following conditions:

- Servicing the EWM when the counter value is less than CMPL value.
- If the EWM counter value reaches the CMPH value, and no EWM service has occurred.
- Servicing the EWM when the counter value is more than CMPL and less than CMPH values and EWM_in signal is asserted.
- If functionality of EWM_in pin is enabled and EWM_in pin is asserted while servicing the EWM.
- After any reset (by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ pin)

On a normal reset, the $\overline{\text{EWM\_out}}$ is asserted. To deassert the $\overline{\text{EWM\_out}}$, set EWMEN bit in the CTRL register to enable the EWM.

If the $\overline{\text{EWM\_out}}$ signal shares its pad with a digital I/O pin, on reset this actual pad defers to being an input signal. It takes the $\overline{\text{EWM\_out}}$ output condition only after you enable the EWM by the EWMEN bit in the CTRL register.

When the $\overline{\text{EWM\_out}}$ pin is asserted, it can only be deasserted by forcing a MCU reset.

Note

$\overline{\text{EWM\_out}}$ pad must be in pull down state when EWM functionality is used and when EWM is under Reset.

30.5.2 The EWM_in Signal

The EWM_in is a digital input signal that allows an external circuit to control the $\overline{\text{EWM\_out}}$ signal. For example, in the application, an external circuit monitors a critical safety function, and if there is fault with this circuit's behavior, it can then actively initiate the $\overline{\text{EWM\_out}}$ signal that controls the gating circuit.

The EWM_in signal is ignored if the EWM is disabled, or if INEN bit of CTRL register is cleared, as after any reset.

On enabling the EWM (setting the CTRL[EWMEN] bit) and enabling EWM_in functionality (setting the CTRL[INEN] bit), the EWM_in signal must be in the deasserted state prior to the CPU servicing the EWM. This ensures that the $\overline{\text{EWM\_out}}$ stays in the deasserted state; otherwise, the $\overline{\text{EWM\_out}}$ pin is asserted.

Note

You must update the CMPH and CMPL registers prior to enabling the EWM. After enabling the EWM, the counter resets to zero, therefore providing a reasonable time after a power-on reset for the external monitoring circuit to stabilize and ensure that the EWM_in pin is deasserted.

30.5.3 EWM Counter

It is an 8-bit ripple counter fed from a clock source that is independent of the peripheral bus clock source. As the preferred time-out is between 1 ms and 100 ms the actual clock source should be in the kHz range.

The counter is reset to zero, after a CPU reset, or a EWM refresh cycle. The counter value is not accessible to the CPU.

30.5.4 EWM Compare Registers

The compare registers CMPL and CMPH are write-once after a CPU reset and cannot be modified until another CPU reset occurs.

The EWM compare registers are used to create a service window, which is used by the CPU to service/refresh the EWM module.

- If the CPU services the EWM when the counter value lies between CMPL value and CMPH value, the counter is reset to zero. This is a legal service operation.
- If the CPU executes a EWM service/refresh action outside the legal service window, $\overline{\text{EWM\_out}}$ is asserted.

It is illegal to program CMPL and CMPH with same value. In this case, as soon as counter reaches (CMPL + 1), $\overline{\text{EWM\_out}}$ is asserted.

30.5.5 EWM Refresh Mechanism

Other than the initial configuration of the EWM, the CPU can only access the EWM by the EWM Service Register. The CPU must access the EWM service register with correct write of unique data within the windowed time frame as determined by the CMPL and CMPH registers. Therefore, three possible conditions can occur:

Table 30-4. EWM Refresh Mechanisms

Condition	Mechanism
A unique EWM service occurs when CMPL < Counter < CMPH.	The software behaves as expected and the counter of the EWM is reset to zero, and $\overline{\text{EWM\_out}}$ pin remains in the deasserted state. **Note:** EWM_in pin is also assumed to be in the deasserted state.
A unique EWM service occurs when Counter < CMPL	The software services the EWM and therefore resets the counter to zero and asserts the $\overline{\text{EWM\_out}}$ pin (irrespective of the EWM_in pin). The $\overline{\text{EWM\_out}}$ pin is expected to gate critical safety circuits.
Counter value reaches CMPH prior to a unique EWM service	The counter value reaches the CMPH value and no service of the EWM resets the counter to zero and assert the $\overline{\text{EWM\_out}}$ pin (irrespective of the EWM_in pin). The $\overline{\text{EWM\_out}}$ pin is expected to gate critical safety circuits.

Any illegal service on EWM has no effect on $\overline{\text{EWM\_out}}$.

30.5.6 EWM Interrupt

When $\overline{\text{EWM\_out}}$ is asserted, an interrupt request is generated to indicate the assertion of the EWM reset out signal. This interrupt is enabled when CTRL[INTEN] is set. Clearing this bit clears the interrupt request but does not affect $\overline{\text{EWM\_out}}$. The $\overline{\text{EWM\_out}}$ signal can be deasserted only by forcing a system reset.

30.5.7 Selecting the EWM counter clock

There are four possible low power clock sources for the EWM counter. Select one of the available clock sources by programming CLKCTRL[CLKSEL].

30.5.8 Counter clock prescaler

The EWM counter clock source can be prescaled by a clock divider, by programming CLKPRESCALER[CLK_DIV]. This divided clock is used to run the EWM counter.

NOTE

The divided clock used to run the EWM counter must be no more than half the frequency of the bus clock.

Chapter 31
Analog Front End (AFE)

31.1 Chip-specific AFE information

31.1.1 Overview

The Analog Front End or AFE is an integrated module that is comprised of ΣΔ ADCs, PGA, filtering and phase compensation blocks. The AFE is responsible for measuring the phase voltage, phase current and neutral current.

31.1.2 Clock Sources

AFE can be clocked by the following clock sources (see the figure below):

- 6.144 MHz clock generated from PLL with 32.768 kHz reference clock input
- MHz crystal oscillator clock output (if crystal clock < 32 MHz). Prescaler on AFE will divide this external clock to generate required range of clock at modulator input (30 kHz to 6.5 MHz).
- Externally supplied clock (30 kHz to 50 MHz) via AFE_CLK pin
  - This pin can additionally be used to interface clock port of external modulators (which provide clock output with data).
  - Prescaler on AFE will divide this external clock to generate required range of clock at modulator input (30 kHz to 6.5 MHz).
- Quad Timer channel 0 output
- 4 MHz IRC from MCG

The above clock sources are available in all MCU power modes where AFE can work, except for VLPS where PLL cannot work. Hence for operating AFE in VLPS, a different clock source would be required.

The selected clock source is prescaled down to the AFE modulator clock operating range (30 kHz to 6.5 MHz in Normal mode, and less than 1.6 MHz for Low Power mode) before enabling AFE operations. The Prescaler divides down the selected clock in powers of 2 (that is 1, 2, 4, 8, 16, 32, 64, 128 and 256). It is important to divide the selected clock by at least 2 to compensate for any duty cycle variations during muxing and clock propagation.

NOTE

- The externally supplied clock is routed directly from the AFE_CLK pad to the AFE prescaler.
- The clock source selection is one time selection and cannot be changed on the fly. AFE needs to be disabled to change the clock and prescaler.
- Using any clock source other than the PLL output clock or oscillator clock will not guarantee AFE performance.
- Refer to the Clocking chapter for details on generation of MCGPLLCLK and muxing options.
- All AFE channels should run on the same clock. The clock selection is done via register bits in AFE and is common to all AFE channels.

The following figure shows the clocking strategy for the AFE. By configuring the SIM_CTRL_REG[AFEOUTCLKSEL]bit, user can select the AFE clock before or prescaler and output this clock to AFE_CLK pin and XBAR.

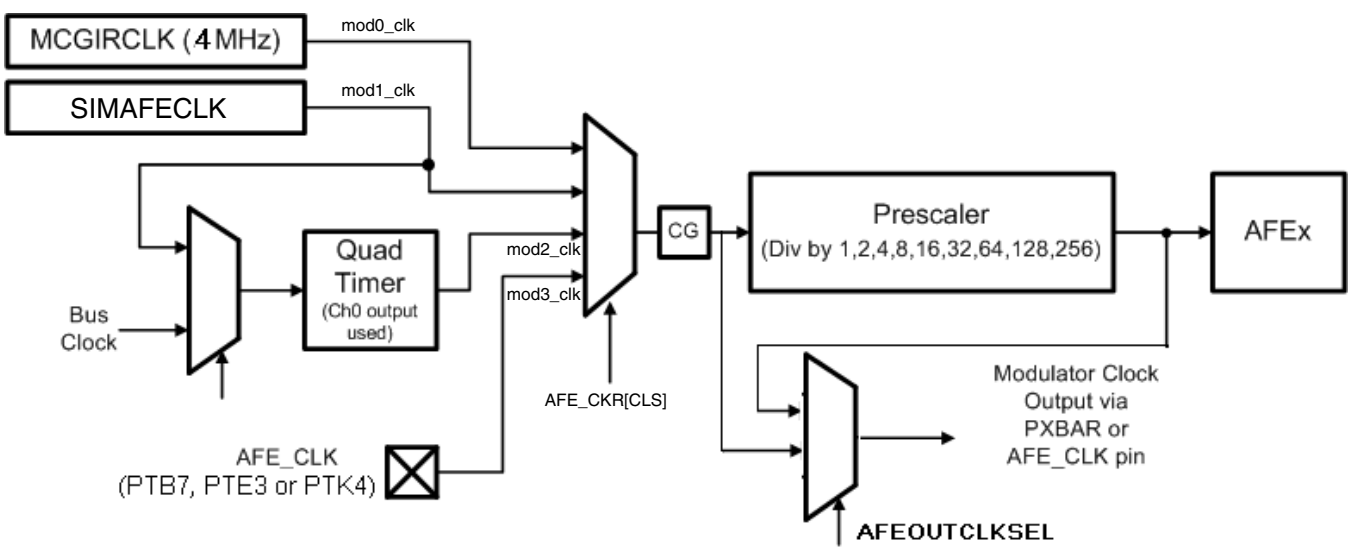


Figure 31-1. AFE Clocking Options

The selected clock is common to all AFE channels (0-3) even if some of the channels use an external modulator. When AFE is disabled, the clock sources should be gated or disabled to save power.

31.2 Introduction

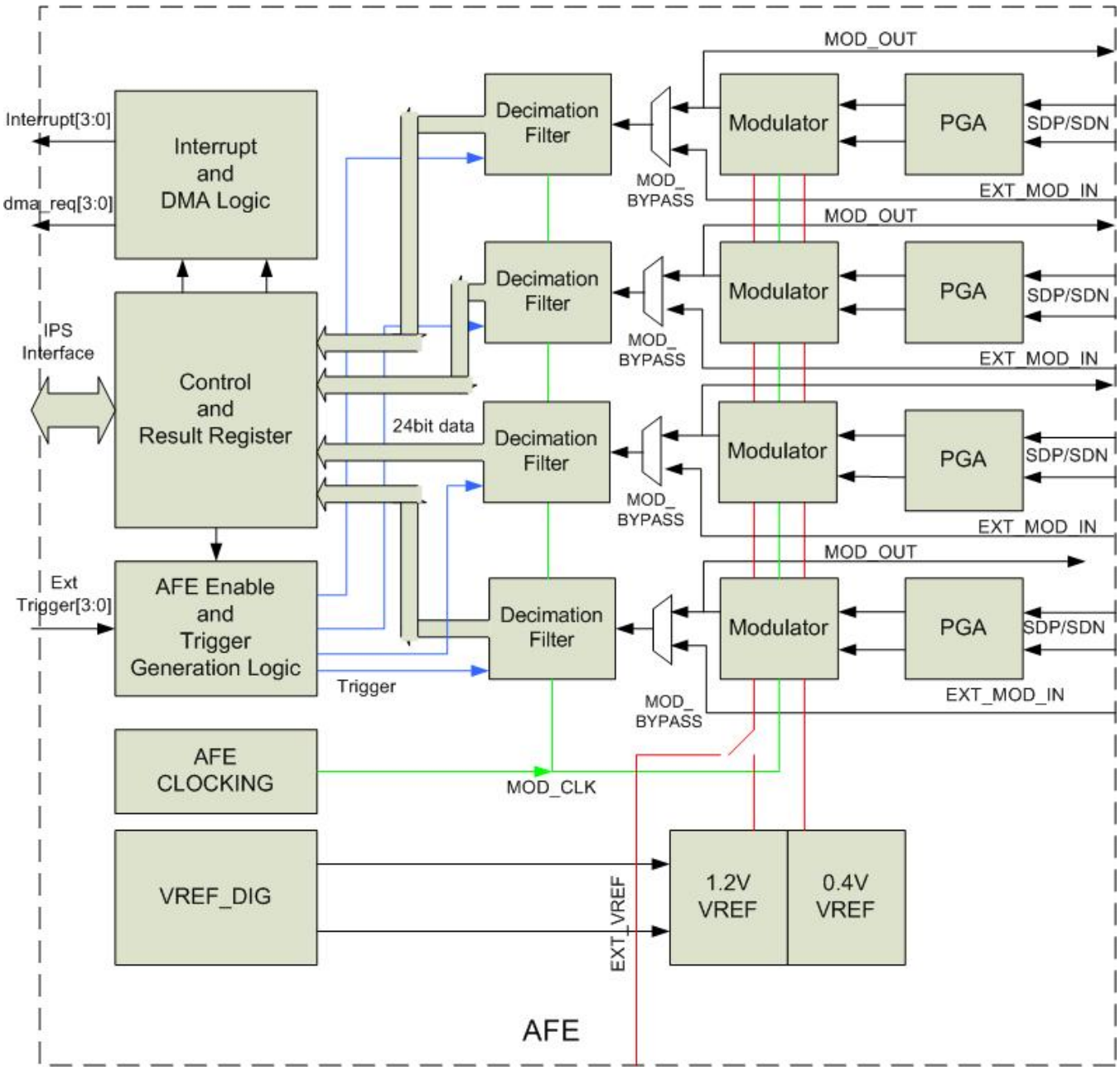
The Analog Front End or AFE is an integrated module that is comprised of four sigma-delta ADCs (all with PGA) referred to as channels and input voltage reference. The converters are based on second-order oversampling sigma-delta modulators and digital decimation filters with selectable over sampling ratio up to 2048. Additional filtering can be done in software.

31.3 Features

- Four 24-bit (after averaging) sigma-delta ADC with PGA
- Option to bypass/disable the PGA
- PGA with 7μV sensitivity (PGA gain = 1 V/V), 7 μV sensitivity (PGA gain = 32 V/V)
- Voltage measurement can operate in both differential and single ended mode. For single ended operation, the differential pin shall be grounded externally on board.
- Input common-mode voltage range of 0 to 0.8V
- Full Operating Voltage Range – 2.7V to 3.3V
- Output sampling rates 3 KHz, 6 KHz, 12 KHz, 24 KHz, 48 KHz, and 96 KHz.
- Software selectable analog gain 1x, 2x, 4x, 8x, 16x, 32x
- Software selectable on-chip reference voltage and common mode voltage generation (1.2 V)
- Software selectable internal or external reference
- ±250 mVp (1 Vpp differential, 0.5 Vpp single ended) input range for all AFE inputs
- Support for interfacing to external modulators when AFE bypass option is enabled.
- Digital logic enables synchronized start of all AFE blocks
- Interrupt and DMA request on Conversion Complete.
- Asynchronous DMA request generation to transfer data in deep sleep(STOP) mode.
- Both Hardware and software trigger can be used to initiate data conversion.
- 30kHz to 6.5 MHz Modulator input frequency.
- Selectable low-power conversion mode.
- Implements phase compensation Upto 0-6.0 degree @(OSR = 2048 & 50 Hz input)
- Allows dynamic change of phase shift without interrupting sampling cycles
- Software selectable clock option for modulator clock.

31.4 Block Diagram

The figure below shows the block diagram of AFE.



NOTE

Decimators doesn't get mod_clk directly. In Modulator engage mode the clock is provided by corresponding modulator which is properly aligned with the modulator output data. In bypass

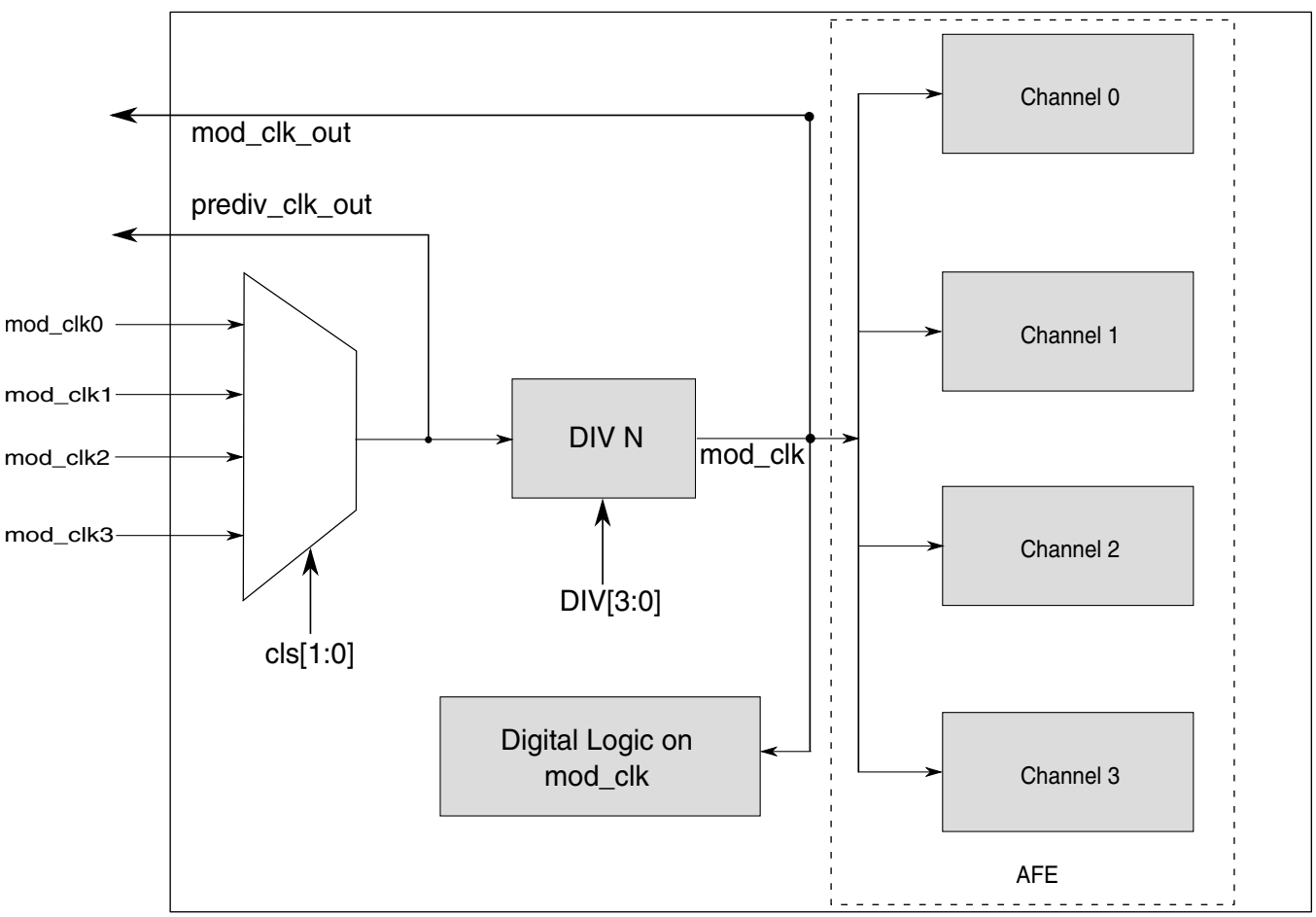
mode the clock can come from external interface too.Please see AFE BYPASS mode diagram for detailed clocking.

31.5 AFE Clocking

The maximum operational frequency of the AFE modulator clock is 6.5 MHz. For optimum performance the clock should have minimum jitter with 50% (approx) duty cycle. Four possible clock sources (selected by programming CLS[1:0]) are provided to be used as MOD_CLK.A clock divider is in place to further divide down the MOD_CLK frequency for low power application. The AFE_CKR[DIV] field selects the divide ratio to generate final mod_clk to AFE.

Fmod_clk = Fclk / 2^ DIV where Fclk is the input clock to divider. The modulator clock frequency must be in the range of 30KHz -6.5 MHz. The clock divide ratio has a range from 1 - 256

Below digram depicts the AFE clocking implementation.



NOTE

Refer to "AFE Chip-specific configuration" section for more information on Clock Sources.

31.6 OSR Select

The oversampling ratio (OSR) is the ratio of the modulator clock frequency (fSDMCK) to the output sample frequency. For example: a 6.144 MHz modulator clock with an OSR = 256 yields an output sample rate of 24 kHz. The OSR field selects the OSR which can be 64, 128, 256, 512, 1024, or 2048. A higher OSR produces a lower sample rate and narrower bandwidth.

31.7 Analog Gain Select

The AGN field selects the front end analog gain applied to the input signal. The selectable gains are x1, x2, x4, x8, x16, x32. Higher gain settings reduce the full scale input of the converter.

31.8 Memory Map and Register Definition

AFE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_0000	Channel0 Configuration Register (AFE_CH0_CFR)	32	R/W	0008_0000h	31.8.1/541
4003_0004	Channel1 Configuration Register (AFE_CH1_CFR)	32	R/W	0008_0000h	31.8.2/544
4003_0008	Channel2 Configuration Register (AFE_CH2_CFR)	32	R/W	0008_0000h	31.8.3/546
4003_000C	Channel3 Configuration Register (AFE_CH3_CFR)	32	R/W	0008_0000h	31.8.4/548
4003_0018	Control Register (AFE_CR)	32	R/W	0040_FA00h	31.8.5/551
4003_001C	Clock Configuration Register (AFE_CKR)	32	R/W	1000_0000h	31.8.6/553
4003_0020	DMA and Interrupt Register (AFE_DI)	32	R/W	0000_0000h	31.8.7/554
4003_002C	Channel0 Delay Register (AFE_CH0_DR)	32	R/W	0000_0000h	31.8.8/555
4003_0030	Channel1 Delay Register (AFE_CH1_DR)	32	R/W	0000_0000h	31.8.9/555
4003_0034	Channel2 Delay Register (AFE_CH2_DR)	32	R/W	0000_0000h	31.8.10/ 556
4003_0038	Channel3 Delay Register (AFE_CH3_DR)	32	R/W	0000_0000h	31.8.11/ 556

Table continues on the next page...

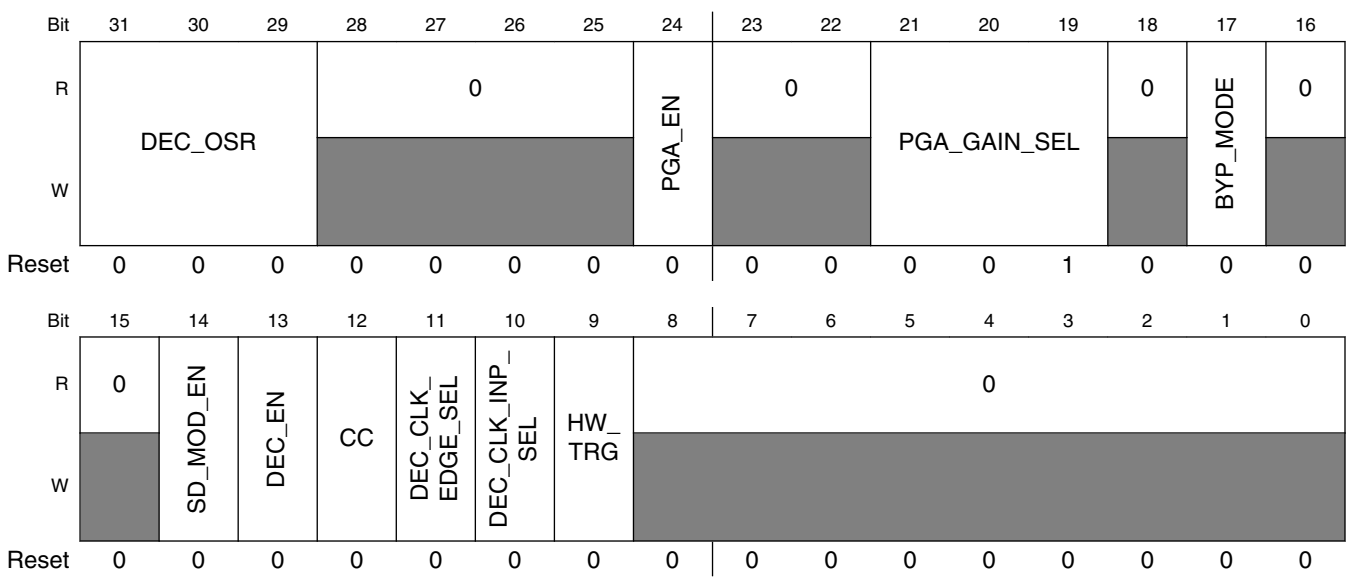
AFE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_0044	Channel0 Result Register (AFE_CH0_RR)	32	R	0000_0000h	31.8.12/ 557
4003_0048	Channel1 Result Register (AFE_CH1_RR)	32	R	0000_0000h	31.8.13/ 557
4003_004C	Channel2 Result Register (AFE_CH2_RR)	32	R	0000_0000h	31.8.14/ 558
4003_0050	Channel3 Result Register (AFE_CH3_RR)	32	R	0000_0000h	31.8.15/ 558
4003_005C	Status Register (AFE_SR)	32	R	0000_0000h	31.8.16/ 560

31.8.1 Channel0 Configuration Register (AFE_CH0_CFR)

Channel0 Configuration Register has various fields to configure channel0 of AFE. Any changes done in this register while the conversion is in progress will come into effect from next conversion only.

Address: 4003_0000h base + 0h offset = 4003_0000h



AFE_CH0_CFR field descriptions

Field	Description
31–29 DEC_OSR	Decimator OverSampling Ratio select This field selects the oversampling ratio. It should not be modified while the filter is operating. The Sinc filter output is scaled based on the OSR to produce a 24 bit signed, two's complement output.. 000 64

Table continues on the next page...

AFE_CH0_CFR field descriptions (continued)

Field	Description
	001 128 010 256 011 512 100 1024 101 2048 110-111 reserved
28–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 PGA_EN	PGA enable It enables the PGA gain stage. 0 PGA disabled 1 PGA enabled
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–19 PGA_GAIN_SEL	PGA Gain Select This field selects the analog gain applied to the input signal. 000 reserved 001 1x (default) 010 2x 011 4x 100 8x 101 16x 110 32x 111 reserved
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 BYP_MODE	AFE Channel0 bypass mode This field is used to disable internal modulators of channel0. External modulators will be used. 0 Normal mode 1 Bypass mode where ADC and PGA of channel0 are disabled.
16–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SD_MOD_EN	Sigma Delta Modulator enable This is used to enable SD ADC1 . 0 SD ADC1 is disabled 1 SD ADC1 is enabled
13 DEC_EN	Decimation Filter enable This is used to enable decimation filter.

Table continues on the next page...

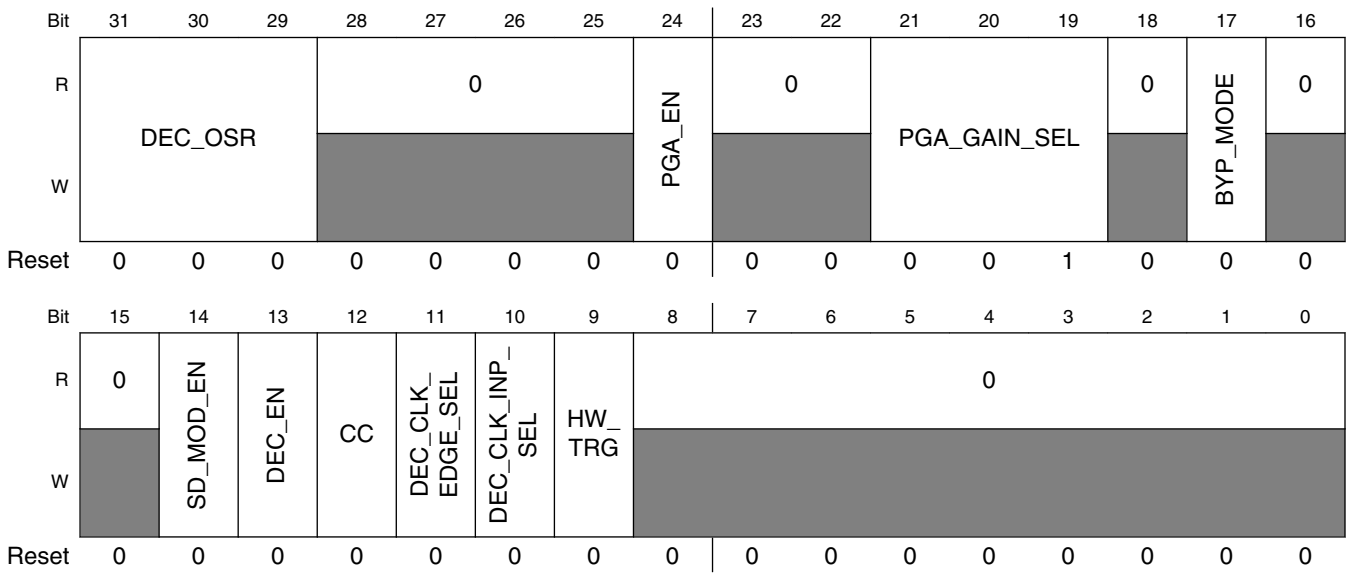
AFE_CH0_CFR field descriptions (continued)

Field	Description
	0 Decimation filter is disabled. 1 Decimation filter is enabled.
12 CC	Continuous Conversion/Single Conversion Mode Select 0 One conversion following a triggering event 1 Continuous conversions following a triggering event.
11 DEC_CLK_ EDGE_SEL	Decimator Clock Edge Select This field is used to select rising edge or falling edge for registering input data by the decimation filter in case of by-pass mode. 0 Posedge will be used. 1 Negedge will be used.
10 DEC_CLK_INP_ SEL	Decimator Clock Input Select This field is used to select on the chip modulator clock or an external clock for the decimator in case of bypass mode. 0 On the chip modulator clock will be used 1 External clock will be used.
9 HW_TRG	Hardware Trigger Select This field is used to select trigger source for conversion. 0 Software trigger select 1 Hardware trigger select
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.2 Channel1 Configuration Register (AFE_CH1_CFR)

Channel1 Configuration Register has various fields to configure channel1 of AFE. Any changes done in this register while the conversion is in progress will come into effect from next conversion only.

Address: 4003_0000h base + 4h offset = 4003_0004h



AFE_CH1_CFR field descriptions

Field	Description
31–29 DEC_OSR	Decimator OverSampling Ratio select This field selects the oversampling ratio. It should not be modified while the filter is operating. The Sinc filter output is scaled based on the OSR to produce a 24 bit signed, two's complement output.. 000 64 001 128 010 256 011 512 100 1024 101 2048 110-111 reserved
28–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 PGA_EN	PGA enable It enables the PGA gain stage. 0 PGA disabled 1 PGA enabled

Table continues on the next page...

AFE_CH1_CFR field descriptions (continued)

Field	Description
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–19 PGA_GAIN_SEL	PGA Gain Select This field selects the analog gain applied to the input signal. 000 reserved 001 1x (default) 010 2x 011 4x 100 8x 101 16x 110 32x 111 reserved
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 BYP_MODE	AFE Channel1 bypass mode This field is used to disable internal modulators of channel1. External modulators will be used. 0 Normal mode 1 Bypass mode where ADC and PGA of channel1 are disabled.
16–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SD_MOD_EN	Sigma Delta Modulator enable This is used to enable SD ADC1 . 0 SD ADC1 is disabled 1 SD ADC1 is enabled
13 DEC_EN	Decimation Filter enable This is used to enable decimation filter. 0 Decimation filter is disabled. 1 Decimation filter is enabled.
12 CC	Continuous Conversion/Single Conversion Mode Select 0 One conversion following a triggering event 1 Continuous conversions following a triggering event.
11 DEC_CLK_ EDGE_SEL	Decimator Clock Edge Select This field is used to select rising edge or falling edge for registering input data by the decimation filter in case of by-pass mode. 0 Posedge will be used. 1 Negedge will be used.
10 DEC_CLK_INP_ SEL	Decimator Clock Input Select

Table continues on the next page...

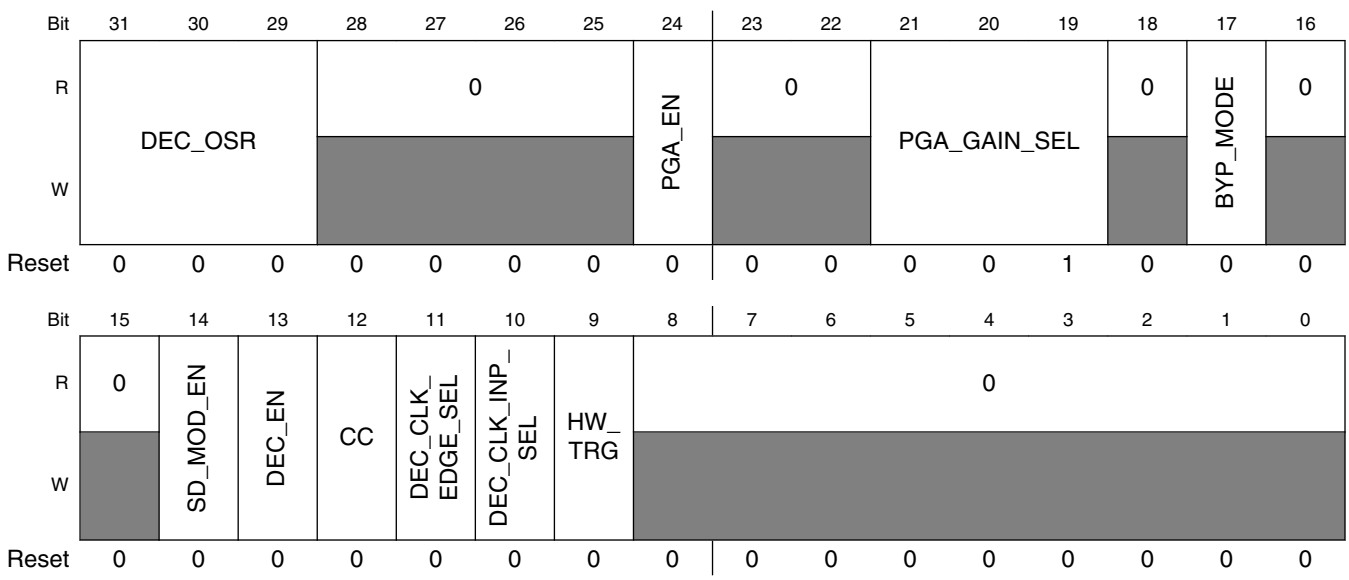
AFE_CH1_CFR field descriptions (continued)

Field	Description
	This field is used to select on the chip modulator clock or an external clock for the decimator in case of bypass mode. 0 On the chip modulator clock will be used 1 External clock will be used.
9 HW_TRG	Hardware Trigger Select This field is used to select trigger source for conversion. 0 Software trigger select 1 Hardware trigger select
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.3 Channel2 Configuration Register (AFE_CH2_CFR)

Channel2 Configuration Register has various fields to configure Channel2 of AFE. Any changes done in this register while the conversion is in progress will come into effect from next conversion only.

Address: 4003_0000h base + 8h offset = 4003_0008h



AFE_CH2_CFR field descriptions

Field	Description
31–29 DEC_OSR	Decimator OverSampling Ratio select This field selects the oversampling ratio. It should not be modified while the filter is operating. The Sinc filter output is scaled based on the OSR to produce a 24 bit signed, two's complement output.

Table continues on the next page...

AFE_CH2_CFR field descriptions (continued)

Field	Description
	000 64 001 128 010 256 011 512 100 1024 101 2048 110-111 reserved
28–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 PGA_EN	It enables the PGA gain stage.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–19 PGA_GAIN_SEL	This field selects the analog gain applied to the input signal. 000 Reserved 001 1x(default) 010 2x 011 4x 100 8x 101 16x 110 32x 111 Reserved
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 BYP_MODE	AFE Channel2 bypass mode This field is used to disable internal modulators of channel2. External modulators will be used. 0 Normal mode 1 Bypass mode where ADC and PGA of channel2 are disabled.
16–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SD_MOD_EN	Sigma Delta Modulator enable This is used to enable SD ADC3. 0 SD ADC3 is disabled 1 SD ADC3 is enabled
13 DEC_EN	Decimation Filter enable This is used to enable decimation filter. 0 Decimation filter is disabled. 1 Decimation filter is enabled.
12 CC	Continuous Conversion/Single Conversion Mode Select 0 One conversion following a triggering event 1 Continuous conversions following a triggering event.

Table continues on the next page...

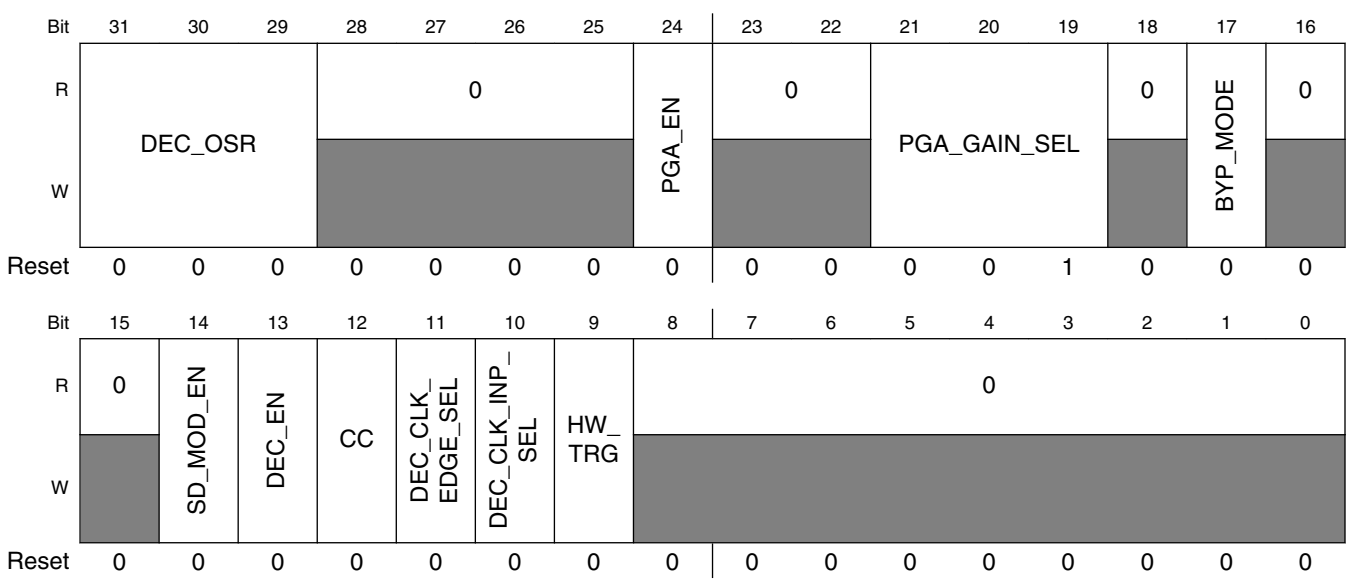
AFE_CH2_CFR field descriptions (continued)

Field	Description
11 DEC_CLK_ EDGE_SEL	Decimator Clock Edge Select This field is used to select rising edge or falling edge for registering input data by the decimation filter in case of by-pass mode. 0 Posedge will be used. 1 Negedge will be used.
10 DEC_CLK_INP_ SEL	Decimator Clock Input Select This field is used to select on the chip modulator clock or an external clock for the decimator in case of bypass mode. 0 On the chip modulator clock will be used. 1 External clock will be used.
9 HW_TRG	Hardware Trigger Select This field is used to select trigger source for conversion. 0 Software trigger select 1 Hardware trigger select
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.4 Channel3 Configuration Register (AFE_CH3_CFR)

Channel3 Configuration Register has various fields to configure Channel3 of AFE. Any changes done in this register while the conversion is in progress will come into effect from next conversion only.

Address: 4003_0000h base + Ch offset = 4003_000Ch



AFE_CH3_CFR field descriptions

Field	Description
31–29 DEC_OSR	Decimator OverSampling Ratio select This field selects the oversampling ratio. It should not be modified while the filter is operating. The Sinc filter output is scaled based on the OSR to produce a 24 bit signed, two's complement output. 000 64 001 128 010 256 011 512 100 1024 101 2048 110-111 reserved
28–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 PGA_EN	It enables the PGA gain stage.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–19 PGA_GAIN_SEL	This field selects the analog gain applied to the input signal. 000 Reserved 001 1x(default) 010 2x 011 4x 100 8x 101 16x 110 32x 111 Reserved
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 BYP_MODE	AFE Channel3 bypass mode This field is used to disable internal modulators of channel3. External modulators will be used. 0 Normal mode 1 Bypass mode where ADC and PGA of channel3 are disabled.
16–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SD_MOD_EN	Sigma Delta Modulator enable This is used to enable SD ADC3. 0 SD ADC3 is disabled 1 SD ADC3 is enabled
13 DEC_EN	Decimation Filter enable This is used to enable decimation filter.

Table continues on the next page...

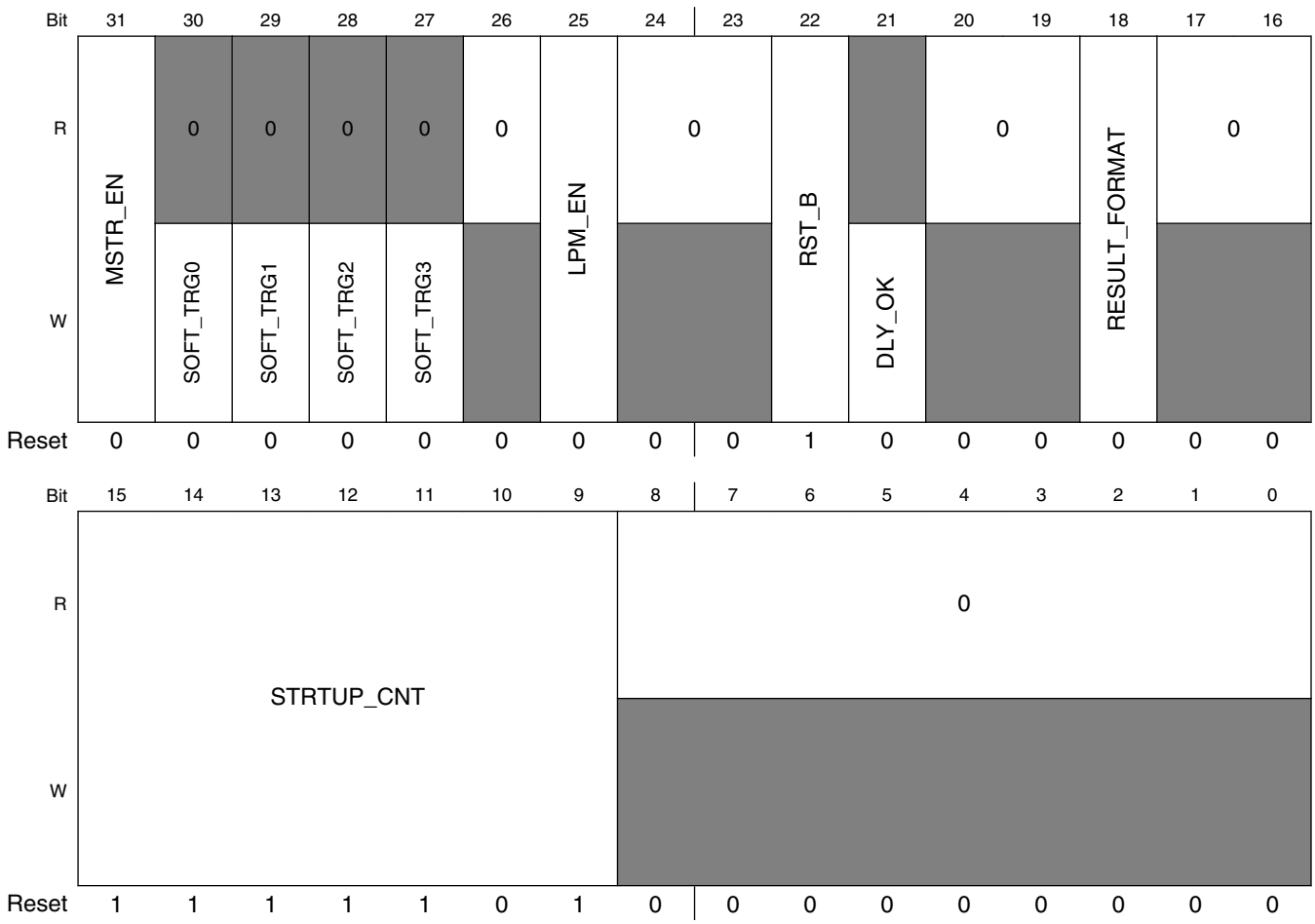
AFE_CH3_CFR field descriptions (continued)

Field	Description
	0 Decimation filter is disabled. 1 Decimation filter is enabled.
12 CC	Continuous Conversion/Single Conversion Mode Select 0 One conversion following a triggering event 1 Continuous conversions following a triggering event.
11 DEC_CLK_ EDGE_SEL	Decimator Clock Edge Select This field is used to select rising edge or falling edge for registering input data by the decimation filter in case of by-pass mode. 0 Posedge will be used. 1 Negedge will be used.
10 DEC_CLK_INP_ SEL	Decimator Clock Input Select This field is used to select on the chip modulator clock or an external clock for the decimator in case of bypass mode. 0 On the chip modulator clock will be used. 1 External clock will be used.
9 HW_TRG	Hardware Trigger Select This field is used to select trigger source for conversion. 0 Software trigger select 1 Hardware trigger select
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.5 Control Register (AFE_CR)

AFE Control Register has fields to control the global features of all channels of AFE.

Address: 4003_0000h base + 18h offset = 4003_0018h



AFE_CR field descriptions

Field	Description
31 MSTR_EN	AFE Master Enable Setting this bit enables all ADCs and filters simultaneously whose SD_MOD_EN and DEC_EN is asserted. If AFE bypass mode is enabled SD_MOD_EN bit will be ignored. 0 All ADCs are disabled. 1 All ADCs and filters will get simultaneously enabled .
30 SOFT_TRG0	Software Trigger0 This field is set to trigger conversion on AFE Channel0.
29 SOFT_TRG1	Software Trigger1 This field is set to trigger conversion on AFE Channel1.

Table continues on the next page...

AFE_CR field descriptions (continued)

Field	Description
28 SOFT_TRG2	Software Trigger2 This field is set to trigger conversion on AFE Channel2.
27 SOFT_TRG3	Software Trigger3 This field is set to trigger conversion on AFE Channel3.
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 LPM_EN	Low power Mode enable 0 AFE will be in normal mode 1 AFE will be in low power mode. Setting this bit reduce the current consumption of ADC and Buffer Amplifier , the max modulator clock frequency is below 1Mhz.
24–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 RST_B	Software Reset RST_B field is used to reset all ADCs, PGAs , Decimation filters and Clock configuration bits. 0 All ADCs, PGAs and Decimation filters are disabled. Clock Configuration bits will be reset. 1 .= All ADCs, PGAs and Decimation filters are enabled.
21 DLY_OK	Delay OK This field should be asserted after all the four delay registers are loaded. DLY values in CHx_DR will become active when DLY_OK =1 and "a new conversion starts". This bit is self clearing.
20–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 RESULT_ FORMAT	Result Format This field can be used to select AFE Result Register formating. 0 Left justified 2's complement 32-bit : SVVVVVVVVVVVVVVVVVVVVVV00000000 where (S= sign bit , V=valid result value, 0=zero) 1 Right justified 2's complement 32-bit : SSSSSSSSSVVVVVVVVVVVVVVVVVVVVV where (S= sign bit , V= valid result value, 0= zero)
17–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–9 STRTUP_CNT	Start up count This field is used to program the start up delay of modulators. Minimum value is two. if set less than that it would be converted to two.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.6 Clock Configuration Register (AFE_CKR)

AFE Clock Configuration Register has fields to control the source of clock, DIV factor and gating of clocks to AFEs.

Address: 4003_0000h base + 1Ch offset = 4003_001Ch

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

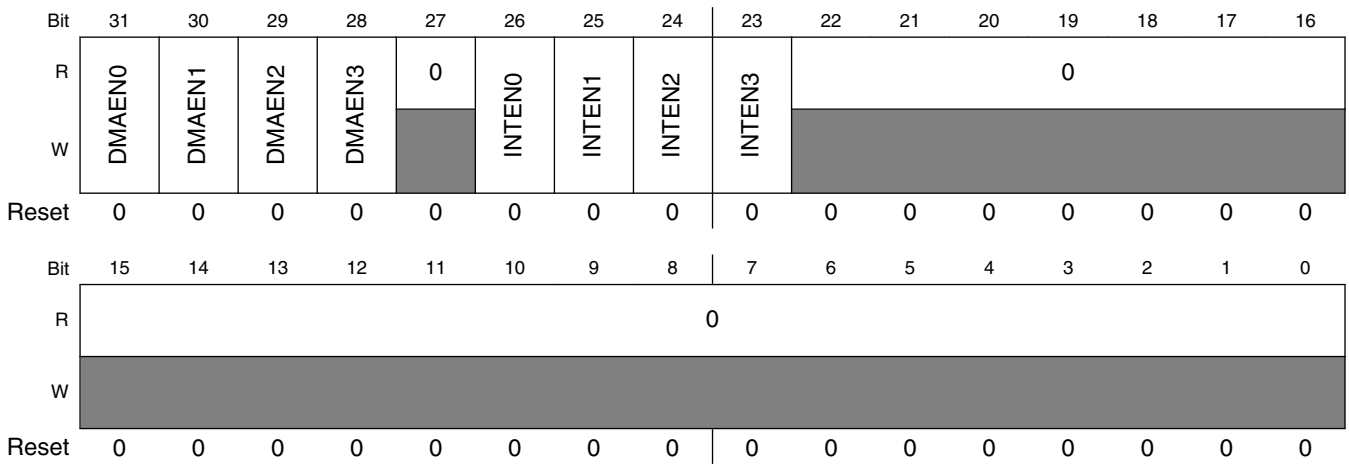
AFE_CKR field descriptions

Field	Description
31–28 DIV	Clock Divider Select The DIV selects the clock divider ratio for the modulator clock. The modulator clock frequency is the source clock divided by division factor. The default value of DIV is 2. 0000 divide by 1 0001 divide by 2 (default) 0010 divide by 4 0011 divide by 8 0100 divide by 16 0101 divide by 32 0110 divide by 64 0111 divide by 128 1xxx divide by 256
27–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–21 CLS	Clock Source Select The CLS bits are used to select one of the four clock sources for modulator clk. 00 mod_clk0 01 mod_clk1 10 mod_clk2 11 mod_clk3
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.7 DMA and Interrupt Register (AFE_DI)

DMA and Interrupt Register has fields to enable DMA and Interrupt requests.

Address: 4003_0000h base + 20h offset = 4003_0020h



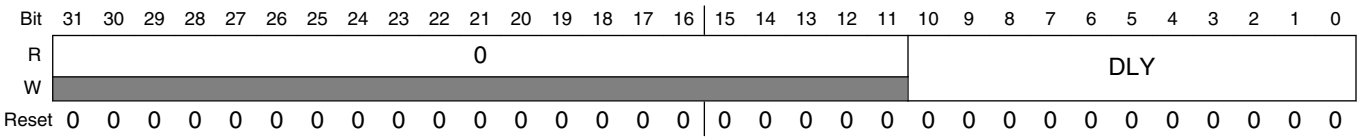
AFE_DI field descriptions

Field	Description
31 DMAEN0	DMA Enable0 The DMA request is asserted for channel0 when the DMAEN0 and COC0 bits are set.
30 DMAEN1	DMA Enable1 The DMA request is asserted for channel1 when the DMAEN1 and COC1 bits are set.
29 DMAEN2	DMA Enable2 The DMA request is asserted for channel2 when the DMAEN2 and COC2 bits are set.
28 DMAEN3	DMA Enable3 The DMA request is asserted for channel3 when the DMAEN3 and COC3 bits are set.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 INTEN0	Interrupt Enable 0 The conversion complete interrupt request is asserted for channel 0 when INTEN0 and COC0 bits are set.
25 INTEN1	Interrupt Enable 1 The conversion complete interrupt request is asserted for channel 1 when INTEN1 and COC1 bits are set.
24 INTEN2	Interrupt Enable 2 The conversion complete interrupt request is asserted for channel 2 when INTEN2 and COC2 bits are set.
23 INTEN3	Interrupt Enable 3 The conversion complete interrupt request is asserted for channel 3 when INTEN3 and COC3 bits are set.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.8.8 Channel0 Delay Register (AFE_CH0_DR)

Channel0 Delay Register contains a 11-bit field used to insert a delay into the trigger response of the decimation filters in order to provide phase compensation between four channels.

Address: 4003_0000h base + 2Ch offset = 4003_002Ch



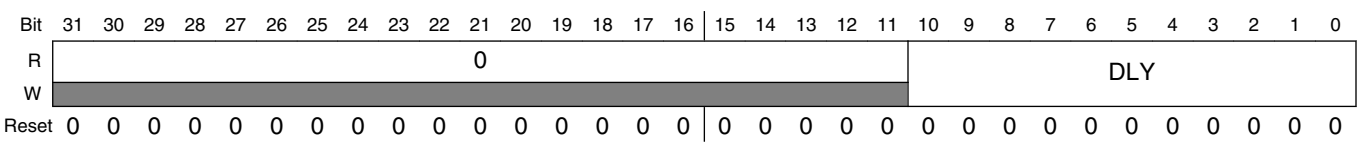
AFE_CH0_DR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	Delay This field can be used to provide phase compensation between four AFE Channels in step of prescaled modulator clock periods. This delay will be seen in every subsequent change to sampling phase after initial sampling.

31.8.9 Channel1 Delay Register (AFE_CH1_DR)

Channel1 Delay Register contains a 11-bit field used to insert a delay into the trigger response of the decimation filters in order to provide phase compensation between four channels.

Address: 4003_0000h base + 30h offset = 4003_0030h



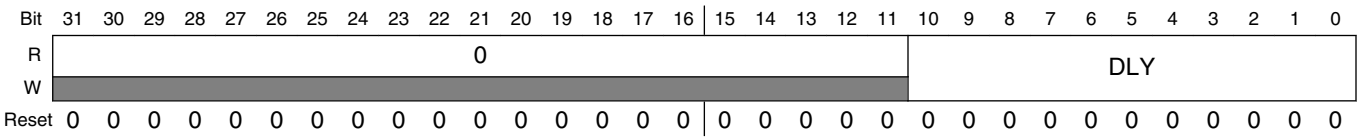
AFE_CH1_DR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	Delay This field can be used to provide phase compensation between four AFE Channels in step of prescaled modulator clock periods. This delay will be seen in every subsequent change to sampling phase after initial sampling.

31.8.10 Channel2 Delay Register (AFE_CH2_DR)

Channel2 Delay Register contains a 11-bit field used to insert a delay into the trigger response of the decimation filters in order to provide phase compensation between four channels.

Address: 4003_0000h base + 34h offset = 4003_0034h



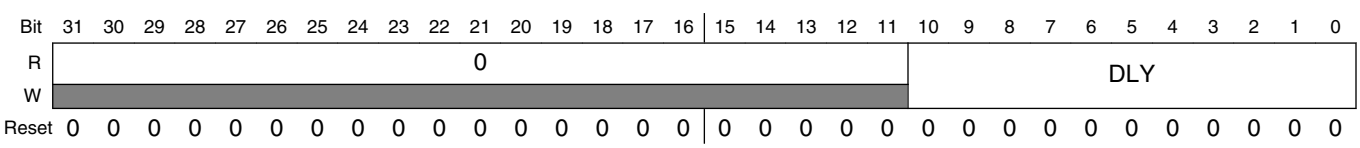
AFE_CH2_DR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	Delay This field can be used to provide phase compensation between four AFE Channels in step of prescaled modulator clock periods. This delay will be seen in every subsequent change to sampling phase after initial sampling.

31.8.11 Channel3 Delay Register (AFE_CH3_DR)

Channel3 Delay Register contains a 11-bit field used to insert a delay into the trigger response of the decimation filters in order to provide phase compensation between four channels.

Address: 4003_0000h base + 38h offset = 4003_0038h



AFE_CH3_DR field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	Delay The DLY field can be used to provide phase compensation between four AFE Channels in step of prescaled modulator clock periods. This delay will be seen in every subsequent change to sampling phase after initial sampling.

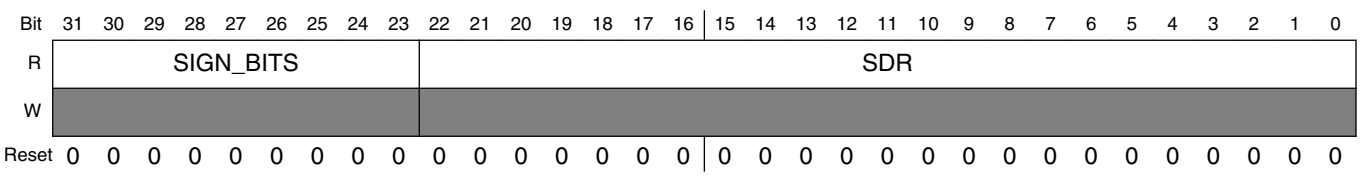
31.8.12 Channel0 Result Register (AFE_CH0_RR)

Depending upon AFE_CR[RESULT_FORMAT] field result can be represented in following ways: where (S= sign bit , V=valid result value, 0=zero)

Left justified 2's complement 32-bit :
SVVVVVVVVVVVVVVVVVVVVVV00000000

Right justified 2's complement 32-bit :
SSSSSSSSSVVVVVVVVVVVVVVVVVVVVV

Address: 4003_0000h base + 44h offset = 4003_0044h



AFE_CH0_RR field descriptions

Field	Description
31–23 SIGN_BITS	Sign Bits This field represents sign bits (bits 31 to 23 are filled with sign bits) if result is right justified or else 31-bit will be signbit and the lowest 8LSBs will be zeroes in case result is left justified. The formatting option can be selected from AFE_CR[RESULT_FORMAT].
SDR	Sample Data Result This field represents valid sample value in 2's complement form.

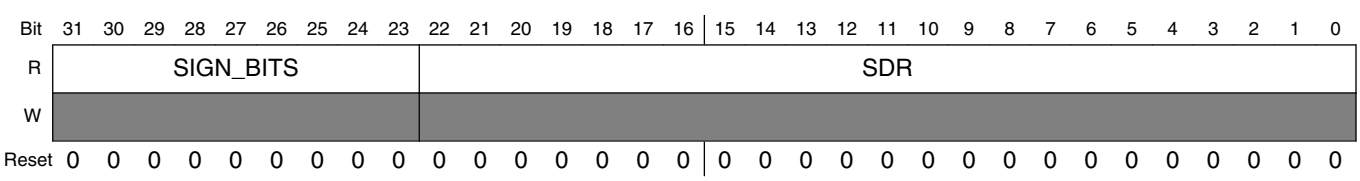
31.8.13 Channel1 Result Register (AFE_CH1_RR)

Depending upon AFE_CR[RESULT_FORMAT] field result can be represented in following ways: where (S= sign bit , V=valid result value, 0=zero)

Left justified 2's complement 32-bit :
SVVVVVVVVVVVVVVVVVVVVVV00000000

Right justified 2's complement 32-bit :
SSSSSSSSVVVVVVVVVVVVVVVVVVVVV

Address: 4003_0000h base + 48h offset = 4003_0048h



AFE_CH1_RR field descriptions

Field	Description
31–23 SIGN_BITS	Sign Bits This field represents sign bits (bits 31 to 23 are filled with sign bits) if result is right justified or else 31-bit will be signbit and the lowest 8LSBs will be zeroes in case result is left justified. The formatting option can be selected from AFE_CR[RESULT_FORMAT].
SDR	Sample Data Result This field represents valid sample value in 2's complement form.

31.8.14 Channel2 Result Register (AFE_CH2_RR)

Depending upon AFE_CR[RESULT_FORMAT] field result can be represented in following ways: where (S= sign bit , V=valid result value, 0=zero)

Left justified 2's complement 32-bit :
SVVVVVVVVVVVVVVVVVVVVVV00000000

Right justified 2's complement 32-bit :
SSSSSSSSSVVVVVVVVVVVVVVVVVVVVV

Address: 4003_0000h base + 4Ch offset = 4003_004Ch

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

AFE_CH2_RR field descriptions

Field	Description
31–23 SIGN_BITS	Sign Bits This field represents sign bits (bits 31 to 23 are filled with sign bits) if result is right justified or else 31-bit will be signbit and the lowest 8LSBs will be zeroes in case result is left justified. The formatting option can be selected from AFE_CR[RESULT_FORMAT].
SDR	Sample Data result This field represents valid sample value in 2's complement form.

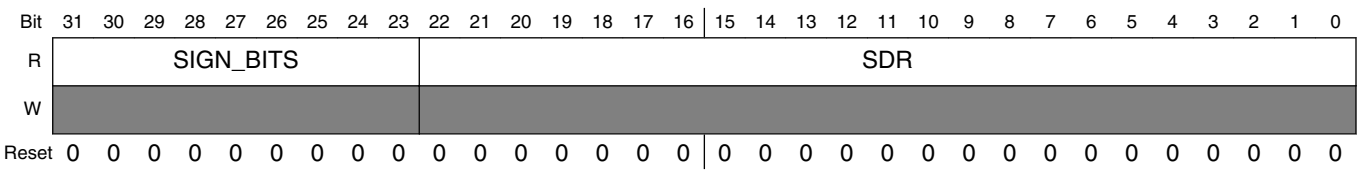
31.8.15 Channel3 Result Register (AFE_CH3_RR)

Depending upon AFE_CR[RESULT_FORMAT] field result can be represented in following ways: where (S= sign bit , V=valid result value, 0=zero)

Left justified 2's complement 32-bit :
SVVVVVVVVVVVVVVVVVVVVVV00000000

Right justified 2's complement 32-bit :
SSSSSSSSSVVVVVVVVVVVVVVVVVVVVV

Address: 4003_0000h base + 50h offset = 4003_0050h



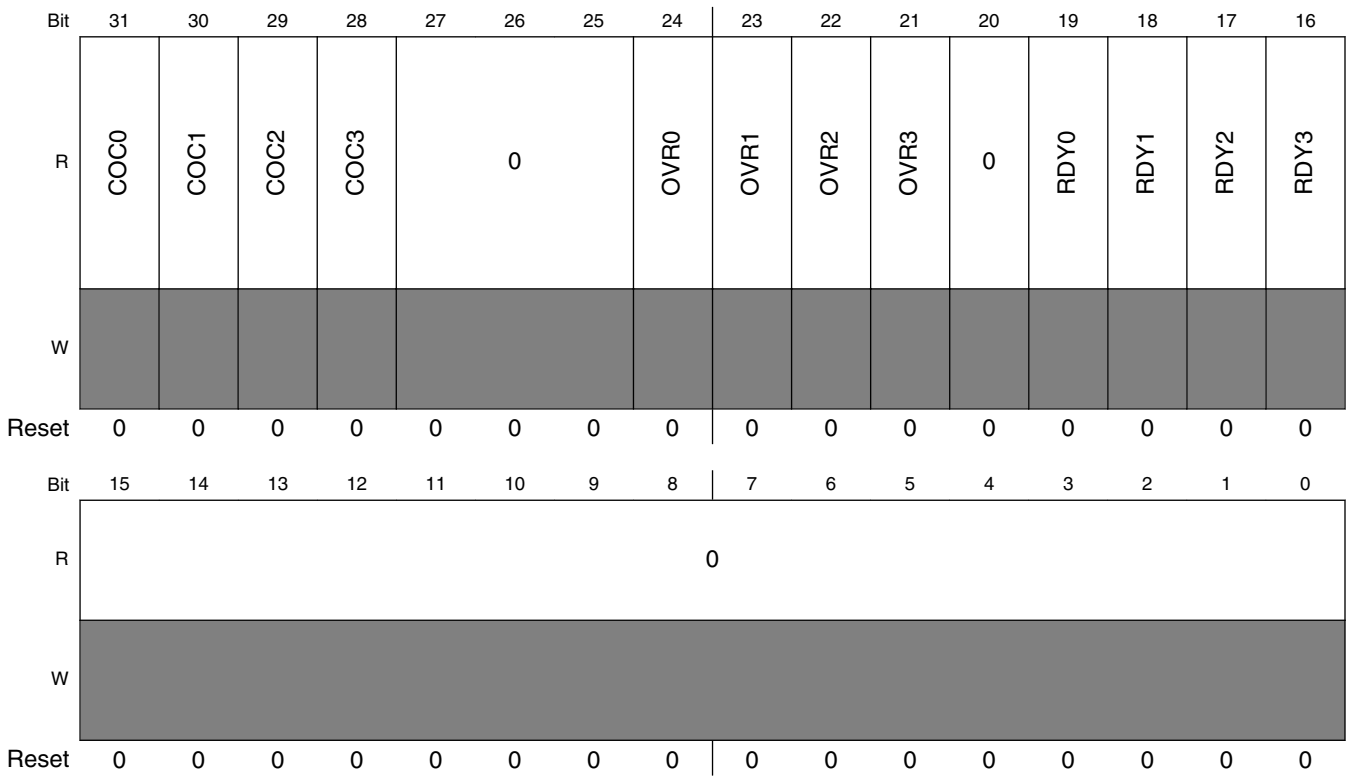
AFE_CH3_RR field descriptions

Field	Description
31–23 SIGN_BITS	Sign Bits This field represents sign bits (bits 31 to 23 are filled with sign bits) if result is right justified or else 31-bit will be signbit and the lowest 8LSBs will be zeroes in case result is left justified. The formatting option can be selected from AFE_CR[RESULT_FORMAT].
SDR	Sample Data result This field represents valid sample value in 2's complement form.

31.8.16 Status Register (AFE_SR)

AFE Status Register has three types of flags COC, OVR, RDY that indicate the status of each channel of AFE.

Address: 4003_0000h base + 5Ch offset = 4003_005Ch



AFE_SR field descriptions

Field	Description
31 COC0	Conversion Complete The COC flag is set when a conversion is completed and cleared on a read of the result register CH0_AFE_RR. Since modulators take some time to start up, this field is set at third sample and first two samples are ignored.
30 COC1	Conversion Complete The COC flag is set when a conversion is completed and cleared on a read of the result register CH1_AFE_RR. Since modulators take some time to start up, this field is set at third sample and first two samples are ignored.
29 COC2	Conversion Complete The COC flag is set when a conversion is completed and cleared on a read of the result register CH2_AFE_RR. Since modulators take some time to start up, this field is set at third sample and first two samples are ignored.
28 COC3	Conversion Complete

Table continues on the next page...

AFE_SR field descriptions (continued)

Field	Description
	The COC flag is set when a conversion is completed and cleared on a read of the result register CH3_AFE_RR. Since modulators take some time to start up, this field is set at third sample and first two samples are ignored.
27–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 OVR0	Overflow Flag This bit is set when the previous data has not been read and new data has already been arrived to be loaded in result register of the AFE Channel0 and cleared on a read of the result register.
23 OVR1	Overflow Flag This bit is set when the previous data has not been read and new data has already been arrived to be loaded in result register of the AFE Channel1 and cleared on a read of the result register
22 OVR2	Overflow Flag This bit is set when the previous data has not been read and new data has already been arrived to be loaded in result register of the AFE Channel2 and cleared on a read of the result register
21 OVR3	Overflow Flag This bit is set when the previous data has not been read and new data has already been arrived to be loaded in result register of the AFE Channel3 and cleared on a read of the result register
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19 RDY0	AFE Ready0 It indicates that AFE Channel0 is ready for conversion. 0 AFE Channel0 is disabled or has not completed its start up period 1 AFE Channel0 is ready to initiate conversions.
18 RDY1	AFE Ready1 It indicates that AFE Channel1 is ready for conversion. 0 AFE Channel1 is disabled or has not completed its start up period 1 AFE Channel1 is ready to initiate conversions.
17 RDY2	AFE Ready2 It indicates that AFE Channel2 is ready for conversion. 0 AFE Channel2 is disabled or has not completed its start up period 1 AFE Channel2 is ready to initiate conversions.
16 RDY3	AFE Ready3 It indicates that AFE Channel3 is ready for conversion. 0 AFE Channel3 is disabled or has not completed its start up period 1 AFE Channel3 is ready to initiate conversions.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.9 Power Modes

AFE supports four power modes

- Normal Run Mode
- Low Power Run Mode
- Wait Mode
- Stop Mode

31.9.1 Normal Run Mode

In Normal run mode AFE is fully functional and maximum clock frequency of operation can be up to 6.5 MHz.In this mode PGA can be optionally bypassed to achieve further low power.

31.9.2 Wait Mode

Operation of the AFE is not affected by the MCU entering wait mode. If conversions have already been initiated before entering wait mode, conversions will continue in wait. Conversions can be initiated while the MCU is in wait mode by means of the hardware trigger. Since the CPU is inactive in wait, a software trigger cannot initiate conversions while in wait. The system must continue to produce the modulator clock in order for conversions to continue during wait. If enabled, the AFE interrupt will wake the MCU from wait mode.

31.9.3 Low Power Run Mode

When the LPM_EN bit is set the AFE enters Low power run mode. In this mode the maximum frequency of the mod_clk should be less than equal to 1.6 MHz where 768 KHz is the nominal value. Setting this mode reduces the current consumption of ADC.When the device enters Very Low power mode(VLP) then AFE automatically enters into low power mode.

31.9.4 STOP Mode

When the device enters stop mode the ADC can be fully operational if the mod_clk is present. AFE is capable to wakeup the device from STOP mode through asynchronous interrupt. AFE is also capable to generate asynchronous DMA request to DMA controller in absence of bus clk. The asynchronous DMA request only gets asserted in STOP mode and remains asserted until synchronous DMA gets asserted in presence of system clock enabled by System Wake-up unit.

31.10 Functional Description

The main components of AFE are analog modulators,programmable gain amplifiers,decimation filters and digital logic comprised of programming model to configure AFE,store result register,clock generation,trigger generation logic and interrupt,dma capability.Modulator clock is provided by the system or external source which is internally prescaled to the correct operating range for AFE.The modulator samples the analog input signal and generates a digital output code at this clock rate.The high rate modulator output is filtered and decimated by the decimation filters which produce 24-bit output samples at a lower rate.Due to the latency of the decimation filter, each time decimation filter is enabled or resynchronization happens it takes three output samples to reach steady state in which fully valid samples are produced.This is the first valid AFE conversion data after it is enabled or re-synchronized.

31.10.1 Start Up

The AFE is disabled by default or when the enable bit in corresponding AFEx_CFR register is cleared. After reset user should enable all required SD-ADCs (SD_MOD_ENx =1), PGAs and Decimation filter and then enbale the global control by setting MSTR_EN bit. Once the configuration is done, AFE will wait for the conversion trigger to start conversion. Once software/hardware (according to TRGx bit) trigger is received the enabled ADC modules go through a 'n' modulator clock cycle start-up period where 'n' should be programmed (AFE_CR[STARTUP_CNT]) to achieve at-least 20 µs startup time. Once this start-up has completed the RDY flag is set and conversions begins. The software/hardware trigger signal is ignored during the start-up period.

31.10.2 Conversion Control

Once a conversion is triggered, the modulator,decimation filters are enabled and the COCx flag is set after the conversion completes.In continuous conversion mode (CC=1), conversions continue after the first sample. Using continuous conversion mode, a series of conversions can be made on at the rate of one sample per OSR modulator clock cycles.In single conversion mode each conversion requires a conversion trigger.

31.10.2.1 Triggering Conversions

Both software and hardware trigger mode is supported to initiate conversion.

- In software trigger mode (TRG=0), writing to AFE_CR[SFT_TRGx] bit will cause a conversion to initiate or restart while a conversion is already in progress.
- In hardware trigger mode (TRG=1), a conversion is initiated/restarted by the assertion of the hardware trigger input signal. The hardware trigger input is ignored during the start-up period. The hardware trigger source is device specific.

NOTE

Conversion modes are mentioned in details in later section.

31.10.2.2 Conversion Complete

When a conversion completes, the new sample value is transferred to the AFEx_RR register and the corresponding COCx bit is set. Several other actions can also be enabled when a conversion completes,for example if the INT_EN bit is set, the interrupt request is asserted, if the DMA bit is set the DMA request is asserted .

The COC bit and interrupt request are all cleared by reading the AFEx_RR register.If the COC bit is still set when a conversion completes and the new sample comes it will overwrite the old sample and set the overflow flag.

31.10.2.3 Total Conversion Time

- In Single conversion mode the total conversion time for a single conversion is (AFE_STARTUP+ 3 x OSR x MOD_CLK) after the trigger event.
- In Continuous conversion mode the conversion time for the first 24 bit sample is (AFE_STARTUP + 3 x OSR x MOD_CLK) after the trigger event,however the subsequent 24bit samples are available after (OSR x MOD_CLK) cycle.

- During any continuous conversion if user opts to resync any conversion by providing trigger , the first conversion result after the resync operation will be available after (AFE_STARTUP time + 3xOSRxMOD_CLK) cycle.
- However if user opts for run time phase compensation for any channel after the current conversion by adjusting Phase delay registers and setting dly_ok signal,the conversion result won't take 3 decimator clock cycle(3xOSRxMOD_CLK).Instead the sampled data will be available after OSRxMOD_CLK cycle.

NOTE

There can be latency of 2-3 mod_clk cycle in conversion time due to synchronization of asynchronous trigger.

31.10.2.4 Timing Diagram

Single conversion: Conversion result (Val1 or Val2) available after (STARTUP_TIME + 3xOSR) mod_clk cycle

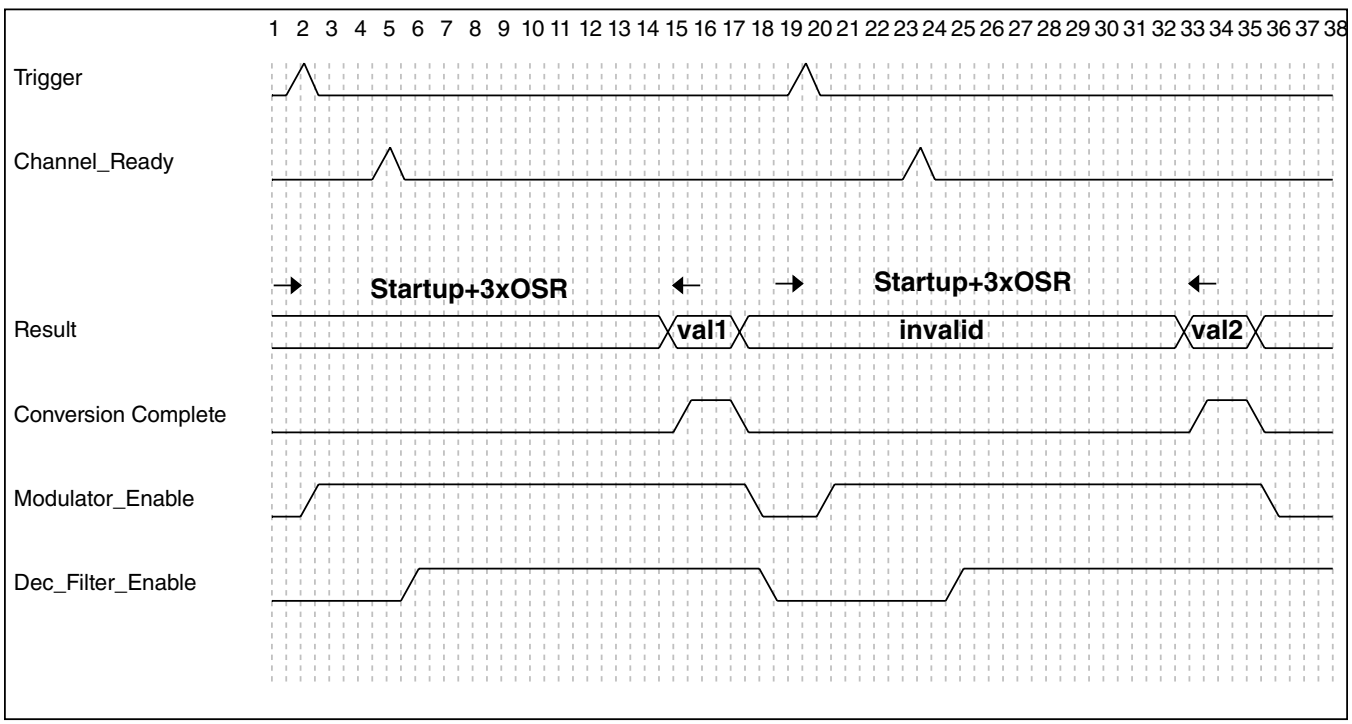


Figure 31-2. Single conversion

Continuous conversion: Val1 is available after (STARTUP_TIME + 3xOSR) mod_clk cycle Val2 to Val5 are available after OSR mod_clk cycle.

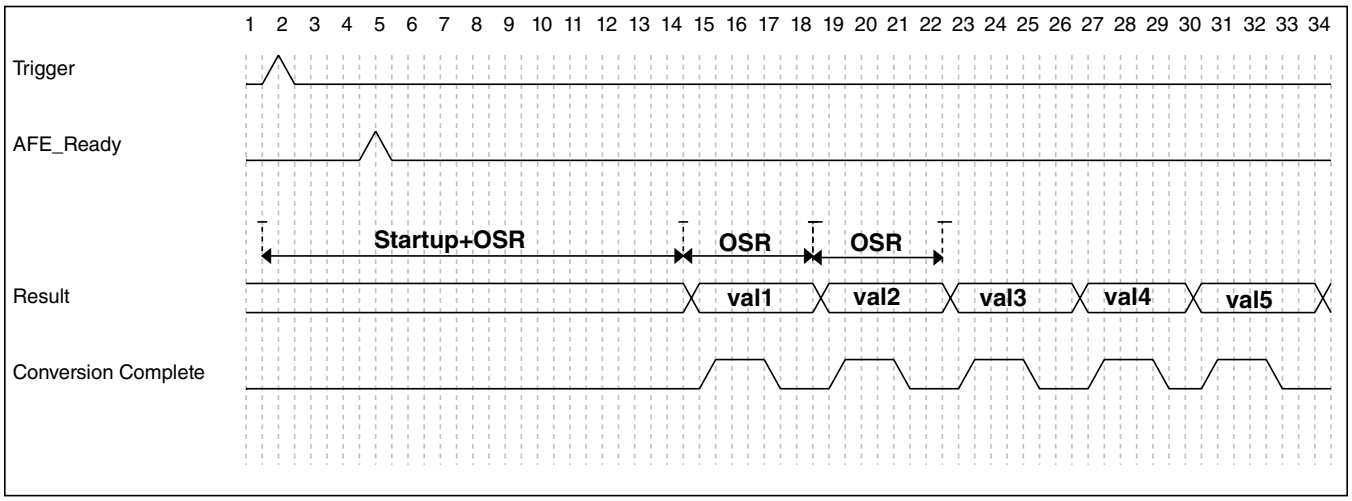


Figure 31-3. Continuous conversion

Continuous conversion with phase delay: Val1 of channel0, 1 are available after (STARTUP_TIME + 3xOSR) mod_clk cycle Val2 of channel0, 1 are available after OSR mod_clk cycle. Val3 of channel1 is available after (OSR+5) mod_clk cycle which signifies input sample of second channel is delayed by 5 mod_clk cycle w.r.t channel0. Val4, Val5 of channel1 are available after(OSR+8) mod_clk cycle which signifies input sample of second channel is delayed by 8 mod_clk cycle w.r.t channel0.

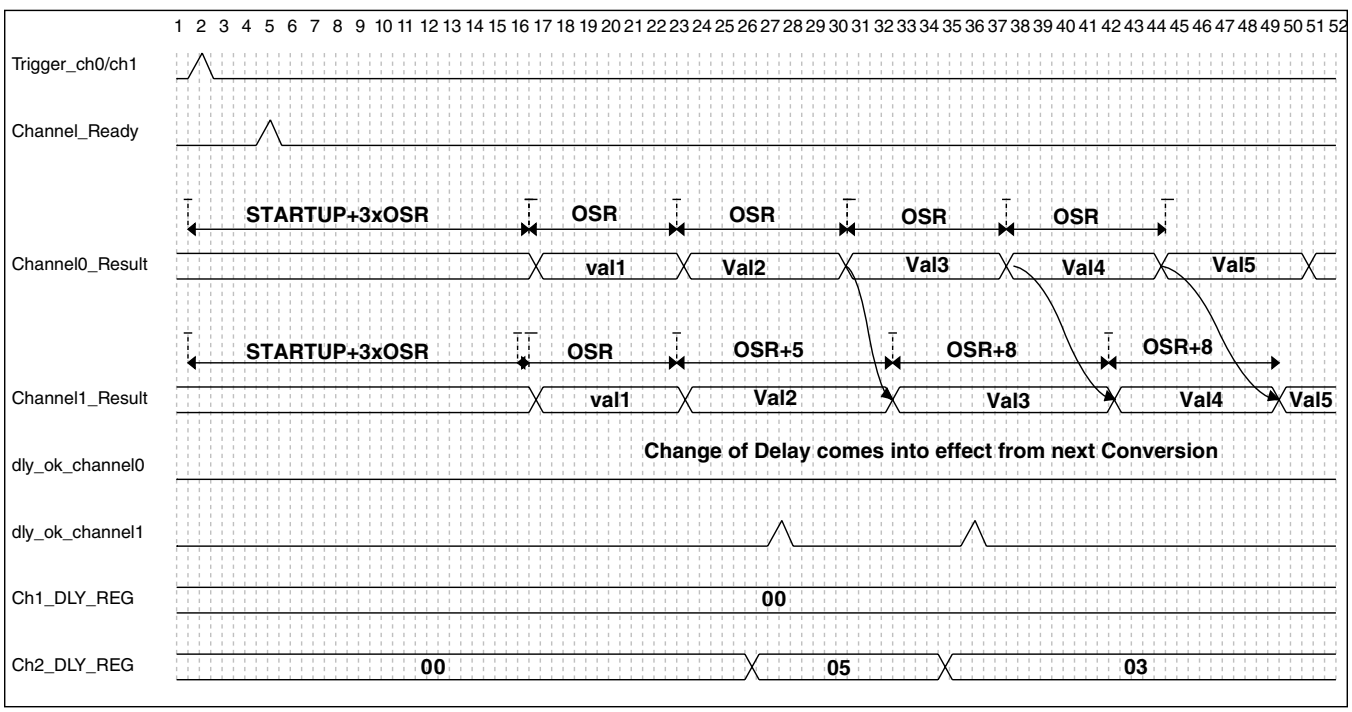


Figure 31-4. Continuous conversion with phase delay

Resynchronization of conversion: In case user initiates trigger during any conversion in progress, system will restart the conversion and first valid data after resynchronization will be available after 3xOSR mod_clk cycle.

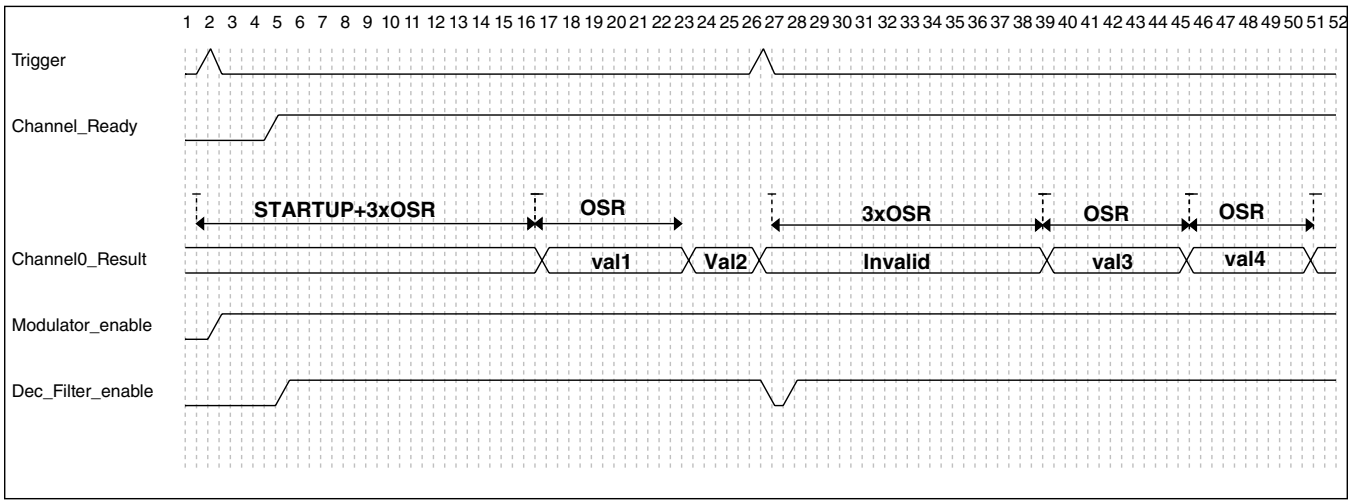


Figure 31-5. Resynchronization of conversion

31.10.2.5 Aborting Conversion

To abort a conversion in progress, disable the ADC by clearing SD_MOD_ENx bit.To abort/disable all the conversions globally ,clear MSTR_EN register.The AFEx_RR register will retain the value of the last completed conversion.

31.10.3 Modes of Conversion

AFE supports both single and continuous conversion mode where each channel can independently be either in single or continuous conversion mode. Below are the steps needed to initiate various conversion supported by AFE.

31.10.3.1 Single Conversion

By default AFE is in single conversion(AFE_CR[CC]=0) mode.After each single conversion modulators,PGAs and Filters get automatically disabled.User should set SW_TRG or hardware trigger to re-initiate another single conversion.Due to latency of decimation filter the valid conversion data becomes available after (AFE_STRUP + 3*OSR) mod_clk cycle.The different approaches to trigger a single conversions are mentioned below.

31.10.3.1.1 Software Trigger

If software trigger is enabled (HW_TRG = 0) then conversion is initiated by setting SW_TRGx bit.The steps are

- Enable and configure the ADCs by programming CHx_CFR register.
- Select AFE mod_clk by configuring AFE_CKR register
- Configure 20us startup delay counter by programming AFE_CR[STRTUP_CNT] register w.r.t mod_clk period.
- If phase compensation is required between the channels,configure the phase difference (in terms of mod_clk cycle) by programming CHx_DR register.
- Once CHx_DR are programmed,set AFE_CR[DLY_OK] bit.
- Set AFE_CR[MSTR_EN] bit.
- Set SOFT_TRGx bit.Setting the bit will initiate conversion after AFE startup delay programmed .
- In single conversion mode due to the latency of AFE filter, the valid result will be available after 3 * OSR modulator clock cycle.
- If Conversion complete DMA is enabled then AFE will generate dma_request to DMA controller after each valid conversion.
- If DMA is disabled and Conversion complete interrupt is enabled AFE will generate interrupt and set COCx flag after each valid conversion.
- While clearing AFE_CR[MSTR_EN] the last conversion result may give invalid result.

NOTE

All the configuration registers should be programmed before setting AFE_CR[MSTR_EN] bit.While the bit is set user should not change or disable any configuration.

31.10.3.1.2 Hardware Trigger

If hardware trigger is enabled (HW_TRG = 1) , a single conversion is initiated by external trigger event.The steps to initiate single conversion using hardware trigger are mentioned below

- Enable and configure the ADCs by programming CHx_CFR register.
- Select AFE mod_clk by configuring AFE_CKR register
- Configure 20us startup delay counter by programming AFE_CR[STARTUP_CNT] w.r.t mod_clk period.
- If phase compensation is required between the channels,configure the phase difference (in terms of mod_clk cycle) by programming CHx_DR register.
- Set DLY_OK flag.
- If already not set ,set AFE_CHX_CFR[HW_TRG] bit.
- Set AFE_CR[MSTR_EN] bit.
- Once the AFE is configured properly,an external pulse trigger will enable all Modulators,PGAs,Filters and its startup period.After the startup period the conversion result will be available after 3*OSR mod clock cycle.

- If Conversion complete DMA is enabled then AFE will generate dma_request to DMA controller.
- If DMA is disabled and Conversion complete interrupt is enabled AFE will generate interrupt and set COCx flag
- All the ADCs,PGA and Filter will be automatically disabled after a valid conversion.
- However for subsequent transfer user doesn't have to re-program CHx_CFR register.
- The subsequent hardware trigger will automatically enable modulators,PGAs and decimation filters for the next conversion.

NOTE

All the configuration registers should be programmed before setting AFE_CR[MSTR_EN] bit.While the bit is set user should not change or disable any configuration.

31.10.3.2 Continuous Conversion

Setting AFE_CR[CC] bit prior to any conversion,initiates continuous conversion.The different approaches to trigger a continuous conversions are mentioned below.

31.10.3.2.1 Soft Trigger

If software trigger is enabled (HW_TRG= 0) then conversion is initiated by SW_TRGx bit. The steps are :

- Enable and configure the channels by programming CHx_CFR register.
- Select AFE mod_clk by configuring AFE_CR register
- Configure 20us startup delay counter by programming AFE_CR[STRTUP_CNT] filed w.r.t mod_clk period.
- If phase compensation is required between the channels,configure the phase difference (in terms of mod_clk cycle) by programming CHx_DR register.
- Set AFE_CR[MSTR_EN] bit.Setting the bit will enable all ADCs and filters after AFE startup delay programmed .
- Set AFE_CR[DLY_OK] filed.
- Set SOFT_TRGx bit. Setting this bit will initiate conversion after AFE start up delay pogrammed.
- Software trigger event during this start up period will be ignored.
- In continuous conversion mode due to the latency of AFE filter, the first valid result will be available after 3 * OSR modulator clock cycle.Subsequent results will be available after OSR modulator clock cycle.
- If Conversion complete DMA is enabled then AFE will generate dma_request to DMA controller after each valid conversion.

- If DMA is disabled and Conversion complete interrupt is enabled AFE will generate interrupt and set COCx flag after each valid conversion.
- To disable continuous conversion globally for all channels, clear AFE_CR[MSTR_EN] bit or disable individual channels Filter(DEC_ENx = 0 or SD_MOD_EN = 0).
- While clearing AFE_CR[MSTR_EN] the last conversion result may give invalid result.

NOTE

All the configuration registers should be programmed before setting AFE_CR[MSTR_EN] bit.While the bit is set user should not change or disable any configuration otherwise it will abort the current conversion.

31.10.3.2.2 Hardware Trigger

If hardware trigger is enabled (HW_TRG = 1) then conversion is initiated by an external trigger pulse.The steps are

- Enable and configure the channels by programming CHx_CFR register.
- Select AFE mod_clk by configuring AFE_CR register
- Configure 20us startup delay counter by programming AFE_CR[STRTUP_CNT] filed w.r.t mod_clk period.
- If phase compensation is required between the channels,configure the phase difference (in terms of mod_clk cycle) by programming CHx_DR register.
- SET AFE_CHX_CFR[HW_TRG] bit(if not already set) to recognize hardware trigger event.
- Set AFE_CR[MSTR_EN] bit.Setting the bit will enable all ADCs and filters after AFE startup delay programmed .
- Set AFE_CR[DLY_OK] filed.
- An external trigger pulse will initiate conversion after the startup period.
- External trigger event during this start-up period will be ignored.
- In continuous conversion mode due to the latency of AFE filter, the first valid result will be available after 3 * OSR modulator clock cycle.Subsequent results will be available after OSR modulator clock cycle.
- While the conversion is in progress any hardware trigger event will restart the conversion and the first conversion data will be available after (AFE STARTUP +3xOSR) mod_clk cycle following the trigger event.
- If Conversion complete DMA is enabled then AFE will generate dma_request to DMA controller after each valid conversion.
- If DMA is disabled and Conversion complete interrupt is enabled AFE will generate interrupt and set COCx flag after each valid conversion.

- To disable continuous conversion globally for all channels, clear AFE_CR[MSTR_EN] bit or disable individual channels Filter(DEC_ENx = 0 or SD_MOD_EN = 0).
- While clearing AFE_CR[MSTR_EN] the last conversion result may give invalid result.

NOTE

All the configuration registers should be programmed before setting AFE_CR[MSTR_EN] bit.While the bit is set user should not change or disable any configuration.

31.10.3.2.3 On-the-fly Phase Delay Adjustment

While continuous conversion is in progress, on-the-fly delay adjustment between channels can be done using these steps.

- If phase compensation is required between the channels, configure the phase difference (in terms of mod_clk cycle) by programming CHx_DR register.
- Select AFE_CR[DLY_OK] bit.
- Internal phase synchronization will be initiated for the next sample with the adjusted delay programmed in CHx_DR register.
- If Conversion complete DMA is enabled then AFE will generate dma_request to DMA controller after each valid conversion.
- If DMA is disabled and Conversion complete interrupt is enabled AFE will generate interrupt and set COCx flag after each valid conversion.
- To disable continuous conversion globally for all channels, clear AFE_CR[MSTR_EN] bit or disable individual channels Filter(DEC_ENx = 0 or SD_MOD_EN = 0).
- While clearing AFE_CR[MSTR_EN] the last conversion result may give invalid result.

NOTE

All the configuration registers should be programmed before setting AFE_CR[MSTR_EN] bit.While the bit is set user should not change or disable any configuration otherwise it will abort the current conversion.

31.10.4 Independent Control for Conversion

Conversion control configuration for all channels are completely independent to each other to enhance the flexibility of AFE functionality.Hence user can configure one channel for single conversion with software trigger mode where the other channels are in continuous conversion mode triggered by either software or external hardware source.Each channel has independent conversion complete indicator to the system.

31.11 Decimation Filter

The decimation filter is a digital block that low-pass filters and decimates the high rate, single bit sigma-delta modulator output to generate a lower rate 24-bit output signal. The decimation filter supports over-sampling ratios (OSR) from 64 to 2048. It includes a third order sinc filter.The sinc filter down samples the signal by the selected OSR.The sinc filter output it scaled based on the OSR and then limited to produce a 24-bit signed, two’s complement output which is stored in corresponding channel's result register.

31.11.1 Sampling Phase Control

When conversion is either HW or SW triggered, regardless of the operation mode, or in continuous mode when new conversion starts then the phase between two or more channels may be adjusted ,if user opts to adjust the sampling phase during filtering operation to allow a precise sampling phase offset to be established and adjusted between two or more channels it is required to update the delay register and set dly_ok at the end to indicate the update of delay register. The delay registers set the initial phase offsets for the channels when they are enabled. When the channels are running, the delay registers are used to add delay to a channel. However the value of the delay register comes into effect from the next sampling period after the current conversion is completed.For example. the set of initial values in delay register are 0,5,10 and 8 for the four channels and we want the final sampling phase delays to be 0,4,12,10. The delays of 1,0,3,3 are updated in corresponding channel's delay register which yield the 1,5,13,11 as absolute delays and have the desired relative delay of 0,4,12,10 respectively.

31.11.2 Frequency response

The overall filter frequency response scales with the output sample rate which is a function of the modulator clock (mod_clk) frequency and the OSR. If the FIR filter is disabled, the frequency response is a $sinc^3$ response, which is $H(z) = [(1 - z^{-OSR}) / (1 - z^{-1})]^3$.

An example of the frequency response with a modulator clock of 6.144 MHz and an OSR=256 is shown in the following figure. In this case, the sample frequency is 24 kHz, so frequency components above 12 kHz are aliased back into the band from 0 to 12 kHz. Additional decimation and droop compensation filtering can done in software.

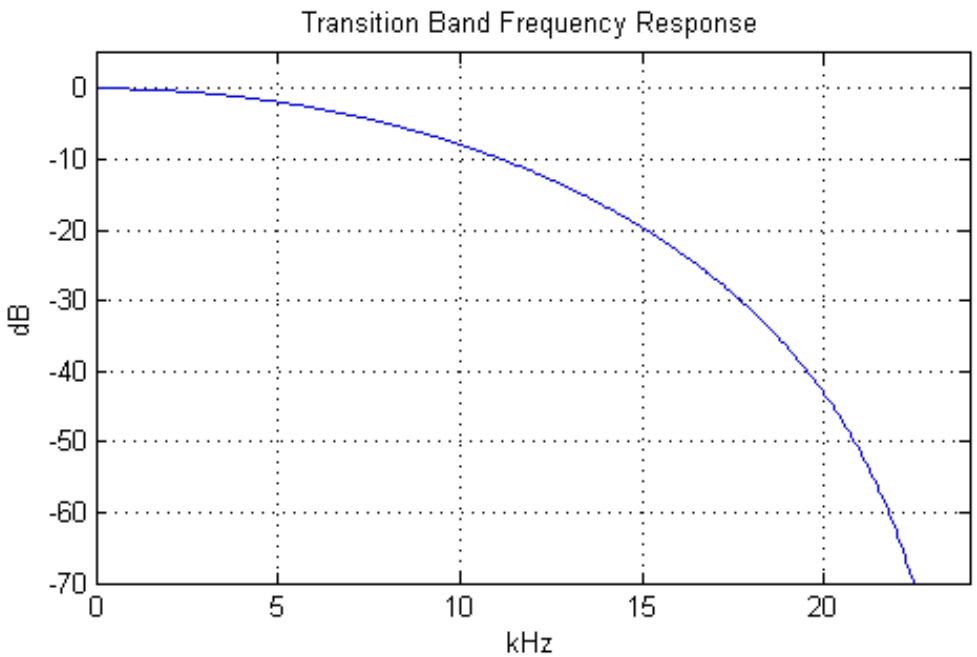


Figure 31-6. Frequency Response, 6.144 MHz, OSR=256

31.12 Modulator Bypass Mode

Modulator bypass mode is set by setting CHn_CR[BYP_MOD] bit. When set, the analog part of the corresponding channel gets automatically disabled.The SINC3 Filter and phase shift compensation block can be used in connection with an external sigma delta modulator. The clock to external sigma delta modulator can be derived from on-chip modulator clock. On the contrary, external sigma delta modulator binary output stream is routed to internal decimation filter.In such a way, any external sigma delta modulator can be used for the measurement along with other built-in sigma delta modulators.The clock is provided either by on-chip mod_clk source or any external clock input which after division fed to decimation filter as well as external Modulator through mod_clk_out port.The module also supports independent control for each channel.So one channel can be used in bypass mode,while the others are in modulator engage mode.

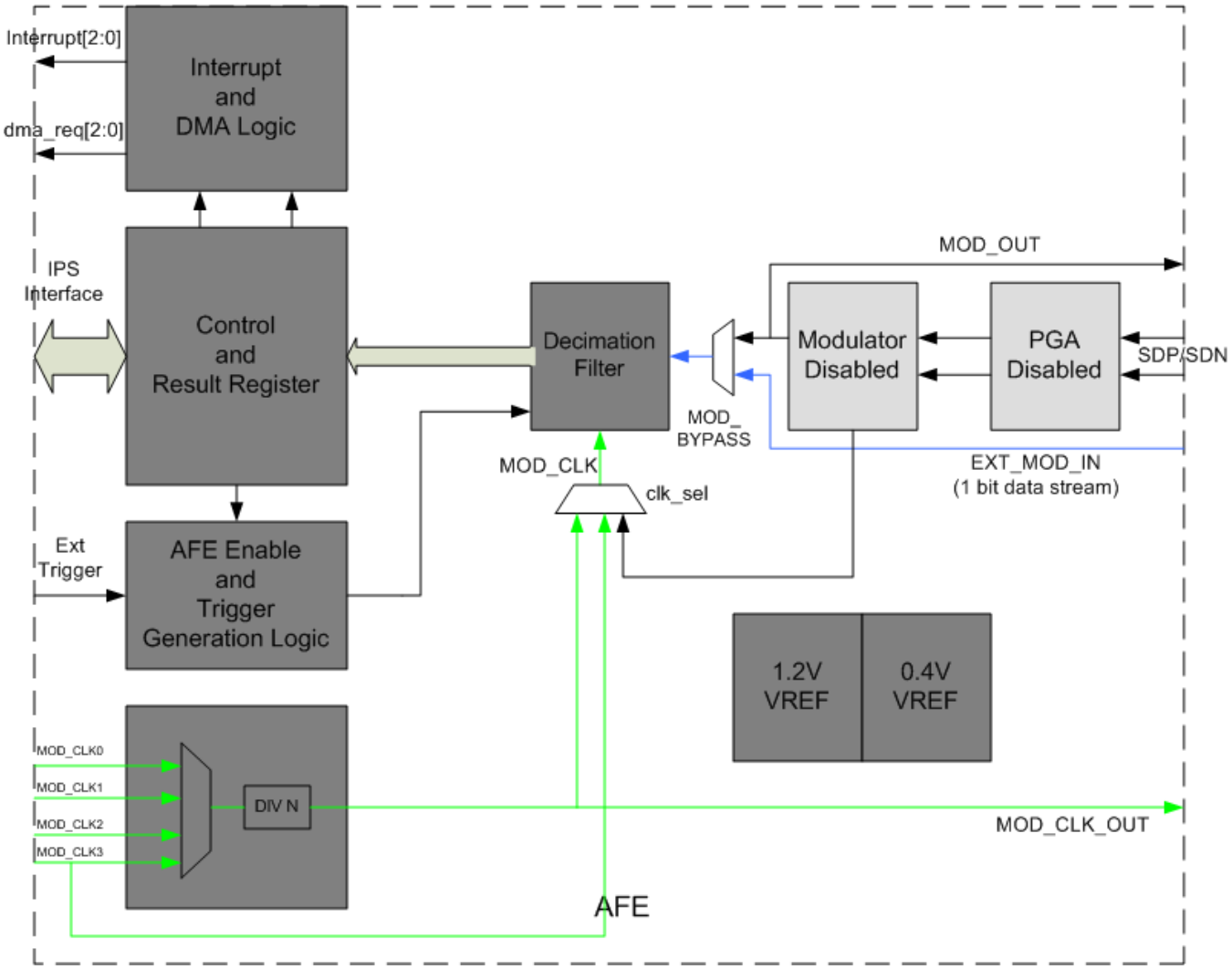


Figure 31-7. Data and Clock Flow in Modulator Bypass Mode

NOTE

The above diagram depicts the data/clock flow for one channel.The same is replicated for each channel.

Chapter 32
Analog-to-Digital Converter (ADC)

32.1 Chip-specific ADC information

32.1.1 Overview

This device contains a 16-bit successive-approximation analog-to-digital converter (SAR ADC) with DMA support. The SAR ADC is for general purpose use. There are no differential channel inputs for the SAR ADC. Only single ended channels are available which are muxed with other digital functions.

32.1.2 Instantiation

This device contains one 16-bit SAR ADC with 16 single ended channels (ADC0_SE0 - ADC0_SE3, ADC0_SE4a - ADC0_SE7a, ADC0_SE8 - ADC0_SE15; number varies on each package) and 4 result registers. The reference voltage option for this ADC are discussed in Reference Options section.

32.1.3 DMA Support on SAR ADC

Applications that may require continuous sampling of the ADC, may have considerable load on the CPU. The ADC supports DMA request functionality for higher performance when the ADC is sampled at a very high rate. The ADC can trigger the DMA (via DMA req) on conversion completion.

32.1.4 Channel Assignments

The following table describes all the connections to SAR ADC channels. Unused channels shall be connected to SAR_VREFL.

NOTE

Channels 0 - 3 do not have a mux in their path while other channels are muxed before being connected to SAR. Channel 0 - 3 are expected to give better performance compared to muxed channels.

Table 32-1. SAR ADC Channel Assignments

ADC Channel(SC1n[ADCH])	Channel	Input signal(SC1n[DIFF]=1)	Input signal(SC1n[DIFF]=0)	Description
00000	DAD0	Reserved	ADC0_SE0	Single ended input from pin
00001	DAD1	Reserved	ADC0_SE1	Single ended input from pin
00010	DAD2	Reserved	ADC0_SE2	Single ended input from pin
00011	DAD3	Reserved	ADC0_SE3	Single ended input from pin
00100	AD4a	Reserved	ADC0_SE4a	Single ended input from pin
00101	AD5a	Reserved	ADC0_SE5a	Single ended input from pin
00110	AD6a	Reserved	ADC0_SE6a	Single ended input from pin
00111	AD7a	Reserved	ADC0_SE7a	Single ended input from pin
00100	AD4b	Reserved	VLL1	LCD VLL1 Voltage
00101	AD5b	Reserved	VCAP1	LCD VCAP1 Voltage
00110	AD6b	Reserved	VCAP2	LCD VCAP2 Voltage
00111	AD7b	Reserved	VLL2	LCD VLL2 Voltage
01000	AD8	Reserved	ADC0_SE8	Single ended input from pin
01001	AD9	Reserved	ADC0_SE9	Single ended input from pin
01010	AD10	Reserved	ADC0_SE10	Single ended input from pin
01011	AD11	Reserved	ADC0_SE11	Single ended input from pin
01100	AD12	Reserved	ADC0_SE12	Single ended input from pin
01101	AD13	Reserved	ADC0_SE13	Single ended input from pin
01110	AD14	Reserved	ADC0_SE14	Single ended input from pin
01111	AD15	Reserved	ADC0_SE15	Single ended input from pin
10000	AD16	Reserved	Reserved	
10001	AD17	Reserved	PMC_VDD	VDD supply from PMC
10010	AD18	Reserved	PMC_VDD_SW	Switched VDD supply from PMC
10011	AD19	Reserved	SAR_VDDA	VDDA supply for SAR
10100	AD20	Reserved	CMP0 DAC Out	CMP0 DAC output voltage
10101	AD21	Reserved	CMP1 DAC Out	CMP1 DAC output voltage
10110	AD22	Reserved	CMP2 DAC Out	CMP2 DAC output voltage

Table continues on the next page...

Table 32-1. SAR ADC Channel Assignments (continued)

ADC Channel(SC1n[ADCH])	Channel	Input signal(SC1n[DIFF]=1)	Input signal(SC1n[DIFF]=0)	Description
10111	AD23	Reserved	VBAT	VBAT input voltage from VBAT pin
11000	AD24	Sense bus(for test)	Sense bus(for test)	
11001	AD25	Reserved	Reserved	
11010	AD26	Temperature Sensor(Diff)	Temperature Sensor(S.E)	
11011	AD27	PMC 1V Bandgap(Diff)	PMC 1V Bandgap(S.E)	
11100	AD28	Reserved	Reserved	
11101	AD29	VREF(Diff)	VREFH(S.E)	
11110	AD30	Reserved	VREFL	
11111	AD31	Module Disabled	Module Disabled	

32.1.5 Reference Options

The reference selection is done from within the SAR ADC. The SAR ADC supports the following references:

- V_{REFH} and V_{REFL} of SAR ADC - connected as the primary reference option. However, these are bonded to SAR_VDDA & SAR_VSSA respectively on the device package
- Bandgap voltage from PMC
- V_{ALTH} and V_{ALTL} (either from off-chip source or internal source for which the selection is controlled from SIM_MISC_CTL[VREFBUFINSEL])

NOTE

Refer to the ADC_SC2[REFSEL] bitfield in Status and Control Register 2 for more information.

32.1.6 Trigger Options

The ADC supports both software and hardware triggers. The hardware trigger options are shown in the table below.

Table 32-2. Hardware trigger options for SAR ADC

	PDB	Quad Timer	PIT	iRTC	LPTIMER

Table continues on the next page...

Table 32-2. Hardware trigger options for SAR ADC (continued)

RUN, VLPR	Yes	Yes	Yes	Yes	Yes
WAIT, VLPW, VLPS, STOP and PSTOP1	No in VLPS & STOP. Yes if module is active in WAIT/ VLPW.	Yes	No in VLPS & STOP. Yes if module is active in WAIT/ VLPW.	Yes	Yes

The selection of trigger can be done via the XBAR. User should clear SIM_MISC_CTL[PDBADCTRG] bit so that XBAR outputs can serve as the SAR ADC hardware trigger. The trigger outputs from XBAR are connected to respective SAR ADC trigger inputs, thereby forcing ADC conversions.

When SIM_MISC_CTL[PDBADCTRG] bit is set to 1, the PDB module serves as the SAR ADC hardware trigger source.

The ADC can be triggered in low power modes (WAIT, VLPW, VLPS, STOP and PSTOP1) using QTMR, LPTIMER or iRTC. This allows the ADC to do conversions in low power modes and store the output in the result register. The ADC generates interrupt when the data is ready in the result register that wakes the system from low power mode.

NOTE

- The timer (QTMR or PDB) channels used for trigger generation must be programmed to generate the trigger pulses (Trigger A, …, Trigger D) in an interleaved fashion. Pulses should not come at the same time.
- Generation of a trigger before conversion completes for the previous trigger is an error condition.

32.1.7 Alternate Clock

The ADC has multiple input clock sources. Selection is determined by the ADCx_CFG1[ADICLK] bitfield. The following table shows the chip-specific clock assignments for this bitfield.

NOTE

The ALTCLK option is only usable when OSCERCLK is in MHz range. A system with OSCERCLK in kHz range has the optional clock source below the minimum ADC clock operating frequency.

NOTE

It is not a usable source in Stop – ADC wrapper logic requires the selection of the ADC internal clock source to operate in Stop.

Table 32-3. ADC Conversion Clock Options

ADCx_CFG1[ADICLK]	ADC defined selection	Chip clock	Note
00	Bus Clock	Bus Clock	
01	ALTCLK2	MCGPLLCLK	1
10	ALTCLK	OSCERCLK	1
11	Asynchronous clock (ADACK)	N/A - sourced from within ADC block	

1. For ADC operation in Compute only, PSTOP1, Stop or VLPS mode, ADACK and the alternate clock sources are the only allowed clock sources.

32.2 Introduction

The 16-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

32.2.1 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to 24 single-ended external analog inputs
- Output modes:
  - single-ended 16-bit, 12-bit, 10-bit, and 8-bit modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion, that is, automatic return to idle after single conversion
- Configurable sample time and conversion speed/power

- Conversion complete/hardware average complete flag and interrupt
- Input clock selectable from up to four sources
- Operation in low-power modes for lower noise
- Asynchronous clock source for lower noise operation with option to output the clock
- Selectable hardware conversion trigger with hardware channel select
- Automatic compare with interrupt for less-than, greater-than or equal-to, within range, or out-of-range, programmable value
- Temperature sensor
- Hardware average function
- Selectable voltage reference: external or alternate
- Self-Calibration mode

32.2.2 Block diagram

The following figure is the ADC module block diagram.

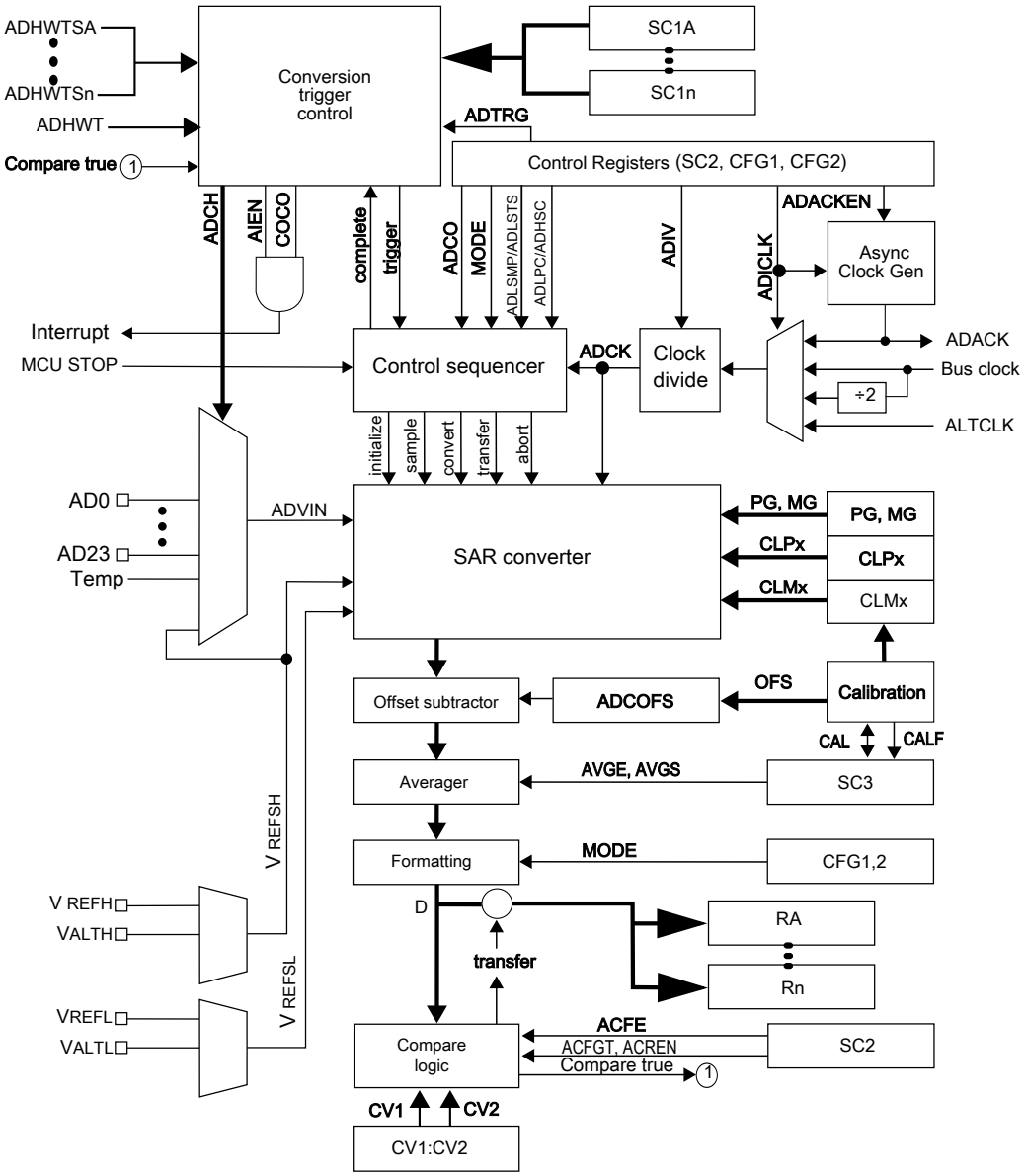


Figure 32-1. ADC block diagram

32.3 ADC signal descriptions

The ADC module supports up to 24 single-ended inputs.

The ADC also requires four supply/reference/ground connections.

NOTE

For the number of channels supported on this device as well as information regarding other chip-specific inputs into the ADC block, see the chip-specific ADC configuration information.

Table 32-4. ADC signal descriptions

Signal	Description	I/O
AD*n*	Single-Ended Analog Channel Inputs	I
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I

32.3.1 Analog Power (V_{DDA})

The ADC analog portion uses V_{DDA} as its power connection. In some packages, V_{DDA} is connected internally to V_{DD}. If externally available, connect the V_{DDA} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDA} for good results.

32.3.2 Analog Ground (V_{SSA})

The ADC analog portion uses V_{SSA} as its ground connection. In some packages, V_{SSA} is connected internally to V_{SS}. If externally available, connect the V_{SSA} pin to the same voltage potential as V_{SS}.

32.3.3 Analog Channel Inputs (ADx)

The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the SC1[ADCH] channel select bits.

32.4 Memory map and register definitions

This section describes the ADC registers.

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_B000	ADC Status and Control Registers 1 (ADC0_SC1A)	32	R/W	0000_001Fh	32.4.1/583
4002_B004	ADC Status and Control Registers 1 (ADC0_SC1B)	32	R/W	0000_001Fh	32.4.1/583
4002_B008	ADC Status and Control Registers 1 (ADC0_SC1C)	32	R/W	0000_001Fh	32.4.1/583
4002_B00C	ADC Status and Control Registers 1 (ADC0_SC1D)	32	R/W	0000_001Fh	32.4.1/583

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_B010	ADC Configuration Register 1 (ADC0_CFG1)	32	R/W	0000_0000h	32.4.2/587
4002_B014	ADC Configuration Register 2 (ADC0_CFG2)	32	R/W	0000_0000h	32.4.3/588
4002_B018	ADC Data Result Register (ADC0_RA)	32	R	0000_0000h	32.4.4/589
4002_B01C	ADC Data Result Register (ADC0_RB)	32	R	0000_0000h	32.4.4/589
4002_B020	ADC Data Result Register (ADC0_RC)	32	R	0000_0000h	32.4.4/589
4002_B024	ADC Data Result Register (ADC0_RD)	32	R	0000_0000h	32.4.4/589
4002_B028	Compare Value Registers (ADC0_CV1)	32	R/W	0000_0000h	32.4.5/590
4002_B02C	Compare Value Registers (ADC0_CV2)	32	R/W	0000_0000h	32.4.5/590
4002_B030	Status and Control Register 2 (ADC0_SC2)	32	R/W	0000_0000h	32.4.6/591
4002_B034	Status and Control Register 3 (ADC0_SC3)	32	R/W	0000_0000h	32.4.7/593
4002_B038	ADC Offset Correction Register (ADC0_OFS)	32	R/W	0000_0004h	32.4.8/594
4002_B03C	ADC Plus-Side Gain Register (ADC0_PG)	32	R/W	0000_8200h	32.4.9/595
4002_B044	ADC Plus-Side General Calibration Value Register (ADC0_CLPD)	32	R/W	0000_000Ah	32.4.10/ 596
4002_B048	ADC Plus-Side General Calibration Value Register (ADC0_CLPS)	32	R/W	0000_0020h	32.4.11/ 596
4002_B04C	ADC Plus-Side General Calibration Value Register (ADC0_CLP4)	32	R/W	0000_0200h	32.4.12/ 597
4002_B050	ADC Plus-Side General Calibration Value Register (ADC0_CLP3)	32	R/W	0000_0100h	32.4.13/ 597
4002_B054	ADC Plus-Side General Calibration Value Register (ADC0_CLP2)	32	R/W	0000_0080h	32.4.14/ 598
4002_B058	ADC Plus-Side General Calibration Value Register (ADC0_CLP1)	32	R/W	0000_0040h	32.4.15/ 598
4002_B05C	ADC Plus-Side General Calibration Value Register (ADC0_CLP0)	32	R/W	0000_0020h	32.4.16/ 599

32.4.1 ADC Status and Control Registers 1 (ADC*x*_SC1*n*)

SC1A is used for both software and hardware trigger modes of operation.

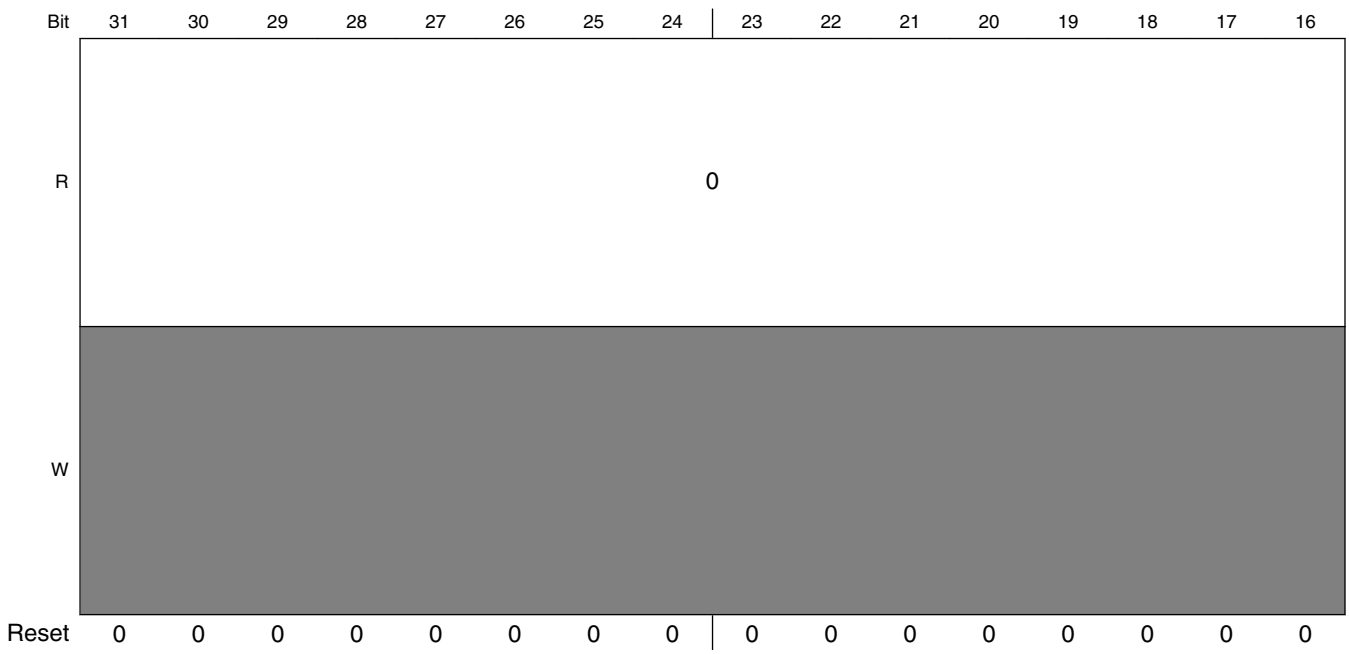
To allow sequential conversions of the ADC to be triggered by internal peripherals, the ADC can have more than one status and control register: one for each conversion. The SC1B–SC1n registers indicate potentially multiple SC1 registers for use only in hardware trigger mode. See the chip configuration information about the number of SC1n registers specific to this device. The SC1n registers have identical fields, and are used in a "ping-pong" approach to control ADC operation.

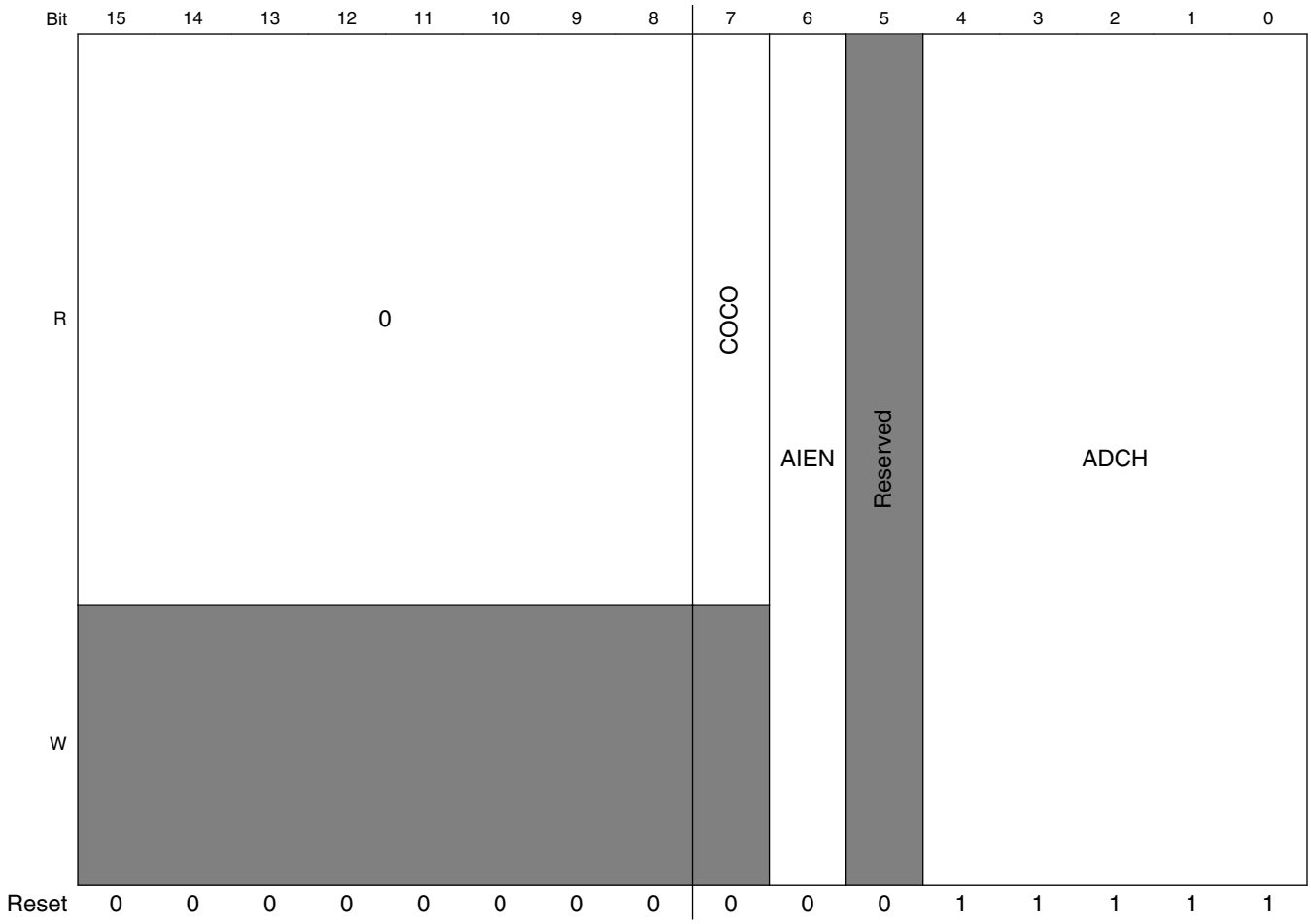
At any one point in time, only one of the SC1n registers is actively controlling ADC conversions. Updating SC1A while SC1n is actively controlling a conversion is allowed, and vice-versa for any of the SC1n registers specific to this MCU.

Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, writes to SC1A subsequently initiate a new conversion, if SC1[ADCH] contains a value other than all 1s (module disabled).

Writing any of the SC1n registers while that specific SC1n register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1n registers are used for software trigger operation and therefore writes to the SC1B–SC1n registers do not initiate a new conversion.

Address: 4002_B000h base + 0h offset + (4d × i), where i=0d to 3d





ADCx_SC1n field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 COCO	Conversion Complete Flag This is a read-only field that is set each time a conversion is completed when the compare function is disabled, or SC2[ACFE]=0 and the hardware average function is disabled, or SC3[AVGE]=0. When the compare function is enabled, or SC2[ACFE]=1, COCO is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled, or SC3[AVGE]=1, COCO is set upon completion of the selected number of conversions (determined by AVGS). COCO in SC1A is also set at the completion of a calibration sequence. COCO is cleared when the respective SC1n register is written or when the respective Rn register is read. 0 Conversion is not completed. 1 Conversion is completed.
6 AIEN	Interrupt Enable Enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt is disabled. 1 Conversion complete interrupt is enabled.

Table continues on the next page...

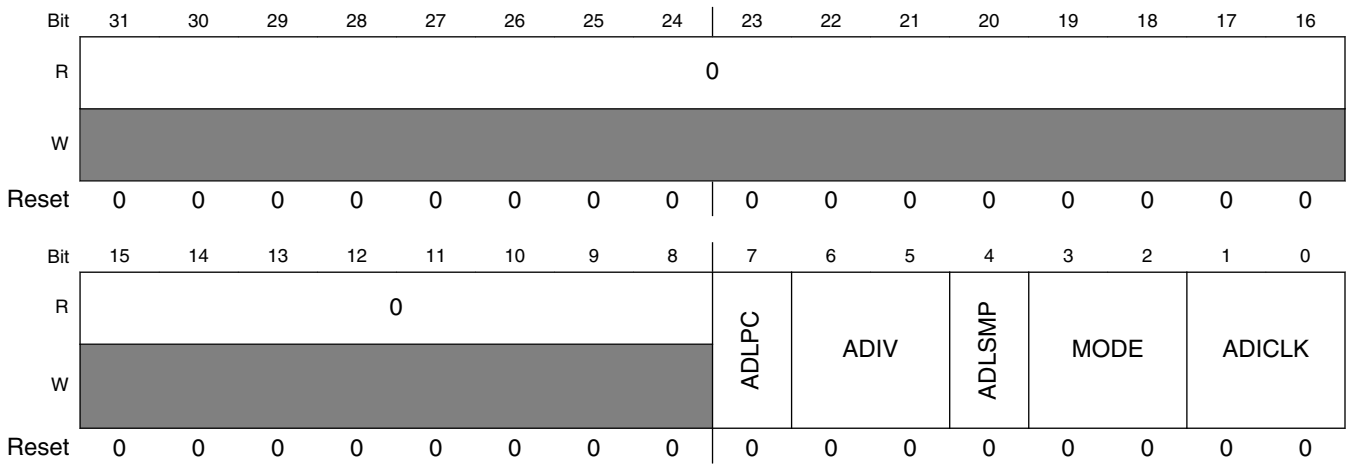
ADC*x*_SC1*n* field descriptions (continued)

Field	Description
5 Reserved	This field is reserved. This reserved bit should not be changed.
ADCH	Input channel select Selects one of the input channels. **NOTE:** Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the actual ADC channel assignments for your device, see the Chip Configuration details. The successive approximation converter subsystem is turned off when the channel select bits are all set, that is, ADCH = 11111. This feature allows explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional single conversion from being performed. It is not necessary to set ADCH to all 1s to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes. 00000 AD0 is selected as input. 00001 AD1 is selected as input. 00010 AD2 is selected as input. 00011 AD3 is selected as input. 00100 AD4 is selected as input. 00101 AD5 is selected as input. 00110 AD6 is selected as input. 00111 AD7 is selected as input. 01000 AD8 is selected as input. 01001 AD9 is selected as input. 01010 AD10 is selected as input. 01011 AD11 is selected as input. 01100 AD12 is selected as input. 01101 AD13 is selected as input. 01110 AD14 is selected as input. 01111 AD15 is selected as input. 10000 AD16 is selected as input. 10001 AD17 is selected as input. 10010 AD18 is selected as input. 10011 AD19 is selected as input. 10100 AD20 is selected as input. 10101 AD21 is selected as input. 10110 AD22 is selected as input. 10111 AD23 is selected as input. 11000 Reserved. 11001 Reserved. 11010 Temp Sensor (single-ended) is selected as input. 11011 Bandgap (single-ended) is selected as input. 11100 Reserved. 11101 V_{REFSH} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 11110 V_{REFSL} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 11111 Module is disabled.

32.4.2 ADC Configuration Register 1 (ADCx_CFG1)

The configuration Register 1 (CFG1) selects the mode of operation, clock source, clock divide, and configuration for low power or long sample time.

Address: 4002_B000h base + 10h offset = 4002_B010h



ADCx_CFG1 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ADLPC	Low-Power Configuration Controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 Normal power configuration. 1 Low-power configuration. The power is reduced at the expense of maximum clock speed.
6–5 ADIV	Clock Divide Select Selects the divide ratio used by the ADC to generate the internal clock ADCK. 00 The divide ratio is 1 and the clock rate is input clock. 01 The divide ratio is 2 and the clock rate is (input clock)/2. 10 The divide ratio is 4 and the clock rate is (input clock)/4. 11 The divide ratio is 8 and the clock rate is (input clock)/8.
4 ADLSMP	Sample Time Configuration Selects between different sample times based on the conversion mode selected. This field adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption if continuous conversions are enabled and high conversion rates are not required. When ADLSMP=1, the long sample time select bits, (ADLSTS[1:0]), can select the extent of the long sample time.

Table continues on the next page...

ADCx_CFG1 field descriptions (continued)

Field	Description
	0 Short sample time. 1 Long sample time.
3–2 MODE	Conversion mode selection Selects the ADC resolution mode. 00 It is single-ended 8-bit conversion. 01 It is single-ended 12-bit conversion . 10 It is single-ended 10-bit conversion. 11 It is single-ended 16-bit conversion..
ADICLK	Input Clock Select Selects the input clock source to generate the internal clock, ADCK. Note that when the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start, when CFG2[ADACKEN]=0, the asynchronous clock is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. 00 Bus clock 01 Bus clock divided by 2(BUSCLK/2) 10 Alternate clock (ALTCLK) 11 Asynchronous clock (ADACK)

32.4.3 ADC Configuration Register 2 (ADCx_CFG2)

Configuration Register 2 (CFG2) selects the special high-speed configuration for very high speed conversions and selects the long sample time duration during long sample mode.

Address: 4002_B000h base + 14h offset = 4002_B014h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0			MUXSEL	ADACKEN	ADHSC	ADLSTS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CFG2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 MUXSEL	ADC Mux Select Changes the ADC mux setting to select between alternate sets of ADC channels. 0 ADxxa channels are selected. 1 ADxxb channels are selected.
3 ADACKEN	Asynchronous Clock Output Enable Enables the asynchronous clock source and the clock source output regardless of the conversion and status of CFG1[ADICLK]. Based on MCU configuration, the asynchronous clock may be used by other modules. See chip configuration information. Setting this field allows the clock to be used even while the ADC is idle or operating from a different clock source. Also, latency of initiating a single or first-continuous conversion with the asynchronous clock selected is reduced because the ADACK clock is already operational. 0 Asynchronous clock output disabled; Asynchronous clock is enabled only if selected by ADICLK and a conversion is active. 1 Asynchronous clock and clock output is enabled regardless of the state of the ADC.
2 ADHSC	High-Speed Configuration Configures the ADC for very high-speed operation. The conversion sequence is altered with 2 ADCK cycles added to the conversion time to allow higher speed conversion clocks. 0 Normal conversion sequence selected. 1 High-speed conversion sequence selected with 2 additional ADCK cycles to total conversion time.
ADLSTS	Long Sample Time Select Selects between the extended sample times when long sample time is selected, that is, when CFG1[ADLSMP]=1. This allows higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 00 Default longest sample time; 20 extra ADCK cycles; 24 ADCK cycles total. 01 12 extra ADCK cycles; 16 ADCK cycles total sample time. 10 6 extra ADCK cycles; 10 ADCK cycles total sample time. 11 2 extra ADCK cycles; 6 ADCK cycles total sample time.

32.4.4 ADC Data Result Register (ADCx_R*n*)

The data result registers (Rn) contain the result of an ADC conversion of the channel selected by the corresponding status and channel control register (SC1A:SC1n). For every status and channel control register, there is a corresponding data result register.

Unused bits in R n are cleared in unsigned right-aligned modes and carry the sign bit (MSB) in sign-extended 2's complement modes.

The following table describes the behavior of the data result registers in the different modes of operation.

Table 32-5. Data result register description

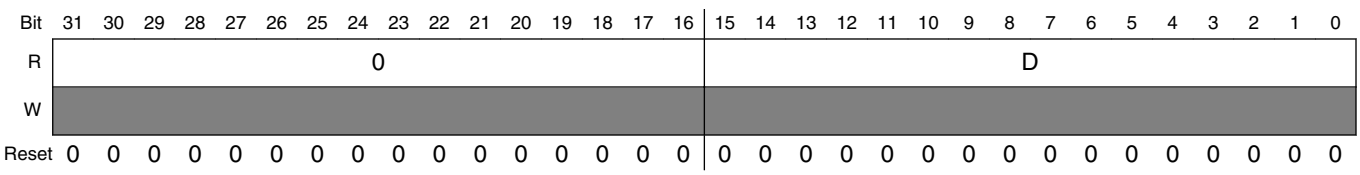
Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
12-bit single-ended	0	0	0	0	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right-justified
10-bit single-ended	0	0	0	0	0	0	D	D	D	D	D	D	D	D	D	D	Unsigned right-justified
8-bit single-ended	0	0	0	0	0	0	0	0	D	D	D	D	D	D	D	D	Unsigned right-justified

NOTE

S: Sign bit or sign bit extension;

D: Data, which is 2's complement data if indicated

Address: 4002_B000h base + 18h offset + (4d × i), where i=0d to 3d



ADCx_Rn field descriptions

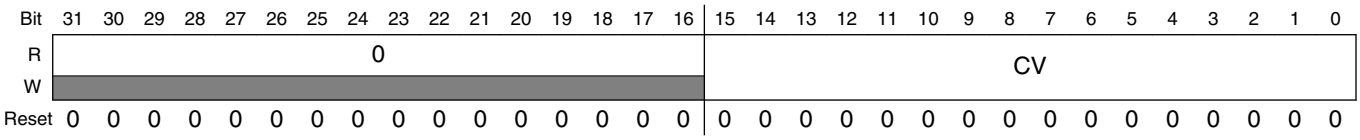
Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
D	Data result

32.4.5 Compare Value Registers (ADCx_CVn)

The Compare Value Registers (CV1 and CV2) contain a compare value used to compare the conversion result when the compare function is enabled, that is, SC2[ACFE]=1. This register is formatted in the same way as the Rn registers in different modes of operation for both bit position definition and value format using unsigned or sign-extended 2's complement. Therefore, the compare function uses only the CVn fields that are related to the ADC mode of operation.

The compare value 2 register (CV2) is used only when the compare range function is enabled, that is, SC2[ACREN]=1.

Address: 4002_B000h base + 28h offset + (4d × i), where i=0d to 1d



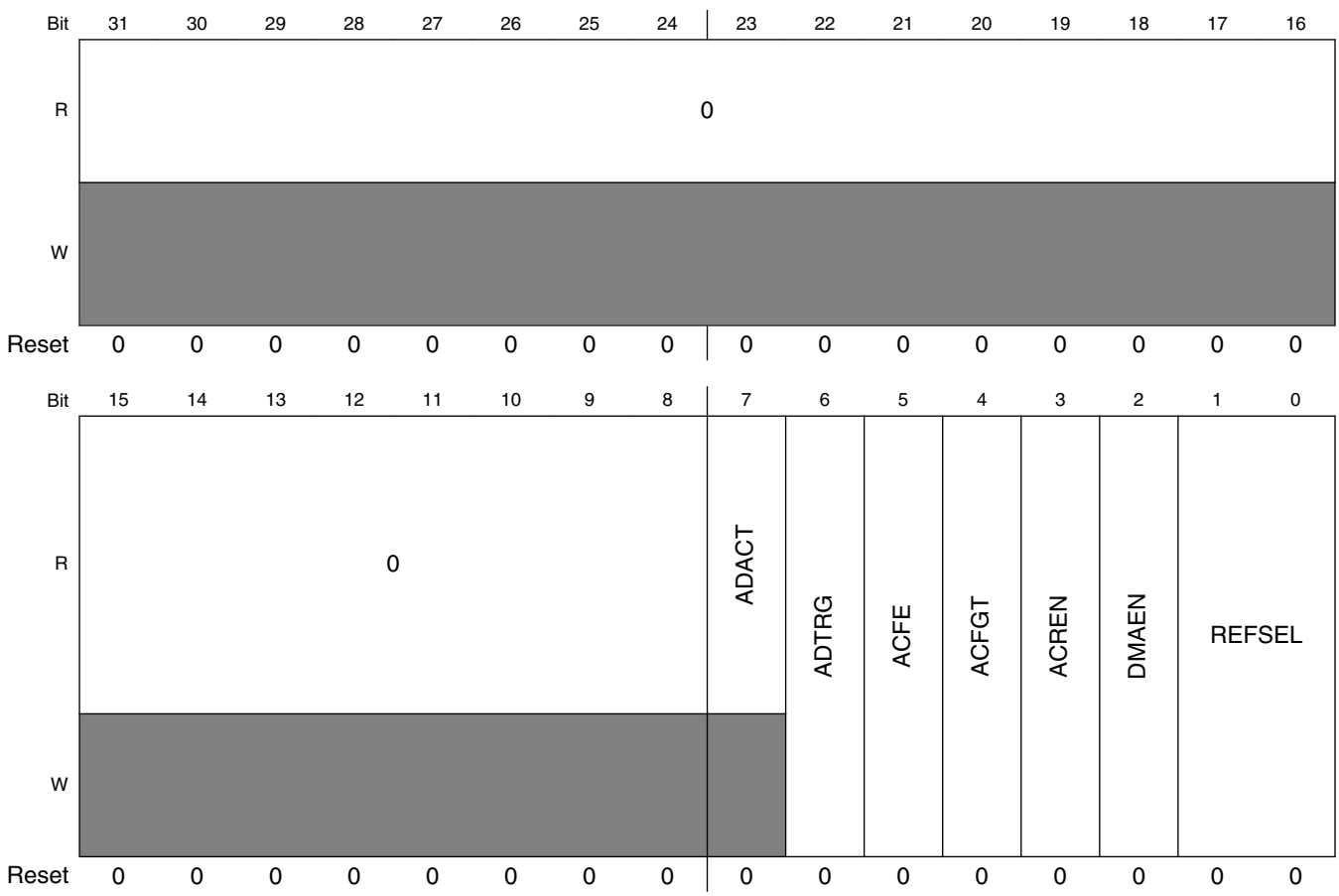
ADC*x*_CV*n* field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CV	Compare Value.

32.4.6 Status and Control Register 2 (ADCx_SC2)

The status and control register 2 (SC2) contains the conversion active, hardware/software trigger select, compare function, and voltage reference select of the ADC module.

Address: 4002_B000h base + 30h offset = 4002_B030h



ADCx_SC2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ADACT	Conversion Active Indicates that a conversion or hardware averaging is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress. 1 Conversion in progress.
6 ADTRG	Conversion Trigger Select Selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: • Software trigger: When software trigger is selected, a conversion is initiated following a write to SC1A. • Hardware trigger: When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input after a pulse of the ADHWTSn input. 0 Software trigger selected. 1 Hardware trigger selected.
5 ACFE	Compare Function Enable Enables the compare function. 0 Compare function disabled. 1 Compare function enabled.
4 ACFGT	Compare Function Greater Than Enable Configures the compare function to check the conversion result relative to the CV1 and CV2 based upon the value of ACREN. ACFE must be set for ACFGT to have any effect. 0 Configures less than threshold, outside range not inclusive and inside range not inclusive; functionality based on the values placed in CV1 and CV2. 1 Configures greater than or equal to threshold, outside and inside ranges inclusive; functionality based on the values placed in CV1 and CV2.
3 ACREN	Compare Function Range Enable Configures the compare function to check if the conversion result of the input being monitored is either between or outside the range formed by CV1 and CV2 determined by the value of ACFGT. ACFE must be set for ACFGT to have any effect. 0 Range function disabled. Only CV1 is compared. 1 Range function enabled. Both CV1 and CV2 are compared.
2 DMAEN	DMA Enable 0 DMA is disabled. 1 DMA is enabled and will assert the ADC DMA request during an ADC conversion complete event noted when any of the SC1n[COCO] flags is asserted.
REFSEL	Voltage Reference Selection Selects the voltage reference source used for conversions. 00 Default voltage reference pin pair, that is, external pins V_{REFH} and V_{REFL}

Table continues on the next page...

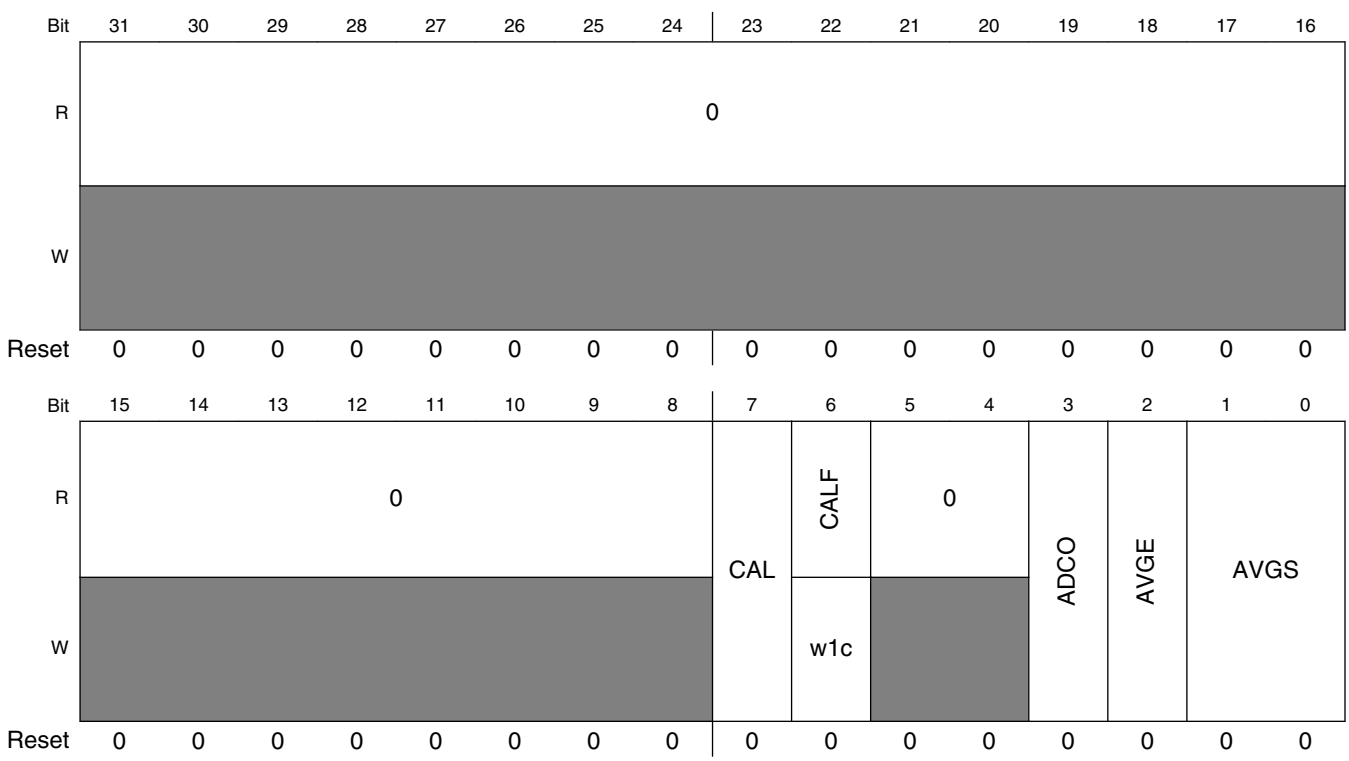
ADCx_SC2 field descriptions (continued)

Field	Description
	01 Alternate reference pair, that is, V_{ALTH} and V_{ALTL} . This pair may be additional external pins or internal sources depending on the MCU configuration. See the chip configuration information for details specific to this MCU 10 Internal bandgap reference and associated ground reference (V_{BGH} and V_{BGL}). Consult the Chip Configuration information for details specific to this MCU. 11 Reserved

32.4.7 Status and Control Register 3 (ADCx_SC3)

The Status and Control Register 3 (SC3) controls the calibration, continuous convert, and hardware averaging functions of the ADC module.

Address: 4002_B000h base + 34h offset = 4002_B034h



ADCx_SC3 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CAL	Calibration Begins the calibration sequence when set. This field stays set while the calibration is in progress and is cleared when the calibration sequence is completed. CALF must be checked to determine the result of the

Table continues on the next page...

ADCx_SC3 field descriptions (continued)

Field	Description
	calibration sequence. Once started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid and CALF will set. Setting CAL will abort any current conversion.
6 CALF	Calibration Failed Flag Displays the result of the calibration sequence. The calibration sequence will fail if SC2[ADTRG] = 1, any ADC register is written, or any stop mode is entered before the calibration sequence completes. Writing 1 to CALF clears it. 0 Calibration completed normally. 1 Calibration failed. ADC accuracy specifications are not guaranteed.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ADCO	Continuous Conversion Enable Enables continuous conversions. 0 One conversion or one set of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion. 1 Continuous conversions or sets of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion.
2 AVGE	Hardware Average Enable Enables the hardware average function of the ADC. 0 Hardware average function disabled. 1 Hardware average function enabled.
AVGS	Hardware Average Select Determines how many ADC conversions will be averaged to create the ADC average result. 00 4 samples averaged. 01 8 samples averaged. 10 16 samples averaged. 11 32 samples averaged.

32.4.8 ADC Offset Correction Register (ADCx_OFS)

The ADC Offset Correction Register (OFS) contains the user-selected or calibration-generated offset error correction value. This register is a 2's complement, left-justified, 16-bit value . The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4002_B000h base + 38h offset = 4002_B038h

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

ADCx_OFS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
OFS	Offset Error Correction Value

32.4.9 ADC Plus-Side Gain Register (ADCx_PG)

The Plus-Side Gain Register (PG) contains the gain error correction for the overall conversion in single-ended mode. PG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between PG[15] and PG[14]. This register must be written by the user with the value described in the calibration procedure. Otherwise, the gain error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4002_B000h base + 3Ch offset = 4002_B03Ch

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

ADCx_PG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PG	Plus-Side Gain

32.4.10 ADC Plus-Side General Calibration Value Register (ADCx_CLPD)

The Plus-Side General Calibration Value Registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLP0[5:0], CLP1[6:0], CLP2[7:0], CLP3[8:0], CLP4[9:0], CLPS[5:0], and CLPD[5:0]. CLPx are automatically set when the self-calibration sequence is done, that is, CAL is cleared. If these registers are written by the user after calibration, the linearity error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4002_B000h base + 44h offset = 4002_B044h

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

ADCx_CLPD field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPD	Calibration Value Calibration Value

32.4.11 ADC Plus-Side General Calibration Value Register (ADCx_CLPS)

For more information, see CLPD register description.

Address: 4002_B000h base + 48h offset = 4002_B048h

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

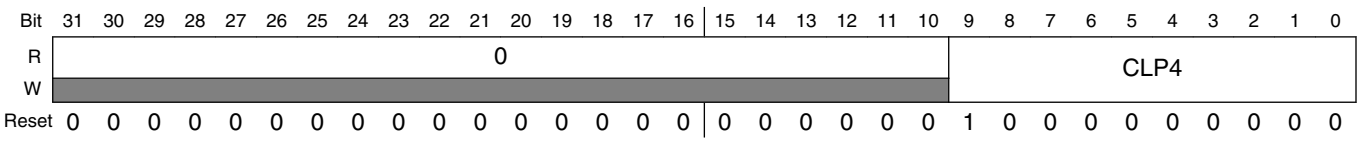
ADCx_CLPS field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPS	Calibration Value Calibration Value

32.4.12 ADC Plus-Side General Calibration Value Register (ADCx_CLP4)

For more information, see CLPD register description.

Address: 4002_B000h base + 4Ch offset = 4002_B04Ch



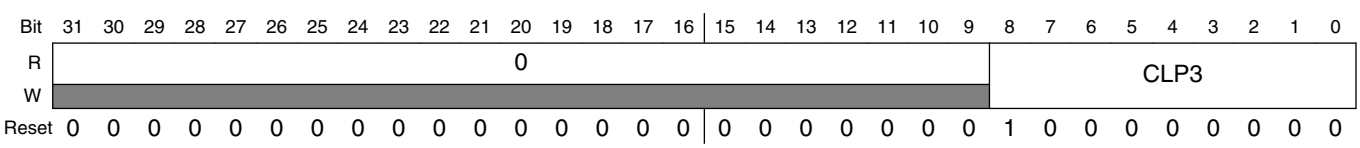
ADCx_CLP4 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP4	Calibration Value Calibration Value

32.4.13 ADC Plus-Side General Calibration Value Register (ADCx_CLP3)

For more information, see CLPD register description.

Address: 4002_B000h base + 50h offset = 4002_B050h



ADCx_CLP3 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

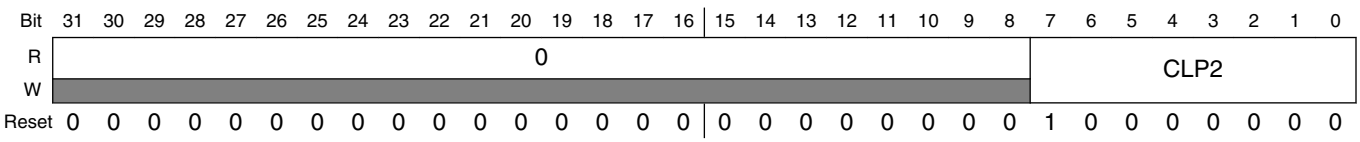
ADCx_CLP3 field descriptions (continued)

Field	Description
CLP3	Calibration Value Calibration Value

32.4.14 ADC Plus-Side General Calibration Value Register (ADCx_CLP2)

For more information, see CLPD register description.

Address: 4002_B000h base + 54h offset = 4002_B054h



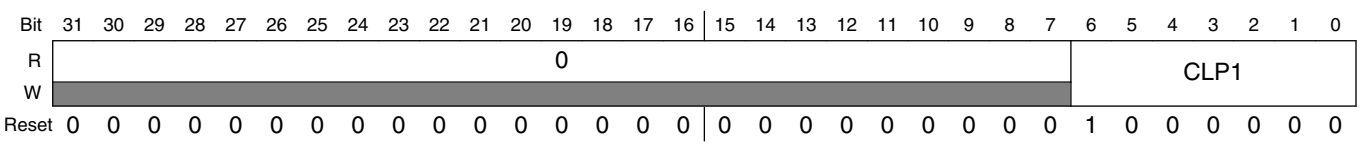
ADCx_CLP2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP2	Calibration Value Calibration Value

32.4.15 ADC Plus-Side General Calibration Value Register (ADCx_CLP1)

For more information, see CLPD register description.

Address: 4002_B000h base + 58h offset = 4002_B058h



ADCx_CLP1 field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP1	Calibration Value Calibration Value

32.4.16 ADC Plus-Side General Calibration Value Register (ADCx_CLP0)

For more information, see CLPD register description.

Address: 4002_B000h base + 5Ch offset = 4002_B05Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CLP0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

ADCx_CLP0 field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP0	Calibration Value Calibration Value

32.5 Functional description

The ADC module is disabled during reset, in Low-Power Stop mode, or when SC1n[ADCH] are all high; see the power management information for details. The module is idle when a conversion has completed and another conversion has not been initiated. When it is idle and the asynchronous clock output enable is disabled, or CFG2[ADACKEN]= 0, the module is in its lowest power state. The ADC can perform an analog-to-digital conversion on any of the software selectable channels. All modes perform conversion by a successive approximation algorithm.

To meet accuracy specifications, the ADC module must be calibrated using the on-chip calibration function.

See Calibration function for details on how to perform calibration.

When the conversion is completed, the result is placed in the Rn data registers. The respective SC1n[COCO] is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, or, when SC1n[AIEN]=1.

The ADC module has the capability of automatically comparing the result of a conversion with the contents of the CV1 and CV2 registers. The compare function is enabled by setting SC2[ACFE] and operates in any of the conversion modes and configurations.

The ADC module has the capability of automatically averaging the result of multiple conversions. The hardware average function is enabled by setting SC3[AVGE] and operates in any of the conversion modes and configurations.

NOTE

For the chip specific modes of operation, see the power management information of this MCU.

32.5.1 Clock select and divide control

One of four clock sources can be selected as the clock source for the ADC module.

This clock source is then divided by a configurable value to generate the input clock ADCK, to the module. The clock is selected from one of the following sources by means of CFG1[ADICLK].

- Bus clock. This is the default selection following reset.
- Bus clock divided by two. For higher bus clock rates, this allows a maximum divide-by-16 of the bus clock using CFG1[ADIV].
- ALTCLK: As defined for this MCU. See the chip configuration information. Conversions are possible using ALTCLK as the input clock source while the MCU is in Normal Stop mode.
- Asynchronous clock (ADACK): This clock is generated from a clock source within the ADC module. When the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start CFG2[ADACKEN]=0, ADACK is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. To avoid the conversion time variability and latency associated with the ADACK clock startup, set CFG2[ADACKEN]=1 and wait the worst-case startup time of 5 μs prior to initiating any conversions using the ADACK clock source. Conversions are possible using ADACK as the input clock source while the MCU is in Normal Stop mode. See Power Control for more information.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC may not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by CFG1[ADIV] and can be divide-by 1, 2, 4, or 8.

32.5.2 Voltage reference selection

The ADC can be configured to accept one of the two voltage reference pairs as the reference voltage (V_{REFSH} and V_{REFSL}) used for conversions.

Each pair contains a positive reference that must be between the minimum Ref Voltage High and V_{DDA}, and a ground reference that must be at the same potential as V_{SSA}. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using SC2[REFSEL]. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

32.5.3 Hardware trigger and channel selects

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when SC2[ADTRG] is set and a hardware trigger select event, ADHWTSn, has occurred.

This source is not available on all MCUs. See the chip-specific ADC information for information on the ADHWT source and the ADHWTSn configurations specific to this MCU.

When an ADHWT source is available and hardware trigger is enabled, that is SC2[ADTRG]=1, a conversion is initiated on the rising-edge of ADHWT after a hardware trigger select event, that is, ADHWTSn, has occurred. If a conversion is in progress when a rising-edge of a trigger occurs, the rising-edge is ignored. In continuous convert configuration, only the initial rising-edge to launch continuous conversions is observed, and until conversion is aborted, the ADC continues to do conversions on the same SCn register that initiated the conversion. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

The hardware trigger select event, ADHWTSn, must be set prior to the receipt of the ADHWT signal. If these conditions are not met, the converter may ignore the trigger or use the incorrect configuration. If a hardware trigger select event is asserted during a conversion, it must stay asserted until the end of current conversion and remain set until the receipt of the ADHWT signal to trigger a new conversion. The channel and status fields selected for the conversion depend on the active trigger select signal:

- ADHWTSA active selects SC1A.
- ADHWTSn active selects SC1n.

Note

Asserting more than one hardware trigger select signal (ADHWTSn) at the same time results in unknown results. To avoid this, select only one hardware trigger select signal (ADHWTSn) prior to the next intended conversion.

When the conversion is completed, the result is placed in the Rn registers associated with the ADHWTSn received. For example:

- ADHWTSA active selects RA register
- ADHWTSn active selects Rn register

The conversion complete flag associated with the ADHWTSn received, that is, SC1n[COCO], is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, that is, SC1[AIEN]=1.

32.5.4 Conversion control

Conversions can be performed as determined by CFG1[MODE] as shown in the description of CFG1[MODE].

Conversions can be initiated by a software or hardware trigger.

In addition, the ADC module can be configured for:

- Low-power operation
- Long sample time
- Continuous conversion
- Hardware average
- Automatic compare of the conversion result to a software determined compare value

32.5.4.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, with SC1n[ADCH] not all 1's, if software triggered operation is selected, that is, when SC2[ADTRG]=0.
- Following a hardware trigger, or ADHWT event, if hardware triggered operation is selected, that is, SC2[ADTRG]=1, and a hardware trigger select event, ADHWTSn, has occurred. The channel and status fields selected depend on the active trigger select signal:
  - ADHWTSA active selects SC1A.

- ADHWTSn active selects SC1n.
- if neither is active, the off condition is selected

Note

Selecting more than one ADHWTSn prior to a conversion completion will result in unknown results. To avoid this, select only one ADHWTSn prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled, that is, when SC3[ADCO] = 1.

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, that is, when SC2[ADTRG] = 0, continuous conversions begin after SC1A is written and continue until aborted. In hardware triggered operation, that is, when SC2[ADTRG] = 1 and one ADHWTSn event has occurred, continuous conversions begin after a hardware trigger event and continue until aborted.

If hardware averaging is enabled, a new conversion is automatically initiated after the completion of the current conversion until the correct number of conversions are completed. In software triggered operation, conversions begin after SC1A is written. In hardware triggered operation, conversions begin after a hardware trigger. If continuous conversions are also enabled, a new set of conversions to be averaged are initiated following the last of the selected number of conversions.

32.5.4.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, Rn. If the compare functions are disabled, this is indicated by setting of SC1n[COCO]. If hardware averaging is enabled, the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed. If the compare function is enabled, the respective SC1n[COCO] sets and conversion result data is transferred only if the compare condition is true. If both hardware averaging and compare functions are enabled, then the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed and the compare condition is true. An interrupt is generated if the respective SC1n[AIEN] is high at the time that the respective SC1n[COCO] is set.

32.5.4.3 Aborting conversions

Any conversion in progress is aborted when:

- Writing to SC1A while it is actively controlling a conversion, aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, a write to SC1A initiates a new conversion if SC1A[ADCH] is equal to a value other than all 1s. Writing to any of the SC1B–SC1n registers while that specific SC1B–SC1n register is actively controlling a conversion aborts the current conversion. The SC1(B-n) registers are not used for software trigger operation and therefore writes to the SC1(B-n) registers do not initiate a new conversion.
- A write to any ADC register besides the SC1A-SC1n registers occurs. This indicates that a change in mode of operation has occurred and the current conversion is therefore invalid.
- The MCU is reset or enters Low-Power Stop modes.
- The MCU enters Normal Stop mode with ADACK or Alternate Clock Sources not enabled.

When a conversion is aborted, the contents of the data registers, Rn, are not altered. The data registers continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset or Low-Power Stop modes, RA and Rn return to their reset states.

32.5.4.4 Power control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, but the asynchronous clock output is disabled, that is CFG2[ADACKEN]=0, the ADACK clock generator also remains in its idle state (disabled) until a conversion is initiated. If the asynchronous clock output is enabled, that is, CFG2[ADACKEN]=1, it remains active regardless of the state of the ADC or the MCU power mode.

Power consumption when the ADC is active can be reduced by setting CFG1[ADLPC]. This results in a lower maximum value for f_{ADCK}.

32.5.4.5 Sample time and total conversion time

For short sample, that is, when CFG1[ADLSMP]=0, there is a 2-cycle adder for first conversion over the base sample time of four ADCK cycles. For high-speed conversions, that is, when CFG2[ADHSC]=1, there is an additional 2-cycle adder on any conversion. The table below summarizes sample times for the possible ADC configurations.

ADC configuration			Sample time (ADCK cycles)	
CFG1[ADLSMP]	CFG2[ADLSTS]	CFG2[ADHSC]	First or Single	Subsequent
0	X	0	6	4
1	00	0	24	
1	01	0	16	
1	10	0	10	
1	11	0	6	
0	X	1	8	6
1	00	1	26	
1	01	1	18	
1	10	1	12	
1	11	1	8	

The total conversion time depends upon:

- The sample time as determined by CFG1[ADLSMP] and CFG2[ADLSTS]
- The MCU bus frequency
- The conversion mode, as determined by CFG1[MODE]
- The high-speed configuration, that is, CFG2[ADHSC]
- The frequency of the conversion clock, that is, f_{ADCK}.

CFG2[ADHSC] is used to configure a higher clock input frequency. This will allow faster overall conversion times. To meet internal ADC timing requirements, CFG2[ADHSC] adds additional ADCK cycles. Conversions with CFG2[ADHSC]=1 take two more ADCK cycles. CFG2[ADHSC] must be used when the ADCLK exceeds the limit for CFG2[ADHSC]=0.

After the module becomes active, sampling of the input begins.

1. CFG1[ADLSMP] and CFG2[ADLSTS] select between sample times based on the conversion mode that is selected.
2. When sampling is completed, the converter is isolated from the input channel and a successive approximation algorithm is applied to determine the digital value of the analog signal.
3. The result of the conversion is transferred to Rn upon completion of the conversion algorithm.

If the bus frequency is less than f_{ADCK}, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled, that is, when CFG1[ADLSMP]=0.

The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by CFG1[ADICLK], and the divide ratio is specified by CFG1[ADIV].

The maximum total conversion time for all configurations is summarized in the equation below. See the following tables for the variables referenced in the equation.

$$\text{ConversionTime} = \text{SFCAdder} + \text{AverageNum} \times (\text{BCT} + \text{LSTAdder} + \text{HSCAdder})$$

Equation 1. Conversion time equation

Table 32-6. Single or first continuous time adder (SFCAdder)

CFG1[ADLSMP]	CFG2[ADACKEN]	CFG1[ADICLK]	Single or first continuous time adder (SFCAdder)
1	x	0x, 10	3 ADCK cycles + 5 bus clock cycles
1	1	11	3 ADCK cycles + 5 bus clock cycles[1]
1	0	11	5 µs + 3 ADCK cycles + 5 bus clock cycles
0	x	0x, 10	5 ADCK cycles + 5 bus clock cycles
0	1	11	5 ADCK cycles + 5 bus clock cycles[1]
0	0	11	5 µs + 5 ADCK cycles + 5 bus clock cycles

1. To achieve this time, CFG2[ADACKEN] must be 1 for at least 5 µs prior to the conversion is initiated.

Table 32-7. Average number factor (AverageNum)

SC3[AVGE]	SC3[AVGS]	Average number factor (AverageNum)
0	xx	1
1	00	4
1	01	8
1	10	16
1	11	32

Table 32-8. Base conversion time (BCT)

Mode	Base conversion time (BCT)
8b single-ended	17 ADCK cycles
10b single-ended	20 ADCK cycles
12b single-ended	20 ADCK cycles
16b single-ended	25 ADCK cycles

Table 32-9. Long sample time adder (LSTAdder)

CFG1[ADLSMP]	CFG2[ADLSTS]	Long sample time adder (LSTAdder)
0	xx	0 ADCK cycles
1	00	20 ADCK cycles
1	01	12 ADCK cycles
1	10	6 ADCK cycles
1	11	2 ADCK cycles

Table 32-10. High-speed conversion time adder (HSCAdder)

CFG2[ADHSC]	High-speed conversion time adder (HSCAdder)
0	0 ADCK cycles
1	2 ADCK cycles

Note

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

32.5.4.6 Conversion time examples

The following examples use the Equation 1 on page 606, and the information provided in Table 32-6 through Table 32-10.

32.5.4.6.1 Typical conversion time configuration

A typical configuration for ADC conversion is:

- 10-bit mode, with the bus clock selected as the input clock source
- The input clock divide-by-1 ratio selected
- Bus frequency of 8 MHz
- Long sample time disabled
- High-speed conversion disabled

The conversion time for a single conversion is calculated by using the Equation 1 on page 606, and the information provided in Table 32-6 through Table 32-10. The table below lists the variables of Equation 1 on page 606.

Table 32-11. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles

Table continues on the next page...

Table 32-11. Typical conversion time (continued)

Variable	Time
AverageNum	1
BCT	20 ADCK cycles
LSTAdder	0
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for a bus clock and an ADCK frequency equal to 8 MHz, the resulting conversion time is 3.75 µs.

32.5.4.6.2 Long conversion time configuration

A configuration for long ADC conversion is:

- The input clock divide-by-8 ratio selected
- Bus frequency of 8 MHz
- Long sample time enabled
- Configured for longest adder
- High-speed conversion disabled
- Average enabled for 32 conversions

The conversion time for this conversion is calculated by using the Equation 1 on page 606, and the information provided in Table 32-6 through Table 32-10. The following table lists the variables of the Equation 1 on page 606.

Table 32-12. Typical conversion time

Variable	Time
SFCAdder	3 ADCK cycles + 5 bus clock cycles
AverageNum	32
BCT	34 ADCK cycles
LSTAdder	20 ADCK cycles
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for bus clock equal to 8 MHz and ADCK equal to 1 MHz, the resulting conversion time is 57.625 µs, that is, AverageNum. This results in a total conversion time of 1.844 ms.

32.5.4.6.3 Short conversion time configuration

A configuration for short ADC conversion is:

- 8-bit Single-Ended mode with the bus clock selected as the input clock source

- The input clock divide-by-1 ratio selected
- Bus frequency of 20 MHz
- Long sample time disabled
- High-speed conversion enabled

The conversion time for this conversion is calculated by using the Equation 1 on page 606, and the information provided in Table 32-6 through Table 32-10. The table below lists the variables of Equation 1 on page 606.

Table 32-13. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	17 ADCK cycles
LSTAdder	0 ADCK cycles
HSCAdder	2

The resulting conversion time is generated using the parameters listed in in the preceding table. Therefore, for bus clock and ADCK frequency equal to 20 MHz, the resulting conversion time is 1.45 µs.

32.5.4.7 Hardware average function

The hardware average function can be enabled by setting SC3[AVGE]=1 to perform a hardware average of multiple conversions. The number of conversions is determined by the AVGS[1:0] bits, which can select 4, 8, 16, or 32 conversions to be averaged. While the hardware average function is in progress, SC2[ADACT] will be set.

After the selected input is sampled and converted, the result is placed in an accumulator from which an average is calculated once the selected number of conversions have been completed. When hardware averaging is selected, the completion of a single conversion will not set SC1n[COCO].

If the compare function is either disabled or evaluates true, after the selected number of conversions are completed, the average conversion result is transferred into the data result registers, Rn, and SC1n[COCO] is set. An ADC interrupt is generated upon the setting of SC1n[COCO] if the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

Note

The hardware average function can perform conversions on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the hardware average is completed if SC1n[AIEN] is set.

32.5.5 Automatic compare function

The compare function can be configured to check whether the result is less than or greater-than-or-equal-to a single compare value, or, if the result falls within or outside a range determined by two compare values.

The compare mode is determined by SC2[ACFGT], SC2[ACREN], and the values in the compare value registers, CV1 and CV2. After the input is sampled and converted, the compare values in CV1 and CV2 are used as described in the following table. There are six Compare modes as shown in the following table.

Table 32-14. Compare modes

SC2[ACFGT]	SC2[ACREN]	ADCCV1 relative to ADCCV2	Function	Compare mode description
0	0	—	Less than threshold	Compare true if the result is less than the CV1 registers.
1	0	—	Greater than or equal to threshold	Compare true if the result is greater than or equal to CV1 registers.
0	1	Less than or equal	Outside range, not inclusive	Compare true if the result is less than CV1 **Or** the result is greater than CV2.
0	1	Greater than	Inside range, not inclusive	Compare true if the result is less than CV1 **And** the result is greater than CV2.
1	1	Less than or equal	Inside range, inclusive	Compare true if the result is greater than or equal to CV1 **And** the result is less than or equal to CV2.
1	1	Greater than	Outside range, inclusive	Compare true if the result is greater than or equal to CV1 **Or** the result is less than or equal to CV2.

With SC2[ACREN] =1, and if the value of CV1 is less than or equal to the value of CV2, then setting SC2[ACFGT] will select a trigger-if-inside-compare-range inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-outside-compare-range, not-inclusive-of-endpoints function.

If CV1 is greater than CV2, setting SC2[ACFGT] will select a trigger-if-outside-compare-range, inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-inside-compare-range, not-inclusive-of-endpoints function.

If the condition selected evaluates true, SC1n[COCO] is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, SC1n[COCO] is not set and the conversion result data will not be transferred to the result register, Rn. If the hardware averaging function is enabled, the compare function compares the averaged result to the compare values. The same compare function definitions apply. An ADC interrupt is generated when SC1n[COCO] is set and the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

Note

The compare function can monitor the voltage on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the compare condition is met.

32.5.6 Calibration function

The ADC contains a self-calibration function that is required to achieve the specified accuracy.

Calibration must be run, or valid calibration values written, after any reset and before a conversion is initiated. The calibration function sets the offset calibration value and the plus-side calibration values. The offset calibration value is automatically stored in the ADC offset correction register (OFS), and the plus-side calibration values are automatically stored in the ADC plus-side calibration registers, CLPx. The user must configure the ADC correctly prior to calibration, and must generate the plus-side gain calibration results and store them in the ADC plus-side gain register (PG) after the calibration function completes.

Prior to calibration, the user must configure the ADC's clock source and frequency, low power configuration, voltage reference selection, sample time, and high speed configuration according to the application's clock source availability and needs. If the application uses the ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected, or multiple calibrations can be done for the different configurations. For best calibration results:

- Set hardware averaging to maximum, that is, SC3[AVGE]=1 and SC3[AVGS]=11 for an average of 32
- Set ADC clock frequency f_{ADCK} less than or equal to 4 MHz
- $V_{REFH}=V_{DDA}$
- Calibrate at nominal voltage and temperature

The input channel, conversion mode continuous function, compare function, resolution mode, and single-ended mode are all ignored during the calibration function.

To initiate calibration, the user sets SC3[CAL] and the calibration will automatically begin if the SC2[ADTRG] is 0. If SC2[ADTRG] is 1, SC3[CAL] will not get set and SC3[CALF] will be set. While calibration is active, no ADC register can be written and no stop mode may be entered, or the calibration routine will be aborted causing SC3[CAL] to clear and SC3[CALF] to set. At the end of a calibration sequence, SC1n[COCO] will be set. SC1n[AIEN] can be used to allow an interrupt to occur at the end of a calibration sequence. At the end of the calibration routine, if SC3[CALF] is not set, the automatic calibration routine is completed successfully.

To complete calibration, the user must generate the gain calibration values using the following procedure:

1. Initialize or clear a 16-bit variable in RAM.
2. Add the plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable.
3. Divide the variable by two.
4. Set the MSB of the variable.
5. The previous two steps can be achieved by setting the carry bit, rotating to the right through the carry bit on the high byte and again on the low byte.
6. Store the value in the plus-side gain calibration register PG.

When calibration is complete, the user may reconfigure and use the ADC as desired. A second calibration may also be performed, if desired, by clearing and again setting SC3[CAL].

Overall, the calibration routine may take as many as 14k ADCK cycles and 100 bus cycles, depending on the results and the clock source chosen. For an 8 MHz clock source, this length amounts to about 1.7 ms. To reduce this latency, the calibration values, which are offset, plus-side gain, and plus-side calibration values, may be stored in flash memory after an initial calibration and recovered prior to the first ADC conversion. This method can reduce the calibration latency to 20 register store operations on all subsequent power, reset, or Low-Power Stop mode recoveries.

Further information on the calibration procedure can be found in the Calibration section of AN3949: ADC16 Calibration Procedure and Programmable Delay Block Synchronization.

32.5.7 User-defined offset function

OFS contains the user-selected or calibration-generated offset error correction value.

This register is a 2’s complement, left-justified. The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

The formatting of the OFS is different from the data result register, Rn, to preserve the resolution of the calibration value regardless of the conversion mode selected. Lower order bits are ignored in lower resolution modes. For example, in 8-bit single-ended mode, OFS[14:7] are subtracted from D[7:0]; OFS[15] indicates the sign (negative numbers are effectively added to the result) and OFS[6:0] are ignored. In 16-bit single-ended mode, there is no field in the OFS corresponding to the least significant result D[0], so odd values, such as -1 or +1, cannot be subtracted from the result.

OFS is automatically set according to calibration requirements once the self-calibration sequence is done, that is, SC3[CAL] is cleared. The user may write to OFS to override the calibration result if desired. If the OFS is written by the user to a value that is different from the calibration value, the ADC error specifications may not be met. Storing the value generated by the calibration function in memory before overwriting with a user-specified value is recommended.

Note

There is an effective limit to the values of offset that can be set by the user. If the magnitude of the offset is too high, the results of the conversions will cap off at the limits.

The offset calibration function may be employed by the user to remove application offsets or DC bias values. OFS may be written with a number in 2's complement format and this offset will be subtracted from the result, or hardware averaged value. To add an offset, store the negative offset in 2's complement format and the effect will be an addition. An offset correction that results in an out-of-range value will be forced to the minimum or maximum value. The minimum value for single-ended conversions is 0x0000.

To preserve accuracy, the calibrated offset value initially stored in OFS must be added to the user-defined offset. For applications that may change the offset repeatedly during operation, store the initial offset calibration value in flash so it can be recovered and added to any user offset adjustment value and the sum stored in OFS.

32.5.8 Temperature sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs.

The following equation provides an approximate transfer function of the temperature sensor.

$$Temp = 25 - \left(\left(V_{TEMP} - V_{TEMP25} \right) \div m \right)$$

Equation 2. Approximate transfer function of the temperature sensor

where:

- V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.
- V_{TEMP25} is the voltage of the temperature sensor channel at 25 °C.
- m is referred as temperature sensor slope in the device data sheet. It is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the V_{TEMP25} and temperature sensor slope values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP}, and compares to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in the preceding equation. If V_{TEMP} is less than V_{TEMP25}, the hot slope value is applied in the preceding equation. ADC Electricals table may only specify one temperature sensor slope value. In that case, the user could use the same slope for the calculation across the operational temperature range.

For more information on using the temperature sensor, see the application note titled *Temperature Sensor for the HCS08 Microcontroller Family* (document AN3031).

32.5.9 MCU wait mode operation

Wait mode is a lower-power consumption Standby mode from which recovery is fast because the clock sources remain active.

If a conversion is in progress when the MCU enters Wait mode, it continues until completion. Conversions can be initiated while the MCU is in Wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two; and ADACK are available as conversion clock sources while in Wait mode. The use of ALTCLK as the conversion clock source in Wait is dependent on the definition of ALTCLK for this MCU. See the Chip Configuration information on ALTCLK specific to this MCU.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Wait mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare trigger is not met, the ADC will return to its idle state and cannot wake the MCU from Wait mode unless a new conversion is initiated by the hardware trigger.

32.5.10 MCU Normal Stop mode operation

Stop mode is a low-power consumption Standby mode during which most or all clock sources on the MCU are disabled.

32.5.10.1 Normal Stop mode with ADACK disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a stop instruction aborts the current conversion and places the ADC in its Idle state. The contents of the ADC registers, including Rn, are unaffected by Normal Stop mode. After exiting from Normal Stop mode, a software or hardware trigger is required to resume conversions.

32.5.10.2 Normal Stop mode with ADACK enabled

If ADACK is selected as the conversion clock, the ADC continues operation during Normal Stop mode. See the chip-specific ADC information for configuration information for this device.

If a conversion is in progress when the MCU enters Normal Stop mode, it continues until completion. Conversions can be initiated while the MCU is in Normal Stop mode by means of the hardware trigger or if continuous conversions are enabled.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Normal Stop mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. The result register, Rn, will contain the data from the first completed conversion that occurred during Normal Stop mode. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an

interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare is not true, the ADC will return to its idle state and cannot wake the MCU from Normal Stop mode unless a new conversion is initiated by another hardware trigger.

32.5.11 MCU Low-Power Stop mode operation

The ADC module is automatically disabled when the MCU enters Low-Power Stop mode.

All module registers contain their reset values following exit from Low-Power Stop mode. Therefore, the module must be re-enabled and re-configured following exit from Low-Power Stop mode.

NOTE

For the chip specific modes of operation, see the power management information for the device.

32.6 Initialization information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module.

The user can configure the module for 16-bit, 12-bit, 10-bit, or 8-bit single-ended resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. For information used in this example, refer to Table 32-9, Table 32-10, and Table 32-11.

Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

32.6.1 ADC module initialization example

32.6.1.1 Initialization sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is:

1. Calibrate the ADC by following the calibration instructions in Calibration function.
2. Update CFG to select the input clock source and the divide ratio used to generate ADCK. This register is also used for selecting sample time and low-power configuration.
3. Update SC2 to select the conversion trigger, hardware or software, and compare function options, if enabled.
4. Update SC3 to select whether conversions will be continuous or completed only once (ADCO) and whether to perform hardware averaging.
5. Update SC1:SC1n registers to enable or disable conversion complete interrupts. Also, select the input channel which can be used to perform conversions.

32.6.1.2 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low-power with a long sample time on input channel 1, where ADCK is derived from the bus clock divided by 1.

CFG1 = 0x98 (%10011000)

```
Bit 7   ADLPC   1      Configures for low power, lowers maximum clock speed.
Bit 6:5 ADIV    00     Sets the ADCK to the input clock ÷ 1.
Bit 4   ADLSMP  1      Configures for long sample time.
Bit 3:2   MODE   10    Selects the single-ended 10-bit conversion.
Bit 1:0   ADICLK   00    Selects the bus clock.
```

SC2 = 0x00 (%00000000)

```
Bit 7   ADACT   0       Flag indicates if a conversion is in progress.
Bit 6   ADTRG   0       Software trigger selected.
Bit 5   ACFE    0       Compare function disabled.
Bit 4   ACFGT   0       Not used in this example.
Bit 3   ACREN   0      Compare range disabled.
 Bit 2 DMAEN 0 DMA request disabled.
Bit 1:0 REFSEL  00     Selects default voltage reference pin pair (External pins
VREFH and VREFL).
```

SC1A = 0x41 (%01000001)

```
Bit 7   COCO    0       Read-only flag which is set when a conversion completes.
Bit 6   AIEN    1       Conversion complete interrupt enabled.

Bit 4:0 ADCH    00001   Input channel 1 selected as ADC input channel.
```

RA = 0xxx

Holds results of conversion.

CV = 0xxx

Holds compare value when compare function enabled.

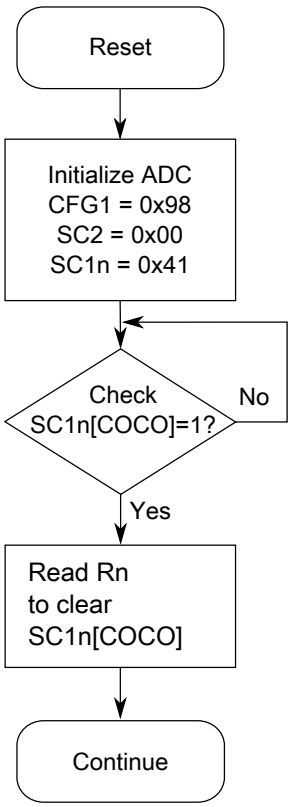


Figure 32-2. Initialization flowchart example

32.7 Application information

The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an ADC.

For guidance on selecting optimum external component values and converter parameters see AN4373: Cookbook for SAR ADC Measurements.

32.7.1 External pins and routing

32.7.1.1 Analog supply pins

Depending on the device, the analog power and ground supplies, V_{DDA} and V_{SSA}, of the ADC module are available as:

- V_{DDA} and V_{SSA} available as separate pins—When available on a separate pin, both V_{DDA} and V_{SSA} must be connected to the same voltage potential as their corresponding MCU digital supply, V_{DD} and V_{SS}, and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.
- V_{SSA} is shared on the same pin as the MCU digital V_{SS}.
- V_{SSA} and V_{DDA} are shared with the MCU digital supply pins—In these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSA} pin. This must be the only ground connection between these supplies, if possible. V_{SSA} makes a good single point ground location.

32.7.1.2 Analog voltage reference pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs used by the converter:

- V_{REFSH} is the high reference voltage for the converter.
- V_{REFSL} is the low reference voltage for the converter.

The ADC can be configured to accept one of two voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference and a ground reference. The two pairs are external, V_{REFH} and V_{REFL} and alternate, V_{ALTH} and V_{ALTL}. These voltage references are selected using SC2[REFSEL]. The alternate voltage reference pair, V_{ALTH} and V_{ALTL}, may select additional external pins or internal sources based on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

In some packages, the external or alternate pairs are connected in the package to V_{DDA} and V_{SSA}, respectively. One of these positive references may be shared on the same pin as V_{DDA} on some devices. One of these ground references may be shared on the same pin as V_{SSA} on some devices.

If externally available, the positive reference may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential. The positive reference must never exceed V_{DDA}. If externally available, the ground reference must be connected to the same voltage potential as V_{SSA}. The voltage reference pairs must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high-frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum, that is, parasitic only.

32.7.1.3 Analog input pins

The external analog inputs are typically shared with digital I/O pins on MCU devices.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used, they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to 0xFFF, which is full scale 12-bit representation, 0x3FF, which is full scale 10-bit representation, or 0xFF, which is full scale 8-bit representation. If the input is equal to or less than V_{REFL}, the converter circuit converts it to 0x000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There is a brief current associated with V_{REFL} when the sampling capacitor is charging.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins must not be transitioning during conversions.

32.7.2 Sources of error

32.7.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy.

RAS + RADIN =SC / (FMAX * NUMTAU * CADIN)

Figure 32-3. Sampling equation

Where:

RAS = External analog source resistance

SC = Number of ADCK cycles used during sample window

CADIN = Internal ADC input capacitance

NUMTAU = -ln(LSBERR / 2^N)

LSBERR = value of acceptable sampling error in LSBs

N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode or 16 in 16-bit mode

Higher source resistances or higher-accuracy sampling is possible by setting CFG1[ADLSMP] and changing CFG2[ADLSTS] to increase the sample window, or decreasing ADCK frequency to increase sample time.

32.7.2.2 Pin leakage error

Leakage on the I/O pins can cause conversion error if the external analog source resistance, R_{AS}, is high. If this error cannot be tolerated by the application, keep R_{AS} lower than V_{REFH} / ($4 \times I_{LEAK} \times 2^N$) for less than 1/4 LSB leakage error, where N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode, or 16 in 16-bit mode.

32.7.2.3 Noise-induced errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 μF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 μF low-ESR capacitor from V_{DDA} to V_{SSA}.
- If inductive isolation is used from the primary supply, an additional 1 μF capacitor is placed from V_{DDA} to V_{SSA}.
- V_{SSA}, and V_{REFL}, if connected, is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in Wait or Normal Stop mode before initiating (hardware-triggered conversions) or immediately after initiating (hardware- or software-triggered conversions) the ADC conversion.

- For software triggered conversions, immediately follow the write to SC1 with a Wait instruction or Stop instruction.
- For Normal Stop mode operation, select ADACK as the clock source. Operation in Normal Stop reduces V_{DD} noise but increases effective conversion time due to stop recovery.

- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in Wait or Normal Stop mode, or I/O activity cannot be halted, the following actions may reduce the effect of noise on the accuracy:

- Place a 0.01 µF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSA}. This improves noise issues, but affects the sample rate based on the external analog source resistance.
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1 LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock, that is, ADACK, and averaging. Noise that is synchronous to ADCK cannot be averaged out.

32.7.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 65536 steps in the 16-bit mode.. Each step ideally has the same height, that is, 1 code, and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N-bit converter, where N can be 16, 12, 10, or 8, defined as 1 LSB, is:

$$1LSB=\left(V_{REFH}\right)/2^{N}$$

Equation 3. Ideal code width for an N-bit converter

There is an inherent quantization error due to the digitization of the result. For 8-bit, 10-bit, or 12-bit conversions, the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2 LSB in 8-bit, 10-bit, or 12-bit modes. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 LSB and the code width of the last (0xFF or 0x3FF) is 1.5 LSB.

For 16-bit conversions, the code transitions only after the full code width is present, so the quantization error is -1 LSB to 0 LSB and the code width of each step is 1 LSB.

32.7.2.5 Linearity errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors, but the system designers must be aware of these errors because they affect overall accuracy:

- Zero-scale error (E_{ZS}), sometimes called offset: This error is defined as the difference between the actual code width of the first conversion and the ideal code width. This is 1/2 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 LSB) is used.
- Full-scale error (E_{FS}): This error is defined as the difference between the actual code width of the last conversion and the ideal code width. This is 1.5 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1 LSB) is used.
- Differential non-linearity (DNL): This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL): This error is defined as the highest-value or absolute value that the running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE): This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

32.7.2.6 Code jitter, non-monotonicity, and missing codes

Analog-to-digital converters are susceptible to three special forms of error:

- Code jitter: Code jitter occurs when a given input voltage converts to one of the two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code, and vice-versa. However, even small amounts of system noise can cause the converter to be indeterminate, between two codes, for a range of input voltages around the transition voltage.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally, the techniques discussed in Noise-induced errors reduces this error.

- Non-monotonicity: Non-monotonicity occurs when, except for code jitter, the converter converts to a lower code for a higher input voltage.
- Missing codes: Missing codes are those values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.

Chapter 33
Comparator (CMP)

33.1 Chip-specific CMP information

33.1.1 Overview

The high speed analog comparator system contains the following:

- A high speed CMOS comparator (HSCMP) with a programmable reference input.
- A 6-bit DAC with a 64-tap resistor ladder network, which provides a selectable voltage reference for applications where voltage reference is needed for internal connection to the HSCMP.
- An analog MUX (ANMUX) that provides a circuit for selecting an analog input signal from eight channels.

33.1.2 Instantiation Information

KM3x_256 microcontrollers has 3 high speed comparator modules with an integrated 6-bit DAC and analog mux. The CMP's 6-bit DAC sub-block supports selection of two voltage references. For KM3x_256 devices, the references implemented are:

- CMP V_{in1} is connected to VREF Output;
- CMP V_{in2} is connected to V_{DD}.

Table 33-1. CMP instantiation

Comparator Features	CMP0	CMP1	CMP2
Number of 6-bit DACs	1	1	1
Analog mux size	8 bits	8 bits	8 bits
Number of CMP OUT pins	1	1	1
External Comparator Inputs	up to 6	up to 6	up to 6

NOTE

Refer to Device Configuration for the number of external inputs supported on each device.

The following table shows the fixed internal and external connections to each CMP and output connections for each CMP instance, respectively. For information about the specific pins to which the input and output signals are assigned on the package, refer to the KM3x_256 Family Pinouts chapter.

NOTE

The "Reserved for Test" input is to be connected to the Analog Test Sense Bus signal.

Table 33-2. CMP Input connections

CMP inputs	CMP0	CMP1	CMP2
CMP Input0	CMP0P0	CMP1P0	CMP2P0
CMP Input1	CMP0P1	CMP1P1	CMP2P1
CMP Input2	CMP0P2	SDADP2	CMP2P2
CMP Input3	CMP0P3	SDADM2	CMP2P3
CMP Input4	CMP0P4	SDADP3	CMP2P4
CMP Input5	CMP0P5	SDADM3	CMP2P5
CMP Input6	Reserved for Test	Reserved for Test	Reserved for Test
CMP Input7	6-bit DAC reference	6-bit DAC reference	6-bit DAC reference

CMP0 output connections are:

- CMP output (CMP0OUT) pin
- Quad Timer input via XBAR
- Any UART Rx input via XBAR (one UART can work at a time)
- EWM_IN input via XBAR
- LLWU_M2IF input
- LPTMR input

CMP1 output connections are:

- CMP output (CMP1OUT) pin
- Quad Timer input via XBAR
- LLWU_M4IF input
- EWM_IN input via XBAR
- LPTMR input

CMP2 output connections are:

- CMP output (CMP2OUT) pin
- Quad Timer input via XBAR

- EWM_IN input via XBAR
- LLWU_M5IF input

CMP0 is used for IrDA operation via UART. CMP1 and CMP2 are used for zero crossing detection (for frequency measurement).

33.2 Introduction

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 6-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference V_{in} into 64 voltage levels. A 6-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/64$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}. The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

33.2.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
    - Sampled
    - Windowed, which is ideal for certain PWM zero-crossing-detection applications
    - Digitally filtered:

    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:
    - Shorter propagation delay at the expense of higher power
    - Low power, with longer propagation delay
- DMA transfer support
    - A comparison event can be selected to trigger a DMA transfer
- Functional in all modes of operation
- The window and filter functions are not available in the following modes:
    - Stop
    - VLPS
    - VLLSx

33.2.2 6-bit DAC key features

The 6-bit DAC has the following features:

- 6-bit resolution
- Selectable supply reference source
- Power Down mode to conserve power when not in use
- Option to route the output to internal comparator input

33.2.3 ANMUX key features

The ANMUX has the following features:

- Two 8-to-1 channel mux
- Operational over the entire supply range

33.2.4 CMP, DAC and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

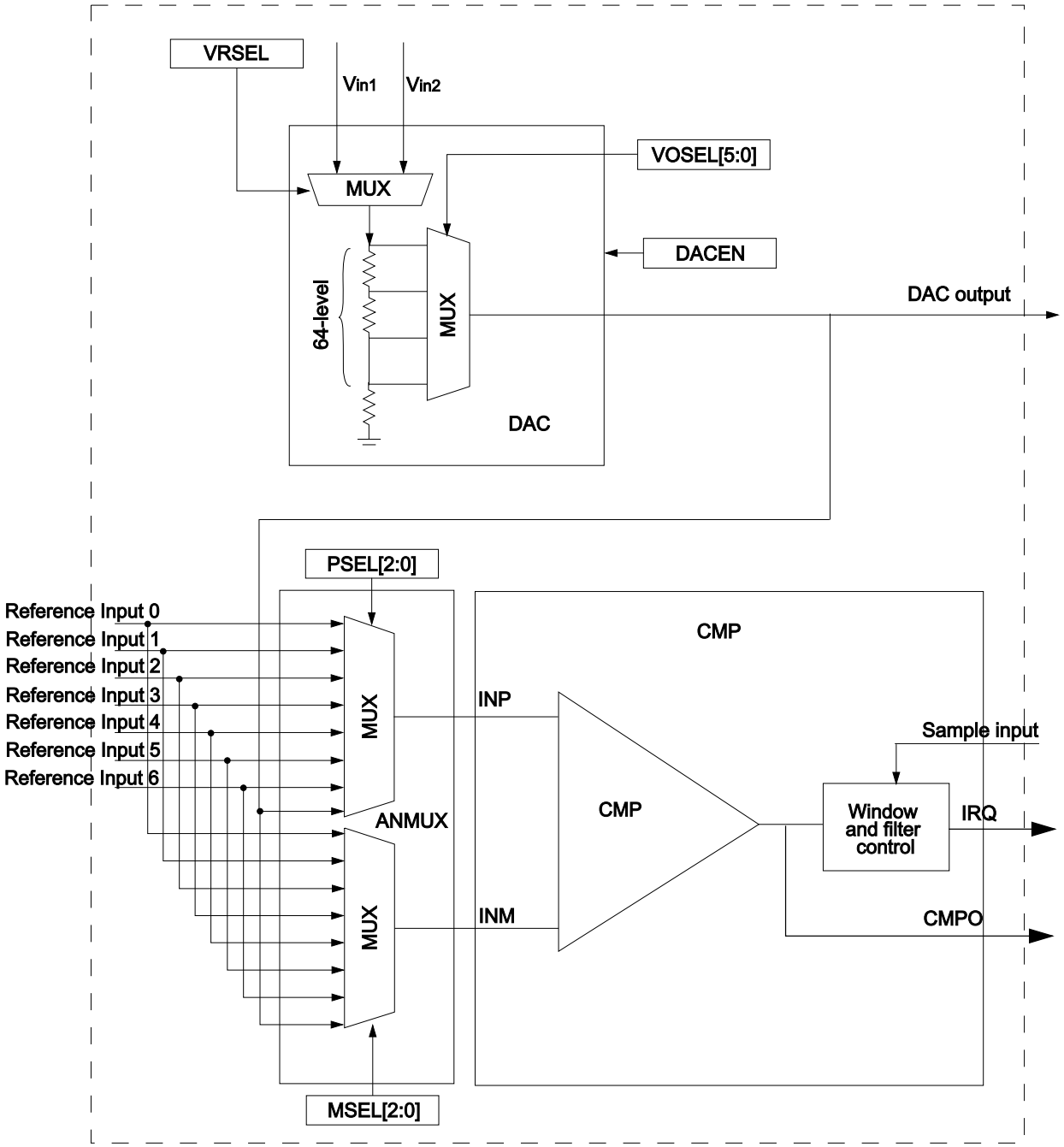


Figure 33-1. CMP, DAC and ANMUX block diagram

33.2.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

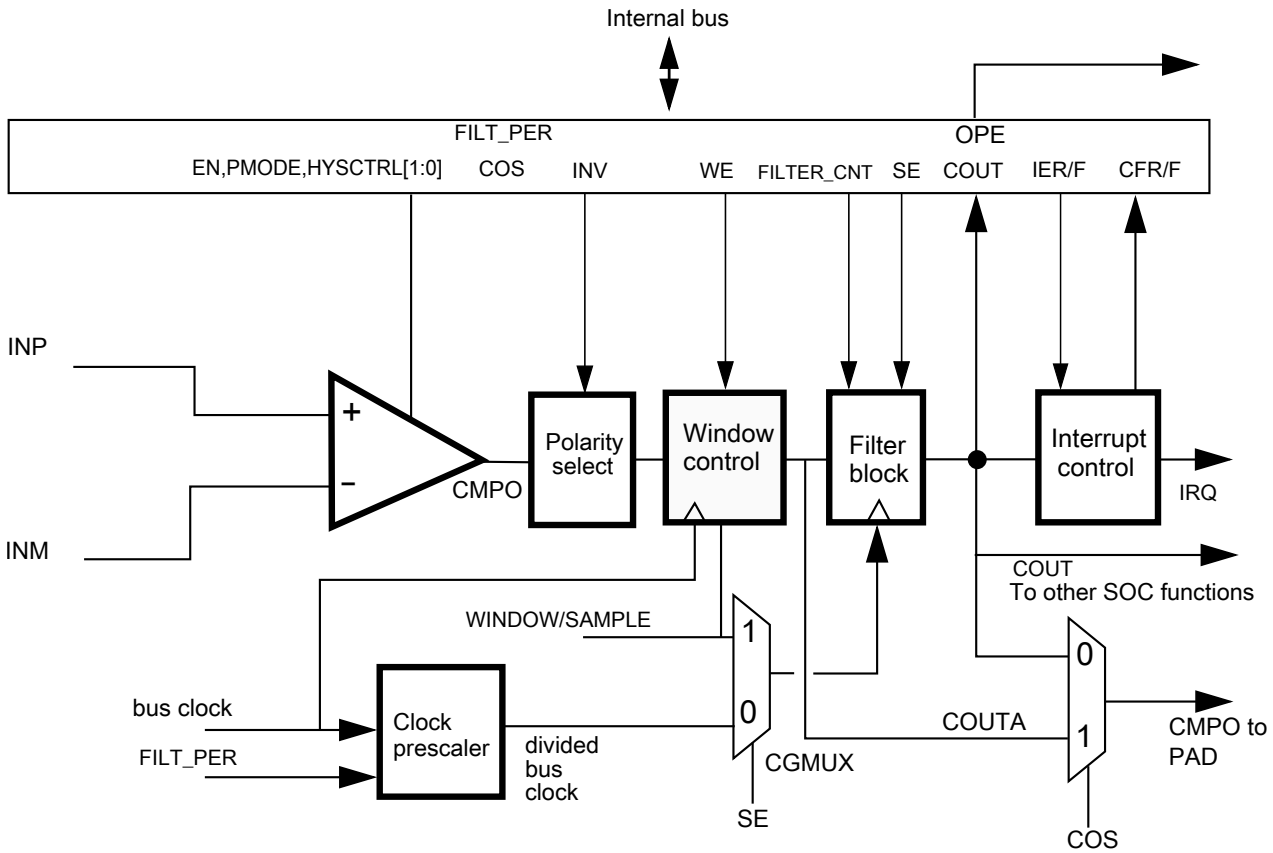


Figure 33-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.
- The Filter block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - If CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which is crossing clock domain boundaries, must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

33.3 Memory map/register definitions

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_2000	CMP Control Register 0 (CMP0_CR0)	8	R/W	00h	33.3.1/631
4007_2001	CMP Control Register 1 (CMP0_CR1)	8	R/W	00h	33.3.2/632
4007_2002	CMP Filter Period Register (CMP0_FPR)	8	R/W	00h	33.3.3/634
4007_2003	CMP Status and Control Register (CMP0_SCR)	8	R/W	00h	33.3.4/634
4007_2004	DAC Control Register (CMP0_DACCR)	8	R/W	00h	33.3.5/635
4007_2005	MUX Control Register (CMP0_MUXCR)	8	R/W	00h	33.3.6/636
4007_2008	CMP Control Register 0 (CMP1_CR0)	8	R/W	00h	33.3.1/631
4007_2009	CMP Control Register 1 (CMP1_CR1)	8	R/W	00h	33.3.2/632
4007_200A	CMP Filter Period Register (CMP1_FPR)	8	R/W	00h	33.3.3/634
4007_200B	CMP Status and Control Register (CMP1_SCR)	8	R/W	00h	33.3.4/634
4007_200C	DAC Control Register (CMP1_DACCR)	8	R/W	00h	33.3.5/635
4007_200D	MUX Control Register (CMP1_MUXCR)	8	R/W	00h	33.3.6/636
4007_2010	CMP Control Register 0 (CMP2_CR0)	8	R/W	00h	33.3.1/631
4007_2011	CMP Control Register 1 (CMP2_CR1)	8	R/W	00h	33.3.2/632
4007_2012	CMP Filter Period Register (CMP2_FPR)	8	R/W	00h	33.3.3/634
4007_2013	CMP Status and Control Register (CMP2_SCR)	8	R/W	00h	33.3.4/634
4007_2014	DAC Control Register (CMP2_DACCR)	8	R/W	00h	33.3.5/635
4007_2015	MUX Control Register (CMP2_MUXCR)	8	R/W	00h	33.3.6/636

33.3.1 CMP Control Register 0 (CMPx_CR0)

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0	0	HYSTCTR	
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count Represents the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency, see the Functional description. 000 Filter is disabled. If SE = 1, then COUT is a logic 0. This is not a legal state, and is not recommended. If SE = 0, COUT = COUTA. 001 One sample must agree. The comparator output is simply sampled. 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level are device-specific. See the Data Sheet of the device for the exact values. 00 Level 0 01 Level 1 10 Level 2 11 Level 3

33.3.2 CMP Control Register 1 (CMPx_CR1)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SE	WE	TRIGM	PMODE	INV	COS	OPE	EN
Reset	0	0	0	0	0	0	0	0

CMPx_CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. 0 Sampling mode is not selected. 1 Sampling mode is selected.

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

Field	Description
6 WE	Windowing Enable At any given time, either SE or WE can be set. 0 Windowing mode is not selected. 1 Windowing mode is selected.
5 TRIGM	Trigger Mode Enable CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1. In addition, the CMP should be enabled. If the DAC is to be used as a reference to the CMP, it should also be enabled. CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare. Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received. See the chip configuration for details about the external timer resource. 0 Trigger mode is disabled. 1 Trigger mode is enabled.
4 PMODE	Power Mode Select See the electrical specifications table in the device Data Sheet for details. 0 Low-Speed (LS) Comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption. 1 High-Speed (HS) Comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption.
3 INV	Comparator INVERT Allows selection of the polarity of the analog comparator function. It is also driven to the COUT output, on both the device pin and as SCR[COUT], when OPE=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select 0 Set the filtered comparator output (CMPO) to equal COUT. 1 Set the unfiltered comparator output (CMPO) to equal COUTA.
1 OPE	Comparator Output Pin Enable 0 CMPO is not available on the associated CMPO output pin. 1 CMPO is available on the associated CMPO output pin.
0 EN	Comparator Module Enable Enables the Analog Comparator module. When the module is not enabled, it remains in the off state, and consumes no power. When the user selects the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.

33.3.3 CMP Filter Period Register (CMPx_FPR)

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

CMPx_FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when CR1[SE]=0. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional description. This field has no effect when CR1[SE]=1. In that case, the external SAMPLE signal is used to determine the sampling period.

33.3.4 CMP Status and Control Register (CMPx_SCR)

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

CMPx_SCR field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DMAEN	DMA Enable Control Enables the DMA transfer triggered from the CMP module. When this field is set, a DMA request is asserted when CFR or CFF is set. 0 DMA is disabled. 1 DMA is enabled.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set.

Table continues on the next page...

CMPx_SCR field descriptions (continued)

Field	Description
	0 Interrupt is disabled. 1 Interrupt is enabled.
3 IEF	Comparator Interrupt Enable Falling Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 CFR	Analog Comparator Flag Rising Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is level sensitive . 0 Rising-edge on COUT has not been detected. 1 Rising-edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is level sensitive . 0 Falling-edge on COUT has not been detected. 1 Falling-edge on COUT has occurred.
0 COUT	Analog Comparator Output Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as CR1[INV] when the Analog Comparator module is disabled, that is, when CR1[EN] = 0. Writes to this field are ignored.

33.3.5 DAC Control Register (CMPx_DACCR)

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read Write	DACEN	VRSEL	VOSEL					
Reset	0	0	0	0	0	0	0	0

CMPx_DACCR field descriptions

Field	Description
7 DACEN	DAC Enable Enables the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
6 VRSEL	Supply Voltage Reference Source Select

Table continues on the next page...

CMPx_DACCR field descriptions (continued)

Field	Description
	0 V_{in1} is selected as resistor ladder network supply reference. 1 V_{in2} is selected as resistor ladder network supply reference.
VOSEL	DAC Output Voltage Select Selects an output voltage from one of 64 distinct levels. `DACO = (V in /64) * (VOSEL[5:0] + 1)` , so the DACO range is from V_{in} /64 to V_{in} .

33.3.6 MUX Control Register (CMPx_MUXCR)

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	Reserved	0	PSEL			MSEL		
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_MUXCR field descriptions

Field	Description
7 Reserved	Bit can be programmed to zero only . This field is reserved.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–3 PSEL	Plus Input Mux Control Determines which input is selected for the plus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7
MSEL	Minus Input Mux Control Determines which input is selected for the minus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0

Table continues on the next page...

CMPx_MUXCR field descriptions (continued)

Field	Description
	001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

33.4 Functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM.

CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

SCR[IER] and SCR[IEF] are used to select the condition which will cause the CMP module to assert an interrupt to the processor. SCR[CFF] is set on a falling-edge and SCR[CFR] is set on rising-edge of the comparator output. The optionally filtered CMPO can be read directly through SCR[COUT].

33.4.1 CMP functional modes

There are the following main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, CR0[FILTER_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 33-3. Comparator sample/filter controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** See the Continuous mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** See the Sampled, Non-Filtered mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode** See the Sampled, Filtered mode (#s 4A & 4B).
4B	1	0	0	> 0x01	> 0x00	
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA. See the Windowed mode (#s 5A & 5B).
5B	1	1	0	X	0x00	
6	1	1	0	0x01	0x01–0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by FILT_PER to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. See the Windowed/Filtered mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal.						

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings must be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This resets the filter to a known state.

33.4.1.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

33.4.1.2 Continuous mode (#s 2A & 2B)

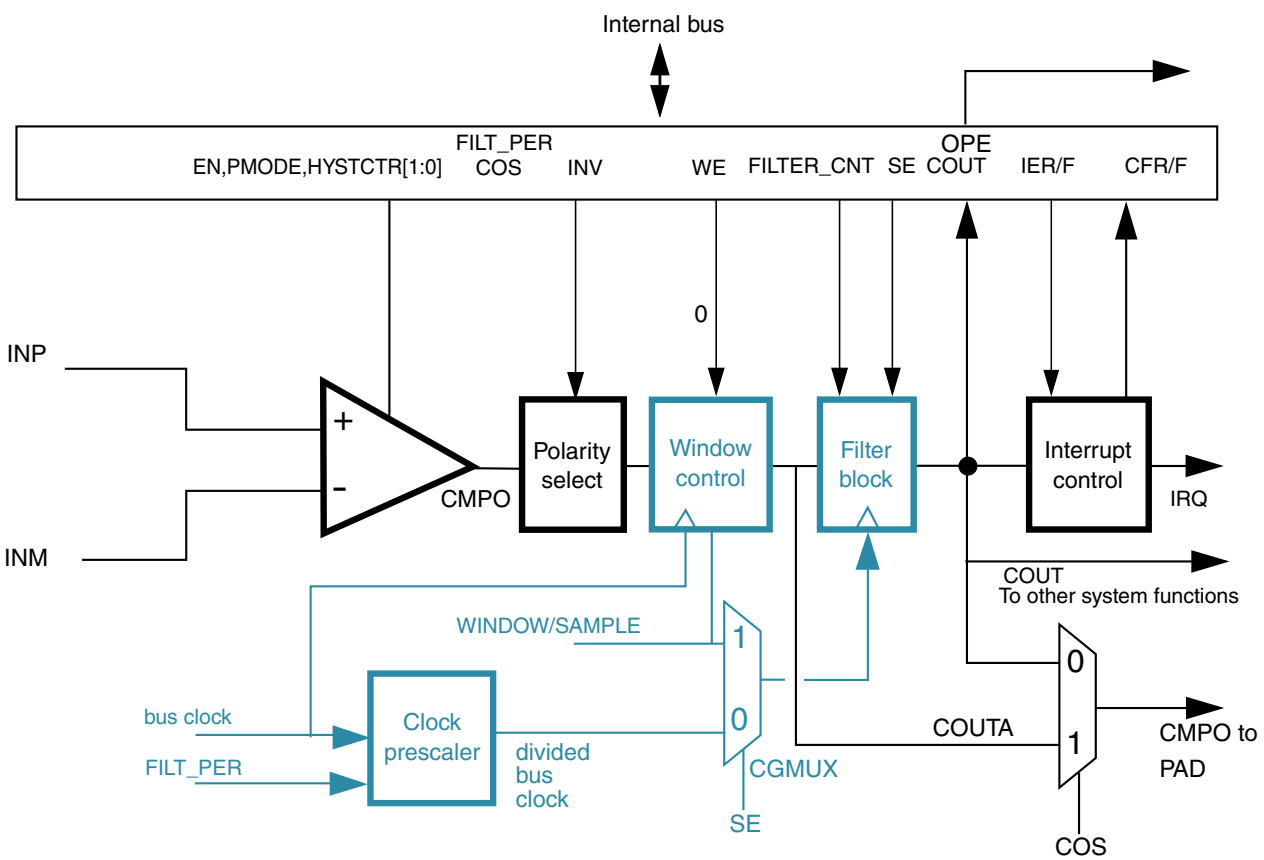


Figure 33-3. Comparator operation in Continuous mode

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations which result in disabling the filter block, see the Filter Block Bypass Logic diagram.

33.4.1.3 Sampled, Non-Filtered mode (#s 3A & 3B)

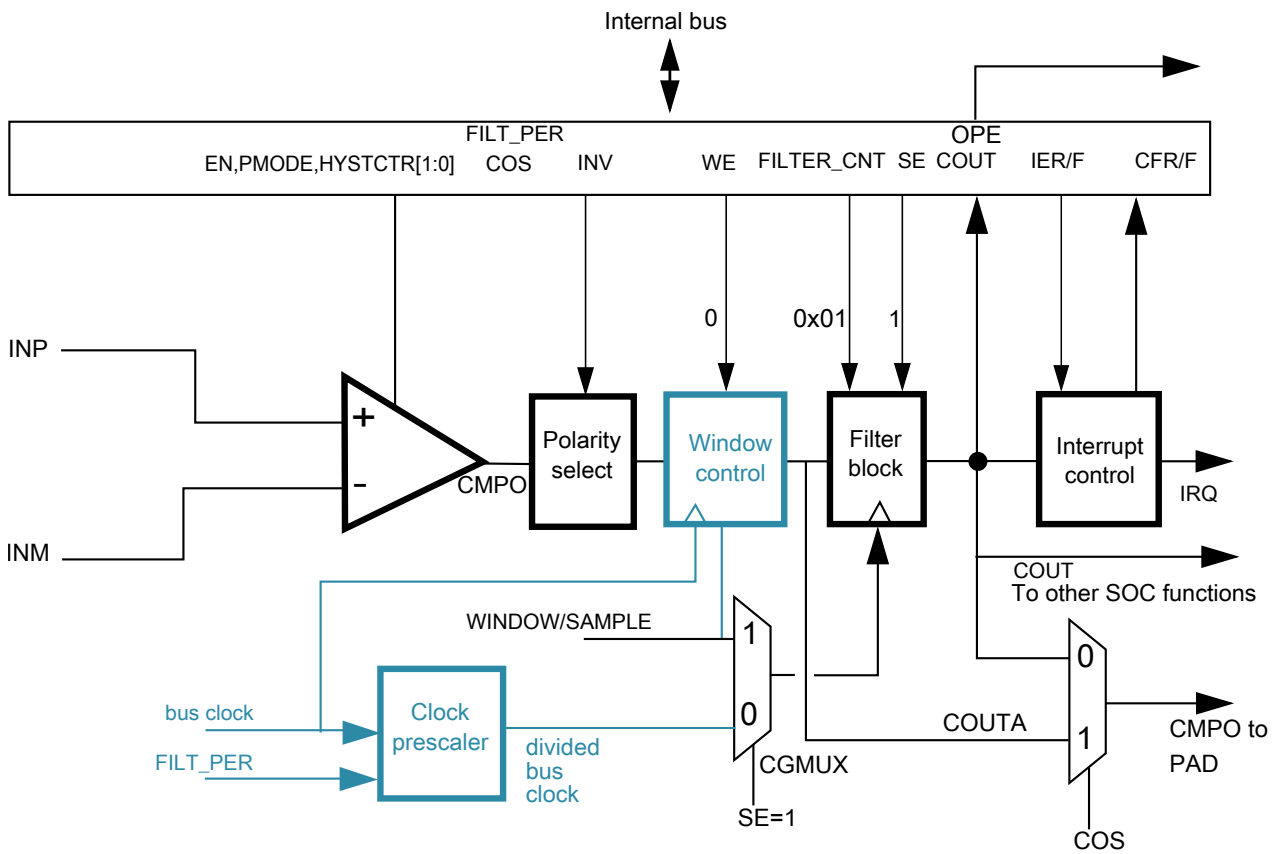


Figure 33-4. Sampled, Non-Filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising-edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

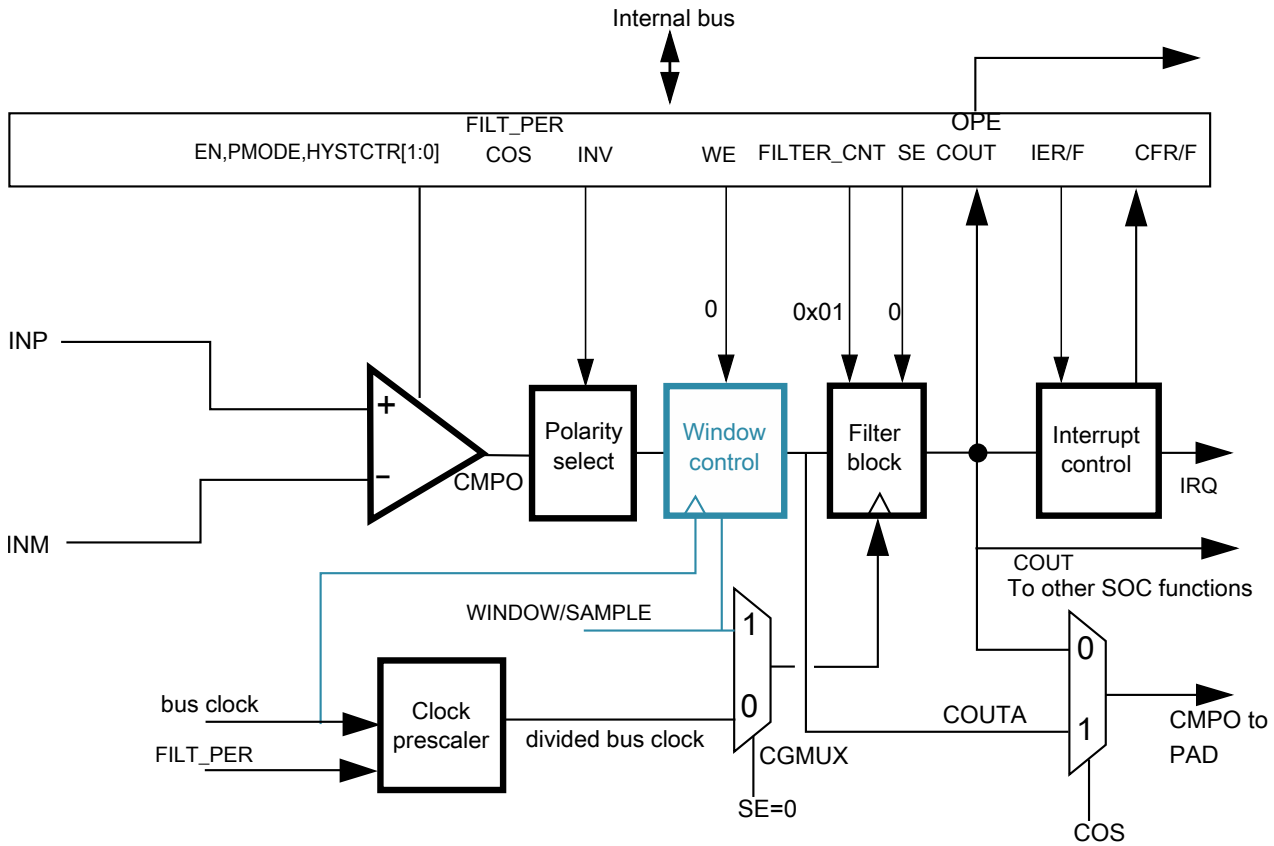


Figure 33-5. Sampled, Non-Filtered (# 3B): sampling interval internally derived

33.4.1.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, CR0[FILTER_CNT]>1, which activates filter operation.

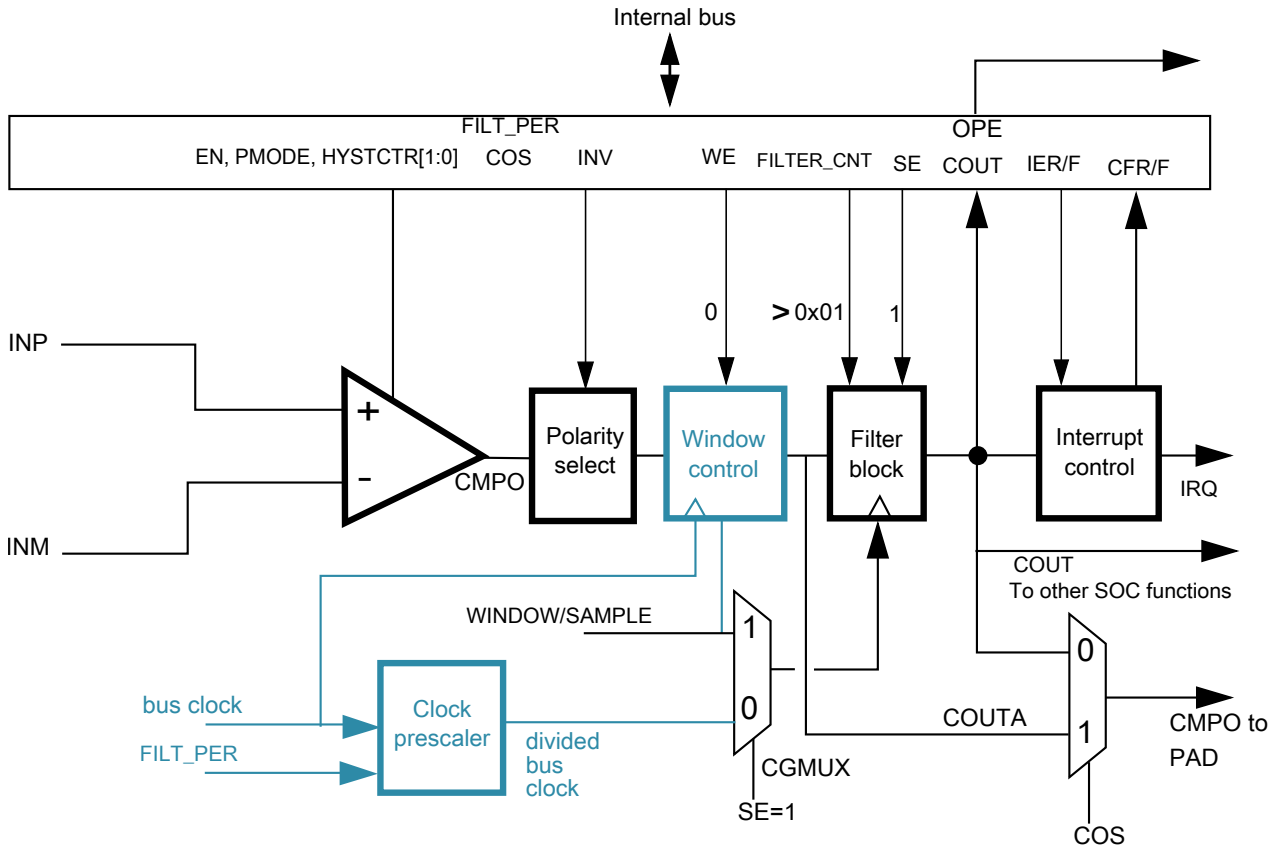


Figure 33-6. Sampled, Filtered (# 4A): sampling point externally driven

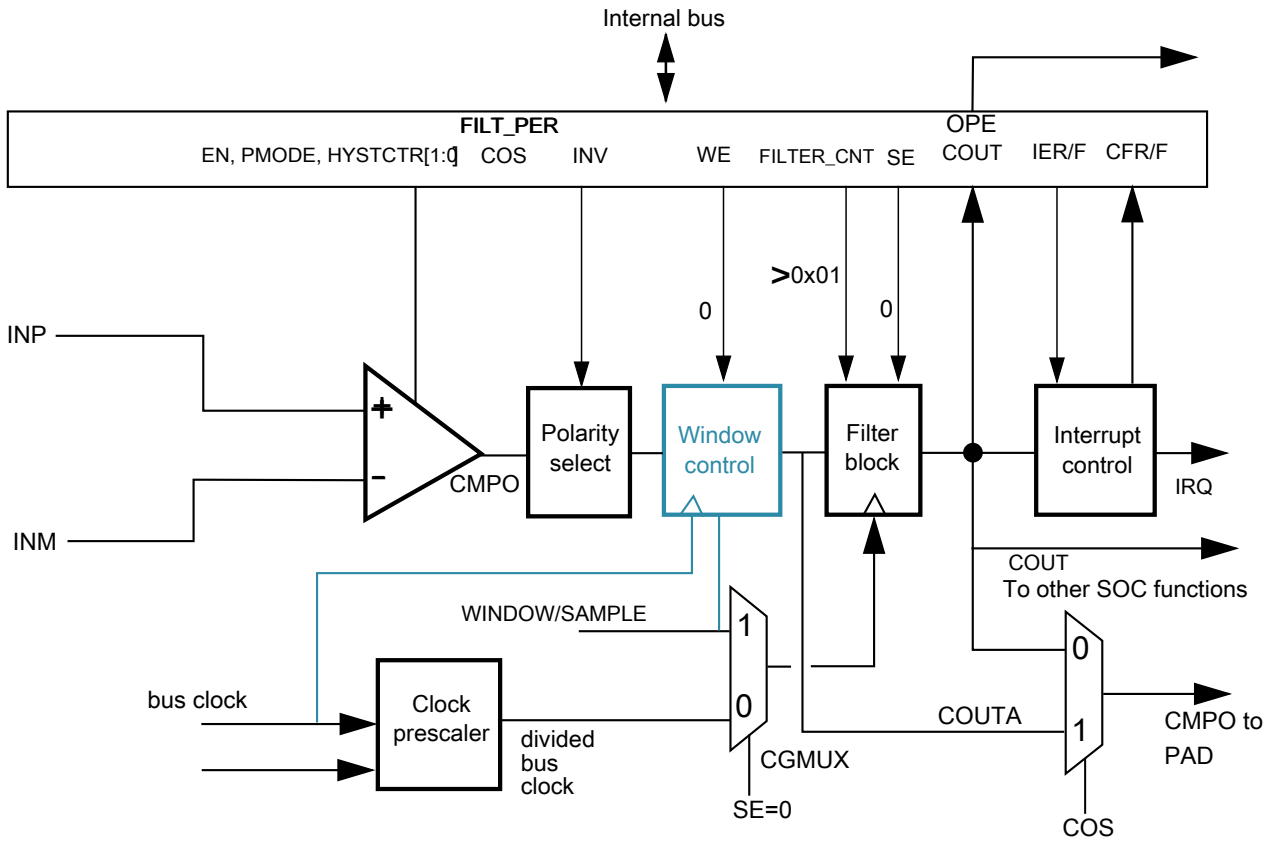


Figure 33-7. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, CR0[FILTER_CNT]>1, which activates filter operation.

33.4.1.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

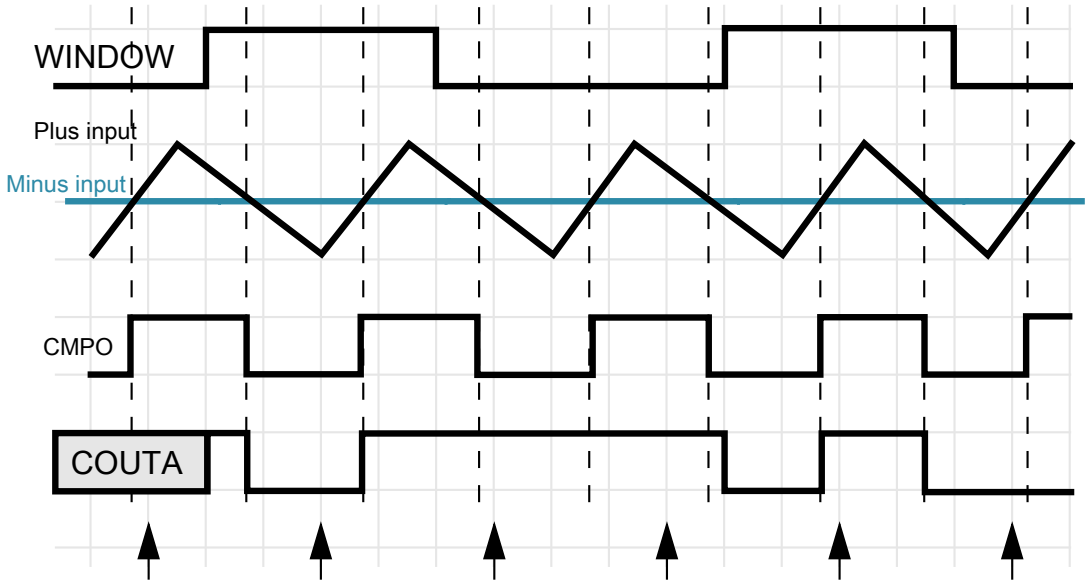


Figure 33-8. Windowed mode operation

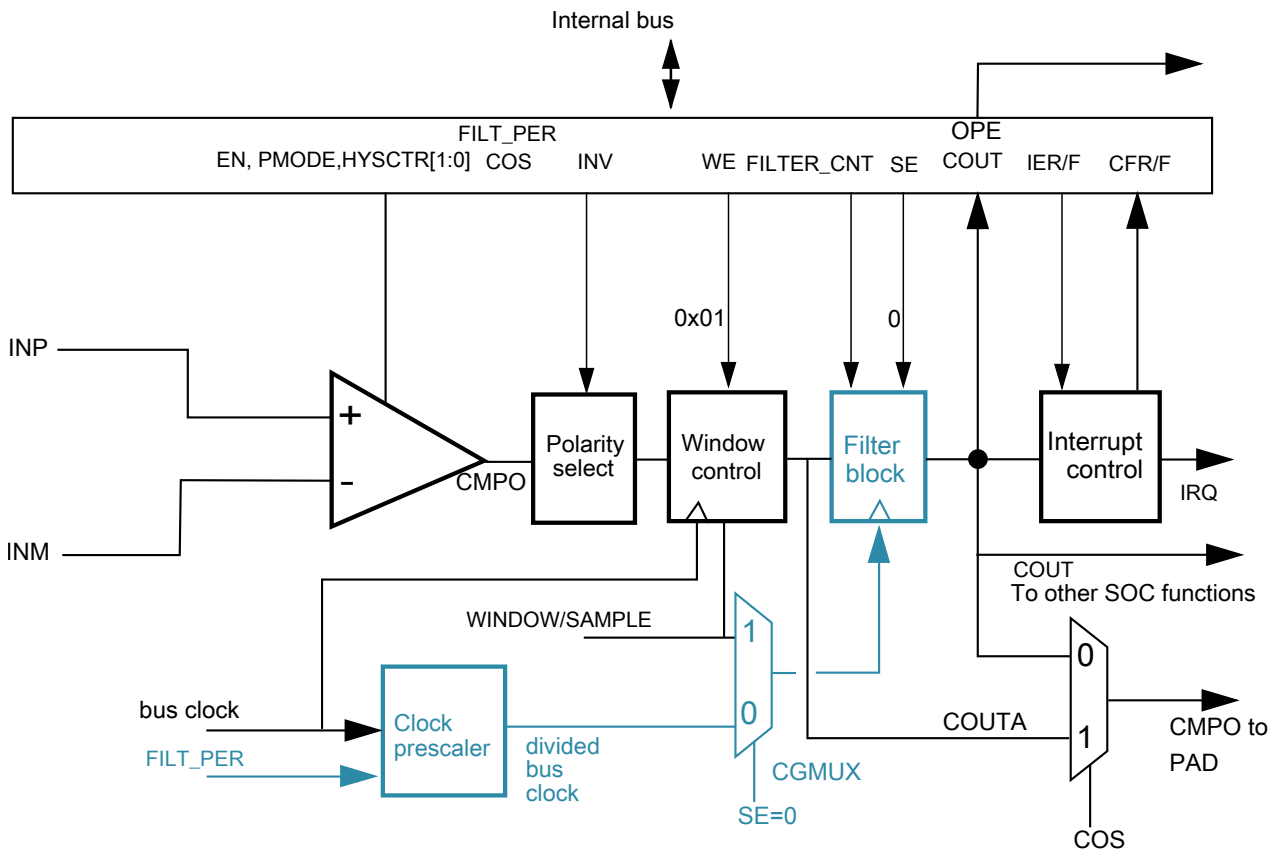


Figure 33-9. Windowed mode

For control configurations which result in disabling the filter block, see Filter Block Bypass Logic diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

33.4.1.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 33-8, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

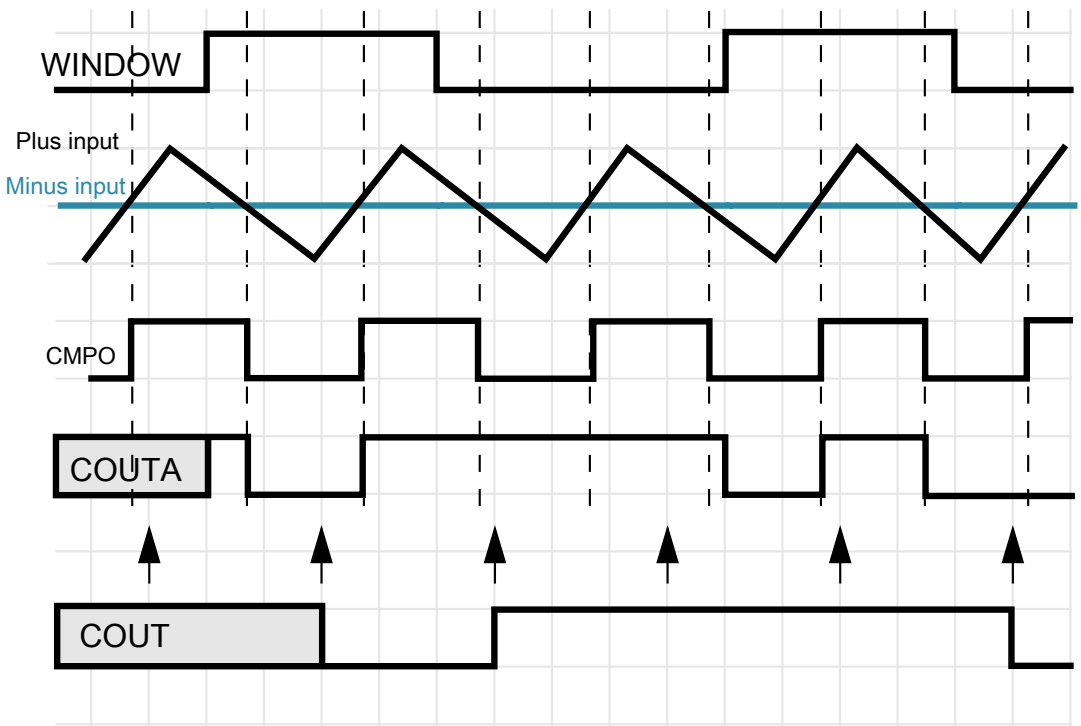


Figure 33-10. Windowed/resampled mode operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER_CNT] must be 1.

33.4.1.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((CR0[FILTER_CNT] * FPR[FILT_PER]) + 1) * bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

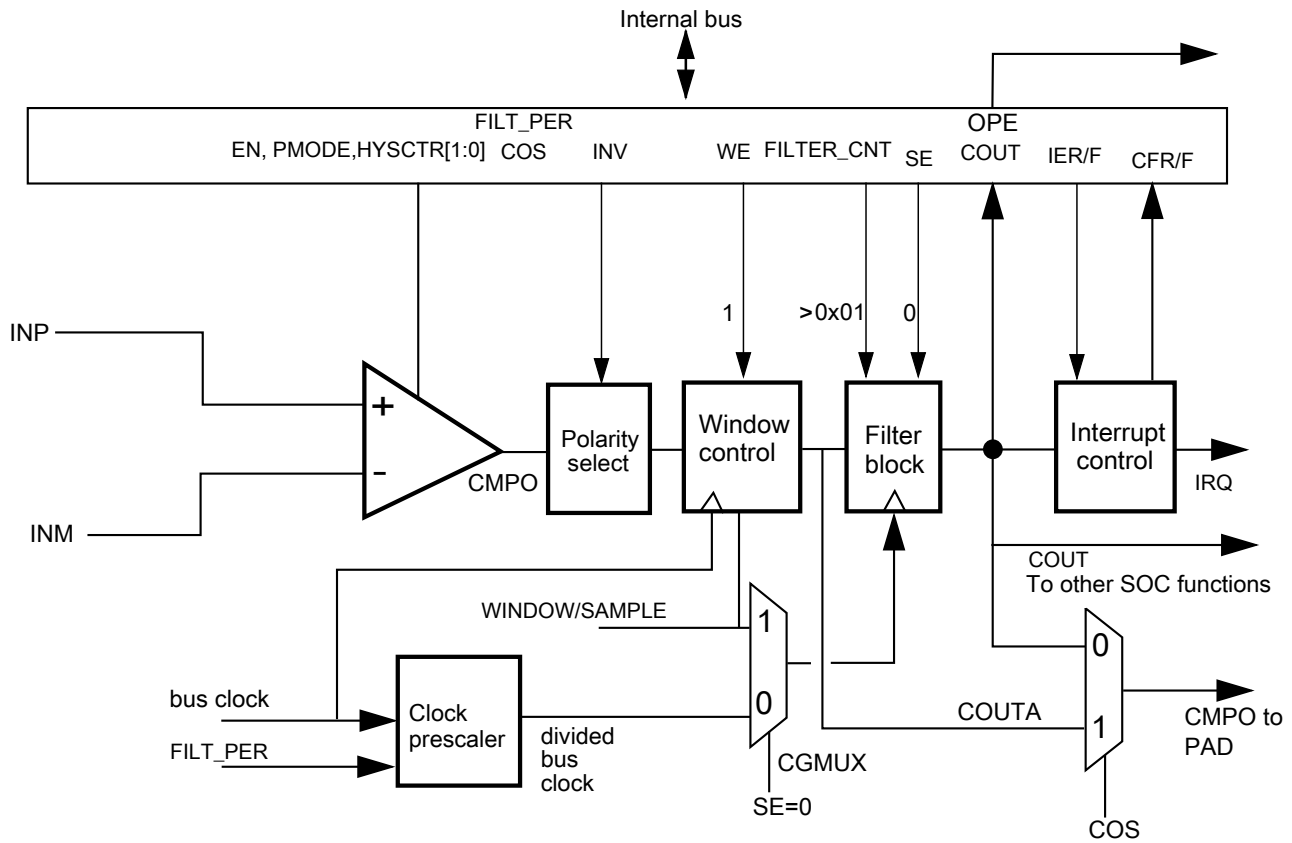


Figure 33-11. Windowed/Filtered mode

33.4.2 Power modes

33.4.2.1 Wait mode operation

During Wait and VLPW modes, the CMP, if enabled, continues to operate normally and a CMP interrupt can wake the MCU.

33.4.2.2 Stop mode operation

Depending on clock restrictions related to the MCU core or core peripherals, the MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In Stop modes, the comparator can be operational in both:

- High-Speed (HS) Comparison mode when CR1[PMODE] = 1
- Low-Speed (LS) Comparison mode when CR1[PMODE] = 0

It is recommended to use the LS mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

33.4.3 Startup and operation

A typical startup sequence is listed here.

- The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. See the Data Sheets for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the Low-pass filter.
- During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and SCR[CFR]/SCR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.
- When programmed for filtering modes, COUT will initially be equal to 0, until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

33.4.4 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT.

Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

33.4.4.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CR0[FILTER_CNT] > 0x01 and
- Setting FPR[FILT_PER] to a nonzero value or setting CR1[SE]=1

If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic 0 when first initalized, and will subsequently change when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed. In other words, SCR[COUT] will be 0 for some initial period, even when COUTA is at logic 1.

Setting both CR1[SE] and FPR[FILT_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed.

33.4.4.2 Latency issues

The value of FPR[FILT_PER] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CR0[FILTER_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CR0[FILTER_CNT].

The values of FPR[FILT_PER] or SAMPLE period and CR0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CR0[FILTER_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

Table 33-4. Comparator sample/filter maximum latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		T_{PD} + (FPR[FILT_PER] * T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] * T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	0x00		$T_{PD} + T_{per}$
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (FPR[FILT_PER] * T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

33.5 CMP interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both.

The following table gives the conditions in which the interrupt request is asserted and deasserted.

When	Then
SCR[IER] and SCR[CFR] are set	The interrupt request is asserted
SCR[IEF] and SCR[CFF] are set	The interrupt request is asserted
SCR[IER] and SCR[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
SCR[IEF] and SCR[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

33.6 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator that deasserts the DMA transfer request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

33.7 CMP Asynchronous DMA support

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

33.8 Digital-to-analog converter

The figure found here shows the block diagram of the DAC module.

It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through the DAC Control Register (DACCR). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down or disabled when not in use. When in Disabled mode, DACO is connected to the analog ground.

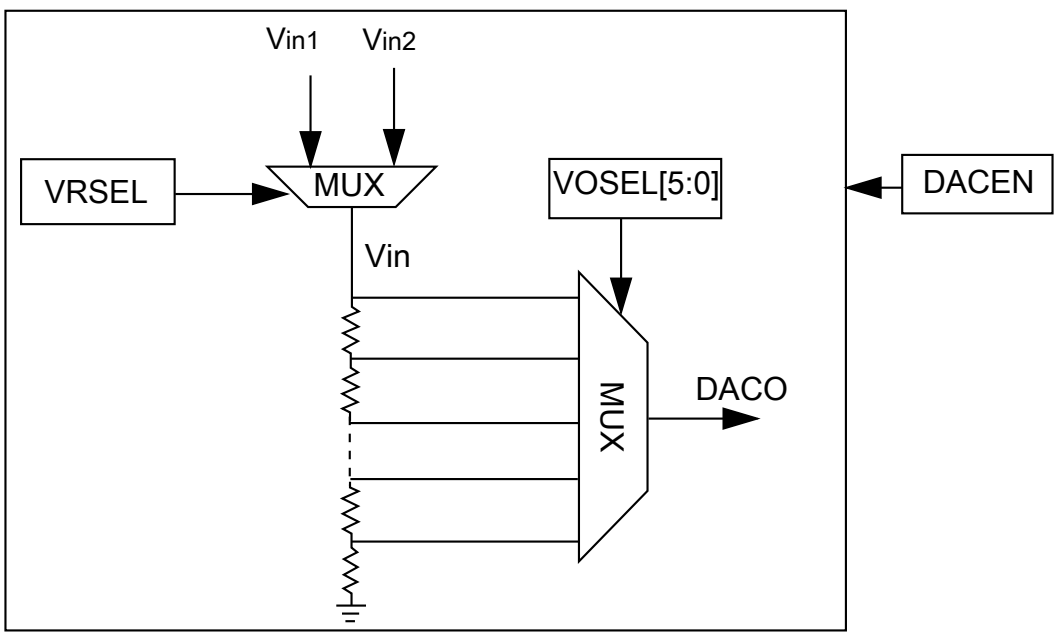


Figure 33-12. 6-bit DAC block diagram

33.9 DAC functional description

This section provides DAC functional description information.

33.9.1 Voltage reference source select

- V_{in1} connects to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} connects to an alternate voltage source

33.10 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

33.11 DAC clocks

This module has a single clock input, the bus clock.

33.12 DAC interrupts

This module has no interrupts.

33.13 CMP Trigger Mode

CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1.

In addition, the CMP must be enabled. If the DAC is to be used as a reference to the CMP, it must also be enabled.

CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare.

Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received.

Chapter 34
Voltage Reference (VREF)

34.1 Chip-specific VREF information

34.1.1 Overview

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

The voltage reference can provide a reference voltage to external peripherals or to analog peripherals, such as Sigma Delta ADCs, SAR and CMP.

34.1.2 Instantiation Information

This device contains one VREF module to produce a 1.2 V reference. This VREF is placed in the VDDA domain .

The VREF is shared between SAR, AFE and DAC (in CMP). The figure below shows the VREF integration and sharing scheme.

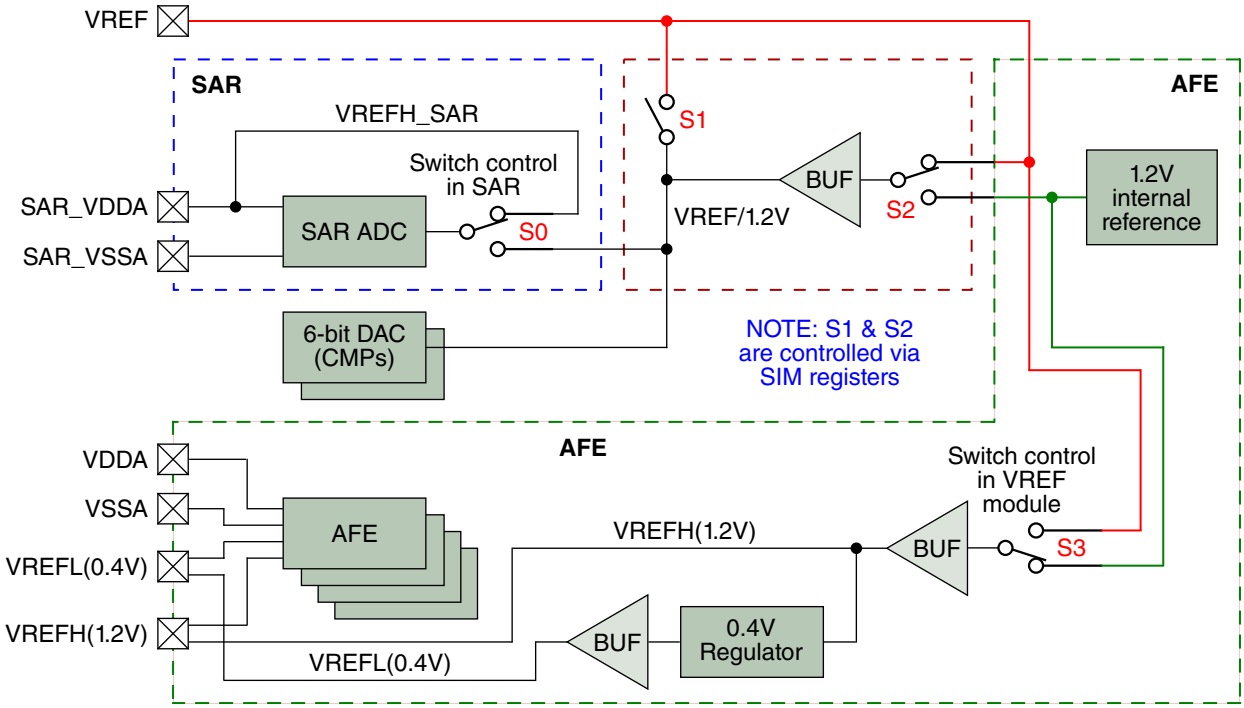


Figure 34-1. VREF Integration

NOTE

The switch S1 is meant for output of 1.2 V VREF onto the SoC pin. When internal 1.2 V VREF is routed through VREF buffer out to VREF pin, then 100 nF capacitor, with good high-frequency characteristics, must be connected between VREF and GNDA. This capacitor must be placed as close as possible to the VREF pin. When using externally supplied VREF, this switch (S1) should be open.

Table 34-1. VREF Sharing Scenarios

	AFE Reference	SAR Reference	6-bit DAC (CMP) Reference	Description
Case I	1.2 V VREF Internal (S3 in current position)	3.3 V SAR_VDDA (S0 in current position)	1.2 V VREF Internal (S2 flipped from current position)	Most common use-case in metering application. Good AFE performance. SAR ADC absolute accuracy driven by tolerance of the LDO regulator (tolerance of typical LDOs is ±2–3%). Internal 1.2 V VREF optionally routed to VREF pin (S1 closed) for use by external circuits.
Case II	1.2 V VREF Internal (S3 in current position)	1.2 V VREF Internal (S0, S2 flipped from current position)	1.2 V VREF Internal (S2 flipped from current position)	Most common use-case in metering application. Good SAR ADC absolute driven by 1.2 V VREF Internal reference (buffered). AFE performance might be impacted by the SAR ADC kick-backs via buffer. Internal 1.2V VREF optionally routed to VREF pin (S1 closed) for use by external circuits.
Case III	1.2 V VREF External	1.2 V VREF External	1.2 V VREF External	Complementary use-case

Table continues on the next page...

Table 34-1. VREF Sharing Scenarios (continued)

	AFE Reference	SAR Reference	6-bit DAC (CMP) Reference	Description
	(S3 flipped from current position)	(S0 flipped from current position, S2 in current position)	(S2 in current position)	
Case IV	1.2 V VREF Internal (S3 in current position)	1.2 V VREF External (S0 flipped from current position, S2 in current position)	1.2 V VREF External (S2 in current position)	AFE reference decoupled from other blocks.

The VREF supports running in tight regulation mode (high power) and low power buffer modes. KM3x_256 family microcontrollers will use this feature in VREF.

NOTE

- S0 corresponds to SAR ADC Voltage Reference Select: ADC_SC2 [REFSEL];
- S1 corresponds to VREF Buffer Output Enable: SIM_MISC_CTL[VREFBUFOUTEN];
- S2 corresponds to VREF Buffer Input Select: SIM_MISC_CTL[VREFBUFINSEL];
- S3 corresponds to Internal and External 1.2 V Reference Select: VREF_VREFL_TRM[VREFL_SEL].

34.2 Introduction

The VREF can be used in applications to provide reference voltages to external devices or used internally as a reference to analog peripherals such as the ADC, DAC, or CMP. The VREF block provides three reference voltages — 1.2 V, 0.8 V, and 0.4 V. VREFH, 1.2 V reference, can be used as an external or internal reference. VREFL, 0.4 V reference, can only be used as a reference for AFE. The 1.2 V and 0.4 V can be combined to provide a differential reference to the AFE. The 0.8 V reference is internal only and is typically used to define the common mode for PGA and Sigma Delta Modulators. In addition it provides a 4μA p-bias source to be used in AFE.

VREFH and VREFL references are output on dedicated output pins when VREF is enabled. VREFH reference output can be trimmed with a resolution of 0.5 mV and the VREFL reference output can be trimmed with a resolution of 10 mV. VREFH_TRIM[5:0] and VREFL_TRIM[2:0] can be used for the purpose respectively.

The following figure is a block diagram of the Voltage Reference.

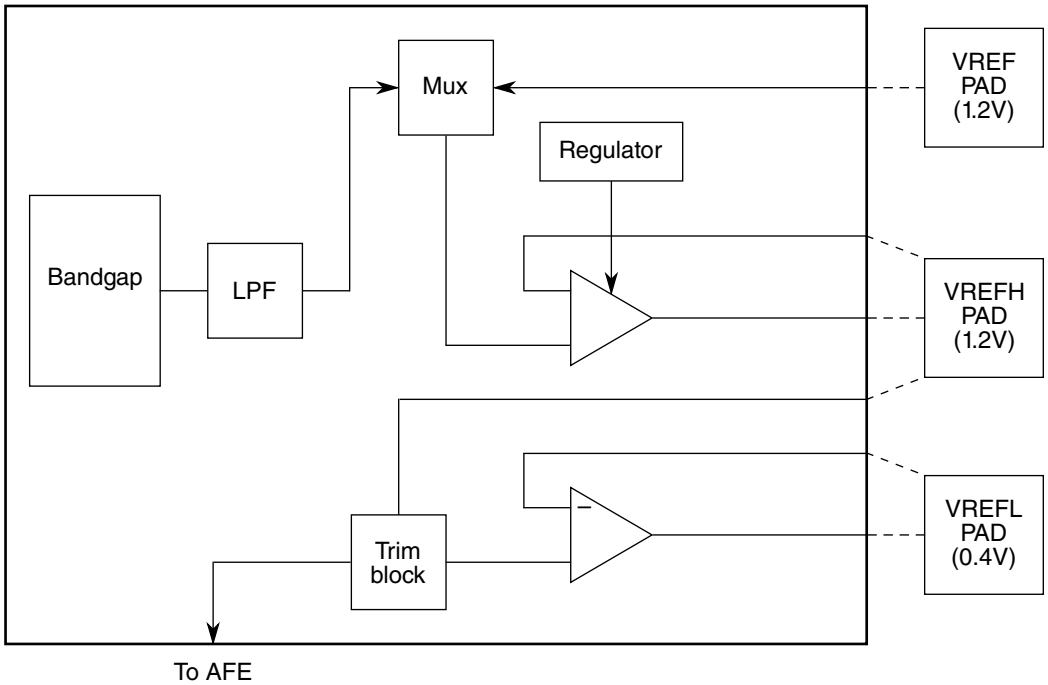


Figure 34-2. Voltage reference block diagram

34.2.1 Features

The Voltage Reference has the following features:

- Programmable trim register with 0.5 mV steps, automatically loaded with factory trimmed value upon reset
- Programmable buffer mode selection:
  - Off
  - Bandgap enabled/standby (output buffer disabled)
  - Low power buffer mode (output buffer enabled)
  - High power buffer mode (output buffer enabled)
- 1.2 V output at room temperature
- Dedicated output pin, VREFH (1.2 V), and VREFL (0.4 V)

34.2.2 Modes of Operation

The Voltage Reference continues normal operation in Run, Wait, and Stop modes. The Voltage Reference can also run in Very Low Power Run (VLPR), Very Low Power Wait (VLPW) and Very Low Power Stop (VLPS). If it is desired to use the VREF regulator in the very low power modes, the system reference voltage must be enabled in these modes. Refer to the chip configuration chapter for information on enabling this mode of operation. Having the VREF regulator enabled does increase current consumption. In very low power modes it may be desirable to disable the VREF regulator to minimize current consumption. Note however that the accuracy of the output voltage will be reduced (by as much as several mVs) when the VREF regulator is not used.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

34.2.3 VREF Signal Descriptions

The following table shows the Voltage Reference signals properties.

Table 34-2. VREF Signal Descriptions

Signal	Description	I/O
VREFH	Internally-generated 1.2 V Voltage Reference output	O
VREFL	Internally-generated 0.4 V Voltage Reference output	O

NOTE

When the VREF output buffers are disabled, the status of the VREFH and VREFL signals is high-impedance.

34.3 Memory Map and Register Definition

VREF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_F000	VREF Trim Register (VREF_VREFH_TRM)	8	R/W	See section	34.3.1/658
4006_F001	VREF Status and Control Register (VREF_VREFH_SC)	8	R/W	00h	34.3.2/659
4006_F005	VREFL TRIM Register (VREF_VREFL_TRM)	8	R/W	See section	34.3.3/660

34.3.1 VREF Trim Register (VREF_VREFH_TRM)

This register contains bits that contain the trim data for the Voltage Reference.

Address: 4006_F000h base + 0h offset = 4006_F000h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved	CHOPEN	TRIM					
Reset	x*	0	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

VREF_VREFH_TRM field descriptions

Field	Description
7 Reserved	This field is reserved. Upon reset this value is loaded with a factory trim value.
6 CHOPEN	Chop oscillator enable. When set, internal chopping operation is enabled and the internal analog offset will be minimized. This bit is set during factory trimming of the VREF voltage. This bit should be written to 1 to achieve the performance stated in the data sheet. 0 Chop oscillator is disabled. 1 Chop oscillator is enabled.
TRIM	Trim bits These bits change the resulting VREF by approximately ± 0.5 mV for each step. **NOTE:** Min = minimum and max = maximum voltage reference output. For minimum and maximum voltage reference output values, refer to the Data Sheet for this chip. 000000 Min .... 111111 Max

34.3.2 VREF Status and Control Register (VREF_VREFH_SC)

This register contains the control bits used to enable the internal voltage reference and to select the buffer mode to be used.

Address: 4006_F000h base + 1h offset = 4006_F001h

Bit	7	6	5	4	3	2	1	0
Read	VREFEN	REGEN	ICOMPEN	0	0	VREFST	MODE_LV	
Write								
Reset	0	0	0	0	0	0	0	0

VREF_VREFH_SC field descriptions

Field	Description
7 VREFEN	Internal Voltage Reference enable This bit is used to enable the bandgap reference within the Voltage Reference module. **NOTE:** After the VREF is enabled, turning off the clock to the VREF module via the corresponding clock gate register will not disable the VREF. VREF must be disabled via this VREFEN bit. 0 The module is disabled. 1 The module is enabled.
6 REGEN	Regulator enable This bit is used to enable the internal 1.75 V regulator to produce a constant internal voltage supply in order to reduce the sensitivity to external supply noise and variation. If it is desired to keep the regulator enabled in very low power modes, refer to the Chip Configuration chapter for a description on how this can be achieved. This bit is set during factory trimming of the VREF voltage. This bit should be written to 1 to achieve the performance stated in the data sheet. 0 Internal 1.75 V regulator is disabled. 1 Internal 1.75 V regulator is enabled.
5 ICOMPEN	Second order curvature compensation enable This bit is set during factory trimming of the VREF voltage. This bit should be written to 1 to achieve the performance stated in the data sheet. 0 Disabled 1 Enabled
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 VREFST	Internal Voltage Reference stable

Table continues on the next page...

VREF_VREFH_SC field descriptions (continued)

Field	Description
	This bit indicates that the bandgap reference within the Voltage Reference module has completed its startup and stabilization. 0 The module is disabled or not stable. 1 The module is stable.
MODE_LV	Buffer Mode selection These bits select the buffer modes for the Voltage Reference module. 00 Bandgap on only, for stabilization and startup 01 High power buffer mode enabled 10 Low-power buffer mode enabled 11 Reserved

34.3.3 VREFL TRIM Register (VREF_VREFL_TRM)

This register contains bits that contain enable and trim data for VREFL (0.4 V) reference. It also contains a bit to select between internal and external 1.2 V reference.

Address: 4006_F000h base + 5h offset = 4006_F005h

Bit	7	6	5	4	3	2	1	0
Read	0			VREFL_SEL	VREFL_EN	VREFL_TRIM		
Write								
Reset	0	0	0	0	0	x*	x*	x*

* Notes:
- x = Undefined at reset.

VREF_VREFL_TRM field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 VREFL_SEL	This bit selects between internal and external 1.2 V reference. 0 Internal reference 1 External reference
3 VREFL_EN	This bit enables the VREFL (0.4 V) reference buffer. 0 Disable 1 Enable
VREFL_TRIM	These bits trim the VREFL reference voltage in steps of 10 mV. The mid voltage of the trim range is 0.4 V at the trim value of 011. **NOTE:** The values 111 and 110 are NOT valid/allowed. **NOTE:** Value of VREFL_TRIM (default 011) is determined by IFR.

34.4 Functional Description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used to buffer the outputs.

The VREFH signal can be used by both internal and external peripherals in low and high power buffer mode. The VREFH and VREFL references together provide a differential referential voltage to the internal ADC. Two 100 nF capacitor must always be connected between VREFH and VREFL and VSSA if the VREF is being used.

The following table shows all possible function configurations of the Voltage Reference.

Table 34-3. Voltage Reference function configurations

SC[VREFEN]	SC[MODE_LV]	Configuration	Functionality
0	X	Voltage Reference disabled	Off
1	00	Voltage Reference enabled, bandgap on only	Startup and standby
1	01	Voltage Reference enabled, high-power buffer on	VREFH available for internal and external use. 100 nF capacitor is required.
1	10	Voltage Reference enabled, low power buffer on	VREFH available for internal and external use. 100 nF capacitor is required.
1	11	Reserved	Reserved

34.4.1 Voltage Reference Disabled, SC[VREFEN] = 0

When SC[VREFEN] = 0, the Voltage Reference is disabled, the bandgap and the output buffers are disabled. The Voltage Reference is in off mode.

34.4.2 Voltage Reference Enabled, SC[VREFEN] = 1

When SC[VREFEN] = 1, the Voltage Reference is enabled, and different modes should be set by the SC[MODE_LV] bits. VREFL reference buffer is enabled by setting VREF_EN bit.

34.4.2.1 SC[MODE_LV]=00

The internal VREF bandgap is enabled to generate an accurate 1.2 V output that can be trimmed with the TRM register's VREFH_TRIM[5:0] bitfield. The bandgap requires some time for startup and stabilization. SC[VREFST] can be monitored to determine if the stabilization and startup is complete.

The output buffer is disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode. If this mode is first selected and the low power or high power buffer mode is subsequently enabled, there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet.

34.4.2.2 SC[MODE_LV] = 01

The internal VREF bandgap is on. The high power buffer is enabled to generate a buffered 1.2 V voltage to VREFH. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from the standby mode (SC[MODE_LV] = 00, SC[VREFEN] = 1) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of Tstup or until SC[VREFST] = 1.

In this mode, a 100 nF capacitor is required to connect between the VREFH pin and VSSA.

34.4.2.3 SC[MODE_LV] = 10

The internal bandgap is on. The low power buffer is enabled to generate a buffered 1.2 V voltage to VREFH. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator inpu0t.

If this mode is entered from the standby mode (SC[MODE_LV] = 00, SC[VREFEN] = 1) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of Tstup or until SC[VREFST] = 1.

In this mode, a 100 nF capacitor is required to connect between the VREFH pin and VSSA.

34.4.2.4 SC[MODE_LV] = 11

Reserved

34.5 Initialization/Application Information

The Voltage Reference requires some time for startup and stabilization. After SC[VREFEN] = 1, wait for 40ms for startup and stabilization.

When the Voltage Reference is already enabled and stabilized, changing SC[MODE_LV] will not clear SC[VREFST] but there will be some startup time before the output voltage at the VREFH pin has settled. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. Also, there will be some settling time when a step change of the load current is applied to the VREFH pin. When the 1.75V VREF regulator is disabled, the VREFH voltage will be more sensitive to supply voltage variation. It is recommended to use this regulator to achieve optimum VREFH performance.

The TRM[CHOPEN]and SC[REGEN] bits are written to 1 during factory trimming of the VREF voltage. These bits should be written to 1 to achieve the performance stated in the device data sheet.

To enable the VREFL reference buffer the bit VREFL_EN should be set to 1. The mid value of 011 should be written to the bits VREFL_TRIM[2:0]

Chapter 35
Multipurpose Clock Generator (MCG)

35.1 Introduction

The multipurpose clock generator (MCG) module provides several clock source choices for the MCU.

The module contains a frequency-locked loop (FLL) and a phase-locked loop (PLL). The FLL is controllable by either an internal or an external reference clock. The PLL is controllable by the external reference clock. The module can select either an FLL or PLL output clock, or a reference clock (internal or external) as a source for the MCU system clock. The MCG operates in conjuction with a crystal oscillator, which allows an external crystal, ceramic resonator, or another external clock source to produce the external reference clock.

35.1.1 Features

Key features of the MCG module are:

- Frequency-locked loop (FLL):
  - Digitally-controlled oscillator (DCO)
  - DCO frequency range is programmable for up to four different frequency ranges.
  - Option to program and maximize DCO output frequency for a low frequency external reference clock source.
  - Option to prevent FLL from resetting its current locked frequency when switching clock modes if FLL reference frequency is not changed.
  - Internal or external reference clock can be used as the FLL source.
  - Can be used as a clock source for other on-chip peripherals.
- Phase-locked loop (PLL):

  - Voltage-controlled oscillator (VCO)
  - Selectable External reference clock is used as the PLL source.
  - Modulo VCO frequency divider
  - Phase/Frequency detector
  - Integrated loop filter
  - Can be used as a clock source for other on-chip peripherals.
- Internal reference clock generator:
  - Slow clock with nine trim bits for accuracy
  - Fast clock with four trim bits
  - Can be used as source clock for the FLL. In FEI mode, only the slow Internal Reference Clock (IRC) can be used as the FLL source.
  - Either the slow or the fast clock can be selected as the clock source for the MCU.
  - Can be used as a clock source for other on-chip peripherals.
- Control signals for the MCG external reference low power oscillator clock generators are provided:
  - HGO, RANGE, EREFS
- External clock from the Crystal Oscillator :
  - Can be used as a source for the FLL and/or the PLL.
  - Can be selected as the clock source for the MCU.
- External clock from the Real Time Counter (RTC):
  - Can only be used as a source for the FLL.
  - Can be selected as the clock source for the MCU.
- External clock monitor with reset and interrupt request capability to check for external clock failure when running in FBE, PEE, BLPE, or FEE modes
- Lock detector with interrupt request capability for use with the PLL
- Internal Reference Clocks Auto Trim Machine (ATM) capability using an external clock as a reference
- Reference dividers for both the FLL are provided

- Reference dividers for the Fast Internal Reference Clock are provided
- MCG PLL Clock (MCGPLLCLK) is provided as a clock source for other on-chip peripherals
- MCG FLL Clock (MCGFLLCLK) is provided as a clock source for other on-chip peripherals
- MCG Fixed Frequency Clock (MCGFFCLK) is provided as a clock source for other on-chip peripherals
- MCG Internal Reference Clock (MCGIRCLK) is provided as a clock source for other on-chip peripherals

This figure presents the block diagram of the MCG module.

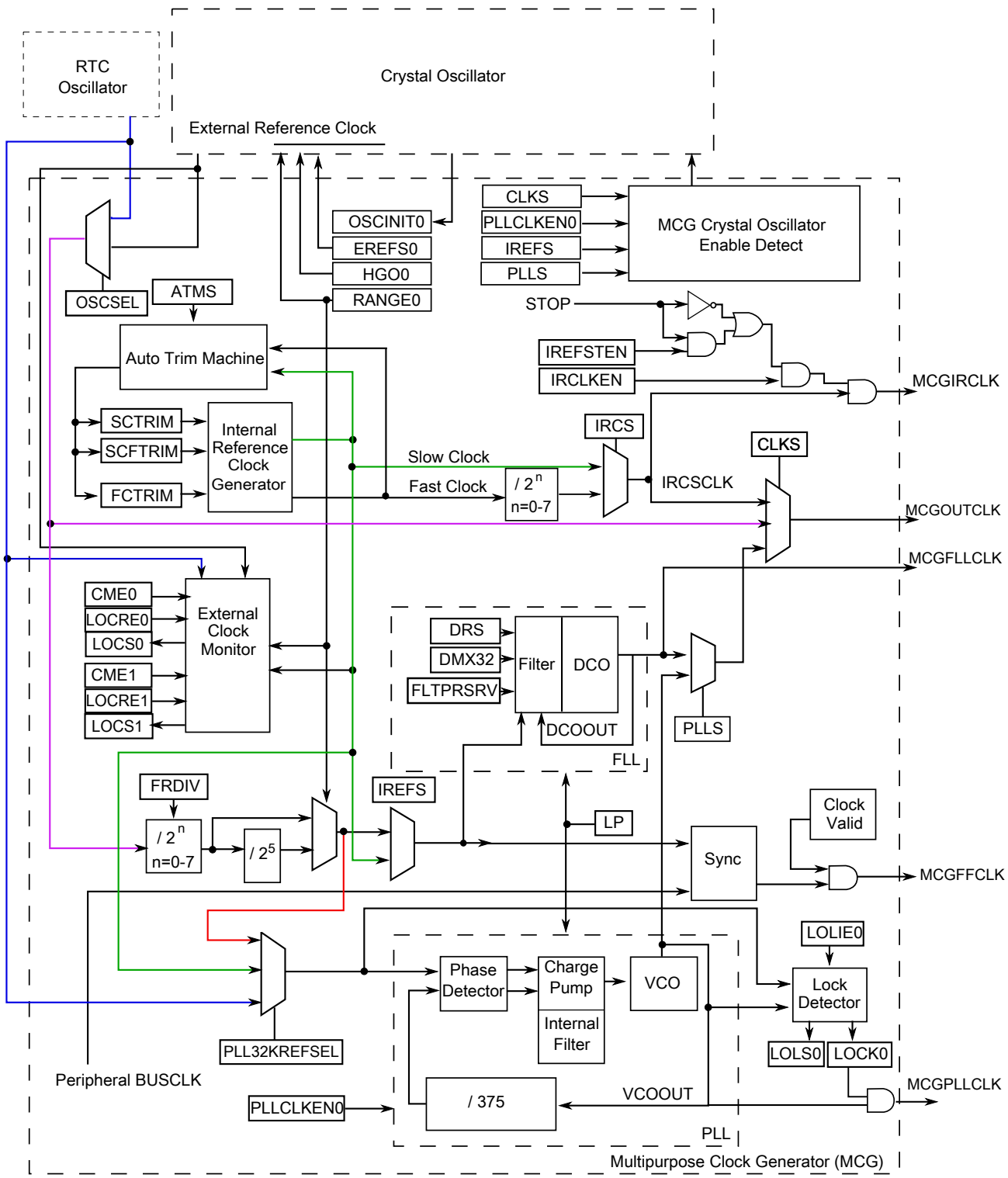


Figure 35-1. Multipurpose Clock Generator (MCG) block diagram

35.1.2 Modes of Operation

The MCG has the following modes of operation: FEI, FEE, FBI, FBE, PBE, PEE, PEI, PBI, BLPI, BLPE, and Stop. For details, see MCG modes of operation.

35.2 External Signal Description

There are no MCG signals that connect off chip.

35.3 Memory Map/Register Definition

This section includes the memory map and register definition.

MCG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_4000	MCG Control 1 Register (MCG_C1)	8	R/W	44h	35.3.1/670
4006_4001	MCG Control 2 Register (MCG_C2)	8	R/W	83h	35.3.2/671
4006_4002	MCG Control 3 Register (MCG_C3)	8	R/W	Undefined	35.3.3/672
4006_4003	MCG Control 4 Register (MCG_C4)	8	R/W	Undefined	35.3.4/673
4006_4004	MCG Control 5 Register (MCG_C5)	8	R/W	00h	35.3.5/674
4006_4005	MCG Control 6 Register (MCG_C6)	8	R/W	08h	35.3.6/675
4006_4006	MCG Status Register (MCG_S)	8	R	15h	35.3.7/676
4006_4008	MCG Status and Control Register (MCG_SC)	8	R/W	02h	35.3.8/677
4006_400A	MCG Auto Trim Compare Value High Register (MCG_ATCVH)	8	R/W	00h	35.3.9/678
4006_400B	MCG Auto Trim Compare Value Low Register (MCG_ATCVL)	8	R/W	00h	35.3.10/ 679
4006_400C	MCG Control 7 Register (MCG_C7)	8	R/W	00h	35.3.11/ 679
4006_400D	MCG Control 8 Register (MCG_C8)	8	R/W	80h	35.3.12/ 680
4006_400E	MCG Control 9 Register (MCG_C9)	8	R/W	00h	35.3.13/ 681
4006_4011	MCG Control 12 Register (MCG_C12)	8	R/W	00h	35.3.14/ 682
4006_4012	MCG Status 2 Register (MCG_S2)	8	R/W	00h	35.3.14/ 682
4006_4013	MCG Test 3 Register (MCG_T3)	8	R/W	00h	35.3.14/ 683

35.3.1 MCG Control 1 Register (MCG_C1)

Address: 4006_4000h base + 0h offset = 4006_4000h

Bit	7	6	5	4	3	2	1	0
Read Write	CLKS		FRDIV			IREFS	IRCLKEN	IREFSTEN
Reset	0	1	0	0	0	1	0	0

MCG_C1 field descriptions

Field	Description
7–6 CLKS	Clock Source Select Selects the clock source for MCGOUTCLK. 00 Encoding 0 — Output of FLL or PLL is selected (depends on PLLS control bit). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Reserved.
5–3 FRDIV	FLL External Reference Divider Selects the amount to divide down the external reference clock for the FLL. The resulting frequency must be in the range 31.25 kHz to 39.0625 kHz (This is required when FLL/DCO is the clock source for MCGOUTCLK. In FBE mode, it is not required to meet this range, but it is recommended in the cases when trying to enter a FLL mode from FBE). 000 If RANGE = 0 or OSCSEL=1 , Divide Factor is 1; for all other RANGE values, Divide Factor is 32. 001 If RANGE = 0 or OSCSEL=1 , Divide Factor is 2; for all other RANGE values, Divide Factor is 64. 010 If RANGE = 0 or OSCSEL=1 , Divide Factor is 4; for all other RANGE values, Divide Factor is 128. 011 If RANGE = 0 or OSCSEL=1 , Divide Factor is 8; for all other RANGE values, Divide Factor is 256. 100 If RANGE = 0 or OSCSEL=1 , Divide Factor is 16; for all other RANGE values, Divide Factor is 512. 101 If RANGE = 0 or OSCSEL=1 , Divide Factor is 32; for all other RANGE values, Divide Factor is 1024. 110 If RANGE = 0 or OSCSEL=1 , Divide Factor is 64; for all other RANGE values, Divide Factor is 1280 . 111 If RANGE = 0 or OSCSEL=1 , Divide Factor is 128; for all other RANGE values, Divide Factor is 1536 .
2 IREFS	Internal Reference Select Selects the reference clock source for the FLL. 0 External reference clock is selected. 1 The slow internal reference clock is selected.
1 IRCLKEN	Internal Reference Clock Enable Enables the internal reference clock for use as MCGIRCLK. 0 MCGIRCLK inactive. 1 MCGIRCLK active.
0 IREFSTEN	Internal Reference Stop Enable Controls whether or not the internal reference clock remains enabled when the MCG enters Stop mode.

Table continues on the next page...

MCG_C1 field descriptions (continued)

Field	Description
	0 Internal reference clock is disabled in Stop mode. 1 Internal reference clock is enabled in Stop mode if IRCLKEN is set or if MCG is in FEI, FBI, or BLPI modes before entering Stop mode.

35.3.2 MCG Control 2 Register (MCG_C2)

Address: 4006_4000h base + 1h offset = 4006_4001h

Bit	7	6	5	4	3	2	1	0
Read Write	LOCRE0	0	RANGE0		HGO0	EREFS0	LP	IRCS
Reset	1	0	0	0	0	0	1	1

MCG_C2 field descriptions

Field	Description
7 LOCRE0	Loss of Clock Reset Enable Determines whether an interrupt or a reset request is made following a loss of OSC0 external reference clock. The LOCRE0 only has an affect when CME0 is set. 0 Interrupt request is generated on a loss of OSC0 external reference clock. 1 Generate a reset request on a loss of OSC0 external reference clock.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 RANGE0	Frequency Range Select Selects the frequency range for the crystal oscillator or external clock source. See the Oscillator (OSC) chapter for more details and the device data sheet for the frequency ranges used. 00 Encoding 0 — Low frequency range selected for the crystal oscillator . 01 Encoding 1 — High frequency range selected for the crystal oscillator . 1X Encoding 2 — Very high frequency range selected for the crystal oscillator .
3 HGO0	High Gain Oscillator Select Controls the crystal oscillator mode of operation. See the Oscillator (OSC) chapter for more details. 0 Configure crystal oscillator for low-power operation. 1 Configure crystal oscillator for high-gain operation.
2 EREFS0	External Reference Select Selects the source for the external reference clock. See the Oscillator (OSC) chapter for more details. 0 External reference clock requested. 1 Oscillator requested.
1 LP	Low Power Select

Table continues on the next page...

MCG_C2 field descriptions (continued)

Field	Description
	Controls whether the FLL or PLL is disabled in BLPI and BLPE modes. In FBE or PBE modes, setting this bit to 1 will transition the MCG into BLPE mode; in FBI mode, setting this bit to 1 will transition the MCG into BLPI mode. In any other MCG mode, LP bit has no affect. 0 FLL or PLL is not disabled in bypass modes. 1 FLL or PLL is disabled in bypass modes (lower power)
0 IRCS	Internal Reference Clock Select Selects between the fast or slow internal reference clock source. 0 Slow internal reference clock selected. 1 Fast internal reference clock selected.

35.3.3 MCG Control 3 Register (MCG_C3)

Address: 4006_4000h base + 2h offset = 4006_4002h

Bit	7	6	5	4	3	2	1	0
Read Write	SCTRIM							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCG_C3 field descriptions

Field	Description
SCTRIM	Slow Internal Reference Clock Trim Setting SCTRIM [1] controls the slow internal reference clock frequency by controlling the slow internal reference clock period. The SCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period. An additional fine trim bit is available in C4 register as the SCFTRIM bit. Upon reset, this value is loaded with a factory trim value. If an SCTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.

1. A value for SCTRIM is loaded during reset from a factory programmed location.

35.3.4 MCG Control 4 Register (MCG_C4)

NOTE

Reset values for DRST and DMX32 bits are 0.

Address: 4006_4000h base + 3h offset = 4006_4003h

Bit	7	6	5	4	3	2	1	0
Read Write	DMX32	DRST_DRS			FCTRIM			SCFTRIM
Reset	0	0	0	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.
- x = Undefined at reset.

MCG_C4 field descriptions

Field	Description
7 DMX32	DCO Maximum Frequency with 32.768 kHz Reference The DMX32 bit controls whether the DCO frequency range is narrowed to its maximum frequency with a 32.768 kHz reference. The following table identifies settings for the DCO frequency range. **NOTE:** The system clocks derived from this source should not exceed their specified maximums.

DRST_DRS	DMX32	Reference Range	FLL Factor	DCO Range
00	0	31.25–39.0625 kHz	640	20–25 MHz
	1	32.768 kHz	732	24 MHz
01	0	31.25–39.0625 kHz	1280	40–50 MHz
	1	32.768 kHz	1464	48 MHz
10	0	31.25–39.0625 kHz	1920	60–75 MHz
	1	32.768 kHz	2197	72 MHz
11	0	31.25–39.0625 kHz	2560	80–100 MHz
	1	32.768 kHz	2929	96 MHz

0 DCO has a default range of 25%.
1 DCO is fine-tuned for maximum frequency with 32.768 kHz reference.

6–5 DRST_DRS	DCO Range Select The DRS bits select the frequency range for the FLL output, DCOOUT. When the LP bit is set, writes to the DRS bits are ignored. The DRST read field indicates the current frequency range for DCOOUT. The DRST field does not update immediately after a write to the DRS field due to internal synchronization between clock domains. See the DCO Frequency Range table for more details. 00 Encoding 0 — Low range (reset default).

Table continues on the next page...

MCG_C4 field descriptions (continued)

Field	Description
	01 Encoding 1 — Mid range. 10 Encoding 2 — Mid-high range. 11 Encoding 3 — High range.
4–1 FCTRIM	Fast Internal Reference Clock Trim Setting FCTRIM [1] controls the fast internal reference clock frequency by controlling the fast internal reference clock period. The FCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period. If an FCTRIM[3:0] value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.
0 SCFTRIM	Slow Internal Reference Clock Fine Trim SCFTRIM [2] controls the smallest adjustment of the slow internal reference clock frequency. Setting SCFTRIM increases the period and clearing SCFTRIM decreases the period by the smallest amount possible. If an SCFTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this bit.

1. A value for FCTRIM is loaded during reset from a factory programmed location.
2. A value for SCFTRIM is loaded during reset from a factory programmed location .

35.3.5 MCG Control 5 Register (MCG_C5)

Address: 4006_4000h base + 4h offset = 4006_4004h

Bit	7	6	5	4	3	2	1	0
Read	0	PLLCLKEN0	PLLSTEN0	0				
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C5 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 PLLCLKEN0	PLL Clock Enable Enables the PLL independent of PLLS and enables the PLL clock for use as MCGPLLCLK. Setting PLLCLKEN 0 will enable the external oscillator if not already enabled. Whenever the PLL is being enabled by means of the PLLCLKEN 0 bit, and the external oscillator is being used as the reference clock, the OSCINIT 0 bit should be checked to make sure it is set. 0 MCGPLLCLK is inactive. 1 MCGPLLCLK is active.
5 PLLSTEN0	PLL Stop Enable Enables the PLL Clock during Normal Stop. In Low Power Stop mode, the PLL clock gets disabled even if PLLSTEN 0 =1. All other power modes, PLLSTEN 0 bit has no affect and does not enable the PLL Clock to run if it is written to 1.

Table continues on the next page...

MCG_C5 field descriptions (continued)

Field	Description
	0 MCGPLLCLK is disabled in any of the Stop modes. 1 MCGPLLCLK is enabled if system is in Normal Stop mode.
Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.

35.3.6 MCG Control 6 Register (MCG_C6)

Address: 4006_4000h base + 5h offset = 4006_4005h

Bit	7	6	5	4	3	2	1	0
Read Write	LOLIE0	PLLS	CME0	CHGPMP_BIAS				
Reset	0	0	0	0	1	0	0	0

MCG_C6 field descriptions

Field	Description
7 LOLIE0	Loss of Lock Interrrupt Enable Determines if an interrupt request is made following a loss of lock indication. This bit only has an effect when LOLS 0 is set. 0 No interrupt request is generated on loss of lock. 1 Generate an interrupt request on loss of lock.
6 PLLS	PLL Select Controls whether the PLL or FLL output is selected as the MCG source when CLKS[1:0]=00. If the PLLS bit is cleared and PLLCLKEN 0 is not set, the PLL is disabled in all modes. If the PLLS is set, the FLL is disabled in all modes. 0 FLL is selected. 1 PLL is selected (PLL reference clock must be in the range of 31.25–39.0625 KHz prior to setting the PLLS bit).
5 CME0	Clock Monitor Enable Enables the loss of clock monitoring circuit for the OSC0 external reference mux select. The LOCRE0 bit will determine if a interrupt or a reset request is generated following a loss of OSC0 indication. The CME0 bit must only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE) . Whenever the CME0 bit is set to a logic 1, the value of the RANGE0 bits in the C2 register should not be changed. CME0 bit should be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur while in Stop mode. CME0 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode. 0 External clock monitor is disabled for OSC0. 1 External clock monitor is enabled for OSC0.
CHGPMP_BIAS	Directly controls the PLL Charge Pump Current. Appropiate selection of this value is imperative to ensure stable operation of the PLL closed loop system. The default value for this field is set to 5'b01000 out of reset which generates a nominal 750 nA charge pump current (Icp).

35.3.7 MCG Status Register (MCG_S)

Address: 4006_4000h base + 6h offset = 4006_4006h

Bit	7	6	5	4	3	2	1	0
Read	LOLS0	LOCK0	PLLST	IREFST	CLKST		OSCINIT0	IRCST
Write								
Reset	0	0	0	1	0	1	0	1

MCG_S field descriptions

Field	Description
7 LOLS0	Loss of Lock Status This bit is a sticky bit indicating the lock status for the PLL. LOLS is set if after acquiring lock, the PLL output frequency has fallen outside the lock exit frequency tolerance, D_{unl} . LOLIE determines whether an interrupt request is made when LOLS is set. LOLRE determines whether a reset request is made when LOLS is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect. 0 PLL has not lost lock since LOLS 0 was last cleared. 1 PLL has lost lock since LOLS 0 was last cleared.
6 LOCK0	Lock Status This bit indicates whether the PLL has acquired lock. Lock detection is only enabled when the PLL is enabled (either through clock mode selection or PLLCLKEN0=1 setting). While the PLL clock is locking to the desired frequency, the MCG PLL clock (MCGPLLCLK) will be gated off until the LOCK bit gets asserted. Loss of PLL reference clock will also cause the LOCK0 bit to clear until the PLL has reacquired lock. Entry into VLPS, or regular Stop with PLLSTEN=0 also causes the lock status bit to clear and stay cleared until the Stop mode is exited and the PLL has reacquired lock. Any time the PLL is enabled and the LOCK0 bit is cleared, the MCGPLLCLK will be gated off until the LOCK0 bit is asserted again. 0 PLL is currently unlocked. 1 PLL is currently locked.
5 PLLST	PLL Select Status This bit indicates the clock source selected by PLLS . The PLLST bit does not update immediately after a write to the PLLS bit due to internal synchronization between clock domains. 0 Source of PLLS clock is FLL clock. 1 Source of PLLS clock is PLL output clock.
4 IREFST	Internal Reference Status This bit indicates the clock source for the FLL reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains. 0 Source of FLL reference clock is the external reference clock. 1 Source of FLL reference clock is the internal reference clock.
3–2 CLKST	Clock Mode Status These bits indicate the current clock mode. The CLKST bits do not update immediately after a write to the CLKS bits due to internal synchronization between clock domains.

Table continues on the next page...

MCG_S field descriptions (continued)

Field	Description
	00 Encoding 0 — Output of the FLL is selected (reset default). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Output of the PLL is selected.
1 OSCINIT0	OSC Initialization This bit, which resets to 0, is set to 1 after the initialization cycles of the crystal oscillator clock have completed. After being set, the bit is cleared to 0 if the OSC is subsequently disabled. See the OSC module's detailed description for more information.
0 IRCST	Internal Reference Clock Status The IRCST bit indicates the current source for the internal reference clock select clock (IRCSCLK). The IRCST bit does not update immediately after a write to the IRCS bit due to internal synchronization between clock domains. The IRCST bit will only be updated if the internal reference clock is enabled, either by the MCG being in a mode that uses the IRC or by setting the C1[IRCLKEN] bit . 0 Source of internal reference clock is the slow clock (32 kHz IRC). 1 Source of internal reference clock is the fast clock (4 MHz IRC).

35.3.8 MCG Status and Control Register (MCG_SC)

Address: 4006_4000h base + 8h offset = 4006_4008h

Bit	7	6	5	4	3	2	1	0
Read	ATME	ATMS	ATMF	FLTPRSRV	FCRDIV			LOCS0
Write			w1c					w1c
Reset	0	0	0	0	0	0	1	0

MCG_SC field descriptions

Field	Description
7 ATME	Automatic Trim Machine Enable Enables the Auto Trim Machine to start automatically trimming the selected Internal Reference Clock. **NOTE:** ATME deasserts after the Auto Trim Machine has completed trimming all trim bits of the IRCS clock selected by the ATMS bit. Writing to C1, C3, C4, and SC registers or entering Stop mode aborts the auto trim operation and clears this bit. 0 Auto Trim Machine disabled. 1 Auto Trim Machine enabled.
6 ATMS	Automatic Trim Machine Select Selects the IRCS clock for Auto Trim Test. 0 32 kHz Internal Reference Clock selected. 1 4 MHz Internal Reference Clock selected.

Table continues on the next page...

MCG_SC field descriptions (continued)

Field	Description
5 ATMF	Automatic Trim Machine Fail Flag Fail flag for the Automatic Trim Machine (ATM). This bit asserts when the Automatic Trim Machine is enabled, ATME=1, and a write to the C1, C3, C4, and SC registers is detected or the MCG enters into any Stop mode. A write to ATMF clears the flag. 0 Automatic Trim Machine completed normally. 1 Automatic Trim Machine failed.
4 FLTPRSRV	FLL Filter Preserve Enable This bit will prevent the FLL filter values from resetting allowing the FLL output frequency to remain the same during clock mode changes where the FLL/DCO output is still valid. (Note: This requires that the FLL reference frequency to remain the same as what it was prior to the new clock mode switch. Otherwise FLL filter and frequency values will change.) 0 FLL filter and FLL frequency will reset on changes to currect clock mode. 1 Fll filter and FLL frequency retain their previous values during new clock mode change.
3–1 FCRDIV	Fast Clock Internal Reference Divider Selects the amount to divide down the fast internal reference clock. The resulting frequency will be in the range 31.25 kHz to 4 MHz (Note: Changing the divider when the Fast IRC is enabled is not supported). 000 Divide Factor is 1 001 Divide Factor is 2. 010 Divide Factor is 4. 011 Divide Factor is 8. 100 Divide Factor is 16 101 Divide Factor is 32 110 Divide Factor is 64 111 Divide Factor is 128.
0 LOCS0	OSC0 Loss of Clock Status The LOCS0 indicates when a loss of OSC0 reference clock has occurred. The LOCS0 bit only has an effect when CME0 is set. This bit is cleared by writing a logic 1 to it when set. 0 Loss of OSC0 has not occurred. 1 Loss of OSC0 has occurred.

35.3.9 MCG Auto Trim Compare Value High Register (MCG_ATCVH)

Address: 4006_4000h base + Ah offset = 4006_400Ah

Bit	7	6	5	4	3	2	1	0
Read Write	ATCVH							
Reset	0	0	0	0	0	0	0	0

MCG_ATCVH field descriptions

Field	Description
ATCVH	ATM Compare Value High Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

35.3.10 MCG Auto Trim Compare Value Low Register (MCG_ATCVL)

Address: 4006_4000h base + Bh offset = 4006_400Bh

Bit	7	6	5	4	3	2	1	0
Read Write	ATCVL							
Reset	0	0	0	0	0	0	0	0

MCG_ATCVL field descriptions

Field	Description
ATCVL	ATM Compare Value Low Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

35.3.11 MCG Control 7 Register (MCG_C7)

Address: 4006_4000h base + Ch offset = 4006_400Ch

Bit	7	6	5	4	3	2	1	0
Read	PLL32KREFSEL		0				0	OSCSEL
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C7 field descriptions

Field	Description
7–6 PLL32KREFSEL	MCG PLL 32Khz Reference Clock Select Selects the 32kHz PLL reference clock. 00 Selects 32 kHz RTC Oscillator. 01 Selects 32 kHz IRC. 10 Selects FLL FRDIV clock. 11 Reserved.
5–2 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MCG_C7 field descriptions (continued)

Field	Description
1 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
0 OSCSEL	MCG OSC Clock Select Selects the MCG FLL external reference clock 0 Selects Oscillator (OSCCLK). 1 Selects 32 kHz RTC Oscillator.

35.3.12 MCG Control 8 Register (MCG_C8)

Address: 4006_4000h base + Dh offset = 4006_400Dh

Bit	7	6	5	4	3	2	1	0
Read	LOCRE1	LOLRE	CME1	COARSE_ LOLIE	0			LOCS1
Write								
Reset	1	0	0	0	0	0	0	0

MCG_C8 field descriptions

Field	Description
7 LOCRE1	Loss of Clock Reset Enable Determines if a interrupt or a reset request is made following a loss of RTC external reference clock. The LOCRE1 only has an affect when CME1 is set. 0 Interrupt request is generated on a loss of RTC external reference clock. 1 Generate a reset request on a loss of RTC external reference clock
6 LOLRE	PLL Loss of Lock Reset Enable Determines if an interrupt or a reset request is made following a PLL loss of lock. 0 Interrupt request is generated on a PLL loss of lock indication. The PLL loss of lock interrupt enable bit must also be set to generate the interrupt request. 1 Generate a reset request on a PLL loss of lock indication.
5 CME1	Clock Monitor Enable1 Enables the loss of clock monitoring circuit for the output of the RTC external reference clock. The LOCRE1 bit will determine whether an interrupt or a reset request is generated following a loss of RTC clock indication. The CME1 bit should be set to a logic 1 when the MCG is in an operational mode that uses the RTC as its external reference clock or if the RTC is operational. CME1 bit must be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur when in Stop mode. CME1 should also be set to a logic 0 before entering VLPR or VLPW power modes. 0 External clock monitor is disabled for RTC clock. 1 External clock monitor is enabled for RTC clock.

Table continues on the next page...

MCG_C8 field descriptions (continued)

Field	Description
4 COARSE_LOLIE	Loss of Coarse Lock Interrrupt Enable Determines if an interrupt request is made following a coarse loss of lock indication. This bit only has an effect when COARSE_LOLS is set. 0 No interrupt request is generated on coarse loss of lock. 1 Generate an interrupt request on coarse loss of lock.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LOCS1	RTC Loss of Clock Status This bit indicates when a loss of clock has occurred. This bit is cleared by writing a logic 1 to it when set. 0 Loss of RTC has not occur. 1 Loss of RTC has occur

35.3.13 MCG Control 9 Register (MCG_C9)

Address: 4006_4000h base + Eh offset = 4006_400Eh

Bit	7	6	5	4
Read	COARSE_LOLS	COARSE_LOCK	0	
Write				
Reset	0	0	0	0

Bit	3	2	1	0
Read	0			0
Write				
Reset	0	0	0	0

MCG_C9 field descriptions

Field	Description
7 COARSE_LOLS	Coarse Loss of Lock Status This bit is a sticky bit indicating the coarse lock status for the PLL. COARSE_LOLS is set if after acquiring a coarse lock, the PLL output frequency has fallen outside the lock exit frequency tolerance, D_{unl} . COARSE_LOLIE determines whether an interrupt request is made when COARSE_LOLS is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect. 0 PLL has not lost lock since COARSE_LOLS was last cleared. 1 PLL has lost lock since COARSE_LOLS was last cleared.
6 COARSE_LOCK	Coarse Lock Status This bit indicates whether the PLL has acquired coarse lock after the first lock sample (Note: LOCK bit asserts after three lock samples and CRS_LOCK will assert after first lock sample). Lock detection is disabled when not operating in either PEI, PBI, PBE or PEE mode unless PLLCLKEN0 =1 and the MCG is

Table continues on the next page...

MCG_C9 field descriptions (continued)

Field	Description
	not configured in BLPI or BLPE mode. Loss of PLL reference clock will also cause the COARSE LOCK PLL has reacquired lock. Entry into LLS, VLPS, or regular Stop with PLLSTEN0=0 also causes the coarse lock status bit to clear and stay cleared until the Stop mode is exited and the PLL has reacquired lock after the first sample. 0 PLL is currently unlocked. 1 PLL is currently locked after first sample.
5–4 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

35.3.14 MCG Control 12 Register (MCG_C12)

Address: 4006_4000h base + 11h offset = 4006_4011h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C12 field descriptions

Field	Description
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

35.3.14 MCG Status 2 Register (MCG_S2)

Address: 4006_4000h base + 12h offset = 4006_4012h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

MCG_S2 field descriptions

Field	Description
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

35.3.14 MCG Test 3 Register (MCG_T3)

Address: 4006_4000h base + 13h offset = 4006_4013h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

MCG_T3 field descriptions

Field	Description
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

35.4 Functional description

35.4.1 MCG mode state diagram

The eleven states of the MCG are shown in the following figure and are described in Table 35-1. The arrows indicate the permitted MCG mode transitions.

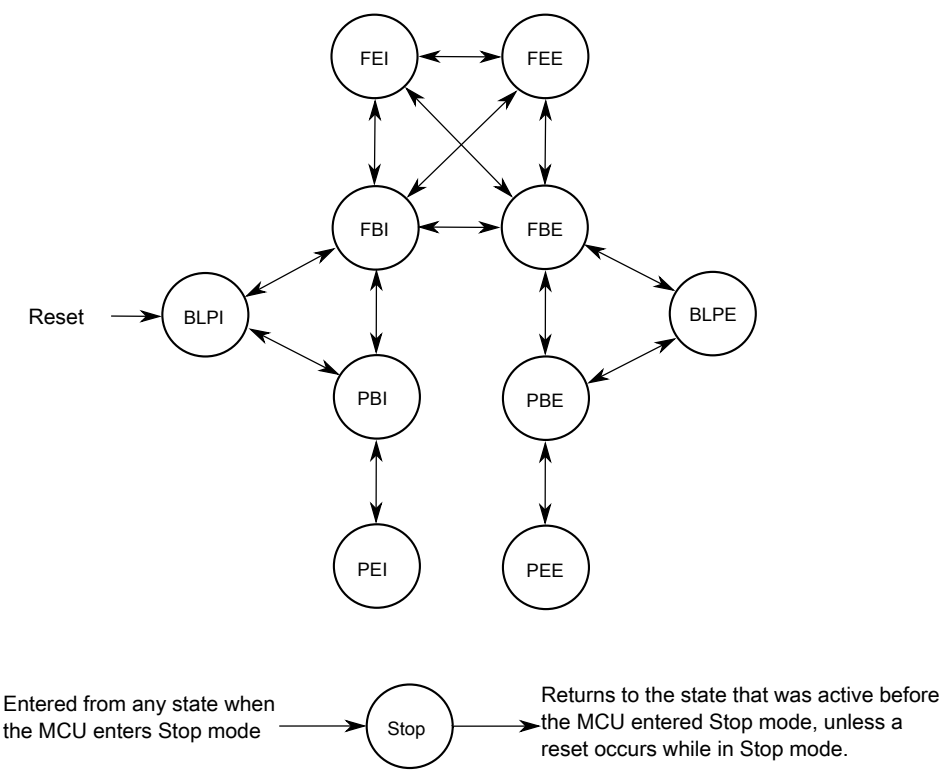


Figure 35-2. MCG mode state diagram

NOTE

- During exits from VLPS when the MCG is in PEE mode, the MCG will reset to PBE clock mode and the C1[CLKS] and S[CLKST] will automatically be set to 2’b10.

  During exits from VLPS when the MCG is in PEI mode, the MCG will reset to PBI clock mode and the C1[CLKS] and S[CLKST] will automatically be set to 2’b01 and C5[PLLSTEN0] will automatically be set to 1'b0.

- If entering Normal Stop mode when the MCG is in PEE mode with PLLSTEN=0, the MCG will reset to PBE clock mode and C1[CLKS] and S[CLKST] will automatically be set to 2’b10.

  If entering Normal Stop mode when the MCG is in PEI mode with C5[PLLSTEN]=0, the MCG will reset to PBI clock mode and C1[CLKS] and S[CLKST] will automatically be set to 2’b01.

35.4.1.1 MCG modes of operation

The MCG operates in one of the following modes.

Note

The MCG restricts transitions between modes. For the permitted transitions, see Figure 35-2.

Table 35-1. MCG modes of operation

Mode	Description
FLL Engaged Internal (FEI)	FLL engaged internal (FEI) is entered when all the following condtions occur: • 00 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 0 is written to C6[PLLS]. In FEI mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the 32 kHz Internal Reference Clock (IRC). The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FEI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
FLL Engaged External (FEE)	FLL engaged external (FEE) mode is entered when all the following conditions occur: • 00 is written to C1[CLKS].

Table continues on the next page...

Table 35-1. MCG modes of operation (continued)

Mode	Description
	• 0 is written to C1[IREFS]. • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz • 0 is written to C6[PLLS]. In FEE mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the external reference clock. The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the external reference frequency, as specified by C1[FRDIV] and C2[RANGE]. See the C4[DMX32] bit description for more details. In FEE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
FLL Bypassed Internal (FBI)	FLL bypassed internal (FBI) mode is entered when all the following conditions occur: • 01 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 0 is written to C6[PLLS] • 0 is written to C2[LP]. In FBI mode, the MCGOUTCLK is derived either from the slow (32 kHz IRC) or fast (4 MHz IRC) internal reference clock, as selected by the C2[IRCS] bit. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the C2[IRCS] selected internal reference clock. The FLL clock (DCOCLK) is controlled by the slow internal reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FBI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
PLL Engaged Internal (PEI)	PLL Engaged Internal (PEI) mode is entered when all the following conditions occur: • 00 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 1 is written to C6[PLLS]. • 01 is written to C7[PLL32KREFSEL]. In PEI mode, the MCGOUTCLK is derived from the PLL clock, which is controlled by the 32 kHz IRC reference clock. The PLL clock frequency locks to a multiplication factor times the internal reference frequency. The FLL is disabled in a low-power state.
PLL Bypassed Internal (PBI)	PLL Bypassed Internal (PBI) mode is entered when all the following conditions occur: • 01 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 1 is written to C6[PLLS]. • 01 is written to C7[PLL32KREFSEL]. • 0 is written to C2[LP]. In PBI mode, MCGOUTCLK is derived from the IRCS clock select; the PLL is operational, but its output clock is not used. This mode is useful to allow the PLL to acquire its target frequency while MCGOUTCLK is driven from the IRCS selected reference clock. The PLL clock frequency locks to a

Table continues on the next page...

Table 35-1. MCG modes of operation (continued)

Mode	Description
	multiplication factor times the PLL reference frequency. In preparation for transition to PEI, the PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.
FLL Bypassed External (FBE)	FLL bypassed external (FBE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz. • 0 is written to C6[PLLS]. • 0 is written to C2[LP]. In FBE mode, the MCGOUTCLK is derived from the OSCSEL external reference clock. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the external reference clock. The FLL clock (DCOCLK) is controlled by the external reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the divided external reference frequency. See the C4[DMX32] bit description for more details. In FBE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
PLL Engaged External (PEE)	PLL Engaged External (PEE) mode is entered when all the following conditions occur: • 00 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C6[PLLS]. • 00 or 10 is written to C7[PLL32KREFSEL]. In PEE mode, the MCGOUTCLK is derived from the output of PLL which is controlled by a selectable external or internal reference clock. The PLL clock frequency locks to a multiplication factor of 375 times the selected PLL reference frequency. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.
PLL Bypassed External (PBE)	PLL Bypassed External (PBE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C6[PLLS]. • 10 is written to C7[PLL32KREFSEL] • 0 is written to C2[LP]. In PBE mode, MCGOUTCLK is derived from the OSCSEL external reference clock; the PLL is operational, but its output clock is not used. This mode is useful to allow the PLL to acquire its target frequency while MCGOUTCLK is driven from the external reference clock. The PLL clock frequency locks to a multiplication factor of 375 times the PLL reference frequency. The FLL is disabled in a low-power state.
Bypassed Low Power Internal (BLPI) [1]	Bypassed Low Power Internal (BLPI) mode is entered when all the following conditions occur: • 01 is written to C1[CLKS]. • 1 is written to C1[IREFS].

Table continues on the next page...

Table 35-1. MCG modes of operation (continued)

Mode	Description
	• 0 is written to C6[PLLS]. • 1 is written to C2[LP]. In BLPI mode, MCGOUTCLK is derived from the internal reference clock. The FLL is disabled and PLL is disabled even if C5[PLLCLKEN] is set to 1.
Bypassed Low Power External (BLPE) [1]	Bypassed Low Power External (BLPE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C2[LP]. In BLPE mode, MCGOUTCLK is derived from the OSCSEL external reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN] is set to 1.
Stop	Entered whenever the MCU enters a Stop state. The power modes are chip specific. For power mode assignments, see the chapter that describes how modules are configured and MCG behavior during Stop recovery. Entering Stop mode, the FLL is disabled, and all MCG clock signals are static except in the following case: MCGPLLCLK is active in Normal Stop mode when PLLSTEN=1 MCGIRCLK is active in Normal Stop mode when all the following conditions become true: • C1[IRCLKEN] = 1 • C1[IREFSTEN] = 1 **NOTE:** • In VLPS Stop Mode, the MCGIRCLK can be programmed to stay enabled and continue running if C1[IRCLKEN] = 1, C1[IREFSTEN]=1, and Fast IRC clock is selected (C2[IRCS] = 1) **NOTE:** • When entering Low Power Stop mode (VLPS) from a PLL Enganged mode (PEE or PEI), on exit the MCG clock mode is forced to PBE clock mode if entering from PEE mode or to PBI if entering from PEI mode. C1[CLKS] and S[CLKST] will be configured to 2'b10if entering from PEE mode or to 2'b01 if entering from PEI mode, C5[PLLSTEN0] will be force to 1'b0 and S[LOCK] bit will be cleared without setting S[LOLS]. • When entering Normal Stop mode from a PLL Engaged mode (PEE or PEI) and if C5[PLLSTEN0]=0, on exit the MCG clock mode is forced to PBE mode if entered from PEE mode or PBI if entered from PBI mode,, the C1[CLKS] and S[CLKST] will be configured to 2'b10 if entering from PEE mode or to 2'b01 if entering from PEI mode and S[LOCK] bit will clear without setting S[LOLS]. If C5[PLLSTEN]=1, the S[LOCK] bit will not get cleared and on exit the MCG will continue to run in PEE mode.

1. **Caution:** If entering VLPR mode, MCG has to be configured and enter BLPE mode or BLPI mode with the Fast IRC clock selected (C2[IRCS]=1). After it enters VLPR mode, writes to any of the MCG control registers that can cause an MCG clock mode switch to a non low power clock mode must be avoided.

NOTE

For the chip-specific modes of operation, see the power management chapter of this MCU.

35.4.1.2 MCG mode switching

C1[IREFS] can be changed at any time, but the actual switch to the newly selected reference clocks is shown by S[IREFST]. When switching between engaged internal and engaged external modes, the FLL will begin locking again after the switch is completed.

C1[CLKS] can also be changed at any time, but the actual switch to the newly selected clock is shown by S[CLKST]. If the newly selected clock is not available, the previous clock will remain selected.

The C4[DRST_DRS] write bits can be changed at any time except when C2[LP] bit is 1. If C4[DRST_DRS] write bits are changed while in FLL engaged internal (FEI) or FLL engaged external (FEE) mode, the MCGOUTCLK switches to the new selected DCO range within three clocks of the selected DCO clock. After switching to the new DCO (indicated by the updated C4[DRST_DRS] read bits), the FLL remains unlocked for several reference cycles. The FLL lock time is provided in the device data sheet as $t_{fll\_acquire}$.

35.4.2 Low-power bit usage

C2[LP] is provided to allow the FLL or PLL to be disabled and thus conserve power when these systems are not being used. C4[DRST_DRS] can not be written while C2[LP] is 1. However, in some applications, it may be desirable to enable the FLL or PLL and allow it to lock for maximum accuracy before switching to an engaged mode. Do this by writing 0 to C2[LP].

35.4.3 MCG Internal Reference Clocks

This module supports two internal reference clocks with nominal frequencies of 32 kHz (slow IRC) and 4 MHz (fast IRC). The fast IRC frequency can be divided down by programming of the FCRDIV to produce a frequency range of 32 kHz to 4 MHz.

35.4.3.1 MCG Internal Reference Clock

The MCG Internal Reference Clock (MCGIRCLK) provides a clock source for other on-chip peripherals and is enabled when C1[IRCLKEN]=1. When enabled, MCGIRCLK is driven by either the fast internal reference clock (4 MHz IRC which can be divided down by the FRDIV factors) or the slow internal reference clock (32 kHz IRC). The IRCS

clock frequency can be re-targeted by trimming the period of its IRCS selected internal reference clock. This can be done by writing a new trim value to the C3[SCTRIM]:C4[SCFTRIM] bits when the slow IRC clock is selected or by writing a new trim value to C4[FCTRIM] when the fast IRC clock is selected. The internal reference clock period is proportional to the trim value written. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) and C4[FCTRIM] (if C2[IRCS]=1) bits affect the MCGOUTCLK frequency if the MCG is in FBI or BLPI modes. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) bits also affect the MCGOUTCLK frequency if the MCG is in FEI mode.

Additionally, this clock can be enabled in Stop mode by setting C1[IRCLKEN] and C1[IREFSTEN], otherwise this clock is disabled in Stop mode.

35.4.4 External Reference Clock

The MCG module can support an external reference clock in all modes. See the device datasheet for external reference frequency range. When C1[IREFS] is set, the external reference clock will not be used by the FLL or PLL. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications will support.

If any of the CME bits are asserted the slow internal reference clock is enabled along with the enabled external clock monitor. For the case when C6[CME0]=1, a loss of clock is detected if the OSC0 external reference falls below a minimum frequency (f_{loc_high} or f_{loc_low} depending on C2[RANGE0]).

NOTE

All clock monitors must be disabled before entering these low-power modes: Stop, VLPS, VLPR, VLPW, and VLLSx.

On detecting a loss-of-clock event, the MCU generates a system reset if the respective LOCRE bit is set. Otherwise the MCG sets the respective LOCS bit and the MCG generates a LOCS interrupt request.

35.4.5 MCG Fixed Frequency Clock

The MCG Fixed Frequency Clock (MCGFFCLK) provides a fixed frequency clock source for other on-chip peripherals; see the block diagram. This clock is driven by either the slow clock from the internal reference clock generator or the external reference clock from the Crystal Oscillator, divided by the FLL reference clock divider. The source of MCGFFCLK is selected by C1[IREFS].

This clock is synchronized to the peripheral bus clock and is valid only when its frequency is not more than 1/8 of the MCGOUTCLK frequency. When it is not valid, it is disabled and held high. The MCGFFCLK is not available when the MCG is in BLPI mode. This clock is also disabled in Stop mode. The FLL reference clock must be set within the valid frequency range for the MCGFFCLK.

35.4.6 MCG PLL clock

The MCG PLL Clock (MCGPLLCLK) is available depending on the device's configuration of the MCG module. For more details, see the clock distribution chapter of this MCU. The MCGPLLCLK is prevented from coming out of the MCG until it is enabled and S[LOCK0] is set.

35.4.7 MCG Auto TRIM (ATM)

The MCG Auto Trim (ATM) is a MCG feature that when enabled, it configures the MCG hardware to automatically trim the MCG Internal Reference Clocks using an external clock as a reference. The selection between which MCG IRC clock gets tested and enabled is controlled by the ATC[ATMS] control bit (ATC[ATMS]=0 selects the 32 kHz IRC and ATC[ATMS]=1 selects the 4 MHz IRC). If 4 MHz IRC is selected for the ATM, a divide by 128 is enabled to divide down the 4 MHz IRC to a range of 31.250 kHz.

When MCG ATM is enabled by writing ATC[ATME] bit to 1, The ATM machine will start auto trimming the selected IRC clock. During the autotrim process, ATC[ATME] will remain asserted and will deassert after ATM is completed or an abort occurs. The MCG ATM is aborted if a write to any of the following control registers is detected : C1, C3, C4, or ATC or if Stop mode is entered. If an abort occurs, ATC[ATMF] fail flag is asserted.

The ATM machine uses the bus clock as the external reference clock to perform the IRC auto-trim. Therefore, it is required that the MCG is configured in a clock mode where the reference clock used to generate the system clock is the external reference clock such as FBE clock mode. The MCG must not be configured in a clock mode where selected IRC ATM clock is used to generate the system clock. The bus clock is also required to be running with in the range of 8–16 MHz.

To perform the ATM on the selected IRC, the ATM machine uses the successive approximation technique to adjust the IRC trim bits to generate the desired IRC trimmed frequency. The ATM SARs each of the ATM IRC trim bits starting with the MSB. For each trim bit test, the ATM uses a pulse that is generated by the ATM selected IRC clock to enable a counter that counts number of ATM external clocks. At end of each trim bit,

the ATM external counter value is compared to the ATCV[15:0] register value. Based on the comparison result, the ATM trim bit under test will get cleared or stay asserted. This is done until all trim bits have been tested by ATM SAR machine.

Before the ATM can be enabled, the ATM expected count needs to be derived and stored into the ATCV register. The ATCV expected count is derived based on the required target Internal Reference Clock (IRC) frequency, and the frequency of the external reference clock using the following formula:

$$\text{ATCV Expected Count Value} = 21*(Fe/Fr)$$

- Fr = Target Internal Reference Clock (IRC) Trimmed Frequency
- Fe = External Clock Frequency

If the auto trim is being performed on the 4 MHz IRC, the calculated expected count value must be multiplied by 128 before storing it in the ATCV register. Therefore, the ATCV Expected Count Value for trimming the 4 MHz IRC is calculated using the following formula.

$$\text{Expected Count Value} = (Fe/Fr)*21*(128)$$

35.5 Initialization / Application information

This section describes how to initialize and configure the MCG module in an application.

The following sections include examples on how to initialize the MCG and properly switch between the various available modes.

35.5.1 MCG module initialization sequence

The MCG comes out of reset configured for BLPI with Fast IRC selected mode.

The internal reference will stabilize in $t_{irefsts}$ microseconds.

35.5.1.1 Initializing the MCG

Because the MCG comes out of reset in FEI mode, the only MCG modes that can be directly switched to upon reset are FEE, FBE, and FBI modes (see Figure 35-2). Reaching any of the other modes requires first configuring the MCG for one of these three intermediate modes. Care must be taken to check relevant status bits in the MCG status register reflecting all configuration changes within each mode.

To change from FEI mode to FEE or FBE modes, follow this procedure:

1. Enable the external clock source by setting the appropriate bits in C2 register.
2. Write to C1 register to select the clock mode.
   - If entering FEE mode, set C1[FRDIV] appropriately, clear C1[IREFS] bit to switch to the external reference, and leave C1[CLKS] at 2'b00 so that the output of the FLL is selected as the system clock source.
   - If entering FBE, clear C1[IREFS] to switch to the external reference and change C1[CLKS] to 2'b10 so that the external reference clock is selected as the system clock source. The C1[FRDIV] bits should also be set appropriately here according to the external reference frequency to keep the FLL reference clock in the range of 31.25 kHz to 39.0625 kHz. Although the FLL is bypassed, it is still on in FBE mode.
   - The internal reference can optionally be kept running by setting C1[IRCLKEN]. This is useful if the application will switch back and forth between internal and external modes. For minimum power consumption, leave the internal reference disabled while in an external clock mode.
3. Once the proper configuration bits have been set, wait for the affected bits in the MCG status register to be changed appropriately, reflecting that the MCG has moved into the proper mode.
   - If the MCG is in FEE, FBE, PEE, PBE, or BLPE mode, and C2[EREFS] was also set in step 1, wait here for S[OSCINIT0] bit to become set indicating that the external clock source has finished its initialization cycles and stabilized.
   - If in FEE mode, check to make sure S[IREFST] is cleared before moving on.
   - If in FBE mode, check to make sure S[IREFST] is cleared and S[CLKST] bits have changed to 2'b10 indicating the external reference clock has been appropriately selected. Although the FLL is bypassed, it is still on in FBE mode.
4. Write to the C4 register to determine the DCO output (MCGFLLCLK) frequency range.
   - By default, with C4[DMX32] cleared to 0, the FLL multiplier for the DCO output is 640. For greater flexibility, if a mid-low-range FLL multiplier of 1280 is desired instead, set C4[DRST_DRS] bits to 2'b01 for a DCO output frequency of 40 MHz. If a mid high-range FLL multiplier of 1920 is desired instead, set the C4[DRST_DRS] bits to 2'b10 for a DCO output frequency of 60 MHz. If a high-range FLL multiplier of 2560 is desired instead, set the C4[DRST_DRS] bits to 2'b11 for a DCO output frequency of 80 MHz.

- When using a 32.768 kHz external reference, if the maximum low-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b00 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 732 will be 24 MHz.
- When using a 32.768 kHz external reference, if the maximum mid-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b01 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 1464 will be 48 MHz.
- When using a 32.768 kHz external reference, if the maximum mid high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b10 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2197 will be 72 MHz.
- When using a 32.768 kHz external reference, if the maximum high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b11 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2929 will be 96 MHz.

5. Wait for the FLL lock time to guarantee FLL is running at new C4[DRST_DRS] and C4[DMX32] programmed frequency.

To change from FEI clock mode to FBI clock mode, follow this procedure:

1. Change C1[CLKS] bits in C1 register to 2'b01 so that the internal reference clock is selected as the system clock source.
2. Wait for S[CLKST] bits in the MCG status register to change to 2'b01, indicating that the internal reference clock has been appropriately selected.
3. Write to the C2 register to determine the IRCS output (IRCSCLK) frequency range.
   - By default, with C2[IRCS] cleared to 0, the IRCS selected output clock is the slow internal reference clock (32 kHz IRC). If the faster IRC is desired, set C2[IRCS] to 1 for a IRCS clock derived from the 4 MHz IRC source.

35.5.2 Using a 32.768 kHz reference

In FEE and FBE modes, if using a 32.768 kHz external reference, at the default FLL multiplication factor of 640, the DCO output (MCGFLLCLK) frequency is 20.97 MHz at low-range.

If C4[DRST_DRS] bits are set to 2'b01, the multiplication factor is doubled to 1280, and the resulting DCO output frequency is 41.94 MHz at mid-low-range. If C4[DRST_DRS] bits are set to 2'b10, the multiplication factor is set to 1920, and the resulting DCO output frequency is 62.91 MHz at mid high-range. If C4[DRST_DRS] bits are set to 2'b11, the multiplication factor is set to 2560, and the resulting DCO output frequency is 83.89 MHz at high-range.

In FBI and FEI modes, setting C4[DMX32] bit is not recommended. If the internal reference is trimmed to a frequency above 32.768 kHz, the greater FLL multiplication factor could potentially push the microcontroller system clock out of specification and damage the part.

35.5.3 MCG mode switching

When switching between operational modes of the MCG, certain configuration bits must be changed in order to properly move from one mode to another.

Each time any of these bits are changed (C6[PLLS], C1[IREFS], C1[CLKS], C2[IRCS], or C2[EREFS], the corresponding bits in the MCG status register (PLLST, IREFST, CLKST, IRCST, or OSCINIT) must be checked before moving on in the application software.

Additionally, care must be taken to ensure that the reference clock divider (C1[FRDIV]) is set properly for the mode being switched to.

In FBE, FEE, FBI, and FEI modes, at any time, the application can switch the FLL multiplication factor between 640, 1280, 1920, and 2560 with C4[DRST_DRS] bits. Writes to C4[DRST_DRS] bits will be ignored if C2[LP]=1.

The table below shows MCGOUTCLK frequency calculations using C1[FRDIV]for each clock mode.

Table 35-2. MCGOUTCLK Frequency Calculation Options

Clock Mode	$f_{MCGOUTCLK}$[1]	Note
FEI (FLL engaged internal)	$f_{int} \times F$	Typical $f_{MCGOUTCLK}$ = 21 MHz immediately after reset.
FEE (FLL engaged external)	$(f_{ext} / FLL\_R) \times F$	f_{ext} / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz

Table continues on the next page...

Table 35-2. MCGOUTCLK Frequency Calculation Options (continued)

Clock Mode	$f_{MCGOUTCLK}$[1]	Note
FBE (FLL bypassed external)	OSCCLK	OSCCLK / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz
FBI (FLL bypassed internal)	MCGIRCLK	Selectable between slow and fast IRC
PEE (PLL engaged external)	OSCCLK × 375	OSCCLK must be in the range of 31.25 to 39.0625 KHz
PBE (PLL bypassed external)	OSCCLK	OSCCLK must be in the range of 31.25 to 39.0625 KHz
PEI (PLL engaged internal)	MCGIRCLK × 375	MCGIRCLK must be in the range of 32.125 to 39.0625 KHz
PBI (PLL bypassed internal)	MCGIRCLK	MCGIRCLK must be in the range of 32.125 to 39.0625 KHz
BLPI (Bypassed low power internal)	MCGIRCLK	Selectable between slow and fast IRC
BLPE (Bypassed low power external)	OSCCLK	

1. FLL_R is the reference divider selected by the C1[FRDIV] bits, F is the FLL factor selected by C4[DRST_DRS] and C4[DMX32] bits .

This section will include several mode switching examples.

35.5.3.1 Example 1: Moving from BLPI to PEE mode with OSC as the source for the external crystal clock: External Crystal = 32 KHz, MCGOUTCLK frequency = 12 MHz

In this example, the MCG will move through the proper operational modes from BLPI to PEE to achieve 12 MHz MCGOUTCLK frequency from 32 KHz external crystal reference. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, BLPI must transition to FBI mode:
   a. C2 = 0x81
      - C2[LP] set to 0.
2. Then, FBI must transition directly to FBE mode:
   a. C1 = 0x80
      - C1[CLKS] set to 2'b10.

- C1[IREFS] set to 1'b0 to select the output of the OSCSEL mux as the external reference clock. In this example because OSCSEL is set to 1'b0, the OSC0 will be the selected as the reference clock.
- FBE: Loop until S[CLKST] are set to 2'b10 and S[IREFST] is set to 1'b0.

3. Then, FBE must transition either directly to PBE mode or first through BLPE mode and then to PBE mode:
   a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.
   b. BLPE/PBE: C6 = 0x44
      - C6[PLLS] set to 1, selects the PLL. In BLPE mode, changing the C6[PLLS] bit only prepares the MCG for PLL usage in PBE mode.
      - C6[CHGPMP_BIAS] set to 5'b0_0100. Appropiate selection of this value is imperative to ensure stable operation of the PLL closed loop system. It is recommended to keep the reset value and not change them.
   c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to PBE mode.
   d. PBE: Loop until S[PLLST] is set, indicating that the current source for the PLLS clock is the PLL.
   e. PBE: Then loop until MCG_C9[COARSE_LOCK] is set, indicating that the PLL has aquired a coarse lock status or loop until S[LOCK] is set, indicating that the PLL has acquired lock for three samples.
4. Lastly, PBE mode transitions into PEE mode:
   a. C1 = 0x00
      - C1[CLKS] set to 2'b00 to select the output of the PLL as the system clock source.
   b. Loop until S[CLKST] are 2'b11, indicating that the PLL output is selected to feed MCGOUTCLK in the current clock mode.
      - Now MCGOUTCLK = [(32 kHz) * 375] = 12 MHz.

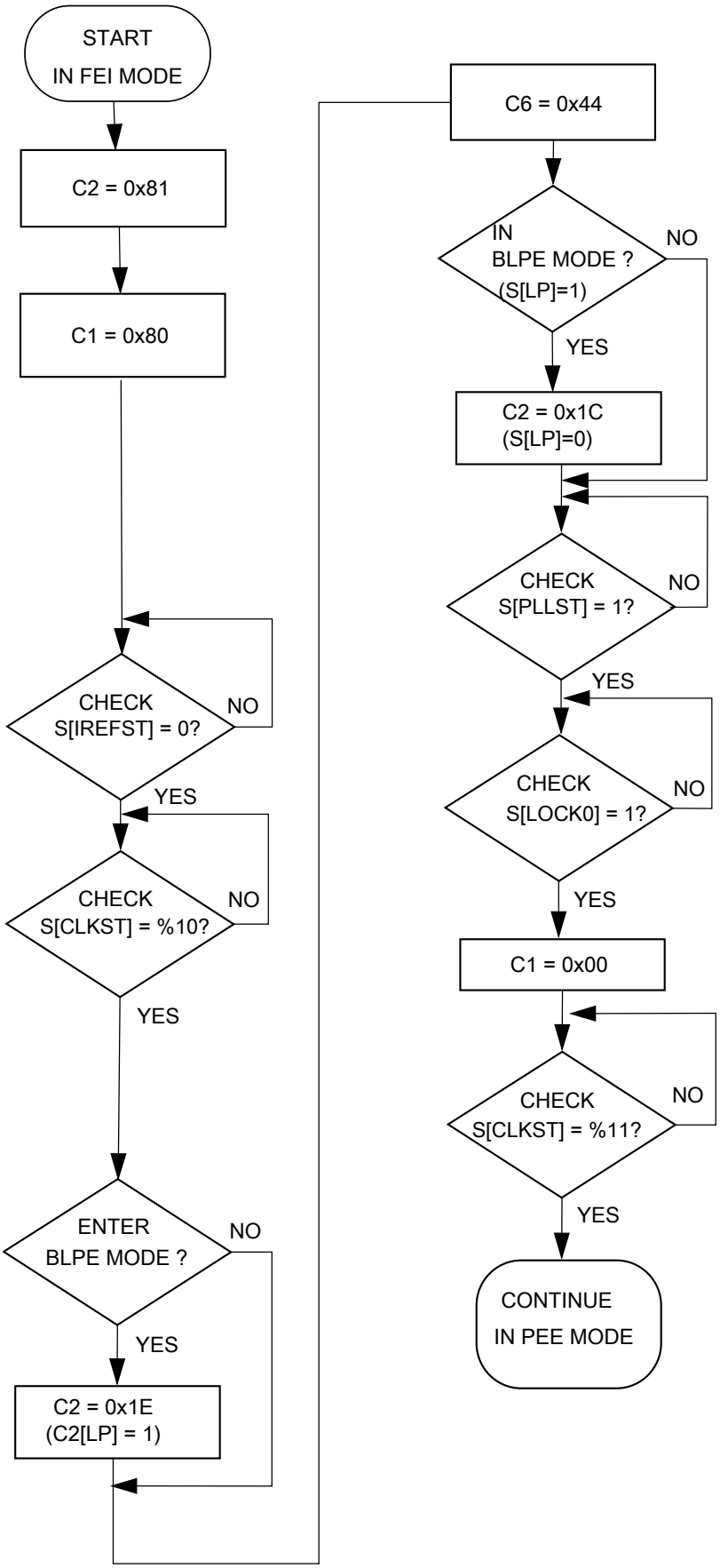


Figure 35-3. Flowchart of BLPI to PEE mode transition using an 32 KHz crystal

35.5.3.2 Example 2: Moving from PEE to BLPI mode: MCGOUTCLK frequency =32 kHz

In this example, the MCG will move through the proper operational modes from PEE mode with a 4 MHz crystal configured for a 48 MHz MCGOUTCLK frequency (see previous example) to BLPI mode with a 32 kHz MCGOUTCLK frequency. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, PEE must transition to PBE mode:
   a. C1 = 0x90
      - C1[CLKS] set to 2'b10 to switch the system clock source to the external reference clock.
   b. Loop until S[CLKST] are 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then, PBE must transition either directly to FBE mode or first through BLPE mode and then to FBE mode:
   a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.
   b. BLPE/FBE: C6 = 0x00
      - C6[PLLS] clear to 0 to select the FLL. At this time, with C1[FRDIV] value of 3'b010, the FLL divider is set to 128, resulting in a reference frequency of 4 MHz / 128 = 31.25 kHz. If C1[FRDIV] was not previously set to 3'b010 (necessary to achieve required 31.25–39.06 kHz FLL reference frequency with an 4 MHz external source frequency), it must be changed prior to clearing C6[PLLS] bit. In BLPE mode,changing this bit only prepares the MCG for FLL usage in FBE mode.
   c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to FBE mode.
   d. FBE: Loop until S[PLLST] is cleared, indicating that the current source for the PLLS clock is the FLL.
3. Next, FBE mode transitions into FBI mode:
   a. C1 = 0x54
      - C1[CLKS] set to 2'b01 to switch the system clock to the internal reference clock.

- C1[IREFS] set to 1 to select the internal reference clock as the reference clock source.
- C1[FRDIV] remain unchanged because the reference divider does not affect the internal reference.

b. Loop until S[IREFST] is 1, indicating the internal reference clock has been selected as the reference clock source.

c. Loop until S[CLKST] are 2'b01, indicating that the internal reference clock is selected to feed MCGOUTCLK.

4. Lastly, FBI transitions into BLPI mode.

a. C2 = 0x02

- C2[LP] is 1
- C2[RANGE], C2[HGO], C2[EREFS], C1[IRCLKEN], and C1[IREFSTEN] bits are ignored when the C1[IREFS] bit is set. They can remain set, or be cleared at this point.

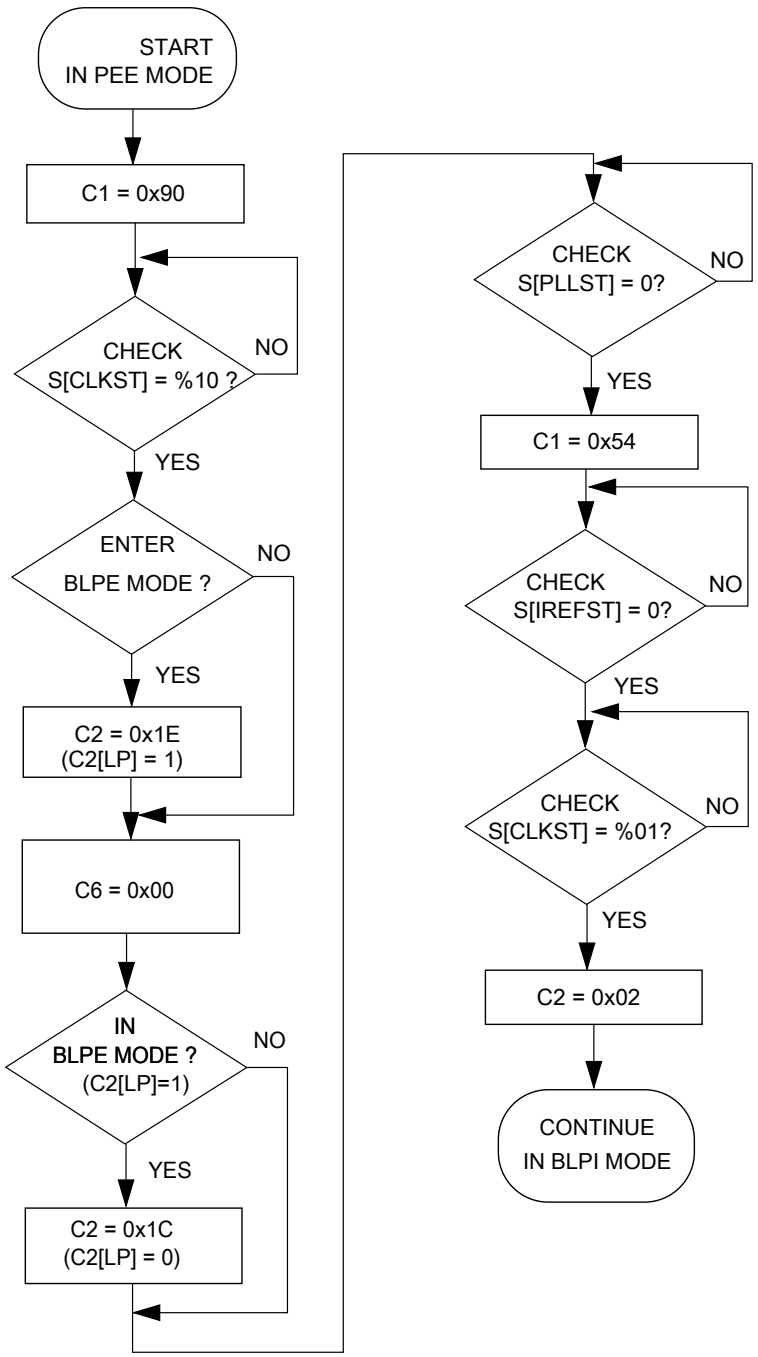


Figure 35-4. Flowchart of PEE to BLPI mode transition using an 4 MHz crystal

Chapter 36
Oscillator (OSC)

36.1 Introduction

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

36.2 Features and Modes

Key features of the module are listed here.

- Supports 32 kHz crystals (Low Range mode)
- Supports 1–8 MHz, crystals and resonators (High Range mode)
- Automatic Gain Control (AGC) to optimize power consumption in high frequency ranges 1–8 MHz, using low-power mode
- High gain option in frequency ranges: 32 kHz, 1–8 MHz,
- Voltage and frequency filtering to guarantee clock frequency and stability
- Optionally external input bypass clock from EXTAL signal directly
- One clock for MCU clock system
- Two clocks for on-chip peripherals that can work in Stop modes

Functional Description describes the module's operation in more detail.

36.3 Block Diagram

The OSC module uses a crystal or resonator to generate three filtered oscillator clock signals.Three clocks are output from OSC module: OSCCLK for MCU system, OSCERCLK for on-chip peripherals, and ERCLK32K. The OSCCLK can only work in run mode. OSCERCLK and ERCLK32K can work in low power modes. For the clock source assignments, refer to the clock distribution information of this MCU.

Refer to the chip configuration details for the external reference clock source in this MCU.

The figure found here shows the block diagram of the OSC module.

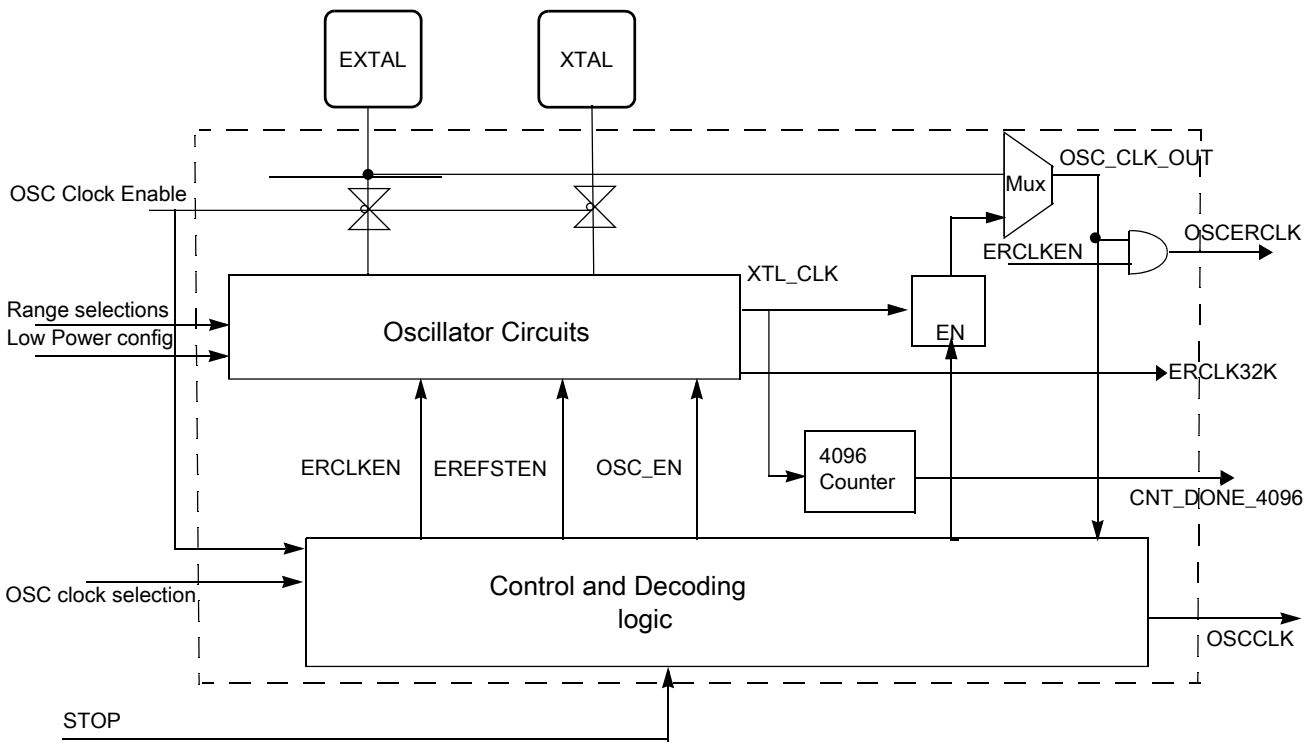


Figure 36-1. OSC Module Block Diagram

36.4 OSC Signal Descriptions

The table found here shows the user-accessible signals available for the OSC module.

Refer to signal multiplexing information for this MCU for more details.

Table 36-1. OSC Signal Descriptions

Signal	Description	I/O
EXTAL	External clock/Oscillator input	I
XTAL	Oscillator output	O

36.5 External Crystal / Resonator Connections

The connections for a crystal/resonator frequency reference are shown in the figures found here.

When using low-frequency, low-power mode, the only external component is the crystal or ceramic resonator itself. In the other oscillator modes, load capacitors (C_x, C_y) and feedback resistor (R_F) are required. In addition, a series resistor (R_S) may be used in high-gain modes. The following table shows all possible connections.

Table 36-2. External Caystal/Resonator Connections

Oscillator Mode	Connections
Low-frequency (32 kHz), low-power	Connection 1[1]
Low-frequency (32 kHz), high-gain	Connection 2/Connection 4[2] / Connection 3[3]
High-frequency (1~32 MHz), low-power	Connection 3[1]
High-frequency (1~32 MHz), high-gain	Connection 3/Connection 4[2]

1. With the low-power mode, the oscillator has the internal feedback resistor R_F. Therefore, the feedback resistor must not be externally with the Connection 3.
2. When the frequency of the crystal is less than 2 MHz and the load capacitors (Cx, Cy) are less than 16 pF, use Connection 4.
3. When the load capacitors (Cx, Cy) are greater than 30 pF, use Connection 3.

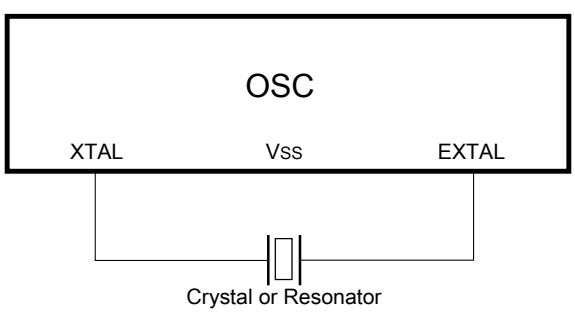


Figure 36-2. Crystal/Ceramic Resonator Connections - Connection 1

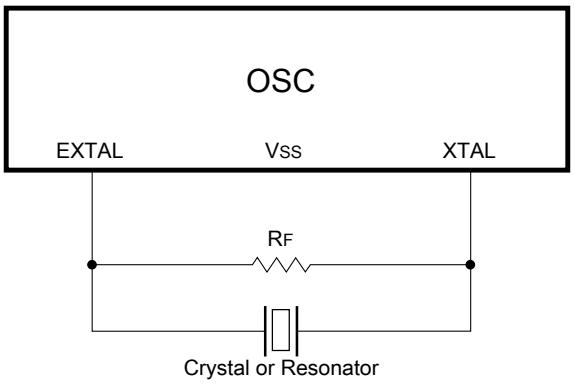


Figure 36-3. Crystal/Ceramic Resonator Connections - Connection 2

NOTE

Connection 1 and Connection 2 should use internal capacitors as the load of the oscillator by configuring the CR[SCxP] bits.

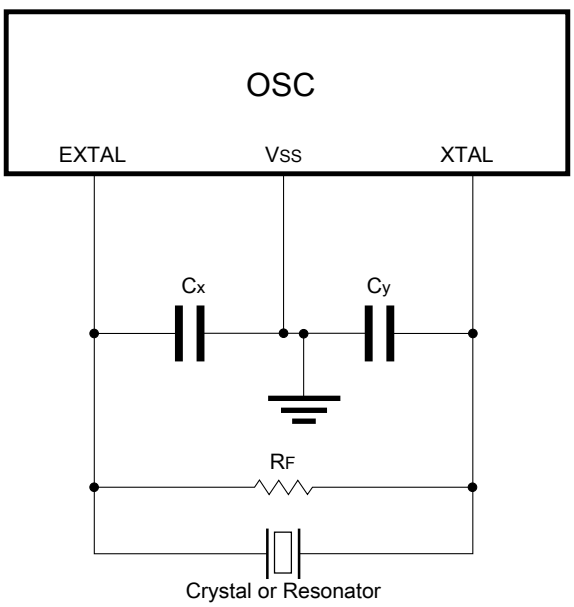


Figure 36-4. Crystal/Ceramic Resonator Connections - Connection 3

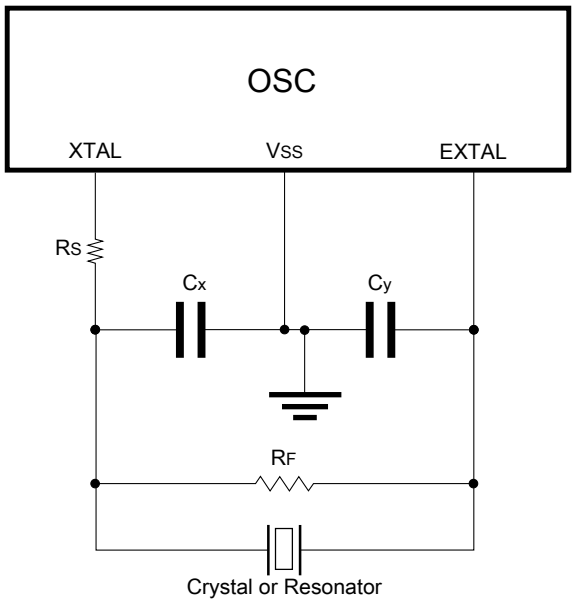


Figure 36-5. Crystal/Ceramic Resonator Connections - Connection 4

36.6 External Clock Connections

In external clock mode, the pins can be connected as shown in the figure found here.

NOTE

XTAL can be used as a GPIO when the GPIO alternate function is configured for it.

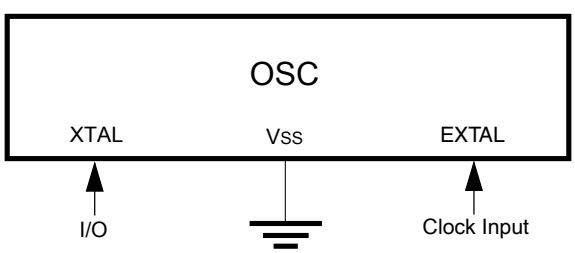


Figure 36-6. External Clock Connections

36.7 Memory Map/Register Definitions

Some oscillator module register bits are typically incorporated into other peripherals such as MCG or SIM.

36.7.1 OSC Memory Map/Register Definition

OSC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6000	OSC Control Register (OSC_CR)	8	R/W	00h	36.7.1.1/ 706

36.7.1.1 OSC Control Register (OSC_CR)

NOTE

After OSC is enabled and starts generating the clocks, the configurations such as low power and frequency range, must not be changed.

Address: 4006_6000h base + 0h offset = 4006_6000h

Bit	7	6	5	4	3	2	1	0
Read	ERCLKEN	0	EREFSTEN	0	SC2P	SC4P	SC8P	SC16P
Write								
Reset	0	0	0	0	0	0	0	0

OSC_CR field descriptions

Field	Description
7 ERCLKEN	External Reference Enable Enables external reference clock (OSCERCLK) . 0 External reference clock is inactive. 1 External reference clock is enabled.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 EREFSTEN	External Reference Stop Enable Controls whether or not the external reference clock (OSCERCLK) remains enabled when MCU enters Stop mode. 0 External reference clock is disabled in Stop mode. 1 External reference clock stays enabled in Stop mode if ERCLKEN is set before entering Stop mode.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SC2P	Oscillator 2 pF Capacitor Load Configure

Table continues on the next page...

OSC_CR field descriptions (continued)

Field	Description
	Configures the oscillator load. 0 Disable the selection. 1 Add 2 pF capacitor to the oscillator load.
2 SC4P	Oscillator 4 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 4 pF capacitor to the oscillator load.
1 SC8P	Oscillator 8 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 8 pF capacitor to the oscillator load.
0 SC16P	Oscillator 16 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 16 pF capacitor to the oscillator load.

36.8 Functional Description

Functional details of the module can be found here.

36.8.1 OSC module states

The states of the OSC module are shown in the following figure. The states and their transitions between each other are described in this section.

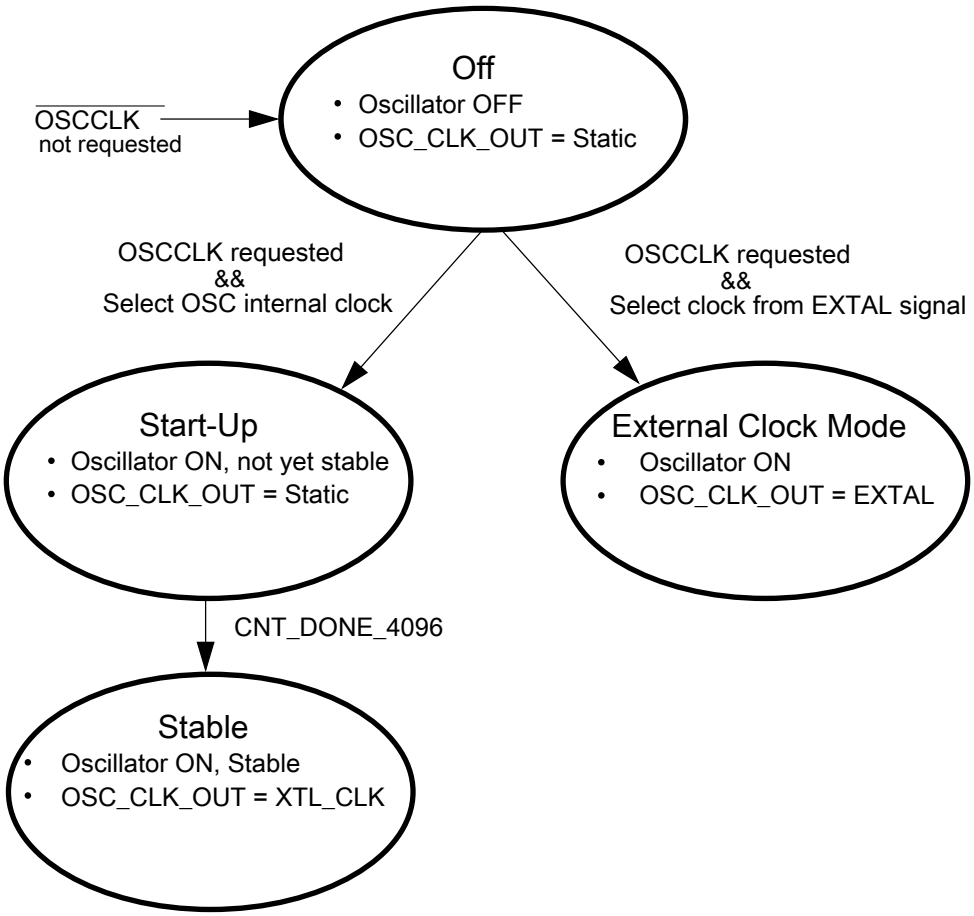


Figure 36-7. OSC Module state diagram

NOTE

XTL_CLK is the clock generated internally from OSC circuits.

36.8.1.1 Off

The OSC enters the Off state when the system does not require OSC clocks. Upon entering this state, XTL_CLK is static unless OSC is configured to select the clock from the EXTAL pad by clearing the external reference clock selection bit. For details regarding the external reference clock source in this MCU, refer to the chip configuration details. The EXTAL and XTAL pins are also decoupled from all other oscillator circuitry in this state. The OSC module circuitry is configured to draw minimal current.

36.8.1.2 Oscillator startup

The OSC enters startup state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit. In this state, the OSC module is enabled and oscillations are starting up, but have not yet stabilized. When the oscillation amplitude becomes large enough to pass through the input buffer, XTL_CLK begins clocking the counter. When the counter reaches 4096 cycles of XTL_CLK, the oscillator is considered stable and XTL_CLK is passed to the output clock OSC_CLK_OUT.

36.8.1.3 Oscillator Stable

The OSC enters stable state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit and the counter reaches 4096 cycles of XTL_CLK (when CNT_DONE_4096 is high). In this state, the OSC module is producing a stable output clock on OSC_CLK_OUT. Its frequency is determined by the external components being used.

36.8.1.4 External Clock mode

The OSC enters external clock state when it is enabled and external reference clock selection bit is cleared. For details regarding external reference clock source in this MCU, see the chip configuration details. In this state, the OSC module is set to buffer (with hysteresis) a clock from EXTAL onto the OSC_CLK_OUT. Its frequency is determined by the external clock being supplied.

36.8.2 OSC module modes

The OSC is a pierce-type oscillator that supports external crystals or resonators operating over the frequency ranges shown in Table 36-3. These modes assume the following conditions: OSC is enabled to generate clocks (OSC_EN=1), configured to generate clocks internally (MCG_C2[EREFS] = 1), and some or one of the other peripherals (MCG, Timer, and so on) is configured to use the oscillator output clock (OSC_CLK_OUT).

Table 36-3. Oscillator modes

Mode	Frequency Range
Low-frequency, high-gain	f_{osc_lo} (32.768 kHz) up to f_{osc_lo} (39.0625 kHz)

Table continues on the next page...

Table 36-3. Oscillator modes (continued)

Mode	Frequency Range
High-frequency mode1, high-gain	$f_{osc\_hi\_1}$ (1 MHz) up to $f_{osc\_hi\_1}$ (8 MHz)
High-frequency mode1, low-power	
High-frequency mode2, high-gain	$f_{osc\_hi\_2}$ (8 MHz) up to $f_{osc\_hi\_2}$ (32 MHz)
High-frequency mode2, low-power	

NOTE

For information about low power modes of operation used in this chip and their alignment with some OSC modes, see the chip's Power Management details.

36.8.2.1 Low-Frequency, High-Gain Mode

In Low-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes.

The oscillator input buffer in this mode is single-ended. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used.

36.8.2.2 Low-Frequency, Low-Power Mode

In low-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used, the internal feedback resistor is connected, and no external resistor should be used.

In this mode, the amplifier inputs, gain-control input, and input buffer input are all capacitively coupled for leakage tolerance (not sensitive to the DC level of EXTAL).

Also in this mode, all external components except for the resonator itself are integrated, which includes the load capacitors and feeback resistor that biases EXTAL.

36.8.2.3 High-Frequency, High-Gain Mode

In high-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels.

36.8.2.4 High-Frequency, Low-Power Mode

In high-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. In this mode, no external resistor should be used.

The oscillator input buffer in this mode is differential. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels.

36.8.3 Counter

The oscillator output clock (OSC_CLK_OUT) is gated off until the counter has detected 4096 cycles of its input clock (XTL_CLK). After 4096 cycles are completed, the counter passes XTL_CLK onto OSC_CLK_OUT. This counting timeout is used to guarantee output clock stability.

36.8.4 Reference clock pin requirements

The OSC module requires use of both the EXTAL and XTAL pins to generate an output clock in Oscillator mode, but requires only the EXTAL pin in External clock mode. The EXTAL and XTAL pins are available for I/O. For the implementation of these pins on this device, refer to the Signal Multiplexing chapter.

36.9 Reset

There is no reset state associated with the OSC module. The counter logic is reset when the OSC is not configured to generate clocks.

There are no sources of reset requests for the OSC module.

36.10 Low power modes operation

When the MCU enters Stop modes, the OSC is functional depending on CR[ERCLKEN] and CR[EREFSETN] bit settings. If both these bits are set, the OSC is in operation.

After waking up from Very Low Leakage Stop (VLLSx) modes, all OSC register bits are reset and initialization is required through software.

36.11 Interrupts

The OSC module does not generate any interrupts.

Chapter 37
RTC Oscillator (OSC32K)

37.1 Introduction

The RTC oscillator module provides the clock source for the RTC. The RTC oscillator module, in conjunction with an external crystal, generates a reference clock for the RTC.

37.1.1 Features and Modes

The key features of the RTC oscillator are as follows:

- Supports 32 kHz crystals with very low power
- Consists of internal feed back resistor
- Consists of internal programmable capacitors as the C_{load} of the oscillator
- Automatic Gain Control (AGC) to optimize power consumption

The RTC oscillator operations are described in detail in Functional Description .

37.1.2 Block Diagram

The following is the block diagram of the RTC oscillator.

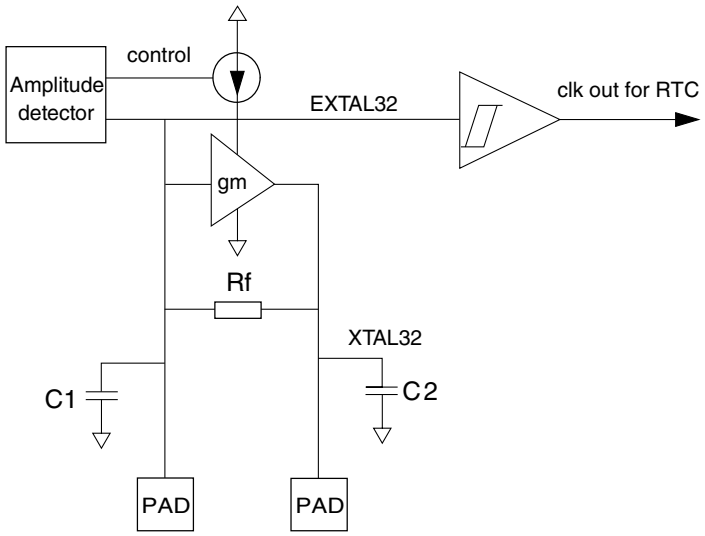


Figure 37-1. RTC Oscillator Block Diagram

37.2 RTC Signal Descriptions

The following table shows the user-accessible signals available for the RTC oscillator. See the chip-level specification to find out which signals are actually connected to the external pins.

Table 37-1. RTC Signal Descriptions

Signal	Description	I/O
EXTAL32	Oscillator Input	I
XTAL32	Oscillator Output	O

37.2.1 EXTAL32 — Oscillator Input

This signal is the analog input of the RTC oscillator.

37.2.2 XTAL32 — Oscillator Output

This signal is the analog output of the RTC oscillator module.

37.3 External Crystal Connections

The connections with a crystal is shown in the following figure. External load capacitors and feedback resistor are not required.

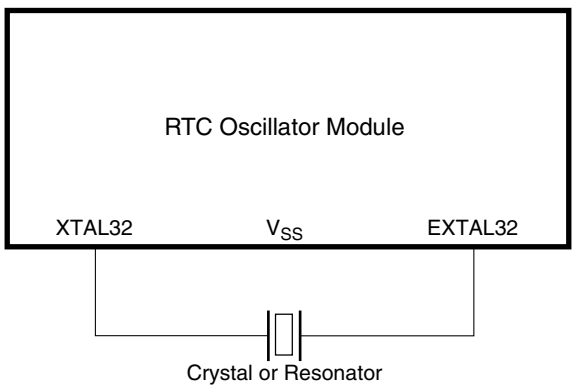


Figure 37-2. Crystal Connections

37.4 Memory Map/Register Descriptions

RTC oscillator control bits are part of the RTC registers. Refer to RTC Control register RTC_GP_DATA_REG in the chip-specific information section for more details.

37.5 Functional Description

As shown in -, the module includes an amplifier which supplies the negative resistor for the RTC oscillator. The gain of the amplifier is controlled by the amplitude detector, which optimizes the power consumption. A schmitt trigger is used to translate the sine-wave generated by this oscillator to a pulse clock out, which is a reference clock for the RTC digital core.

The oscillator includes an internal feedback resistor of approximately 100 MΩ between EXTAL32 and XTAL32.

In addition, there are two programmable capacitors with this oscillator, which can be used as the Cload of the oscillator. The programmable range is from 0pF to 30pF.

37.6 Reset Overview

There is no reset state associated with the RTC oscillator.

37.7 Interrupts

The RTC oscillator does not generate any interrupts.

Chapter 38
Independent Real Time Clock (iRTC)

38.1 Chip-specific iRTC information

38.1.1 Clock Sources

The iRTC has the following clock sources:

- **32 kHz Oscillator Clock -** This oscillator is in the RTC VBAT domain and will always be enabled and supplying clock to RTC. *This is the default clock source*.
- **32 kHz IRC -** This is the 32 kHz IRC present in MCG block of the SoC. Since this clock source is not present in the RTC VBAT domain, RTC can no longer keep time if VDD is powered OFF or in those low power modes where the IRC is OFF or disabled.

The logic for switching the clock will reside in the wrapper integrating the analog and digital iRTC blocks to be present in the RTC VBAT domain. The selection bits and oscillator bypass bits can be taken from the General Purpose Register inside the iRTC. The bits of this registers are available for use at the wrapper level.

NOTE

Since there is no way to disable/halt iRTC counters, there could be a glitch in operation when clock source is changed. Hence clock source should be selected before setting the time & date counters.

38.1.2 RTC General Purpose Data Register (RTC_GP_DATA_REG)

Bit	15	14	13	12	11	10	9	8
Read Write	CFG15	CFG14	CFG13	CFG12	CFG11	CFG10	CFG9	CFG8
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	CFG7	CFG6	CFG5	CFG4	CFG3	CFG2	CFG1	CFG0
Reset	0	0	0	0	0	0	0	0

Figure 38-1. RTC General Purpose Data Register

Table 38-1. RTC General Purpose Data Register Table

Bit Number	Description
CFG0	32 kHz RTC OSC Control. This bit can be used to disable clock when writing to RTC registers. '1'—Disables the oscillator '0'—Enables the oscillator
CFG1	Switched capacitor 2 pF enable. Enables 2 pF capacitor inside oscillator to be used as load capacitance. '1'—Enables capacitor '0'—Disables capacitor
CFG2	Switched capacitor 4 pF enable. Enables 4 pF capacitor inside oscillator to be used as load capacitance. '1'—Enables capacitor '0'—Disables capacitor
CFG3	Switched capacitor 8 pF enable. Enables 8 pF capacitor inside oscillator to be used as load capacitance. '1'—Enables capacitor '0'—Disables capacitor
CFG4	Switched capacitor 16 pF enable. Enables 16 pF capacitor inside oscillator to be used as load capacitance. '1'—Enables capacitor '0'—Disables capacitor
CFG5	Reserved
CFG6	Reserved
CFG7	Boot mode override bit.

Table continues on the next page...

Table 38-1. RTC General Purpose Data Register Table (continued)

Bit Number	Description
	This bit can be used to override the boot to happen in VLPR mode. The effect of this bit can be seen on next system reset. '1'—Boot in VLPR '0'—Boot in RUN
CFG8	Reserved
CFG9	Reserved
CFG10	Reserved
CFG11	Reserved
CFG12	Reserved
CFG13	Reserved
CFG14	Reserved
CFG15	Reserved

38.2 Introduction

This block is a low power module that provides time keeping and calendaring functions and additionally provides protection against tampering (external or internal tamper events), protection against spurious memory/register updates and battery operation. A Standby RAM is provided if the CPU wants to store any data that has to be retained when in battery operation mode.

38.2.1 Features

This block supports the following features:

- Designed for low power
  - Time and Date counters are rippled with respect to each other to prevent simultaneous toggling
- Basic Clock functions
  - Separate counters for Days, Hour, Minutes and Seconds
  - Calendaring support – Separate counters for Months, Year and Day of the Week

        - Automatic adjustment for Day Light Saving with user defined parameters
        - Automatic month and leap year adjustment
    - External clock support to run the counters in the case user wishes to provide externally compensated 1Hz clock.
- RTC utilizes 'local time' which implicitly contains the time zone offset
- Programmable alarm with interrupt. Alarm is output from RTC in case MCU wants to use it as a wake up event
- 8 periodic interrupts (Sampling Timer Interrupts)
- Hardware Compensation to compensate 1 Hz clock (to the counters) against frequency variations in oscillator clock due to temperature or crystal characteristics. Correction factor calculated by firmware. (Programmable correction factor).
- 16-bit CPU register programming interface with protection against run-away code
- Reset to the RTC block is generated when battery supply is removed and powered up later.
- Battery operation (Standby Mode) ensures seamless RTC operation when CPU power is removed
- Option to output the buffered 32.768 kHz clock or the compensated 1 Hz clock.
- 32 Bytes of Battery Backed Up RAM
- Enhanced Tamper Detection
    - Tamper Detection to detect illegal access into the system.
    - Tamper sources can be individually enabled/disabled and polarity can be controlled
    - External tamper sources are filtered for noise & glitches by the RTC. Internal tamper events are not filtered.
    - Configurable width of noise or glitch pulses that can be filtered out
        - Minimum 488 µs to 2 s depending on the tamper filter clock that is chosen.

38.2.2 Modes of Operation

This section describes the RTC modes of operation.

38.2.2.1 Wait Mode

In Wait mode, the RTC is fully operational. Since the RTC runs off a 32.768 kHz clock, which is not gated in Wait mode; RTC 's time keeping functions and other functions that depend on the 32.768 kHz clock, continue to operate independent of CPU. Only the register block's clock is gated. No register contents are lost.

38.2.2.2 Stop Mode

In Stop mode, the RTC is fully operational and behaves in the same way as in Wait mode.

38.2.2.3 VBAT mode

In the VBAT mode, the RTC is fully operational. This is because the RTC runs off a 32.768 kHz clock, which is not gated in VBAT mode. The RTC 's time keeping functions and other functions that depend on the 32.768 kHz clock continue to operate independent of the CPU. Only the register block's clock is gated. No register contents are lost. After coming out of VBAT mode, the register content is not accessible for two osc_clk cycles due to synchronization.

38.2.3 Design Overview

The RTC block provides basic time keeping functions through seconds, minutes, hours counters, and calendaring functions via date, day-of-week, month and year counters; along with automatic adjustments for leap year and day light saving. Reading these counters indicates the current date and time and writing to these registers sets the date and time as provided by the user.

The alarm is set for specific hour, minute and second. When the time counters match the configured alarm hour, minute and second settings, the alarm is set and an interrupt to CPU is generated, if the alarm interrupt is enabled. The alarm can additionally be configured to match days, months and year to generate the alarm interrupt. The alarm signal has been brought out on an MCU pin to allow wake up or control of external devices on board.

RTC module also provides 8 sampling timer interrupts.

The registers in the RTC module are configured via the CPU register programming interface. A protection mechanism is built in to the RTC to protect its registers against spurious writes by any run-away code. The protection mechanism requires the CPU to write a specific sequence of codes to the STATUS[7:6] bits to allows write access to the registers .On completing the update of registers the CPU should write "10" to STATUS[7:6] bits to enable the write protection. After unlocking the registers, the CPU has a window of 2 seconds for updating the register space. On power on reset a window of 15 seconds is allowed for the CPU to configure the RTC after which the registers are locked. Any updates beyond the unlock window would require the CPU to unlock the registers, again.

The RTC battery supply maintains normal RTC functionality when the MCU power (VDD) is removed. The battery supply allows RTC to keep functioning in case CPU is completely turned off. Reset to the RTC block is generated only when the battery supply is removed and powered up again.

RTC can detect intrusion via its tamper detection and logging mechanism. RTC supports 4 tamper events all individually configurable and any change of state will indicate a tamper that will get stored in the RTC tamper status and control register.Removal of battery after power up is also considered as a tamper event.The external tamper inputs pass through a low pass filter (digital) to prevent accidental assertion by noise or glitch pulses. The duration of pulses to be filtered out is configurable. Any tamper detected will cause an interrupt to the CPU. RTC has a FIFO based queue structure implemented in the design to support the logging of up to 4 tamper time stamp entries. Tamper detection logic and its interrupt are enabled on reset. Once a tamper is detected and event stored, a new event for the same tamper will not be stored until the CPU acknowledges the tamper status.

For detailed description on the complete functionality of RTC, refer to Functional Description.

38.3 Signal Description

This section lists all pad inputs to the RTC block.

38.3.1 EXTAL32K, XTAL32K

These pins serve as the connection pins to the external 32.768 kHz crystal.

38.3.2 TAMPER[2:0]

This device contains three external tamper pins. Each pin can act as an input or output depending upon whether active or passive tamper is enabled.

38.3.3 VBAT

The external battery or standby supply is provided via this pin.

38.4 Memory Map and Registers

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level. The address offset is defined at the module level.

NOTE

Initially, a 32 kHz clock is needed to initialize the RTC. This clock can later be gated while programming the RTC registers.

NOTE

The use of the fields of the General Purpose Data Register (RTC_GP_DATA_REG) is specific to the MCU. See the Chip-specific information section for a description of this register. Software reset has no effect on the contents of this register.

RTC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_0000	RTC Year and Month Counters Register (RTC_YEARMON)	16	R/W	0001h	38.4.1/724
4005_0002	RTC Days and Day-of-Week Counters Register (RTC_DAYS)	16	R/W	0001h	38.4.2/726
4005_0004	RTC Hours and Minutes Counters Register (RTC_HOURMIN)	16	R/W	0000h	38.4.3/727
4005_0006	RTC Seconds Counters Register (RTC_SECONDS)	16	R/W	0000h	38.4.4/727
4005_0008	RTC Year and Months Alarm Register (RTC_ALM_YEARMON)	16	R/W	0000h	38.4.5/728
4005_000A	RTC Days Alarm Register (RTC_ALM_DAYS)	16	R/W	0000h	38.4.6/729
4005_000C	RTC Hours and Minutes Alarm Register (RTC_ALM_HOURMIN)	16	R/W	0000h	38.4.7/729
4005_000E	RTC Seconds Alarm Register (RTC_ALM_SECONDS)	16	R/W	0000h	38.4.8/730

Table continues on the next page...

RTC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_0010	RTC Control Register (RTC_CTRL)	16	R/W	See section	38.4.9/731
4005_0012	RTC Status Register (RTC_STATUS)	16	R/W	0008h	38.4.10/ 733
4005_0014	RTC Interrupt Status Register (RTC_ISR)	16	w1c	0001h	38.4.11/ 735
4005_0016	RTC Interrupt Enable Register (RTC_IER)	16	R/W	0001h	38.4.12/ 738
4005_0020	RTC General Purpose Data Register (RTC_GP_DATA_REG)	16	R/W	0000h	38.4.13/ 741
4005_0022	RTC Daylight Saving Hour Register (RTC_DST_HOUR)	16	R/W	0000h	38.4.14/ 743
4005_0024	RTC Daylight Saving Month Register (RTC_DST_MONTH)	16	R/W	0000h	38.4.15/ 744
4005_0026	RTC Daylight Saving Day Register (RTC_DST_DAY)	16	R/W	0000h	38.4.16/ 745
4005_0028	RTC Compensation Register (RTC_COMPEN)	16	R/W	0000h	38.4.17/ 745
4005_0032	RTC Tamper Status and Control Register (RTC_TAMPER_SCR)	16	R/W	See section	38.4.18/ 746
4005_0034	RTC Tamper 01 Filter Configuration Register (RTC_FILTER01_CFG)	16	R/W	0000h	38.4.19/ 747
4005_0036	RTC Tamper 2 Filter Configuration Register (RTC_FILTER2_CFG)	16	R/W	0000h	38.4.20/ 748
4005_0042	RTC Control 2 Register (RTC_CTRL2)	16	R/W	See section	38.4.21/ 750

38.4.1 RTC Year and Month Counters Register (RTC_YEARMON)

This register stores the value of the month and year counters. The year field does not store the actual year value but the offset in years from a hardcoded BASE YEAR taken as the year 2112. This is a signed value and the ranges from –128 to +127. The software should program the offset from that base year into this register. For example, if the current year is 2007, then this will be represented in this register as –105 or 0x97. The actual year value can be found by adding the BASE YEAR and the offset in the YEARMON[15:8] register. Hence the range of year supported will be 1984 (2112 – 128) to 2239 (2112 + 127).

Hence for year calculation:

Actual Year = Base Year (for example, 2112) + Offset Year

The month register stores the count value of the months register. Writing to this register loads the months counter with this new value. The valid values are mentioned in table below. Both month and year are unaffected on software reset.

User software should first determine the state of the INVAL bit in the STATUS (bit 0) to determine that the counters are stable before their value can be read or changed. The assertion of INVAL bit ensures that no operation is done at the boundary of a second when counters change value.

Address: 4005_0000h base + 0h offset = 4005_0000h

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

RTC_YEARMON field descriptions

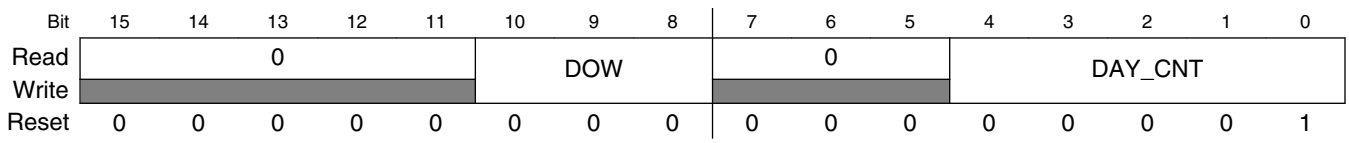
Field	Description
15–8 YROFST	Year Offset Count Value These bits indicate the offset in years from the base year (hard coded as 2112) and do not show the actual year value. This is a signed value. Valid values are –128 to 127. With the Base Year as 2112 and if the value of YEAR field is 0x10, the actual year will be 2112 + 16 = 2128.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MON_CNT	These bits give the value of the Months Counter . Valid Values are: 0 Illegal Value 1 January 2 February 3 March 4 April 5 May 6 June 7 July 8 August 9 September 10 October 11 November 12 December 13 Illegal Value 14 Illegal Value 15 Illegal Value

38.4.2 RTC Days and Day-of-Week Counters Register (RTC_DAYS)

This read/write register shows the current value of the day-of-week counter and days counter. Reading this register returns the latest value of the counter s . Writing to this register loads the value to the day-of-week and days counters and the counters continue to count from this new value. The day-of-week is not calculated automatically and should be written by CPU. This register is unaffected by software reset.

User software should first determine the state of the INVAL bit in the STATUS (bit 0) to determine that the counters are stable before their value can be read or changed. The assertion of INVAL bit ensures that no operation is done at the boundary of a second when counters change value.

Address: 4005_0000h base + 2h offset = 4005_0002h



RTC_DAYS field descriptions

Field	Description
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 DOW	Day of Week Counter Value. 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thrusday 5 Friday 6 Saturday 7 Reserved
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DAY_CNT	Days Counter Value. Valid values are ‘1’ to ‘31’.

38.4.3 RTC Hours and Minutes Counters Register (RTC_HOURMIN)

This register is used to program the hours and minutes counter. It can be read anytime to get the current value of the counters. Only power-on reset can reset this register. Hours counter can be set anything between 0 and 23. Minutes counter can be set anything between 0 and 59. This register is unaffected by software reset.

User software should first determine the state of the INVAL bit in the STATUS (bit 0) to determine that the counters are stable before their value can be read or changed. The assertion of INVAL bit ensures that no operation is done at the boundary of a second when counters change value.

Address: 4005_0000h base + 4h offset = 4005_0004h

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

RTC_HOURMIN field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 HOUR_CNT	Hours Counter Value. Valid count values are 0 to 23.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MIN_CNT	Minutes Counter Value. Valid count values are 0 to 59.

38.4.4 RTC Seconds Counters Register (RTC_SECONDS)

This register is used to program the seconds counter. It can be read anytime to get the current value of the counter. Only power-on reset can reset this register. Seconds counter can be set anything between 0 and 59 both included. This register is unaffected by software reset.

User software should first determine the state of the INVAL bit in the STATUS (bit 0) to determine that the counters are stable before their value can be read or changed. The assertion of INVAL bit ensures that no operation is done at the boundary of a second when counters change value.

Address: 4005_0000h base + 6h offset = 4005_0006h

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

RTC_SECONDS field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEC_CNT	Seconds Counter Value. Valid count values are 0 to 59.

38.4.5 RTC Year and Months Alarm Register (RTC_ALM_YEARMON)

This register is used to configure the months and year setting of the alarm. The alarm setting can be read or written anytime. This register is reset to its default state on software reset. Alarm interrupt bit is set when all values of alarm seconds, minutes, hours, days, month, and year match their respective counter values.

User software can configure the type of alarm using the ALM_TYPE bits in the CTRL register.

Address: 4005_0000h base + 8h offset = 4005_0008h

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

RTC_ALM_YEARMON field descriptions

Field	Description
15–8 ALM_YEAR	Year Value for Alarm. Same as Years Offset Value in RTC Year and Month Counters Register (RTC_YEARMON) .
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ALM_MON	Months Value for Alarm. Same as Months Counter Value in RTC Year and Month Counters Register (RTC_YEARMON) .

38.4.6 RTC Days Alarm Register (RTC_ALM_DAYS)

The days alarm register is used to configure the day setting of the alarm. The alarm setting can be read or written anytime. This register is reset to its default state on software reset. Alarm interrupt bit is set when all values of alarm seconds, minutes, hours, days, month and year match their respective counter values.

User software can configure the type of alarm using the ALM_TYPE bits in the CTRL register.

Address: 4005_0000h base + Ah offset = 4005_000Ah

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

RTC_ALM_DAYS field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ALM_DAY	Days Value for Alarm. Same as Days Counter Value in RTC Days and Day-of-Week Counters Register (RTC_DAYS) .

38.4.7 RTC Hours and Minutes Alarm Register (RTC_ALM_HOURMIN)

The hours and minutes alarm register is used to configure the hour and minute setting of the alarm. The alarm setting can be read or written anytime. This register is reset to default state on software reset.

User software can configure the type of alarm using the ALM_TYPE bits in the CTRL register.

Address: 4005_0000h base + Ch offset = 4005_000Ch

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

RTC_ALM_HOURMIN field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 ALM_HOUR	Hours Value for Alarm. Same as Hours Counter Value in RTC Hours and Minutes Counters Register (RTC_HOURMIN) .
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ALM_MIN	Minutes Value for Alarm. Same as Minutes Counter Value in RTC Hours and Minutes Counters Register (RTC_HOURMIN) .

38.4.8 RTC Seconds Alarm Register (RTC_ALM_SECONDS)

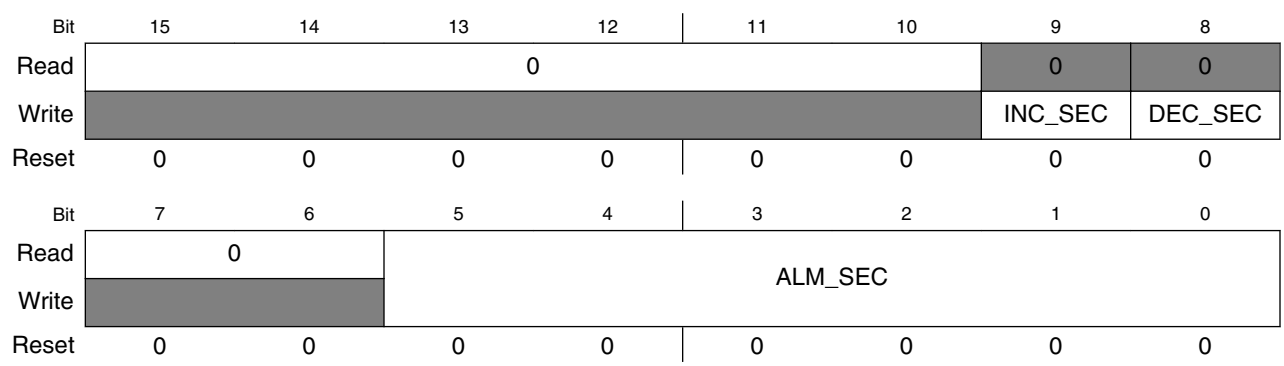
The seconds alarm register is used to configure the seconds setting of the alarm. The alarm setting can be read or written anytime. This register is reset to default value on software reset.

Bits 9:8 provide option to the user software to perform correction on seconds counter to compensate for the leap seconds. Write to these bits adds or subtracts 1 from the seconds counter and read returns zeros.

User software should first determine the state of the INVAL bit in the STATUS (bit 0) to determine that the counters are stable before they can be incremented or decremented. The assertion of INVAL bit ensures that no operation is done at the boundary of a second when counters change value.

User software can configure the type of alarm using the ALM_TYPE bits in the CTRL register.

Address: 4005_0000h base + Eh offset = 4005_000Eh



RTC_ALM_SECONDS field descriptions

Field	Description
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 INC_SEC	Increment Seconds Counter by 1. This bit controls the increment of seconds counter incase the software wants to make corrections to the seconds counter to compensate for the leap seconds or to perform fine trimming of time when needed. Write to this bit increments the seconds counter and then the bit gets cleared on next posedge.
8 DEC_SEC	Decrement Seconds Counter by 1. This bit controls the decrement of seconds counter incase the software wants to make corrections to the seconds counter to compensate for the leap seconds or to perform fine trimming of time when needed. Write to this bit has decrements the seconds counter and then the bit gets cleared on next posedge of bus clock.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ALM_SEC	Seconds Value for Alarm. Same as Seconds Counter Value in RTC Seconds Counters Register (RTC_SECONDS) .

38.4.9 RTC Control Register (RTC_CTRL)

This is the control register and governs all operations being done inside the RTC. This register is used to specify the software reset, daylight controls, and the type of alarm function needed.

Address: 4005_0000h base + 10h offset = 4005_0010h

Bit	15	14	13	12	11	10	9	8
Read	0	CLKOUT			0			0
Write								SWR
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0	DST_EN	0	TIMER_ STB_MASK	ALM_MATCH		COMP_EN	FINEEN
Write								
Reset	0	0	0	0	0	0	0	0

RTC_CTRL field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–13 CLKOUT	RTC Clock Output Selection.

Table continues on the next page...

RTC_CTRL field descriptions (continued)

Field	Description
	Selects which clock to output from SoC for use outside RTC. 00 No Output Clock 01 Fine 1 Hz Clock 10 32.768 kHz Clock 11 Coarse 1 Hz Clock
12–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 SWR	Software Reset bit. Self clearing bit. Asserting this field clears the contents of alarm, interrupt (status and enable except tamper interrupt enable bit) registers, STATUS[CMP_DONE], and STATUS[BUS_ERR] and has no effect on DST, calendaring,Standby time and tamper detect registers. 0 Software Reset cleared. 1 Software Reset asserted.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DST_EN	Daylight Saving Enable. The date and time for daylight saving changes are stored in the Daylight Saving Registers. These registers can be changed when this bit is 0. Once this bit is set, those registers cannot be changed and when time and date match the values in those registers, daylight adjustment will happen. To disable Daylight Saving function, this bit should be set to 0. 0 Disabled. Daylight saving changes are not applied. Daylight saving registers can be modified. 1 Enabled. Daylight saving changes are applied.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 TIMER_STB_ MASK	Sampling timer clocks mask 1 Sampling clocks are gated in standby mode 0 Sampling clocks are not gated when in standby mode
3–2 ALM_MATCH	Alarm Match bits. These bits define the type of alarm function. These bits select which time and calendar counters will be used for matching and generate an alarm. 00 Only Seconds, Minutes, and Hours matched. 01 Only Seconds, Minutes, Hours, and Days matched. 10 Only Seconds, Minutes, Hours, Days, and Months matched. 3 Only Seconds, Minutes, Hours, Days, Months, and Year (offset) matched.
1 COMP_EN	Compensation enable bit 1'b0:- Coarse Compensation is disabled. 1'b1:- Coarse Compensation is enabled. Note:- If both the bits are meant to be set "1'b1" hardware will not let the COMP_EN bit to be written 1.
0 FINEEN	Fine compensation enable bit

Table continues on the next page...

RTC_CTRL field descriptions (continued)

Field	Description
	1 Fine compensation is enabled. 0 Fine compensation is disabled

38.4.10 RTC Status Register (RTC_STATUS)

This register indicates the status of various processes going inside the RTC. This register also helps the user software to read time or date register when their values are stable and not changing. Bus Error bit get cleared by writing '1' to them. Software Reset resets the whole register (except the RST_SRC bit) to its default state.

All memory mapped registersare protected against spurious updates by the write protect mechanism. To unlock the registers, a specific pattern (as mentioned in above table) has to be written in the write enable bits (WE[1:0]) to enable or disable write protection.. The WE[1:0] bits are the only bits that are freely writeable by user software. The write enable bits are self clearing bits that always return zeros on read.

Address: 4005_0000h base + 12h offset = 4005_0012h

Bit	15	14	13	12	11	10	9	8
Read	0				CMP_DONE	0	0	BUS_ERR
Write					w1c			w1c
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0		CMP_INT	0	RST_SRC	CPU_LOW_VOLT	WRITE_PROT_EN	INVAL_BIT
Write	WE							
Reset	0	0	0	0	1	0	0	0

RTC_STATUS field descriptions

Field	Description
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 CMP_DONE	Compensation Done bit. This bit indicates that current compensation cycle is complete. The bit gets cleared by writing 1 to it. Done bit is asserted seven 32.768 kHz clock cycles before actual compensation interval completes so that back to back compensation can be enabled.

Table continues on the next page...

RTC_STATUS field descriptions (continued)

Field	Description
	0 Compensation busy or not enabled. 1 Compensation completed.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 BUS_ERR	Bus Error bit. This bit indicates that a read or write cycle was initiated by software when the INVAL_BIT was set. Write access to time /date registers gets nullified (terminate normally) and no register value gets changed. Read during INVAL bit asserted returns 16'hFFFF. No Transfer Error is asserted. This bit gets cleared by writing 1 to it. 0 Read and Write accesses are normal. 1 Read or Write accesses occurred when INVAL_BIT was asserted.
7–6 WE	Write Enable bits. These bits control the entry and exit of the write protection mode. Both registers are protected by the write protection mechanism. These are self clearing bits. Reads will return zeros. **NOTE:** When the registers are unlocked, they remain in this unlocked state for a time of 2 seconds after which they get locked automatically. After power-on-reset, the registers come out as unlocked but they get locked automatically 15 seconds after power on. 00, 01, 11, 10 Disable Write Protection - Registers are unlocked. 10 Enable Write Protection – Registers are locked.
5 CMP_INT	Compensation Interval bit. This read-only status bit is asserted for a time equal to compensation interval seconds (as configured by user). This bit is used by MCU to calculate the interrupts serviced during this interval and perform corrections in case of deviations (i.e. Calibration). This bit toggles on every start of new compensation interval and is either 0 or 1 during the entire duration. This bit will not toggle if compensation logic has been disabled by MCU.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 RST_SRC	Reset Source bit. This bit indicates to the user software the cause for reset to the part and subsequent boot up of CPU. This bit is asserted only on the power on reset that is generated within RTC (that is Power On Reset when both VBAT and VDD are powered up) and this indicates that part has been reset. On entering standby mode, this bit is cleared which indicates the part is booting after Standby Mode Exit, when the standby mode is actually exited. 0 Part was reset due to Standby Mode Exit (that is when VDD is powered up and VBAT was not powered down at all). 1 Part was reset due to Power-On Reset (that is Power On Reset when both VBAT and VDD are powered up).
2 CPU_LOW_ VOLT	CPU Low Voltage Warning status bit.

Table continues on the next page...

RTC_STATUS field descriptions (continued)

Field	Description
	This bit is asserted when the MCU/CPU power falls below the read only threshold when all write cycles are terminated normally and no change is done to the registers. Registers are read only. Transfer Error is not asserted on write access. 0 CPU in Normal Operating Voltage. 1 CPU Voltage is below Normal Operating Voltage. RTC Registers in read-only mode.
1 WRITE_PROT_EN	Write Protect Enable status bit. This read-only bit indicates that registers are in locked mode and write to them is disabled. Any write access made to the register space when write protection is enabled (that is in locked mode) will cause the transfer error signal to be asserted. 0 Registers are unlocked and can be accessed. 1 Registers are locked and in read-only mode.
0 INVAL_BIT	Invalidate CPU read/write access bit. This read-only bit indicates the time /date counters are invalid or changing and thus should not be read/ written to. This bit is asserted for 1 oscillator clock cycle before and after the 1 Hz (seconds clock) boundary/edge. Write access to time /date registers gets nullified (terminate normally) and no register value gets changed. Read during INVAL bit asserted returns 0xFFFF. No Transfer Error is asserted. 0 Time /Date Counters can be read/written. Time /Date is valid. 1 Time /Date Counter values are changing or Time /Date is invalid and cannot be read or written.

38.4.11 RTC Interrupt Status Register (RTC_ISR)

NOTE

For the sampling timer interrupt status bits [14:6], refer to chip configuration chapter for the applicable sampling timer frequencies.

This register indicates the status of the various real-time clock interrupts. When an event of the types included in this register occurs then the bit will be set in this register regardless of its corresponding interrupt enable bit being set. The status bits are cleared by writing a value of 1, which also clears the interrupt. Interrupts may occur while the system clock is idle or in standby mode. When the system enters the active power mode, interrupt will be indicated to the CPU. The first event of the Sampling Timer interrupts after Power-on-Reset should not used to qualify any periodic interval. However, the correct periodic interval (that is 512 Hz or 256 Hz, and so on) should be determined using two sampling timer interrupts. The time between two interrupt would always be the correct time period.

Tamper Interrupt Status is set on reset as POR is generated when battery and CPU power are unavailable and either is powered up. Removal of battery is considered as a tamper and hence this bit is set on reset.

The status register is also cleared on software reset except for the tamper status which remains unaffected .

NOTE

Sampling interrupts from 512 Hz to 2 Hz are generated using uncompensated clock. Only 1 Hz is compensated.

Address: 4005_0000h base + 14h offset = 4005_0014h

Bit	15	14	13	12	11	10	9	8
Read	IS_512HZ	IS_256HZ	IS_128HZ	IS_64HZ	IS_32HZ	IS_16HZ	IS_8HZ	IS_4HZ
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	IS_2HZ	IS_1HZ	MIN_IS	HOUR_IS	DAY_IS	ALM_IS	0	TAMPER_IS
Write	w1c	w1c	w1c	w1c	w1c	w1c		
Reset	0	0	0	0	0	0	0	1

RTC_ISR field descriptions

Field	Description
15 IS_512HZ	512 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
14 IS_256HZ	256 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
13 IS_128HZ	128 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
12 IS_64HZ	64 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
11 IS_32HZ	32 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.

Table continues on the next page...

RTC_ISR field descriptions (continued)

Field	Description
10 IS_16HZ	16 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
9 IS_8HZ	8 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
8 IS_4HZ	4 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
7 IS_2HZ	2 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
6 IS_1HZ	1 Hz Interval Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
5 MIN_IS	Minutes Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
4 HOUR_IS	Hours Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
3 DAY_IS	Days Interrupt Status bit. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
2 ALM_IS	Alarm Interrupt Status bit. This bit indicates that the alarm value programmed matches the counter values. 0 Interrupt is de-asserted. 1 Interrupt is asserted.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TAMPER_IS	Tamper Interrupt Status bit. This field is cleared when TAMPER_SCR[TMPR_STS] is cleared.

Table continues on the next page...

RTC_ISR field descriptions (continued)

Field	Description
	0 Interrupt is de-asserted. 1 Interrupt is asserted (Default on reset) .

38.4.12 RTC Interrupt Enable Register (RTC_IER)

For the sampling timer interrupt enable bits [14:6], refer to chip configuration chapter for the applicable sampling timer frequencies.

The real-time clock interrupt enable register (IER) is used to enable/disable the various real-time clock interrupts. De-asserting an interrupt enable bit has no effect on the assertion of its corresponding status bit.

Alarm interrupt is asserted on counters matching the alarm setting done in the memory map. The counters matched for the alarm interrupt are selected based on the Alarm Type set in CTRL[3:2] bits. The various types of alarm available are as per the following table. Only one alarm type can be used at a time.

Table 38-2. Alarm Match Table

ALM_MATCH[1:0] (CTRL[3:2])	Counters Matched	Alarm Type
00	Seconds, Minutes, and Hours	Daily
01	Seconds, Minutes, Hours, and Days	Monthly
10	Seconds, Minutes, Hours, Days, and Months	Yearly
11	Seconds, Minutes, Hours, Days, Months, and Year	One Time

A common interrupt is generated by the block. The user software should read the status register in the interrupt service routine to determine which interrupt has occurred.

The tamper detect interrupt enable is an exception as it is enabled after power on reset. All interrupts enables except tamper interrupt enable are reset to default state on software reset.

Address: 4005_0000h base + 16h offset = 4005_0016h

Bit	15	14	13	12	11	10	9	8
Read Write	IE_512HZ	IE_256HZ	IE_128HZ	IE_64HZ	IE_32HZ	IE_16HZ	IE_8HZ	IE_4HZ
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	IE_2HZ	IE_1HZ	MIN_IE	HOUR_IE	DAY_IE	ALM_IE	0	TAMPER_IE
Write								
Reset	0	0	0	0	0	0	0	1

RTC_IER field descriptions

Field	Description
15 IE_512HZ	512 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
14 IE_256HZ	256 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
13 IE_128HZ	128 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
12 IE_64HZ	64 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
11 IE_32HZ	32 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
10 IE_16HZ	16 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
9 IE_8HZ	8 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
8 IE_4HZ	4 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
7 IE_2HZ	2 Hz Interval Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
6 IE_1HZ	1 Hz Interval Interrupt Enable bit.

Table continues on the next page...

RTC_IER field descriptions (continued)

Field	Description
	0 Interrupt is disabled. 1 Interrupt is enabled.
5 MIN_IE	Minutes Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
4 HOUR_IE	Hours Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 DAY_IE	Days Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 ALM_IE	Alarm Interrupt Enable bit. This bit indicates that the alarm value programmed matches the counter values. 0 Interrupt is disabled. 1 Interrupt is enabled.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TAMPER_IE	Tamper Interrupt Enable bit. 0 Interrupt is disabled. 1 Interrupt is enabled (Default on reset).

38.4.13 RTC General Purpose Data Register (RTC_GP_DATA_REG)

The use of the bits in this register are specific to the MCU. See the Chip-specific section for description of this register. Software reset has no effect on the contents of this register.

Address: 4005_0000h base + 20h offset = 4005_0020h

Bit	15	14	13	12	11	10	9	8
Read	0	0	0	0	0	0	0	0
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	CFG7	0	0	CFG4	CFG3	CFG2	CFG1	CFG0
Write								
Reset	0	0	0	0	0	0	0	0

RTC_GP_DATA_REG field descriptions

Field	Description
15 Reserved	Reserved This read-only field is reserved and always has the value 0.
14 Reserved	Reserved This read-only field is reserved and always has the value 0.
13 Reserved	Reserved This read-only field is reserved and always has the value 0.
12 Reserved	Reserved This read-only field is reserved and always has the value 0.
11 Reserved	Reserved This read-only field is reserved and always has the value 0.
10 Reserved	Reserved This read-only field is reserved and always has the value 0.
9 Reserved	Reserved This read-only field is reserved and always has the value 0.
8 Reserved	Reserved This read-only field is reserved and always has the value 0.
7 CFG7	Boot mode override bit This bit can be used to override the boot to happen in VLPR mode. The effect of this bit can be seen on next system reset.

Table continues on the next page...

RTC_GP_DATA_REG field descriptions (continued)

Field	Description
	0 Boot in RUN 1 Boot in VLPR
6 Reserved	Reserved This read-only field is reserved and always has the value 0.
5 Reserved	Reserved This read-only field is reserved and always has the value 0.
4 CFG4	Switched capacitor 16 pF enable Enables 16 pF capacitor inside oscillator to be used as load capacitance. 0 Disables capacitor 1 Enables capacitor
3 CFG3	Switched capacitor 8 pF enable Enables 8 pF capacitor inside oscillator to be used as load capacitance. 0 Disables capacitor 1 Enables capacitor
2 CFG2	Switched capacitor 4 pF enable Enables 4 pF capacitor inside oscillator to be used as load capacitance. 0 Disables capacitor 1 Enables capacitor
1 CFG1	Switched capacitor 2 pF enable Enables 2 pF capacitor inside oscillator to be used as load capacitance. 0 Disables capacitor 1 Enables capacitor
0 CFG0	32 kHz RTC OSC Control This bit can be used to disable clock when writing to RTC registers. 0 Enables the oscillator 1 Disables the oscillator

38.4.14 RTC Daylight Saving Hour Register (RTC_DST_HOUR)

This register stores the time in hours when the Daylight Saving has to be applied or reversed. This register is programmable when the DST_EN bit in CTRL register is 'Zero'. When DST_EN bit is set, the contents of this register cannot be changed. The user software should program the correct hour value (0 – 23) as per the regional settings. For example, if the Daylight Saving starts at 2:00 AM on March 25 and ends at 2:00 AM on October 28 in 2007 then the time at which the RTC advances or falls back is actually 1:59 AM. Hence the user software should program 1 for the hour count value (and not 2!) i.e. write 0x0101 in this register. 59 minute count is automatically checked inside RTC and hence not required to be programmed. This register has no effect on software reset.

Address: 4005_0000h base + 22h offset = 4005_0022h

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

RTC_DST_HOUR field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 DST_START_HOUR	Daylight Saving Time (DST) Hours Start Value. This is the hour value for the time when DST comes into effect. Same as Hours Counter Value in the RTC Hours and Minutes Counters Register (RTC_HOURMIN) .
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DST_END_HOUR	Daylight Saving Time (DST) Hours End Value. This is the hour value for the time when DST is reversed. Same as Hours Counter Value in the RTC Hours and Minutes Counters Register (RTC_HOURMIN) .

38.4.15 RTC Daylight Saving Month Register (RTC_DST_MONTH)

This register stores the month when the Daylight Saving has to be applied or reversed. This register is programmable when the DST_EN bit in CTRL register is Zero. When DST_EN bit is set, the contents of this register cannot be changed. The CPU should program the correct month value (1 – 12) as per the regional settings. For example, if the Daylight Saving starts at March 25 and ends at October 28 in 2007. Hence the CPU should write 0x030A in this register. This register has no effect on software reset.

Address: 4005_0000h base + 24h offset = 4005_0024h

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

RTC_DST_MONTH field descriptions

Field	Description
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 DST_START_MONTH	Daylight Saving Time (DST) Month Start Value. This is the month value for the time when DST comes in to effect. See the RTC Year and Month Counters Register (RTC_YEARMON) .
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DST_END_MONTH	Daylight Saving Time (DST) Month End Value. This is the month value for the time when DST is reversed. See the RTC Year and Month Counters Register (RTC_YEARMON) .

38.4.16 RTC Daylight Saving Day Register (RTC_DST_DAY)

This register stores the day when the Daylight Saving has to be applied or reversed. This register is programmable when the DST_EN bit in CTRL register is Zero. When DST_EN bit is set, the contents of this register cannot be changed. The CPU should program the correct day value (1 – 31) as per the regional settings. For example, if the Daylight Saving starts at March 25 and ends at October 28 in 2007. Hence the CPU should write 0x191C in this register. This register is unaffected by software reset.

Address: 4005_0000h base + 26h offset = 4005_0026h

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

RTC_DST_DAY field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 DST_START_DAY	Daylight Saving Time (DST) Day Start Value. This is the day value for the time when DST comes into effect.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DST_END_DAY	Daylight Saving Time (DST) Day End Value. This is the day value for the time when DST is reversed.

38.4.17 RTC Compensation Register (RTC_COMPEN)

The compensation register stores the compensation value that will be used by the compensation block to correct the 1 Hz clock.

Address: 4005_0000h base + 28h offset = 4005_0028h

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

RTC_COMPEN field descriptions

Field	Description
COMPEN_VAL	Compensation Value This register stores the compensation parameters. The definition of this field is dependent on the setting of RTC_CTRL[FINEEN]. If FINEEN = 0: • Compensation/Correction Value (COMPEN[7:0]) - Compensation/Correction Value is a 2's complement value by which the 1 Hz Clock is modified (during its generation) by either adding or removing RTC Oscillator clock cycles. • Compensation Interval (COMPEN[15:8]) - Compensation Interval is the duration in seconds over which the correction is applied. This is the time in which the addition or removal of 32.768 kHz clock cycles is done thereby ensuring that the compensation interval is close to the interval obtained with an ideal 1 Hz clock. If FINEEN = 1 • Integral Compensation Value (COMPEN[15:12])- This is a 2's complement value of the integer part of correction or compensation value that has to be adjusted in every 1 second period. This value is expressed in terms of number of clock cycles of the RTC Oscillator clock. • COMPEN[11:7] -- Should be zero • Fraction Compensation Value (COMPEN[6:0]) - This is the fractional part of the correction or compensation value that has to be adjusted. This value is expressed as number of clock cycles of a fixed 4.194304 MHz clock. This value is always a positive number.

38.4.18 RTC Tamper Status and Control Register (RTC_TAMPER_SCR)

For the implementation of this register in the MCU, refer to chip configuration chapter.

This register stores the tamper event and provides control to the user to enable or disable each tamper individually. No tamper is disabled automatically unless corresponding control bit is de-asserted by software. The tamper statuses are stored as active high. The tamper status bits store the tamper event irrespective of their corresponding control bit being asserted or not. The control bits gate the assertion of tamper interrupt bit in ISR. These status and controls bits combined assert the tamper interrupt in the ISR register. The tamper interrupt will be cleared when all above Status bits are cleared.

Address: 4005_0000h base + 32h offset = 4005_0032h

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

RTC_TAMPER_SCR field descriptions

Field	Description
15–12 Reserved	This field is reserved.

Table continues on the next page...

RTC_TAMPER_SCR field descriptions (continued)

Field	Description
11–8 TMPR_STS	Tamper Status Bit Indicates if a tamper event was detected or not. Writing '1' to this field clears the tamper status.
7–4 Reserved	This field is reserved.
TMPR_EN	Tamper Control Controls the generation of tamper interrupt from corresponding tamper status bit. Enabled on reset.

38.4.19 RTC Tamper 01 Filter Configuration Register (RTC_FILTER01_CFG)

This register must be configured once during the initial startup of application and must not be changed on the fly to avoid any erratic behavior. Changing the register on the fly might lead to missing of tamper events or assertion of false tamper events.

Address: 4005_0000h base + 34h offset = 4005_0034h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	POL0	CLK_SEL0			FIL_DUR0				POL1	CLK_SEL1			FIL_DUR1			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RTC_FILTER01_CFG field descriptions

Field	Description
15 POL0	Tamper Detect Input Bit 0 Polarity Control This bit controls the polarity of tamper detect input bit 0 (TAMPER[0]). 0 Tamper detect input bit 0 is active high. 1 Tamper detect input bit 0 is active low.
14–12 CLK_SEL0	Tamper Filter 0 Clock Select This bit is a write once bit and selects the clock source for tamper filter for tamper detect input bit 0. 000 32 kHz clock 001 512 Hz clock 010 128 Hz clock 011 64 Hz clock 100 16 Hz clock 101 8 Hz clock 110 4 Hz clock 111 2 Hz clock

Table continues on the next page...

RTC_FILTER01_CFG field descriptions (continued)

Field	Description
11–8 FIL_DUR0	Tamper Detect Bit 0 Filter Duration This bit indicates the number of tamper filter clock cycles for which the TAMPER[0] signal should remains stable before being detected as a tamper. These bits are used by the tamper filtering operation. With the tamper duration set to Zero any tamper detected on the tamper pins will directly set the tamper status and interrupt bits. Caution is required when making the tamper filter duration equal to 0 as any glitches on the tamper pins will cause a tamper interrupt. 0 Filtering operation disabled. 1-1111 Number of tamper filter clock cycles to be counted when tamper is asserted.
7 POL1	Tamper Detect Input Bit 1 Polarity Control This bit controls the polarity of tamper detect input bit 1 (TAMPER[1]). 0 Tamper detect input bit 1 is active high. 1 Tamper detect input bit 1 is active low.
6–4 CLK_SEL1	Tamper Filter 1 Clock Select This bit is a write once bit and selects the clock source for tamper filter for tamper detect input bit 1. 000 32 kHz clock 001 512 Hz clock 010 128 Hz clock 011 64 Hz clock 100 16 Hz clock 101 8 Hz clock 110 4 Hz clock 111 2 Hz clock
FIL_DUR1	Tamper Detect Bit 1 Filter Duration This bit indicates the number of tamper filter clock cycles for which the TAMPER[1] signal should remains stable before being detected as a tamper. These bits are used by the tamper filtering operation. With the tamper duration set to Zero any tamper detected on the tamper pins will directly set the tamper status and interrupt bits. Caution is required when making the tamper filter duration equal to 0 as any glitches on the tamper pins will cause a tamper interrupt. 0 Filtering operation disabled. 1 to 1111 Number of tamper filter clock cycles to be counted when tamper is asserted.

38.4.20 RTC Tamper 2 Filter Configuration Register (RTC_FILTER2_CFG)

This register must be configured once during the initial startup of application and must not be changed on the fly to avoid any erratic behavior. Changing the register on the fly might lead to missing of tamper events or assertion of false tamper events.

Tamper 3 is an internal tamper detect bit and hence does not require any filtering.

Address: 4005_0000h base + 36h offset = 4005_0036h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	POL2	CLK_SEL2			FIL_DUR2				0							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RTC_FILTER2_CFG field descriptions

Field	Description
15 POL2	Tamper Detect Input Bit 2 Polarity Control This bit controls the polarity of tamper detect input bit 2 (TAMPER[2]). 0 Tamper detect input bit 2 is active high. 1 Tamper detect input bit 2 is active low.
14–12 CLK_SEL2	Tamper Filter 2 Clock Select This bit is a write once bit and selects the clock source for tamper filter for tamper detect input bit 2. 000 32 kHz clock 001 512 Hz clock 010 128 Hz clock 011 64 Hz clock 100 16 Hz clock 101 8 Hz clock 110 4 Hz clock 111 2 Hz clock
11–8 FIL_DUR2	Tamper Detect Bit 2 Filter Duration This bit indicates the number of tamper filter clock cycles for which the TAMPER[2] signal should remains stable before being detected as a tamper. These bits are used by the tamper filtering operation. With the tamper duration set to Zero any tamper detected on the tamper pins will directly set the tamper status and interrupt bits. Caution is required when making the tamper filter duration equal to 0 as any glitches on the tamper pins will cause a tamper interrupt. 0 Filtering operation disabled. 1 to 1111 Number of tamper filter clock cycles to be counted when tamper is asserted.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

38.4.21 RTC Control 2 Register (RTC_CTRL2)

RTC control 2 register defines the attributes for the configuration of tamper pins.

Address: 4005_0000h base + 42h offset = 4005_0042h

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	WAKEUP_MODE	WAKEUP_STATUS		0	0			TAMP_CFG_OVER
Write								
Reset	1	0	0	0	0	0	0	0

RTC_CTRL2 field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 WAKEUP_MODE	Wakeup Mode 0 Tamper pin 0 is used as the tamper pin. 1 Tamper pin 0 is used as a wakeup/hibernation pin.
6–5 WAKEUP_STATUS	Wakeup Status 00 The wakeup/hibernation pin is in HiZ mode. 01 The wakeup/hibernation pin is at logic 0. MCU is in sleep mode. 10 The wakeup/ hibernation pin is at logic 1. MCU is in sleep mode. 11 Reserved
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TAMP_CFG_OVER	Tamper Configuration Over **NOTE:** Set this field to 1 only when all the tamper_cfg registers are programmed. 0 Tamper filter processing disabled. 1 Tamper filter processing enabled. **NOTE:** To enable the tamper feature, this bitfield should be set.

38.5 Functional Description

38.5.1 Time and Calendaring Functions

RTC performs and controls all chronological functions as mentioned below:

- Implements all counters for date and time and their related control logic
- Leap Year calculation and adjusting the day count accordingly
- Increment/Decrement of counters for adjustment of leap seconds
- Tracking the number of days in a month
- Automatic Daylight adjustment for time
- Alarm generation with selectable matching of different counters

Dynamic modifications to the date counters are done based on a) Leap Year b) Month and c) Daylight Saving. Additionally, the user software can add or subtract a second to take care of Leap Seconds Adjustment. All changes are hardware controlled and triggered by software. A leap year is defined as a year in which the year value is divisible by 4 and 400 and will have an extra day in February. Daylight savings are done in different regions of the world to shift the local time according to summer or winters. Time is shifted at a pre-defined time decided by the regional conditions and programmed by the user software into the RTC and RTC automatically adjusts the time using hardware alarms for daylight saving. Day counter is also adjusted for months having 28/29/30/31 days.

Alarm generation is done by the RTC block. The matching of counters is controlled by ALM_MATCH bits inside the control register to give various alarm options of daily, monthly, yearly or one-time alarms.

Note

Setting an hour alarm value that coincides with the Daylight Saving Start Hour value will not generate an alarm as when Daylight Saving comes into effect, the hour counter is incremented by 2 instead or normal 1. For example, setting ALM_HOURMIN = 0x1500 (hour alarm value = 15 or 3 PM) and DST_HOUR = 0x1411 (DST Start Hour value = 14 or 2 PM), will cause the hour counter to go from 14 to 16. The hour

counter will not be 15 and hence the alarm which was set for 15 (or 3 PM) will not trigger. User software must take into account Daylight Saving changes when setting alarm values.

NOTE

The user software must program an alarm time equal to the daylight saving end time that is fallback time and when the alarm interrupt comes, the user software must clear the DSTEN bit in control register, else correct operation might not occur.

38.5.2 RTC Compensation Logic

The Compensation Logic provides an accurate and wide range of compensation which is suitable for many crystals, and can correct a wide range crystal offsets with offsets being as low as 0.119 PPM. The same hardware logic supports both temperature compensation and frequency compensation.

There are two important components in the compensation logic viz., Coarse Compensation and Fine Compensation. The coarse compensation provides accurate clock to RTC's internal time & date counters while fine compensation generate an accurate 1 Hz clock output (via MCU pin) with high resolution clock edge placement (up to 0.88 ppm) and near 50% duty cycle.

38.5.2.1 Enabling the required compensation logic

Coarse compensation is enabled by writing to RTC_CTRL[COMPEN] and Fine compensation is enabled by writing to RTC_CTRL[FINEEN]. However, when FINEEN = '1', COMPEN must be equal to '1'. Setting both bits to '0' will disable all compensation and this is the default state out of reset.

In addition, compensation parameters need to be provided to allow required compensation logic to run. The parameters are provided by writing to the RTC_COMPEN register.

NOTE

In RTC_CTRL, if FINEEN = 0, then before changing RTC_COMPEN for the first time, disable the compensation using RTC_CTRL[COMP_EN]. Successive changes can be done without disabling compensation. If FINEEN =1, then there is an option to either clear previously accumulated fractional compensation value by disabling compensation from RTC_CTRL, program new values, and re-enable compensation

and start afresh or to overwrite the current values which leads to adding of previously accumulated fractional compensation value to programmed fractional compensation value. The integer part is always overwritten.

38.5.2.2 Compensation Parameters

Compensation logic requires the user software to provide compensation parameters in the RTC_COMPEN register in order to generate accurate 1 Hz clock. These parameters are defined depending on the type of compensation enabled.

The compensation parameters (when FINEEN = '0') are defined as:

- **Compensation/Correction Value:** Compensation/Correction Value is a 2's complement value by which the 1 Hz Clock is modified (during its generation) by either adding or removing RTC Oscillator clock cycles.
- **Compensation Interval:** Compensation Interval is the duration in seconds over which the correction is applied. This is the time in which the addition or removal of 32.768 kHz clock cycles is done thereby ensuring that the compensation interval is close to the interval obtained with an ideal 1 Hz clock.

The compensation parameters (when FINEEN = '1') are defined as:

- **Integral Compensation Value:** This is a 2's complement value of the integer part of correction or compensation value that has to be adjusted in every 1 second period. This value is expressed in terms of number of clock cycles of the RTC Oscillator clock
- **Fraction Compensation Value:** This is the fractional part of the correction or compensation value that has to be adjusted. This value is expressed as number of clock cycles of a fixed 4.194304 MHz clock that have to be added. This value is always a positive number.

NOTE

When FINEEN = '1', the compensation interval is by default set to "1" indicating that the compensation will be done at every second.

38.5.2.3 Coarse Compensation Logic

The coarse compensation logic provides the accurate 1 Hz clock pulses to the time and date counters and the coarse 1 Hz clock to the fine compensation logic.

When FINEEN = '0': RTC compensates the clock over the provided compensation interval. The addition or removal of clocks is done in a single 1 Hz period, leaving the other 1 Hz periods of the compensation interval same. This addition or removal of RTC Oscillator clock cycles adjusts the 1 Hz clock in a way to keep the overall compensation interval time close to the same interval being measured with an ideal clock.

When FINEN = '1': The integral part of compensation value (or correction value) is added or removed every 1 Hz period while the fraction part is accumulated and adjusted when accumulated value equals 1 RTC Oscillator clock cycle period.

To perform crystal offset compensation, the user software computes the compensation parameters external to RTC and using the details about crystal characteristics (ageing, drift, etc), computes the correction factor and programs it in the two's complement format in the RTC Compensation Register (RTC_COMPEN). Based on the values written in the RTC_COMPEN register, the compensation is performed.

To perform temperature compensation the user software can maintain a look-up-table in its memory which lists the change in frequency of 32.768 kHz crystal clock for each degree change in temperature.The CPU wakes up periodically to measure the external temperature via a temperature sensor connected to an A/D converter. The user software uses the look-up-table to determine the compensation factor and writes the value to be compensated (in terms of number of 32.768 kHz clock cycles in 2's complement format) in the RTC Compensation Register (COMPEN). Based on the new values written the compensation logic adjusts the clock from the next compensation interval.

The user software must compute a common compensation factor if it detects a variation in 32.768 kHz clock due to both temperature and crystal characteristics.

Vital Statistics

- Range of Compensation Interval: 1 second to 255 seconds. If FINEEN=0, programming a 0 disables compensation. If FINEEN=1, this interval is always 1 second.
- Range of Compensation: -128 to +127 (number of 32.768 kHz clock cycles)
- Selection Criteria: Compensation is done only when enabled by user software. User software can disable compensation by programming a zero compensation interval or setting RTC_CTRL[COMPEN] = '0'

38.5.2.4 Fine Compensation Logic

The fine compensation logic takes the coarse 1 Hz clock (from coarse compensation block) and generates accurate 1 Hz clock output with accurate clock edge placement and near 50% duty cycle.

The fine compensation runs when FINEEN = '1' and uses the MCU's IRC clock to adjust the fractional part of the compensation value every 1 Hz period. This provides an accurate edge placement on 1 Hz clock.

The fine compensation module automatically adjusts the fractional compensation value for any variation in the IRC clock.

NOTE

Since the IRC clock is generated on MCU, this clock will get disabled when MCU power is OFF or in certain low power modes. In this case, the coarse 1 Hz clock is output from the MCU. The coarse 1 Hz clock is not a 50% duty cycle clock.

38.5.3 Write Protection Mechanism

This logic protects the RTC Registers from any spurious updates that can happen from a run-away code. The logic monitors the values written to WE[1:0] bits of the STATUS register. By default unconditional write access is allowed to these bits only. For writing the locking and unlocking sequence, 8-bit access should be given on RTC_STATUS[7:0].

To enable write protection, "10" is written on these bits. To disable write protection, the sequence "00 – 01 – 11 – 10" is written onto these bits.

After a power on reset, the write-protect mechanism is disabled, allowing the user software to configure the RTC block. Once configuration is complete, the user code should enable the write protection mode. If this is not done by the user software, the registers are put into write protect mode 15 seconds after power on. In case the write protect mode is unlocked to update registers, then the write protect mode is enabled automatically 2 seconds after unlock, if not locked by user software.

Any write access made to the registers when write protection is enabled (i.e. registers in locked mode) will cause the transfer error signal to be asserted. Reads are allowed at all times.

NOTE

1. Always check RTC_STATUS[WRITE_ PROT_EN] after running unlocking and locking sequence and re-run the

sequence if RTC_STATUS[WRITE_ PROT_EN] is not in the required state.
2. Two consecutive unlocking sequences will lock the registers. Once unlocked, running an unlock sequence before the timeout will lock RTC registers.

38.5.4 Tamper Detection and Logging

A mechanism has been provided in RTC to detect and log any intrusion made to the system. The tamper logic supports up to 4 tamper signal inputs which can be used for detecting either internal tamper events (i.e. battery removal, etc) or external tamper events (i.e. off chip tamper switches).

Internal Tamper Events: Battery removal when MCU is powered OFF

The detection of battery removal during system power off is done using a flop that is asserted on power-on reset. Since there is no difference between a proper shutdown and this tamper condition, this is considered as a tamper always. The firmware needs to differentiate between the tamper condition and normal power up. One way is to ignore this tamper interrupt when the SoC or application is in Service mode and simply reset the tamper interrupt. For other conditions it will be taken as a tamper.

External Tamper Events: Off-Chip Tamper Indication

An external off-chip tamper switch is also used to monitor tampering external to the SoC. For example, a signal can be used to indicate that the case housing the SoC/board is opened or not. Since these events are detected off chip, these tamper inputs are pre-condition in the SoC logic (i.e. conversion to digital or level shifting, etc) before being input to RTC.

External tamper switches are prone to noise and can cause false tamper indication if not filtered. Noise filtering is present in RTC for all external tamper inputs. The duration for which a tamper signal should be asserted to be indicated as tamper is programmable by user software in the register space. This duration is programmable to support a variety of tamper switches. The filtered signals are then used to generate the tamper status and interrupt signals. The polarity of the tamper inputs can be configured to be active high or active low.

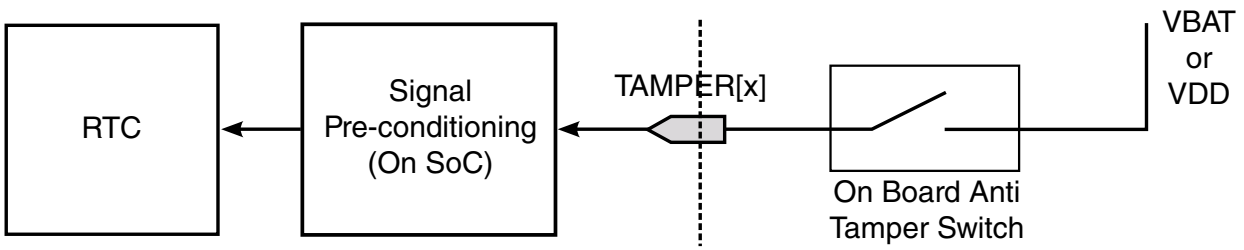


Figure 38-2. External Tamper Detection

38.5.4.1 Tamper Detection Flow

The tampering logic stores only the first occurrence of tamper from a pin till that tamper status is acknowledged by the CPU. As soon as the CPU reads the queue entry it has to clear the status of the corresponding pin to enable logging of subsequent tampers from that pin. It may choose to disable that tamper bit if it does not want further tamper logging from that tamper pin.

Following steps describe how tamper is logged in the registers and tamper queue, and how the software acknowledges the tamper status indication:

- Tamper Interrupt Enable bit is also asserted on POR and an interrupt indication to CPU.
- When CPU acknowledges the tamper interrupt by writing 1'b1 to the tamper interrupt status bit (TTSR_SCR[8]), the tamper status bits are cleared.
- Internal tamper event (detected by SoC logic) are simply stored in their corresponding status bits and time stamp is captured.
- All tamper signals asserted externally are filtered inside the tamper block.
- Tamper status bit is asserted when the filter counter matches the programmed filter duration.
- Tamper status bit is asserted in the tamper status and control register irrespective the tamper control bit is enabled or not. Time stamp is also logged for the first occurrence of the tamper event. The first occurrence of each tamper along with time stamp get pushed into the tamper queue. Until current tamper status bit is acknowledged by CPU, next status and time stamp for that tamper is ignored.
- The tamper interrupt is asserted for the tamper status bits that have the corresponding interrupt enable bits set.

- Tamper interrupt status bit (in ISR) is cleared when all tamper status bits (in TTSR_SCR) are cleared by writing 1 to them.
- The software must read the tamper queue entry first and then clear the corresponding status bit from the TAMPER_SCR register.

NOTE

When programing the tamper parameters such as filter duration, clock selection, polarity, etc, the user software must gate the 32k clock to RTC, program these parameters and then enable clock. While programming the parameters, the clock selection must be changed at the last. It is assumed that time and date will be set after this and tamper parameters will not be required to be changed again later.

38.5.5 RTC Isolation

To prevent leakage and erratic behavior, RTC isolates its CPU register programming interface and other control signals. There are two levels of isolation provided:

- CPU is in certain low power modes where LVD is disabled (except VLPR).
- When VDD < LVD, both read and writes to all registers are blocked. Isolation is enabled to prevent leakage from signals coming from the powered off domain of SoC.

The voltage threshold detection is done by an analog block (outside digital RTC) which monitors the voltage levels.

38.5.6 Wakeup Mode

To enable the wakeup functionality of RTC, set CTRL2[WAKEUP_MODE]. If this field is set to "1", the tamper_out[0] pin is by default set to the wakeup functionality. This pin is driven by CTRL2[WAKEUP_STATUS].

NOTE

If the tamper pin is in HiZ mode, the CPU is in normal functioning mode. If CTRL2[WAKEUP_STATUS] is changed to 01 or 10, then the CPU needs to put the system in the sleep mode.

The use case scenario is as follows:

1. Set the required wakeup interrupt sources from the RTC_IER register. Select the required event, that is, {(Alarm & select) OR (Tamper & select) OR (1sec period & select)}.
2. The CTRL2[WAKEUP_STATUS] field is by default in HiZ mode.
3. Based on whether LDO or PMOS is used, switch the CTRL2[WAKEUP_STATUS] field to 0 or 1.
4. When the selected event(interrupt) is detected, the wakeup/hibernation pin (tamper_out[0]) changes back to HiZ mode.

Chapter 39
Low-Power Timer (LPTMR)

39.1 Chip-specific LPTMR information

39.1.1 Overview

The Low-Power Timer (LPTMR) can operate as either a real time interrupt or as a pulse accumulator, and operate in all of the chip's power modes (including VLLSx).

39.1.2 Instantiation Information

This device has one LPTMR module with one 16-bit channel.

There are three LPTMR pins: LPTMR0_ALT1, LPTMR0_ALT2, and LPTMR0_ALT3. Each of these are connected to LPTMR and can be selected by module's CSR[TPS] field with the settings 01, 10, and 11, respectively. The LPTMR0_ALTx pin source selection can be achieved by configuring SIM_SOPT1CFG[LPTMRSEL1], SIM_SOPT1CFG[LPTMRSEL2] and SIM_SOPT1CFG[LPTMRSEL3].

Either of the CMP output can also be connected to LPTMR by setting each module's CSR[TPS] field to 00. This configuration allows pulse counting of the CMP output. Selection between CMP0, CMP1 and CMP2 can be achieved by using SIM_SOPT1CFG[LPTMRSEL0].

Table 39-1. Selecting the timer pin to be used by LPTMR in the pulse counter mode

LPTMR_CSR[TPS]	LPTMR
0b00	Pin LPT_ALT0 selected (CMP0, CMP1 or CMP2 output)[1]

Table continues on the next page...

Table 39-1. Selecting the timer pin to be used by LPTMR in the pulse counter mode (continued)

LPTMR_CSR[TPS]	LPTMR
0b01	Pin LPT_ALT1 selected (External pin LPTMR0)
0b10	Pin LPT_ALT2 selected (External pin LPTMR1)
0b11	Pin LPT_ALT3 selected (External pin LPTMR2)

1Selection between CMP0, CMP1 and
. CMP2 controlled via
SIM_SOPT1CFG[LPTMRSEL0].

39.1.3 Clock Options

LPTMR's PSR[PCS] field controls the selection of external clock options.

- Setting PSR[PCS] to 00 selects the MCGIRCLK internal reference clock (not available in low leakage power modes).
- Setting PSR[PCS] to 01 selects the internal 1 kHz LPO clock.
- Setting PSR[PCS] to 10 selects OSCCLK32K, the 32.768 kHz clock from OSC32K. This connection is optimized for minimal power consumption in stop modes.
- Setting PSR[PCS] to 11 selects the ERCLK

39.2 Introduction

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

39.2.1 Features

The features of the LPTMR module include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low-power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare

- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

39.2.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 39-2. Modes of operation

Modes	Description
Run	The LPTMR operates normally.
Wait	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.
Stop	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.
Low-Leakage	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.
Debug	The LPTMR operates normally in Pulse Counter mode, but counter does not increment in Time Counter mode.

39.3 LPTMR signal descriptions

Table 39-3. LPTMR signal descriptions

Signal	I/O	Description
LPTMR0_ALT*n*	I	Pulse Counter Input pin

39.3.1 Detailed signal descriptions

Table 39-4. LPTMR interface—detailed signal descriptions

Signal	I/O	Description	
LPTMR_ALT*n*	I	Pulse Counter Input The LPTMR can select one of the input pins to be used in Pulse Counter mode.	
		State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment.

Table continues on the next page...

Table 39-4. LPTMR interface—detailed signal descriptions (continued)

Signal	I/O	Description	
			Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.
		Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.

39.4 Memory map and register definition

LPTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_C000	Low Power Timer Control Status Register (LPTMR_CSR)	32	R/W	0000_0000h	39.4.1/764
4003_C004	Low Power Timer Prescale Register (LPTMR_PSR)	32	R/W	0000_0000h	39.4.2/766
4003_C008	Low Power Timer Compare Register (LPTMR_CMR)	32	R/W	0000_0000h	39.4.3/767
4003_C00C	Low Power Timer Counter Register (LPTMR_CNR)	32	R/W	0000_0000h	39.4.4/768

39.4.1 Low Power Timer Control Status Register (LPTMR_CSR)

Address: 4003_C000h base + 0h offset = 4003_C000h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TCF	TIE	TPS		TPP	TFC	TMS	TEN
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPTMR_CSR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 TCF	Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it.

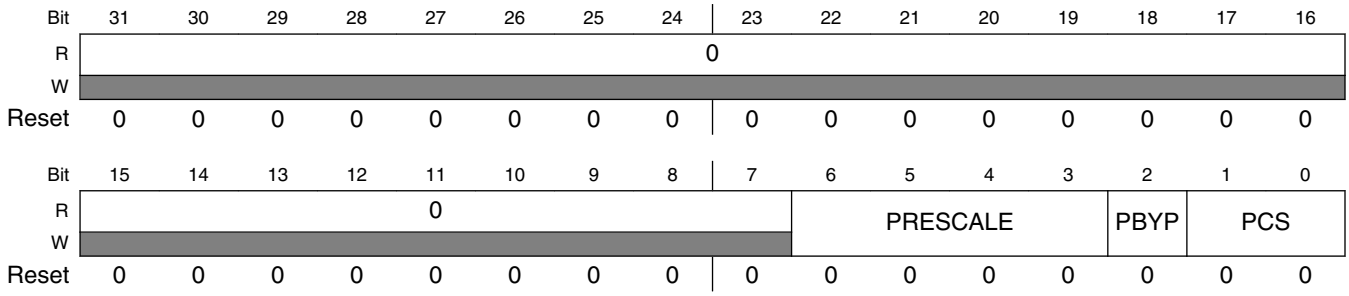
Table continues on the next page...

LPTMR_CSR field descriptions (continued)

Field	Description
	0 The value of CNR is not equal to CMR and increments. 1 The value of CNR is equal to CMR and increments.
6 TIE	Timer Interrupt Enable When TIE is set, the LPTMR Interrupt is generated whenever TCF is also set. 0 Timer interrupt disabled. 1 Timer interrupt enabled.
5–4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. TPS must be altered only when the LPTMR is disabled. The input connections vary by device. See the for information on the connections to these inputs. 00 Pulse counter input 0 is selected. 01 Pulse counter input 1 is selected. 10 Pulse counter input 2 is selected. 11 Pulse counter input 3 is selected.
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled. 0 Pulse Counter input source is active-high, and the CNR will increment on the rising-edge. 1 Pulse Counter input source is active-low, and the CNR will increment on the falling-edge.
2 TFC	Timer Free-Running Counter When clear, TFC configures the CNR to reset whenever TCF is set. When set, TFC configures the CNR to reset on overflow. TFC must be altered only when the LPTMR is disabled. 0 CNR is reset whenever TCF is set. 1 CNR is reset on overflow.
1 TMS	Timer Mode Select Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled. 0 Time Counter mode. 1 Pulse Counter mode.
0 TEN	Timer Enable When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, the LPTMR is enabled. While writing 1 to this field, CSR[5:1] must not be altered. 0 LPTMR is disabled and internal logic is reset. 1 LPTMR is enabled.

39.4.2 Low Power Timer Prescale Register (LPTMR_PSR)

Address: 4003_C000h base + 4h offset = 4003_C004h



LPTMR_PSR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–3 PRESCALE	Prescale Value Configures the size of the Prescaler in Time Counter mode or width of the glitch filter in Pulse Counter mode. PRESCALE must be altered only when the LPTMR is disabled. 0000 Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001 Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010 Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011 Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100 Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges. 0101 Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110 Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111 Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000 Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001 Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010 Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011 Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100 Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101 Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges.

Table continues on the next page...

LPTMR_PSR field descriptions (continued)

Field	Description
	1110 Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111 Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.
2 PBYP	Prescaler Bypass When PBYP is set, the selected prescaler clock in Time Counter mode or selected input source in Pulse Counter mode directly clocks the CNR. When PBYP is clear, the CNR is clocked by the output of the prescaler/glitch filter. PBYP must be altered only when the LPTMR is disabled. 0 Prescaler/glitch filter is enabled. 1 Prescaler/glitch filter is bypassed.
PCS	Prescaler Clock Select Selects the clock to be used by the LPTMR prescaler/glitch filter. PCS must be altered only when the LPTMR is disabled. The clock connections vary by device. **NOTE:** See the chip configuration details for information on the connections to these inputs. 00 Prescaler/glitch filter clock 0 selected. 01 Prescaler/glitch filter clock 1 selected. 10 Prescaler/glitch filter clock 2 selected. 11 Prescaler/glitch filter clock 3 selected.

39.4.3 Low Power Timer Compare Register (LPTMR_CMR)

Address: 4003_C000h base + 8h offset = 4003_C008h

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

LPTMR_CMR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COMPARE	Compare Value When the LPTMR is enabled and the CNR equals the value in the CMR and increments, TCF is set and the hardware trigger asserts until the next time the CNR increments. If the CMR is 0, the hardware trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the CMR must be altered only when TCF is set.

39.4.4 Low Power Timer Counter Register (LPTMR_CNR)

Address: 4003_C000h base + Ch offset = 4003_C00Ch

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

LPTMR_CNR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNTER	Counter Value

39.5 Functional description

39.5.1 LPTMR power and reset

The LPTMR remains powered in all power modes, including low-leakage modes. If the LPTMR is not required to remain operating during a low-power mode, then it must be disabled before entering the mode.

The LPTMR is reset only on global Power On Reset (POR) or Low Voltage Detect (LVD). When configuring the LPTMR registers, the CSR must be initially written with the timer disabled, before configuring the PSR and CMR. Then, CSR[TIE] must be set as the last step in the initialization. This ensures the LPTMR is configured correctly and the LPTMR counter is reset to zero following a warm reset.

39.5.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of the four clocks. The clock source must be enabled before the LPTMR is enabled.

NOTE

The clock source selected need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency f_{LPTMR} defined in the device datasheet.

39.5.3 LPTMR prescaler/glitch filter

The LPTMR prescaler and glitch filter share the same logic which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

39.5.3.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When the LPTMR is enabled, the CNR will increment every 2^2 to 2^{16} prescaler clock cycles. After the LPTMR is enabled, the first increment of the CNR will take an additional one or two prescaler clock cycles due to synchronization logic.

39.5.3.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When the LPTMR is enabled, the first increment will take an additional one or two prescaler clock cycles due to synchronization logic.

39.5.3.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When the LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output will also deassert.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising-edges	The glitch filter output will also assert.

NOTE

The input is only sampled on the rising clock edge.

The CNR will increment each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every 2^2 to 2^{16} prescaler clock edges. When first enabled, the glitch filter will wait an additional one or two prescaler clock edges due to synchronization logic.

39.5.3.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

39.5.4 LPTMR compare

When the CNR equals the value of the CMR and increments, the following events occur:

- CSR[TCF] is set.
- LPTMR interrupt is generated if CSR[TIE] is also set.
- LPTMR hardware trigger is generated.
- CNR is reset if CSR[TFC] is clear.

When the LPTMR is enabled, the CMR can be altered only when CSR[TCF] is set. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

39.5.5 LPTMR counter

The CNR increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when the LPTMR is disabled or if the counter register overflows. If CSR[TFC] is cleared, then the CNR is also reset whenever CSR[TCF] is set.

The CNR continues incrementing when the core is halted in Debug mode when configured for Pulse Counter mode, the CNR will stop incrementing when the core is halted in Debug mode when configured for Time Counter mode.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

39.5.6 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

When	Then
The CMR is set to 0 with CSR[TFC] clear	The LPTMR hardware trigger will assert on the first compare and does not deassert.
The CMR is set to a nonzero value, or, if CSR[TFC] is set	The LPTMR hardware trigger will assert on each compare and deassert on the following increment of the CNR.

39.5.7 LPTMR interrupt

The LPTMR interrupt is generated whenever CSR[TIE] and CSR[TCF] are set. CSR[TCF] is cleared by disabling the LPTMR or by writing a logic 1 to it.

CSR[TIE] can be altered and CSR[TCF] can be cleared while the LPTMR is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low-power mode, including the low-leakage modes, provided the LPTMR is enabled as a wakeup source.

Chapter 40
Periodic Interrupt Timer (PIT)

40.1 Chip-specific PIT information

40.1.1 Overview

The periodic interrupt timer (PIT) is a 32-bit timer that provides precise interrupts at regular intervals with minimal processor intervention. The timer can count down from the value written in the PIT modulus register, or it can be a free-running down-counter.

40.1.2 Instantiation Information

This device contains two PIT modules.

40.1.3 PIT/DMA Periodic Trigger Assignments

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

Table 40-1. PIT channel assignments for periodic DMA triggering

DMA Mux Number	PIT Channel
DMA Channel 0	PIT0 Channel 0
DMA Channel 1	PIT0 Channel 1
DMA Channel 2	PIT1 Channel 0
DMA Channel 3	PIT1 Channel 1

40.1.4 PIT/ADC Triggers

PIT triggers are selected as ADCx trigger sources by configuring the XBAR.

40.2 Introduction

The PIT module is an array of timers that can be used to raise interrupts and trigger DMA channels.

40.2.1 Block diagram

The following figure shows the block diagram of the PIT module.

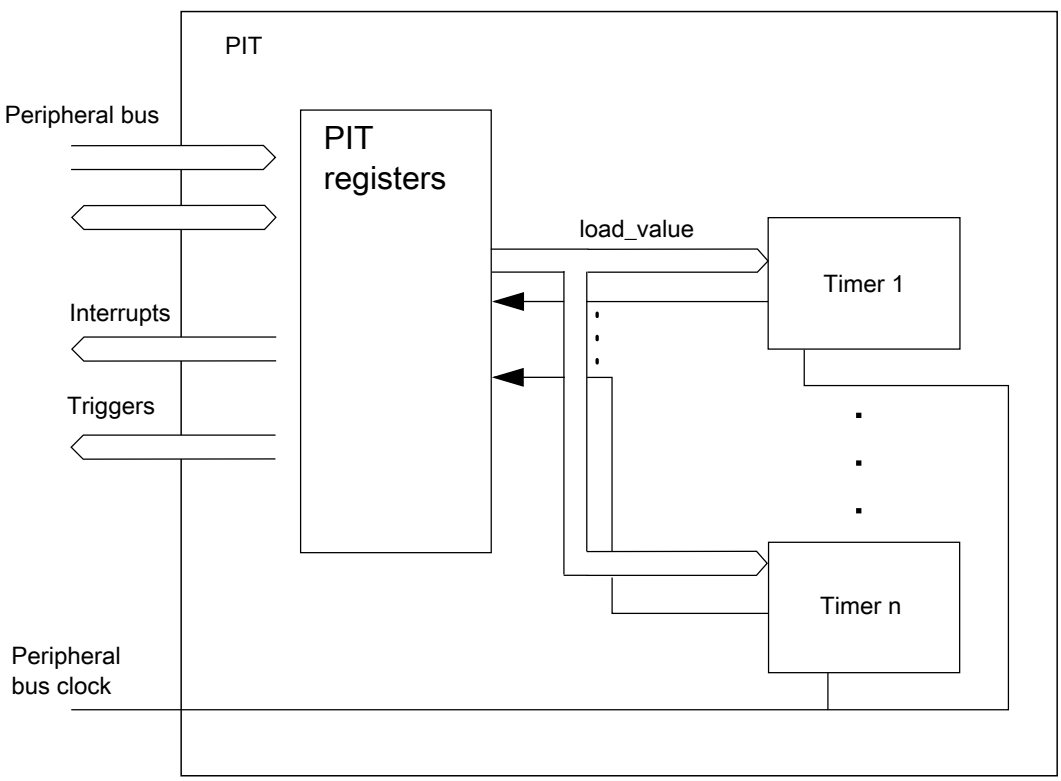


Figure 40-1. Block diagram of the PIT

NOTE

See the chip-specific PIT information for the number of PIT channels used in this MCU.

40.2.2 Features

The main features of this block are:

- Ability of timers to generate DMA trigger pulses
- Ability of timers to generate interrupts
- Maskable interrupts
- Independent timeout periods for each timer

40.3 Signal description

The PIT module has no external pins.

40.4 Memory map/register description

This section provides a detailed description of all registers accessible in the PIT module.

- Reserved registers will read as 0, writes will have no effect.
- See the chip-specific PIT information for the number of PIT channels used in this MCU.

PIT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_D000	PIT Module Control Register (PIT0_MCR)	32	R/W	0000_0000h	40.4.1/776
4002_D100	Timer Load Value Register (PIT0_LDVAL0)	32	R/W	0000_0000h	40.4.2/777
4002_D104	Current Timer Value Register (PIT0_CVAL0)	32	R	0000_0000h	40.4.3/777
4002_D108	Timer Control Register (PIT0_TCTRL0)	32	R/W	0000_0000h	40.4.4/778
4002_D10C	Timer Flag Register (PIT0_TFLG0)	32	R/W	0000_0000h	40.4.5/779
4002_D110	Timer Load Value Register (PIT0_LDVAL1)	32	R/W	0000_0000h	40.4.2/777
4002_D114	Current Timer Value Register (PIT0_CVAL1)	32	R	0000_0000h	40.4.3/777
4002_D118	Timer Control Register (PIT0_TCTRL1)	32	R/W	0000_0000h	40.4.4/778
4002_D11C	Timer Flag Register (PIT0_TFLG1)	32	R/W	0000_0000h	40.4.5/779
4002_E000	PIT Module Control Register (PIT1_MCR)	32	R/W	0000_0000h	40.4.1/776
4002_E100	Timer Load Value Register (PIT1_LDVAL0)	32	R/W	0000_0000h	40.4.2/777
4002_E104	Current Timer Value Register (PIT1_CVAL0)	32	R	0000_0000h	40.4.3/777
4002_E108	Timer Control Register (PIT1_TCTRL0)	32	R/W	0000_0000h	40.4.4/778
4002_E10C	Timer Flag Register (PIT1_TFLG0)	32	R/W	0000_0000h	40.4.5/779

Table continues on the next page...

PIT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_E110	Timer Load Value Register (PIT1_LDVAL1)	32	R/W	0000_0000h	40.4.2/777
4002_E114	Current Timer Value Register (PIT1_CVAL1)	32	R	0000_0000h	40.4.3/777
4002_E118	Timer Control Register (PIT1_TCTRL1)	32	R/W	0000_0000h	40.4.4/778
4002_E11C	Timer Flag Register (PIT1_TFLG1)	32	R/W	0000_0000h	40.4.5/779

40.4.1 PIT Module Control Register (PITx_MCR)

This register enables or disables the PIT timer clocks and controls the timers when the PIT enters the Debug mode.

Access: User read/write

Address: Base address + 0h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													Reserved	MDIS	FRZ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PITx_MCR field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved.
1 MDIS	Module Disable - (PIT section) Disables the standard timers. This field must be enabled before any other setup is done. 0 Clock for standard PIT timers is enabled. 1 Clock for standard PIT timers is disabled.
0 FRZ	Freeze Allows the timers to be stopped when the device enters the Debug mode. 0 Timers continue to run in Debug mode. 1 Timers are stopped in Debug mode.

40.4.2 Timer Load Value Register (PITx_LDVAL*n*)

These registers select the timeout period for the timer interrupts.

Access: User read/write

Address: Base address + 100h offset + (16d × i), where i=0d to 1d

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

PITx_LDVAL*n* field descriptions

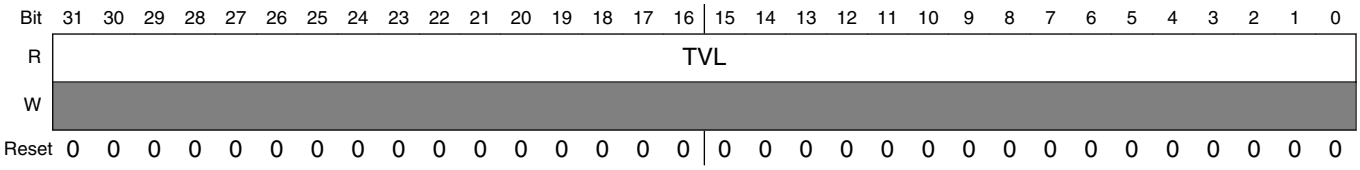
Field	Description
TSV	Timer Start Value Sets the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer; instead the value will be loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

40.4.3 Current Timer Value Register (PITx_CVAL*n*)

These registers indicate the current timer position.

Access: User read only

Address: Base address + 104h offset + (16d × i), where i=0d to 1d



PITx_CVALn field descriptions

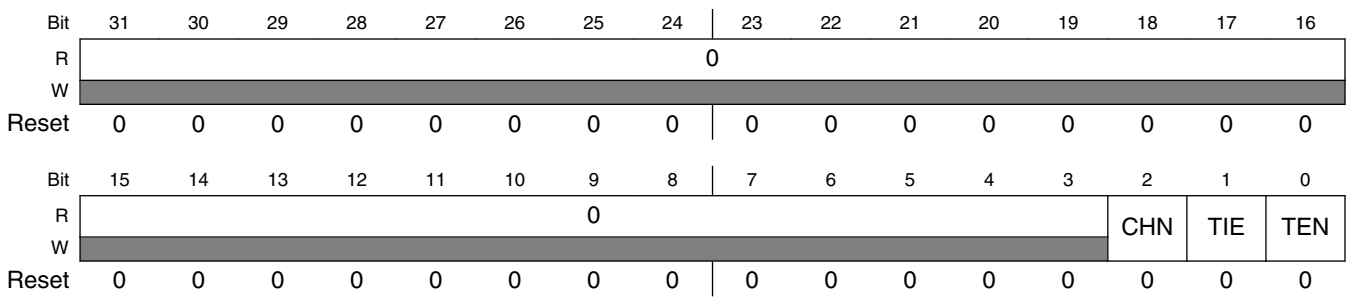
Field	Description
TVL	Current Timer Value Represents the current timer value, if the timer is enabled. **NOTE:** • If the timer is disabled, do not use this field as its value is unreliable. • The timer uses a downcounter. The timer values are frozen in Debug mode if MCR[FRZ] is set.

40.4.4 Timer Control Register (PITx_TCTRLn)

These registers contain the control bits for each timer.

Access: User read/write

Address: Base address + 108h offset + (16d × i), where i=0d to 1d



PITx_TCTRLn field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CHN	Chain Mode When activated, Timer n-1 needs to expire before timer n can decrement by 1. Timer 0 cannot be chained. 0 Timer is not chained. 1 Timer is chained to previous timer. For example, for Channel 2, if this field is set, Timer 2 is chained to Timer 1.
1 TIE	Timer Interrupt Enable

Table continues on the next page...

PITx_TCTRL*n* field descriptions (continued)

Field	Description
	When an interrupt is pending, or, TFLGn[TIF] is set, enabling the interrupt will immediately cause an interrupt event. To avoid this, the associated TFLGn[TIF] must be cleared first. 0 Interrupt requests from Timer n are disabled. 1 Interrupt will be requested whenever TIF is set.
0 TEN	Timer Enable Enables or disables the timer. 0 Timer n is disabled. 1 Timer n is enabled.

40.4.5 Timer Flag Register (PITx_TFLG*n*)

These registers hold the PIT interrupt flags.

Access: User read/write

Address: Base address + 10Ch offset + (16d × i), where i=0d to 1d

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TIF
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PITx_TFLG*n* field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TIF	Timer Interrupt Flag Sets to 1 at the end of the timer period. Writing 1 to this flag clears it. Writing 0 has no effect. If enabled, or, when TCTRLn[TIE] = 1, TIF causes an interrupt request. 0 Timeout has not yet occurred. 1 Timeout has occurred.

40.5 Functional description

This section provides the functional description of the module.

40.5.1 General operation

This section gives detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses and interrupts. Each interrupt is available on a separate interrupt line.

40.5.1.1 Timers

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their LDVAL registers, count down to 0 and then load the respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.

All interrupts can be enabled or masked by setting TCTRLn[TIE]. A new interrupt can be generated only after the previous one is cleared.

If desired, the current counter value of the timer can be read via the CVAL registers.

The counter period can be restarted, by first disabling, and then enabling the timer with TCTRLn[TEN]. See the following figure.

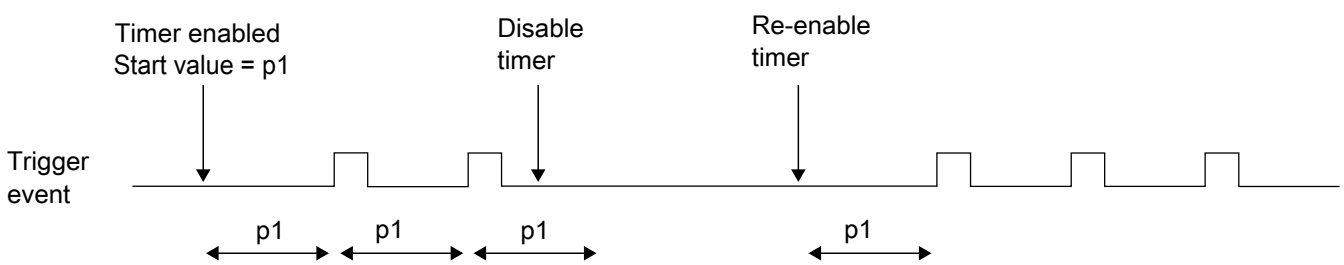


Figure 40-2. Stopping and starting a timer

The counter period of a running timer can be modified, by first disabling the timer, setting a new load value, and then enabling the timer again. See the following figure.

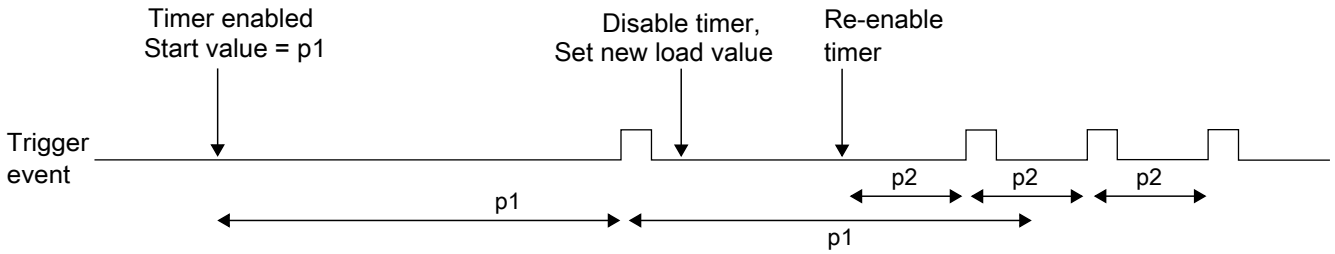


Figure 40-3. Modifying running timer period

It is also possible to change the counter period without restarting the timer by writing LDVAL with the new load value. This value will then be loaded after the next trigger event. See the following figure.

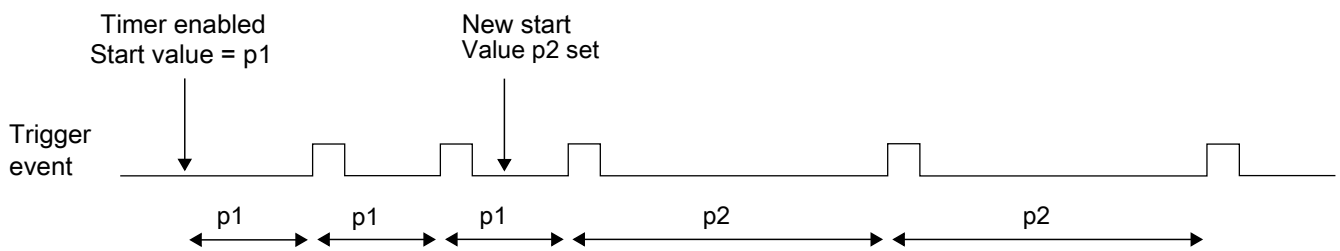


Figure 40-4. Dynamically setting a new load value

40.5.1.2 Debug mode

In Debug mode, the timers will be frozen based on MCR[FRZ]. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system, for example, the timer values, and then continue the operation.

40.5.2 Interrupts

All the timers support interrupt generation. See the MCU specification for related vector addresses and priorities.

Timer interrupts can be enabled by setting TCTRLn[TIE]. TFLGn[TIF] are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to the corresponding TFLGn[TIF].

40.5.3 Chained timers

When a timer has chain mode enabled, it will only count after the previous timer has expired. So if timer n-1 has counted down to 0, counter n will decrement the value by one. This allows to chain some of the timers together to form a longer timer. The first timer (timer 0) cannot be chained to any other timer.

40.6 Initialization and application information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.
- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

The PIT module must be activated by writing a 0 to MCR[MDIS].

The 50 MHz clock frequency equates to a clock period of 20 ns. Timer 1 needs to trigger every 5.12 ms/20 ns = 256,000 cycles and Timer 3 every 30 ms/20 ns = 1,500,000 cycles. The value for the LDVAL register trigger is calculated as:

LDVAL trigger = (period / clock period) -1

This means LDVAL1 and LDVAL3 must be written with 0x0003E7FF and 0x0016E35F respectively.

The interrupt for Timer 1 is enabled by setting TCTRL1[TIE]. The timer is started by writing 1 to TCTRL1[TEN].

Timer 3 shall be used only for triggering. Therefore, Timer 3 is started by writing a 1 to TCTRL3[TEN]. TCTRL3[TIE] stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;


// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1


// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3 for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

40.7 Example configuration for chained timers

In the example configuration:

- The PIT clock has a frequency of 100 MHz.
- Timers 1 and 2 are available.
- An interrupt shall be raised every 1 minute.

The PIT module needs to be activated by writing a 0 to MCR[MDIS].

The 100 MHz clock frequency equates to a clock period of 10 ns, so the PIT needs to count for 6000 million cycles, which is more than a single timer can do. So, Timer 1 is set up to trigger every 6 s (600 million cycles). Timer 2 is chained to Timer 1 and programmed to trigger 10 times.

The value for the LDVAL register trigger is calculated as number of cycles-1, so LDVAL1 receives the value 0x23C345FF and LDVAL2 receives the value 0x00000009.

The interrupt for Timer 2 is enabled by setting TCTRL2[TIE], the Chain mode is activated by setting TCTRL2[CHN], and the timer is started by writing a 1 to TCTRL2[TEN]. TCTRL1[TEN] needs to be set, and TCTRL1[CHN] and TCTRL1[TIE] are cleared.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;


// Timer 2
PIT_LDVAL2 = 0x00000009; // setup Timer 2 for 10 counts
PIT_TCTRL2 = TIE; // enable Timer 2 interrupt
PIT_TCTRL2 |= CHN; // chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN; // start Timer 2


// Timer 1
PIT_LDVAL1 = 0x23C345FF; // setup Timer 1 for 600 000 000 cycles
PIT_TCTRL1 = TEN; // start Timer 1
```

Chapter 41
Quad Timer (QTMR)

41.1 Chip-specific QTMR information

41.1.1 Overview

The quad-timer module provides four timer channels with a variety of controls affecting both individual and multi-channel features. Specific features include up/down count, cascading of counters, programmable modulo, count once/repeated, counter preload, compare registers with preload, shared use of input signals, prescaler controls, independent capture/compare, fault input control, programmable input filters, and multi-channel synchronization.

41.1.2 Instantiation Information

This device has one instance of Quad Timer with 4 timer channels (QTMR0_TMR0, QTMR0_TMR1, QTMR0_TMR2 and QTMR0_TMR3).

When the QTMR primary clock source is other than the bus clock, the QTMR register access will be limited. Under such case, it is recommended that QTMR register configuration is performed with BUSCLK before QTMR clock being switched to other clock source.

By setting and clearing the SIM_CTRL_REG[TMRFREEZE] bit, user can reset the QTMR TMRx counter (TMRx_CNTH / TMRx_CNTL) and the OFLAG.

NOTE

One TMR_ENBL register per module (4 channels). Hence, only TMR0_ENBL is applicable on this device. Ignore TMR1_ENBL, TMR2_ENBL, and TMR3_ENBL in the "TMR memory map".

41.2 Overview

Each timer module (TMR) contains four identical counter/timer groups. Each 16-bit counter/timer group contains a prescaler, a counter, a load register, a hold register, a capture register, two compare registers, two status and control registers, and one control register. All of the registers except the prescaler are read/writable.

NOTE

This document uses the terms "Timer" and "Counter" interchangeably because the counter/timers may perform either or both tasks.

The load register provides the initialization value to the counter when the counter's terminal value has been reached.

The hold register captures the counter's value when other counters are being read. This feature supports the reading of cascaded counters.

The capture register enables an external signal to take a "snap shot" of the counter's current value.

The COMP1 and COMP2 registers provide the values to which the counter is compared. If a match occurs, the OFLAG (TMR Output signal) can be set, cleared, or toggled. At match time, an interrupt is generated if enabled, and the new compare value is loaded into the COMP1 or COMP2 registers from CMPLD1 and CMPLD2 if enabled.

The prescaler provides different time bases useful for clocking the counter/timer.

The counter provides the ability to count internal or external events.

Within a timer module (set of four timer/counters), the input pins are shareable.

41.3 Features

The TMR module design includes these distinctive features:

- Four 16-bit counters/timers
- Count up/down
- Counters are cascadable
- Programmable count modulo

- Max count rate equals peripheral clock/2 for external clocks
- Max count rate equals peripheral clock for internal clocks
- Count once or repeatedly
- Counters are preloadable
- Compare registers are preloadable (available with compare load feature)
- Counters can share available input pins
- Separate prescaler for each counter
- Each counter has capture and compare capability
- Programmable operation during debug mode
- Inputs may act as fault inputs
- Programmable input filter
- Counting start can be synchronized across counters

41.4 Modes of Operation

The TMR module design operates in only a single mode of operation: Functional Mode. The various counting modes are detailed in the Functional Description.

41.5 Block Diagram

Each of the timer/counter groups within the quad-timer are shown in this figure.

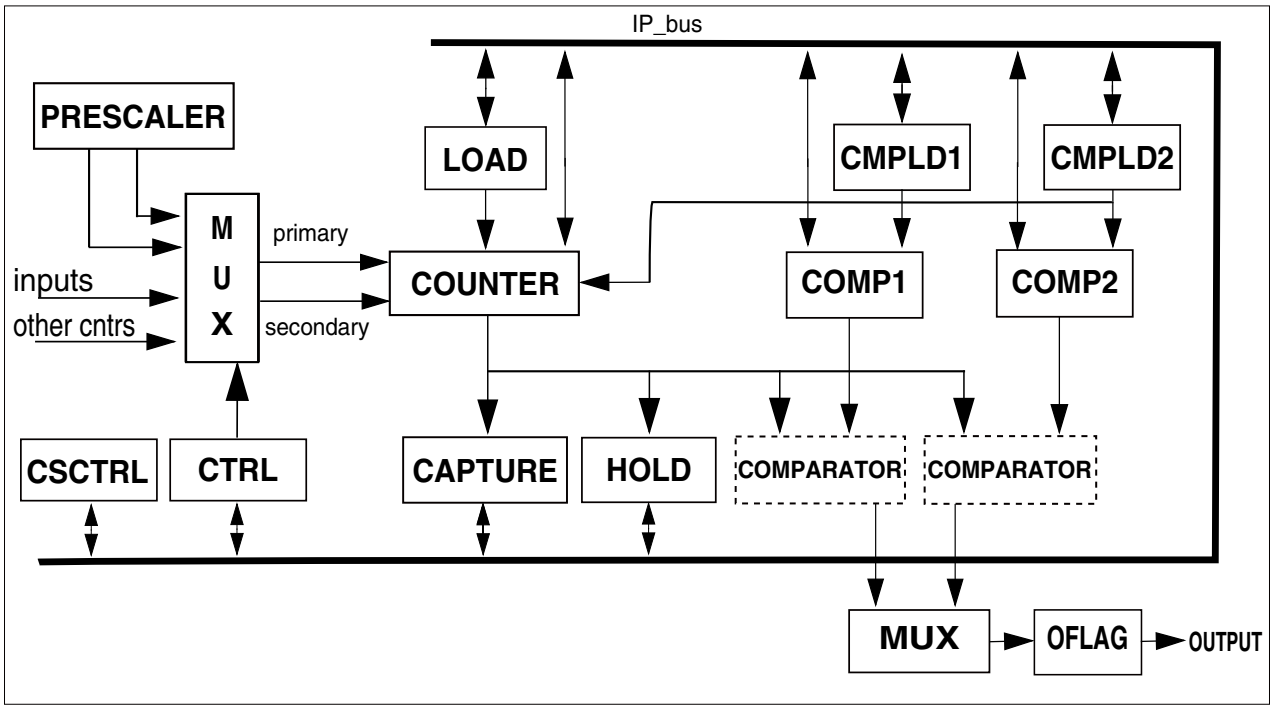


Figure 41-1. Quad Timer Block Diagram

41.6 Memory Map and Registers

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level and the address offset is defined at the module level. Make certain to check which quad timer is available on the chip being used, and which timer channels have external I/O.

TMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_7000	Timer Channel Compare Register 1 (TMR0_COMP1)	16	R/W	0000h	41.6.1/790
4005_7002	Timer Channel Compare Register 2 (TMR0_COMP2)	16	R/W	0000h	41.6.2/791
4005_7004	Timer Channel Capture Register (TMR0_CAPT)	16	R/W	0000h	41.6.3/791
4005_7006	Timer Channel Load Register (TMR0_LOAD)	16	R/W	0000h	41.6.4/791
4005_7008	Timer Channel Hold Register (TMR0_HOLD)	16	R/W	0000h	41.6.5/792
4005_700A	Timer Channel Counter Register (TMR0_CNTR)	16	R/W	0000h	41.6.6/792
4005_700C	Timer Channel Control Register (TMR0_CTRL)	16	R/W	0000h	41.6.7/792
4005_700E	Timer Channel Status and Control Register (TMR0_SCTRL)	16	R/W	0000h	41.6.8/795

Table continues on the next page...

TMR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_7010	Timer Channel Comparator Load Register 1 (TMR0_CMPLD1)	16	R/W	0000h	41.6.9/796
4005_7012	Timer Channel Comparator Load Register 2 (TMR0_CMPLD2)	16	R/W	0000h	41.6.10/ 797
4005_7014	Timer Channel Comparator Status and Control Register (TMR0_CSCTRL)	16	R/W	0000h	41.6.11/ 797
4005_7016	Timer Channel Input Filter Register (TMR0_FILT)	16	R/W	0000h	41.6.12/ 799
4005_7018	TMR0_RESERVED	16		0000h	41.6.13/ 0
4005_701E	Timer Channel Enable Register (TMR0_ENBL)	16	R/W	000Fh	41.6.13/ 800
4005_8000	Timer Channel Compare Register 1 (TMR1_COMP1)	16	R/W	0000h	41.6.1/790
4005_8002	Timer Channel Compare Register 2 (TMR1_COMP2)	16	R/W	0000h	41.6.2/791
4005_8004	Timer Channel Capture Register (TMR1_CAPT)	16	R/W	0000h	41.6.3/791
4005_8006	Timer Channel Load Register (TMR1_LOAD)	16	R/W	0000h	41.6.4/791
4005_8008	Timer Channel Hold Register (TMR1_HOLD)	16	R/W	0000h	41.6.5/792
4005_800A	Timer Channel Counter Register (TMR1_CNTR)	16	R/W	0000h	41.6.6/792
4005_800C	Timer Channel Control Register (TMR1_CTRL)	16	R/W	0000h	41.6.7/792
4005_800E	Timer Channel Status and Control Register (TMR1_SCTRL)	16	R/W	0000h	41.6.8/795
4005_8010	Timer Channel Comparator Load Register 1 (TMR1_CMPLD1)	16	R/W	0000h	41.6.9/796
4005_8012	Timer Channel Comparator Load Register 2 (TMR1_CMPLD2)	16	R/W	0000h	41.6.10/ 797
4005_8014	Timer Channel Comparator Status and Control Register (TMR1_CSCTRL)	16	R/W	0000h	41.6.11/ 797
4005_8016	Timer Channel Input Filter Register (TMR1_FILT)	16	R/W	0000h	41.6.12/ 799
4005_8018	TMR1_RESERVED	16		0000h	41.6.13/ 0
4005_801E	Timer Channel Enable Register (TMR1_ENBL)	16	R/W	000Fh	41.6.13/ 800
4005_9000	Timer Channel Compare Register 1 (TMR2_COMP1)	16	R/W	0000h	41.6.1/790
4005_9002	Timer Channel Compare Register 2 (TMR2_COMP2)	16	R/W	0000h	41.6.2/791
4005_9004	Timer Channel Capture Register (TMR2_CAPT)	16	R/W	0000h	41.6.3/791
4005_9006	Timer Channel Load Register (TMR2_LOAD)	16	R/W	0000h	41.6.4/791
4005_9008	Timer Channel Hold Register (TMR2_HOLD)	16	R/W	0000h	41.6.5/792
4005_900A	Timer Channel Counter Register (TMR2_CNTR)	16	R/W	0000h	41.6.6/792
4005_900C	Timer Channel Control Register (TMR2_CTRL)	16	R/W	0000h	41.6.7/792
4005_900E	Timer Channel Status and Control Register (TMR2_SCTRL)	16	R/W	0000h	41.6.8/795
4005_9010	Timer Channel Comparator Load Register 1 (TMR2_CMPLD1)	16	R/W	0000h	41.6.9/796

Table continues on the next page...

TMR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_9012	Timer Channel Comparator Load Register 2 (TMR2_CMPLD2)	16	R/W	0000h	41.6.10/ 797
4005_9014	Timer Channel Comparator Status and Control Register (TMR2_CSCTRL)	16	R/W	0000h	41.6.11/ 797
4005_9016	Timer Channel Input Filter Register (TMR2_FILT)	16	R/W	0000h	41.6.12/ 799
4005_9018	TMR2_RESERVED	16		0000h	41.6.13/ 0
4005_901E	Timer Channel Enable Register (TMR2_ENBL)	16	R/W	000Fh	41.6.13/ 800
4005_A000	Timer Channel Compare Register 1 (TMR3_COMP1)	16	R/W	0000h	41.6.1/790
4005_A002	Timer Channel Compare Register 2 (TMR3_COMP2)	16	R/W	0000h	41.6.2/791
4005_A004	Timer Channel Capture Register (TMR3_CAPT)	16	R/W	0000h	41.6.3/791
4005_A006	Timer Channel Load Register (TMR3_LOAD)	16	R/W	0000h	41.6.4/791
4005_A008	Timer Channel Hold Register (TMR3_HOLD)	16	R/W	0000h	41.6.5/792
4005_A00A	Timer Channel Counter Register (TMR3_CNTR)	16	R/W	0000h	41.6.6/792
4005_A00C	Timer Channel Control Register (TMR3_CTRL)	16	R/W	0000h	41.6.7/792
4005_A00E	Timer Channel Status and Control Register (TMR3_SCTRL)	16	R/W	0000h	41.6.8/795
4005_A010	Timer Channel Comparator Load Register 1 (TMR3_CMPLD1)	16	R/W	0000h	41.6.9/796
4005_A012	Timer Channel Comparator Load Register 2 (TMR3_CMPLD2)	16	R/W	0000h	41.6.10/ 797
4005_A014	Timer Channel Comparator Status and Control Register (TMR3_CSCTRL)	16	R/W	0000h	41.6.11/ 797
4005_A016	Timer Channel Input Filter Register (TMR3_FILT)	16	R/W	0000h	41.6.12/ 799
4005_A018	TMR3_RESERVED	16		0000h	41.6.13/ 0
4005_A01E	Timer Channel Enable Register (TMR3_ENBL)	16	R/W	000Fh	41.6.13/ 800

41.6.1 Timer Channel Compare Register 1 (TMRx_COMP1)

Address: Base address + 0h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	COMPARISON_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMRx_COMP1 field descriptions

Field	Description
COMPARISON_1	Comparison Value 1 This read/write register stores the value used for comparison with the counter value in count up mode.

41.6.2 Timer Channel Compare Register 2 (TMRx_COMP2)

Address: Base address + 2h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	COMPARISON_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMRx_COMP2 field descriptions

Field	Description
COMPARISON_2	Comparison Value 2 This read/write register stores the value used for comparison with the counter value in count down mode or alternating compare mode.

41.6.3 Timer Channel Capture Register (TMRx_CAPT)

Address: Base address + 4h offset

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

TMRx_CAPT field descriptions

Field	Description
CAPTURE	Capture Value This read/write register stores the value captured from the counter.

41.6.4 Timer Channel Load Register (TMRx_LOAD)

Address: Base address + 6h offset

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

TMRx_LOAD field descriptions

Field	Description
LOAD	Timer Load Register This read/write register stores the value used to initialize the counter after counter compare.

41.6.5 Timer Channel Hold Register (TMRx_HOLD)

Address: Base address + 8h offset

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

TMRx_HOLD field descriptions

Field	Description
HOLD	This read/write register stores the counter's values of specific channels whenever any of the four counters within a module is read.

41.6.6 Timer Channel Counter Register (TMRx_CNTR)

Address: Base address + Ah offset

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

TMRx_CNTR field descriptions

Field	Description
COUNTER	This read/write register is the counter for the corresponding channel in a timer module.

41.6.7 Timer Channel Control Register (TMRx_CTRL)

Address: Base address + Ch offset

Bit	15	14	13	12	11	10	9	8
Read Write	CM			PCS				SCS
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	SCS	ONCE	LENGTH	DIR	COINIT	OUTMODE		
Reset	0	0	0	0	0	0	0	0

TMRx_CTRL field descriptions

Field	Description
15–13 CM	Count Mode These bits control the basic counting and behavior of the counter. 000 No operation 001 Count rising edges of primary source[1] 010 Count rising and falling edges of primary source[2] 011 Count rising edges of primary source while secondary input high active 100 Quadrature count mode, uses primary and secondary sources 101 Count rising edges of primary source; secondary source specifies direction[3] 110 Edge of secondary source triggers primary count until compare 111 Cascaded counter mode (up/down)[4]
12–9 PCS	Primary Count Source These bits select the primary count source. **NOTE:** A timer selecting its own output for input is not a legal choice. The result is no counting. 0000 Counter 0 input pin 0001 Counter 1 input pin 0010 Counter 2 input pin 0011 Counter 3 input pin 0100 Counter 0 output 0101 Counter 1 output 0110 Counter 2 output 0111 Counter 3 output 1000 IP bus clock divide by 1 prescaler 1001 IP bus clock divide by 2 prescaler 1010 IP bus clock divide by 4 prescaler 1011 IP bus clock divide by 8 prescaler 1100 IP bus clock divide by 16 prescaler 1101 IP bus clock divide by 32 prescaler 1110 IP bus clock divide by 64 prescaler 1111 IP bus clock divide by 128 prescaler
8–7 SCS	Secondary Count Source These bits identify the external input pin to be used as a count command or timer command. The selected input can trigger the timer to capture the current value of CNTR . The selected input can also be used to specify the count direction. The selected signal can also be used as a fault input when CSCTRL[FAULT] is set. The polarity of the signal can be inverted by SCTRL[IPS]. 00 Counter 0 input pin 01 Counter 1 input pin 10 Counter 2 input pin 11 Counter 3 input pin
6 ONCE	Count Once This bit selects continuous or one shot counting mode.

Table continues on the next page...

TMRx_CTRL field descriptions (continued)

Field	Description
	0 Count repeatedly. 1 Count until compare and then stop. If counting up, a successful compare occurs when the counter reaches a COMP1 value. If counting down, a successful compare occurs when the counter reaches a COMP2 value. When output mode $4 is used, the counter re-initializes after reaching the COMP1 value, continues to count to the COMP2 value, and then stops.
5 LENGTH	Count Length This bit determines whether the counter: • counts to the compare value and then re-initializes itself to the value specified in the LOAD (or CMPLD2) register, or • continues counting past the compare value to the binary roll over. 0 Count until roll over at $FFFF and continue from $0000. 1 Count until compare, then re-initialize. If counting up, a successful compare occurs when the counter reaches a COMP1 value. If counting down, a successful compare occurs when the counter reaches a COMP2 value. When output mode $4 is used, alternating values of COMP1 and COMP2 are used to generate successful comparisons. For example, the counter counts until a COMP1 value is reached, re-initializes, counts until COMP2 value is reached, re-initializes, counts until COMP1 value is reached, and so on.
4 DIR	Count Direction This bit selects either the normal count direction up, or the reverse direction, down. 0 Count up. 1 Count down.
3 COINIT	Co-Channel Initialization This bit enables another counter/timer within the module to force the re-initialization of this counter/timer when it has an active compare event. 0 Co-channel counter/timers cannot force a re-initialization of this counter/timer 1 Co-channel counter/timers may force a re-initialization of this counter/timer
OUTMODE	Output Mode These bits determine the mode of operation for the OFLAG output signal. 000 Asserted while counter is active 001 Clear OFLAG output on successful compare 010 Set OFLAG output on successful compare 011 Toggle OFLAG output on successful compare 100 Toggle OFLAG output using alternating compare registers 101 Set on compare, cleared on secondary source input edge 110 Set on compare, cleared on counter rollover 111 Enable gated clock output while counter is active

1. Rising edges are counted only when SCTRL[IPS] = 0. Falling edges are counted when SCTRL[IPS] = 1. If the primary count source is IP bus clock divide by 1, only rising edges are counted regardless of the value of SCTRL[IPS].
2. IP bus clock divide by 1 cannot be used as a primary count source in edge count mode.
3. Rising edges are counted only when SCTRL[IPS] = 0. Falling edges are counted when SCTRL[IPS] = 1.
4. The primary count source must be set to one of the counter outputs.

41.6.8 Timer Channel Status and Control Register (TMRx_SCTRL)

Address: Base address + Eh offset

Bit	15	14	13	12	11	10	9	8
Read	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	CAPTURE_MODE		MSTR	EEOF	VAL	0	OPS	OEN
Write						FORCE		
Reset	0	0	0	0	0	0	0	0

TMRx_SCTRL field descriptions

Field	Description
15 TCF	Timer Compare Flag This bit is set when a successful compare occurs. This bit is cleared by writing a zero to this bit location.
14 TCFIE	Timer Compare Flag Interrupt Enable This bit (when set) enables interrupts when TCF is set.
13 TOF	Timer Overflow Flag This bit is set when the counter rolls over its maximum value $FFFF or $0000 (depending on count direction). This bit is cleared by writing a zero to this bit location.
12 TOFIE	Timer Overflow Flag Interrupt Enable This bit (when set) enables interrupts when TOF is set.
11 IEF	Input Edge Flag This bit is set when CAPTMODE is enabled and a proper input transition occurs (on an input selected as a secondary count source) while the count mode does not equal 000. This bit is cleared by writing a zero to this bit position. **NOTE:** Setting the input polarity select bit (IPS) changes the edge to be detected. Also, the control register's secondary count source determines which external input pin is monitored by the detection circuitry.
10 IEFIE	Input Edge Flag Interrupt Enable This bit (when set) enables interrupts when IEF is set.
9 IPS	Input Polarity Select This bit (when set) inverts the input signal polarity.
8 INPUT	External Input Signal This read-only bit reflects the current state of the external input pin selected via the secondary count source after application of IPS and filtering.
7–6 CAPTURE_ MODE	Input Capture Mode

Table continues on the next page...

TMRx_SCTRL field descriptions (continued)

Field	Description
	These bits specify the operation of the capture register as well as the operation of the input edge flag. The input source is the secondary count source. 00 Capture function is disabled 01 Load capture register on rising edge (when IPS=0) or falling edge (when IPS=1) of input 10 Load capture register on falling edge (when IPS=0) or rising edge (when IPS=1) of input 11 Load capture register on both edges of input
5 MSTR	Master Mode This bit (when set) enables the compare function's output to be broadcasted to the other counters/timers in the module. This signal then can be used to re-initialize the other counters and/or force their OFLAG signal outputs.
4 EEOF	Enable External OFLAG Force This bit (when set) enables the compare from another counter/timer within the same module to force the state of this counter's OFLAG output signal.
3 VAL	Forced OFLAG Value This bit determines the value of the OFLAG output signal when software triggers a FORCE command.
2 FORCE	Force OFLAG Output This write only bit forces the current value of VAL to be written to the OFLAG output. This bit always reads as a zero. VAL and FORCE can be written simultaneously in a single write operation. Write to FORCE only if the counter is disabled. Setting this bit while the counter is enabled may yield unpredictable results.
1 OPS	Output Polarity Select This bit determines the polarity of the OFLAG output signal. 0 True polarity. 1 Inverted polarity.
0 OEN	Output Enable This bit determines the direction of the external pin. 0 The external pin is configured as an input. 1 The OFLAG output signal is driven on the external pin. Other timer groups using this external pin as their input see the driven value. The polarity of the signal is determined by OPS.

41.6.9 Timer Channel Comparator Load Register 1 (TMRx_CMPLD1)

Address: Base address + 10h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	COMPARATOR_LOAD_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMRx_CMPLD1 field descriptions

Field	Description
COMPARATOR_LOAD_1	This read/write register is the comparator 1 preload value for the COMP1 register for the corresponding channel in a timer module.

41.6.10 Timer Channel Comparator Load Register 2 (TMRx_CMPLD2)

Address: Base address + 12h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	COMPARATOR_LOAD_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TMRx_CMPLD2 field descriptions

Field	Description
COMPARATOR_LOAD_2	This read/write register is the comparator 2 preload value for the COMP2 register for the corresponding channel in a timer module.

41.6.11 Timer Channel Comparator Status and Control Register (TMRx_CSCTRL)

Address: Base address + 14h offset

Bit	15	14	13	12	11	10	9	8
Read	DBG_EN		FAULT	ALT_LOAD	ROC	TCI	UP	OFLAG
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	TCF2EN	TCF1EN	TCF2	TCF1	CL2		CL1	
Write								
Reset	0	0	0	0	0	0	0	0

TMRx_CSCTRL field descriptions

Field	Description
15–14 DBG_EN	Debug Actions Enable These bits allow the TMR module to perform certain actions in response to the chip entering debug mode. 00 Continue with normal operation during debug mode. (default) 01 Halt TMR counter during debug mode. 10 Force TMR output to logic 0 (prior to consideration of SCTRL[OPS]). 11 Both halt counter and force output to 0 during debug mode.

Table continues on the next page...

TMRx_CSCTRL field descriptions (continued)

Field	Description
13 FAULT	Fault Enable The selected secondary input acts as a fault signal so that the timer OFLAG is cleared when the secondary input is set. When the secondary input is used in this mode, there is no resynchronization of the input so that there is a combinational path to clear the OFLAG. Fault inputs less than two clock periods wide will not be latched. Latched faults will be cleared the next time that the counter logic sets the OFLAG. 0 Fault function disabled. 1 Fault function enabled.
12 ALT_LOAD	Alternative Load Enable This bit allows for an alternative method for loading the counter during modulo counting. Normally, the counter can be loaded only with the value from the LOAD register. When this bit is set, the counter is loaded from the LOAD register when counting up and a match with COMP1 occurs, or the counter is loaded from the CMPLD2 register when counting down and a match with COMP2 occurs. 0 Counter can be re-initialized only with the LOAD register. 1 Counter can be re-initialized with the LOAD or CMPLD2 registers depending on count direction.
11 ROC	Reload on Capture This bit enables the capture function to cause the counter to be reloaded from the LOAD register. 0 Do not reload the counter on a capture event. 1 Reload the counter on a capture event.
10 TCI	Triggered Count Initialization Control This bit is used during triggered count mode, CTRL[CM] = 110, to enable the counter to be re-initialized when a second trigger occurs while the counter is still counting. Normally, the second trigger causes the counting to stop/pause until a third trigger occurs. With this bit set, a second trigger event causes the counter to re-initialize and continue counting. 0 Stop counter upon receiving a second trigger event while still counting from the first trigger event. 1 Reload the counter upon receiving a second trigger event while still counting from the first trigger event.
9 UP	Counting Direction Indicator This read-only bit is used during quadrature count mode, CTRL[CM] = 100, to read the direction of the last count. CTRL[DIR] reverses the sense of this bit. 0 The last count was in the DOWN direction. 1 The last count was in the UP direction.
8 OFLAG	Output flag This read only bit shows the state of the Timer's internal OFLAG signal prior to consideration of polarity or debug mode.
7 TCF2EN	Timer Compare 2 Interrupt Enable An interrupt is issued when both this bit and TCF2 are set.
6 TCF1EN	Timer Compare 1 Interrupt Enable An interrupt is issued when both this bit and TCF1 are set.
5 TCF2	Timer Compare 2 Interrupt Flag

Table continues on the next page...

TMRx_CSCTRL field descriptions (continued)

Field	Description
	When set, this bit indicates a successful comparison of the timer and the the COMP2 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing a zero to this bit location.
4 TCF1	Timer Compare 1 Interrupt Flag When set, this bit indicates a successful comparison of the timer and the the COMP1 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing a zero to this bit location.
3–2 CL2	Compare Load Control 2 These bits control when COMP2 is preloaded with the value from CMPLD2. 00 Never preload 01 Load upon successful compare with the value in COMP1 10 Load upon successful compare with the value in COMP2 11 Reserved
CL1	Compare Load Control 1 These bits control when COMP1 is preloaded with the value from CMPLD1. 00 Never preload 01 Load upon successful compare with the value in COMP1 10 Load upon successful compare with the value in COMP2 11 Reserved

41.6.12 Timer Channel Input Filter Register (TMRx_FILT)

The FILT register programs the values for the filtering of the corresponding input without regard for the fact that any timer channel can use the input as a count source.

Input filter considerations:

- Set the FILT_PER value such that the sampling period is larger than the period of the expected noise. In this way, a noise spike will corrupt only one sample. Choose the FILT_CNT value to reduce the probability that noisy samples cause an incorrect transition to be recognized. The probability of an incorrect transistion is defined as the probability of an incorrect sample raised to the power of (FILT_CNT + 3).
- The values of FILT_PER and FILT_CNT must also be balanced against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of (((FILT_CNT + 3) x FILT_PER) + 2) IP bus clock periods.

Address: Base address + 16h offset

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

TMRx_FILT field descriptions

Field	Description
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FILT_CNT	Input Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. A value of 0x0 represents 3 samples. A value of 0x7 represents 10 samples. The value of FILT_CNT affects the input latency.
FILT_PER	Input Filter Sample Period These bits represent the sampling period (in IP bus clock cycles) of the TMR input signals. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. The value of FILT_PER affects the input latency. When changing values for FILT_PER from one non-zero value to another non-zero value, write a value of zero first to clear the filter.

41.6.13 Timer Channel Enable Register (TMRx_ENBL)

Address: Base address + 1Eh offset

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

TMRx_ENBL field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ENBL	Timer Channel Enable These bits enable the prescaler (if it is being used) and counter in each channel. Multiple ENBL bits can be set at the same time to synchronize the start of separate counters. If an ENBL bit is set, then the corresponding channel starts its counter as soon as the CTRL[CM] field has a value other than 0. When an ENBL bit is clear, the corresponding counter maintains its current value. 0 Timer channel is disabled. 1 Timer channel is enabled. (default)

41.7 Functional Description

41.7.1 General

The counter/timer has two basic modes of operation: it can count internal or external events, or it can count an internal clock source while an external input signal is asserted, thus timing the width of the external input signal.

- The counter can count the rising, falling, or both edges of the selected input pin.
- The counter can decode and count quadrature encoded input signals.
- The counter can count up and down using dual inputs in a "count with direction" format.
- The counter's terminal count value (modulo) is programmable.
    - The value that is loaded into the counter after reaching its terminal count is programmable.
- The counter can count repeatedly, or it can stop after completing one count cycle.
- The counter can be programmed to count to a programmed value and then immediately reinitialize, or it can count through the compare value until the count "rolls over" to zero.

The external inputs to each counter/timer are shareable among each of the four counter/timers within the module. The external inputs can be used as:

- Count commands
- Timer commands
- They can trigger the current counter value to be "captured"
- They can be used to generate interrupt requests

The polarity of the external inputs are selectable.

The primary output of each timer/counter is the output signal OFLAG. The OFLAG output signal can be:

- Set, cleared, or toggled when the counter reaches the programmed value.

- The OFLAG output signal may be output to an external pin instead of having that pin serve as a timer input.
- The OFLAG output signal enables each counter to generate square waves, PWM, or pulse stream outputs.
- The polarity of the OFLAG output signal is selectable.

Any counter/timer can be assigned as a master. A master's compare signal can be broadcast to the other counter/timers within the module. The other counters can be configured to reinitialize their counters and/or force their OFLAG output signals to predetermined values when a master's counter/timer compare event occurs.

41.7.2 Usage of Compare Registers

The dual compare registers (COMP1 and COMP2) provide a bidirectional modulo count capability. The COMP1 register is used when the counter is *counting up*, and the COMP2 register is used when the counter is *counting down*. Alternating compare mode is the only exception.

The COMP1 register should be set to the desired maximum count value or FFFFh to indicate the maximum unsigned value prior to roll-over, and the COMP2 register should be set to the minimum count value or 0000h to indicate the minimum unsigned value prior to roll-under.

If CTRL[OUTMODE] is set to 100, the OFLAG will toggle while using alternating compare registers. In this variable frequency PWM mode, the COMP2 value defines the desired pulse width of the on time, and the COMP1 register defines the off time.

Use caution when changing COMP1 and COMP2 while the counter is active. If the counter has already passed the new value, it will count to FFFFh or 0000h, roll over, then begin counting toward the new value. The check is: Count=CMPx, *not* Count > COMP1 or Count < COMP2.

The use of the CMPLD1 and CMPLD2 registers to compare values will help to minimize this problem.

41.7.3 Usage of Compare Load Registers

The CMPLD1, CMPLD2, and CSCTRL registers offer a high degree of flexibility for loading compare registers with user-defined values on different compare events. To ensure correct functionality while using these registers we strongly suggest using the following method described in this section.

The purpose of the compare load feature is to allow quicker updating of the compare registers. In the past, a compare register could be updated using interrupts. However, because of the latency between an interrupt event occurring and the service of that interrupt, there was the possibility that the counter may have already counted past the new compare value by the time the compare register was updated by the interrupt service routine. The counter would then continue counting until it rolled over and reached the new compare value.

To address this, the compare registers are now updated in hardware in the same way the counter register is re-initialized to the value stored in the load register. The compare load feature allows the user to calculate new compare values and store them in to the comparator load registers. When a compare event occurs, the new compare values in the comparator load registers are written to the compare registers eliminating the use of software to do this.

The compare load feature is intended to be used in variable frequency PWM mode. The COMP1 register determines the pulse width for the logic low part of OFLAG and COMP2 determines the pulse width for the logic high part of OFLAG. The period of the waveform is determined by the COMP1 and COMP2 values and the frequency of the primary clock source. See the following figure.

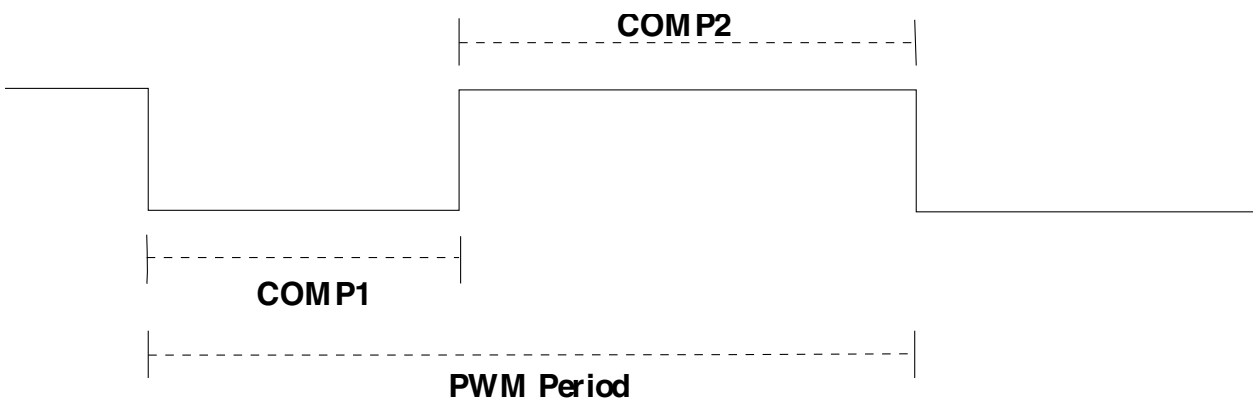


Figure 41-2. Variable PWM Waveform

Should we desire to update the duty cycle or period of the above waveform, we would need to update the COMP1 and COMP2 values using the compare load feature.

41.7.4 Usage of the Capture Register

The capture register stores a copy of the counter's value when an input edge (positive, negative, or both) is detected. After a capture event occurs, no further updating of the capture register will occur until the SCTRL[IEF] (input edge flag) is cleared by writing a zero to the SCTRL[IEF].

41.7.5 Functional Modes

The selected external count signals are sampled at the TMR's base clock rate and then run through a transition detector. The maximum count rate is one-half of the TMR's base clock rate. Internal clock sources can be used to clock the counters at the TMR's base clock rate.

If a counter is programmed to count to a specific value and then stop, the CTRL[CM] field is cleared when the count terminates.

41.7.5.1 Stop Mode

If CTRL[CM] is set to '000', the counter is inert. No counting will occur. Stop mode will also disable the interrupts caused by input transitions on a selected input pin.

41.7.5.2 Count Mode

If CTRL[CM] is set to '001', the counter will count the rising edges of the selected clock source. This mode is useful for generating periodic interrupts for timing purposes, or counting external events such as "widgets" on a conveyor belt passing a sensor. If the selected input is inverted by setting SCTRL[IPS] (input polarity select), then the negative edge of the selected external input signal is counted.

Example: 41.7.5.2.1 Count Pulses from External Source

```
//        (See Processor Expert PulseAccumulator bean.)
//    This example uses TMR1 to count pulse (actually counts rising edges of the pulse)
//      from an external source (QT3).
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0600);           /* Set up mode */
```

```
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);              /* Reset counter register */
  setReg(TMR1_LOAD,0x00);              /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x01);  /* Run counter */
}
```

Example: 41.7.5.2.2 Generate Periodic Interrupt By Counting Internal Clocks

```
//         (See Processor Expert TimerInt bean.)
//   This example generates an interrupt every 100ms,
//     assuming the chip is operating at 60 MHz.
//
// It does this by using the IP_bus_clk divided by 128 as the counter clock source.
// The counter then counts to 46874 where it matches the COMP1 value.
// At that time an interrupt is generated, the counter is reloaded and
// the next COMP1 value is loaded from CMPLD1.
//
void TimerInt_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0x20);              /* Stop all functions of the timer */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
  setReg(TMR0_COMP1,46874);           /* Set up compare 1 register */
  setReg(TMR0_CMPLD1,46874);          /* Also set the compare preload register */
  /* TMR0_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,TCF2EN=0,TCF1EN=1,
                  TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR0_CSCTRL,0x41);           /* Enable compare 1 interrupt and */
                                      /*   compare 1 preload */
  setRegBitGroup(TMR0_CTRL,PCS,0xF); /* Primary Count Source to IP_bus_clk / 128 */
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setRegBitGroup(TMR0_CTRL,CM,0x01); /* Run counter */
}
```

41.7.5.3 Edge-Count Mode

If CTRL[CM] is set to '010', the counter will count both edges of the selected external clock source. This mode is useful for counting the changes in the external environment, such as a simple encoder wheel.

Example: 41.7.5.3.1 Count Both Edges of External Source Signal

```
//       (See Processor Expert PulseAccumulator bean.)
//  This example uses TMR1 to count pulse (actually counts rising edges of the pulse)
//    from an external source (QT3).
```

```
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0600);           /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x02);  /* Run counter */
}
```

41.7.5.4 Gated-Count Mode

If CTRL[CM] is set to '011', the counter will count while the selected secondary input signal is high. This mode is used to time the duration of external events. If the selected input is inverted by setting SCTRL[IPS] (input polarity select), then the counter will count while the selected secondary input is low.

Example: 41.7.5.4.1 Capture Duration of External Pulse

```
//       (See Processor Expert PulseAccumulator bean.)
//   This example uses TMR1 to determine the duration of an external pulse.
//
//   The IP_bus clock is used as the primary counter. If the duration of the
//     external pulse is longer than 0.001 seconds one of the other IP_bus clock
//     dividers can be used. If the pulse duration is longer than 0.128 seconds
//     an external clock source will have to be used as the primary clock source.
//
void Pulse1_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=8,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x1080);            /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x03);  /* Run counter */
}
```

41.7.5.5 Quadrature-Count Mode

If CTRL[CM] is set to '100', the counter will decode the primary and secondary external inputs as quadrature encoded signals. Quadrature signals are usually generated by rotary or linear sensors used to monitor movement of motor shafts or mechanical equipment. The quadrature signals are square waves that are 90 degrees out of phase. The decoding of quadrature signal provides both count and direction information.

This figure shows a timing diagram illustrating the basic operation of a quadrature incremental position encoder.

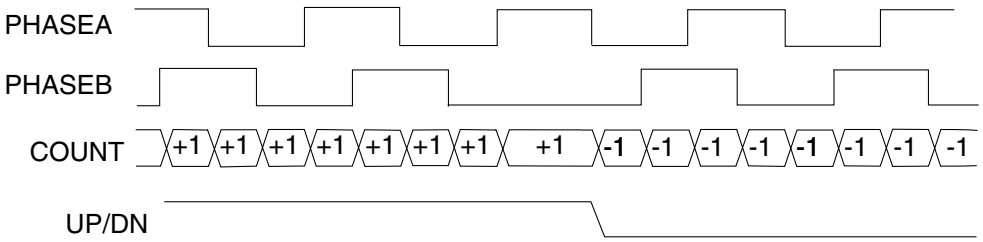


Figure 41-3. Quadrature Incremental Position Encoder

Example: 41.7.5.5.1 Quadrature Count Mode Example

```
//       (See Processor Expert PulseAccumulator bean.)
//  This example uses TMR0 for counting states of a quadrature position encoder.
//
//  Timer input 0 is used as the primary count source (PHASEA).
//  Timer input 1 is used as the secondary count source (PHASEB).
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMRC0_CTRL,0x80);              /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
  setReg(TMR0_COMP1,0xFFFF);           /* Set up compare 1 register */
  setReg(TMR0_COMP2,0x00);             /* Set up compare 2 register */
  /* TMR0_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR0_CSCTRL,0x00);
  setRegBitGroup(TMR0_CTRL,CM,0x04);  /* Run counter */
}
```

41.7.5.6 Quadrature-Count Mode with Index Input

As an extension to the quadrature count mode discussed in the previous paragraph, some rotary shafts have a HOME or INDEX indicator. This would be a third input to the timer that is used to reset the timer's counter.

In this example, channel 0 is used to decode the quadrature inputs, but it doesn't actually count. Because its upper and lower limits are both set to 0, its output is cascaded count up and count down signals each time the quadrature inputs indicate a change in count. Channel 1 works in cascaded count mode receiving its counting instructions from channel 0. When an input capture event occurs, channel 1 is programmed to reset its counter value. The channel 1 counter contains the position value for the shaft.

Example: 41.7.5.6.1 Quadrature Count Mode with Index Input Example

```
//       (See Processor Expert PulseAccumulator bean.)
//  This example uses TMR0and TMR1 for counting states of a quadrature position encoder.
//
// Timer input 0 is used as the primary count source (PHASEA).
// Timer input 1 is used as the secondary count source (PHASEB).
// Timer input 2 is used as the index input source (INDEX).
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=1,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0xA0);             /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
  setReg(TMR0_COMP1,0x00);           /* Set up compare 1 register */
  setReg(TMR0_COMP2,0x00);             /* Set up compare 2 register */
  /* TMR0_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR0_CSCTRL,0x00);
/* TMR1_CTRL: CM=7,PCS=100,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0xEA00);           /* Set up capture edge */
/* TMR1_SCTRL: Capture_Mode=10 */
  setReg(TMR1_SCTRL,0x0080);
/* TMR1_CSTRL: ROC=1 */
  setReg(TMR1_CTRL,0x0800);           /* Set up reload on capture */
setRegBitGroup(TMR0_CTRL,CM,0x04);  /* Run counter */
}
```

41.7.5.7 Signed-Count Mode

If CTRL[CM] is set to '101', the counter counts the primary clock source while the selected secondary source provides the selected count direction (up/down).

Example: 41.7.5.7.1 Signed Count Mode Example

```
//       (See Processor Expert PulseAccumulator bean.)
//  This example uses TMR0 for signed mode counting.
//
//  Timer input 2 is used as the primary count source.
//  Timer input 1 is used to determine the count direction.
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=2,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0x0480);             /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x1000);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
```

```
  setRegBitGroup(TMR0_CTRL,CM,0x05);  /* Run counter */
}
```

41.7.5.8 Triggered-Count Mode 1

If CSCTRL[TCI] is clear and CTRL[CM] is set to '110', the counter will begin counting the primary clock source after a positive transition (negative edge if SCTRL[IPS]=1) of the secondary input occurs. The counting will continue until a compare event occurs or another positive input transition is detected. If a second input transition occurs before a terminal count is reached, counting will stop and SCTRL[TCF] (timer compare flag) will be set. Subsequent secondary input transitions will continue to restart and stop the counting until a compare event occurs.

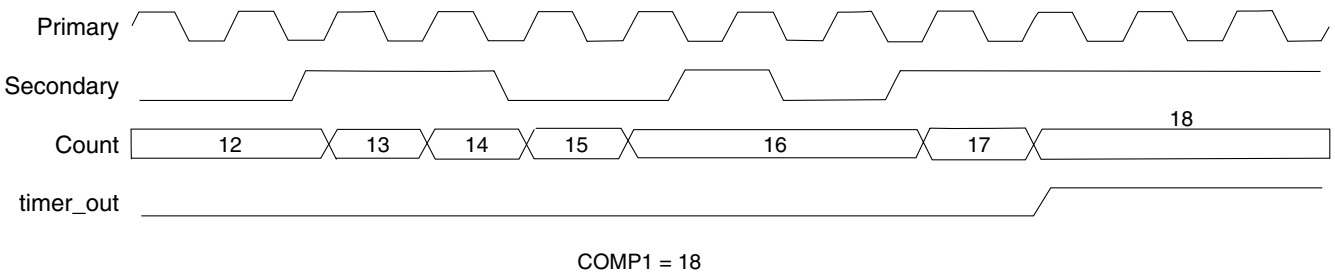


Figure 41-4. Triggered Count Mode 1 (CTRL[LENGTH]=0)

Example: 41.7.5.8.1 Triggered Count Mode 1 Example

```
//        (See Processor Expert PulseAccumulator bean.)
//   This example uses TMR1 for triggered mode counting.
//
//   Timer input 3 is used as the primary count source.
//   Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0700);             /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);
  setReg(TMR1_CNTR,0x00);               /* Reset counter register */
  setReg(TMR1_LOAD,0x00);               /* Reset load register */
  setReg(TMR1_COMP1,0x0012);             /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x00);
  setRegBitGroup(TMR1_CTRL,CM,0x06);    /* Run counter */
}
```

41.7.5.9 Triggered-Count Mode 2

If CSCTRL[TCI] is set and CTRL[CM] is set to '110', the counter will begin counting the primary clock source after a positive transition (negative edge if SCTRL[IPS]=1) of the secondary input occurs. The counting will continue until a compare event occurs or another positive input transition is detected. If a second input transition occurs before a terminal count was reached, the counter will reload and continue counting. When CSCTRL[TCI] is set, the OFLAG output mode, CTRL[OUTMODE], should probably be set to '101' (cleared on init, set on compare) to ensure the output will be in a known state after the second input transition and subsequent reload takes place.

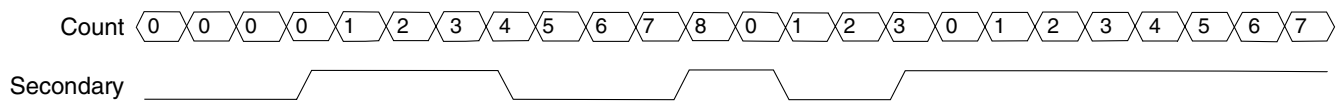


Figure 41-5. Triggered Count Mode 2 (CTRL[LENGTH]=0)

Example: 41.7.5.9.1 Triggered Count Mode 2Example

```
//        (See Processor Expert PulseAccumulator bean.)
//   This example uses TMR1 for triggered mode counting.
//
//   Timer input 3 is used as the primary count source.
//   Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0700);           /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);

  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setReg(TMR1_COMP1,0x0012);           /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x00);
  setRegBitGroup(TMR1_CTRL,CM,0x06); /* Run counter */
}
```

41.7.5.10 One-Shot Mode

If CTRL[CM] is set to '110', and the counter is set to reinitialize at a compare event (CTRL[LENGTH]=1),and CTRL[OUTMODE] is set to '101' (cleared on init, set on compare), the counter works in a one-shot mode. An external event causes the counter to count, and when the terminal count is reached, the output is asserted. This delayed output can be used to provide timing delays.

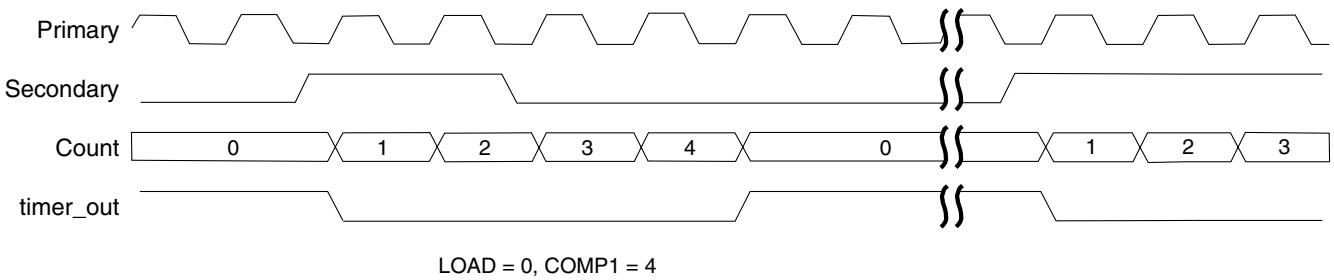


Figure 41-6. One-Shot Mode (CTRL[LENGTH]=1)

Example: 41.7.5.10.1 One-Shot Mode Example

```
//   (See Processor Expert PulseAccumulator bean.)
//  This example uses TMR1 for one-shot mode counting.
//
//  Timer input 3 is used as the primary count source.
//  Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=5 */
  setReg(TMR1_CTRL,0x0725);            /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
              Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);
  setReg(TMR1_CNTR,0x00);              /* Reset counter register */
  setReg(TMR1_LOAD,0x00);              /* Reset load register */
  setReg(TMR1_COMP1,0x0004);            /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x00);
  setRegBitGroup(TMR1_CTRL,CM,0x06);  /* Run counter */
}
```

41.7.5.11 Cascade-Count Mode

If CTRL[CM] is set to '111', the counter's input is connected to the output of another selected counter. The counter will count up and down as compare events occur in the selected source counter. This cascade or daisy-chained mode enables multiple counters to be cascaded to yield longer counter lengths. When operating in cascade mode, a special

high-speed signal path is used between modules rather than the OFLAG output signal. If the selected source counter is counting up and it experiences a compare event, the counter will be incremented. If the selected source counter is counting down and it experiences a compare event, the counter will be decremented.

Up to four counters may be cascaded to create a 64-bit wide synchronous counter. Check the data sheet to see if there are any frequency limits for cascaded counting mode.

Whenever any counter is read within a counter module, all of the counters' values within the module are captured in their respective hold registers. This action supports the reading of a cascaded counter chain. First read any counter of a cascaded counter chain, then read the hold registers of the other counters in the chain. The cascaded counter mode is synchronous.

Note

It is possible to connect counters together by using the other (non-cascade) counter modes and selecting the outputs of other counters as a clock source. In this case, the counters are operating in a ripple mode, where higher order counters will transition a clock later than a purely synchronous design.

Example: 41.7.5.11.1 Generate Periodic Interrupt Cascading Two Counters

```
//    (See Processor Expert TimerInt bean.)
// This example generates an interrupt every 30 seconds,
//    assuming the chip is operating at 60 MHz.
//
// To do this, counter 2 is used to count 60,000 IP_bus clocks, which means it
//    will compare and reload every 0.001 seconds.
// Counter 3 is cascaded and used to count the 0.001 second ticks and
//    generate the desired interrupt interval.
//
void TimerInt_Init(void)
{
// Set counter 2 to count IP_bus clocks
   /* TMR2_CTRL: CM=0,PCS=8,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
   setReg(TMR2_CTRL,0x1020);           /* Stop all functions of the timer */
// Set counter 3 as cascaded and to count counter 2 outputs
  /* TMR3_CTRL: CM=7,PCS=6,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR3_CTRL,0xEC20);            /* Set up cascade counter mode */
  /* TMR3_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR3_SCTRL,0x00);
  /* TMR2_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR2_SCTRL,0x00);
  setReg(TMR3_CNTR,0x00);             /* Reset counter register */
  setReg(TMR2_CNTR,0x00);
  setReg(TMR3_LOAD,0x00);             /* Reset load register */
  setReg(TMR2_LOAD,0x00);
  setReg(TMR3_COMP1, 30000);           /* milliseconds in 30 seconds */
  setReg(TMR3_CMPLD1,30000);
  setReg(TMR2_COMP1, 60000);          // Set to cycle every milisecond
```

```
  setReg(TMR2_CMPLD1,60000);
  /* TMR3_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=1,TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR3_CSCTRL,0x41);           /* Enable compare 1 interrupt and */
                                       /* compare 1 preload              */
  /* TMR2_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR2_CSCTRL,0x01);           /* Enable Compare 1 preload */
  setRegBitGroup(TMR2_CTRL,CM,0x01);  /* Run counter */
}
```

41.7.5.12 Pulse-Output Mode

If CTRL[CM]=001, and CTRL[OUTMODE] is set to 111' (gated clock output), and CTRL[ONCE] is set, then the counter will output a pulse stream of pulses that has the same frequency of the selected clock source, and the number of output pulses is equal to the compare value minus the init value. This mode is useful for driving step motor systems.

Note

This does not work if CTRL[PCS] is set to 1000 (IP_bus/1).

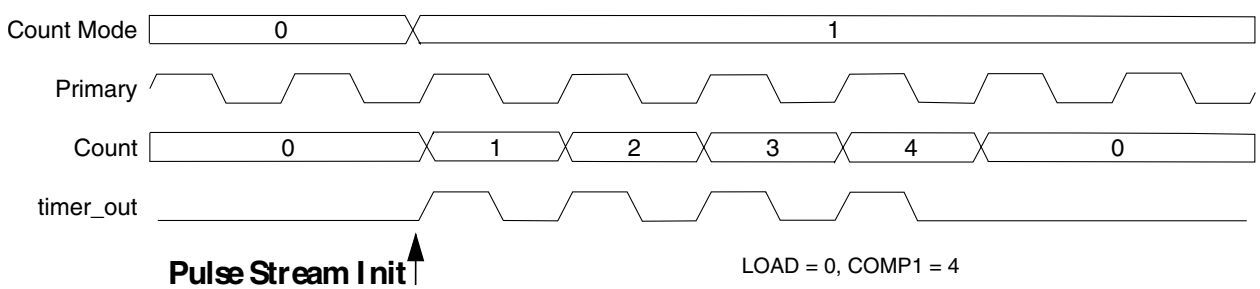


Figure 41-7. Pulse Output Mode

Example: 41.7.5.12.1 Pulse Outputs Using Two Counters

```
//     (See Processor Expert PulseStream bean.)
// This example generates six 10ms pulses, from QT1 output.
// Assuming the chip is operating at 60 MHz.
//
// To do this, timer 3 is used to generate a clock with a period of 10ms.
//
// Timer 1 is used to gate these clocks and count the number of pulses that have
//  been generated.
//
void PulseStream_Init(void)
{
// Select IP_bus_clk/16 as the clock source for Timer 3
  /* TMR3_CTRL: CM=0,PCS=0x0C,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=3 */
   setReg(TMR3_CTRL,0x1823);             /* Set up mode */
  /* TMR3_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
```

```
                Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
   setReg(TMR3_SCTRL,0x00);
   setReg(TMR3_LOAD,0x00);              /* Reset load register */
   setReg(TMR3_COMP1,37500);            /* (16 * 37500 ) / 60e6 = 0.01 sec */
 /* TMR3_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR3_CSCTRL,0x00);             /* Set up comparator control register */



// Timer 3 output is the clock source for this timer.
  /* TMR1_CTRL: CM=0,PCS=7,SCS=0,ONCE=1,LENGTH=1,DIR=0,COINIT=0,OUTMODE=7 */
  setReg(TMR1_CTRL,0x0E67);             /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=1 */
   setReg(TMR1_SCTRL,0x01);
   setReg(TMR1_CNTR,0x00);              /* Reset counter register */
   setReg(TMR1_LOAD,0x00);              /* Reset load register */
   setReg(TMR1_COMP1,0x04);              /* Set up compare 1 register */

// set to interrupt after the last pulse
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=1,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x40);             /* Set up comparator control register */
// Finally, start the counters running
  setReg(TMR3_CNTR,0);                  /* Reset counter */
  setRegBitGroup(TMR3_CTRL,CM,0x01);    /* Run source clock counter */
  setRegBitGroup(TMR1_CTRL,CM,0x01);    /* Run counter */
}
```

41.7.5.13 Fixed-Frequency PWM Mode

If CTRL[CM]=001, count through roll-over (CTRL[LENGTH]=0), continuous count (CTRL[ONCE]=0) and CTRL[OUTMODE] is '110' (set on compare, cleared on counter roll-over), then the counter output yields a pulse-width modulated (PWM) signal with a frequency equal to the count clock frequency divided by 65,536 and a pulse-width duty cycle equal to the compare value divided by 65,536. This mode of operation is often used to drive PWM amplifiers used to power motors and inverters.

Example: 41.7.5.13.1 Fixed-Frequency PWM Mode Example

```
//    (See Processor Expert PWM bean.)
// This example uses TMR0 for Fixed-Frequency PWM mode timing.
//
// The timer will count IP_bus clocks continuously until it rolls over.
// This results in a PWM period of 65536 / 60e6 = 1092.267 usec
//
// Initially, an output pulse width of 25 usec is generated ( 1500 / 60e6 )
//   giving a PWM ratio of 1500 / 65536 = 2.289%
// This pulse width can be changed by changing the COMP1 register value (using CMPLD1).
//
void PWM1_Init(void)
{
  setReg(TMR0_CNTR,0);                  /* Reset counter */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
```

```
                Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=1,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,0x05);              /* Enable output */
  setReg(TMR0_COMP1,1500);          /* Store initial value to the duty-compare register */
  /* TMR0_CTRL: CM=1,PCS=8,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=6 */
  setReg(TMR0_CTRL,0x3006);       /* Run counter */
}
```

41.7.5.14 Variable-Frequency PWM Mode

If CTRL[CM]=001, count until compare (CTRL[LENGTH]=1), continuous count (CTRL[ONCE] = 0) and CTRL[OUTMODE] is '100' (toggle OFLAG and alternate compare registers), then the counter output yields a pulse-width modulated (PWM) signal whose frequency and pulse width is determined by the values programmed into the COMP1 and COMP2 registers, and the input clock frequency. This method of PWM generation has the advantage of allowing almost any desired PWM frequency and/or constant on or off periods. This mode of operation is often used to drive PWM amplifiers used to power motors and inverters. The CMPLD1 and CMPLD2 registers are especially useful for this mode, as they allow the programmer time to calculate values for the next PWM cycle while the PWM current cycle is underway.

To set up the TMR to run in variable frequency PWM mode with compare preload please use the following setup for the specific counter you would like to use. When performing the setup, update the TMR_CTRL register last because the counter will start counting if the count mode is changed to any value other than 000 (assuming the primary count source is already active).

Timer Control Register (CTRL)

- CM=001 (count rising edges of primary source)
- PCS=1000 (IP bus clock for best granularity for waveform timing)
- SCS=Any (ignored in this mode)
- ONCE=0 (want to count repeatedly)
- LENGTH=1 (want to count until compare value is reached and re-initialize counter register)
- DIR=Any (user's choice. The compare register values must be chosen carefully to account for things like roll-under, etc.)
- COINIT=0 (user can set this if they need this function)
- OUTMODE=100 (toggle OFLAG output using alternating compare registers)

Timer Status and Control Register (SCTRL)

- OEN = 1 (output enable to allow OFLAG output to be put on an external pin. Set this bit as needed.)
- OPS = Any (user's choice)
- Make sure the rest of the bits are cleared for this register. We will enable interrupts in the comparator status and control register instead of in this register.

Comparator Status and Control Register (CSCTRL)

- TCF2EN=1 (allow interrupt to be issued when CSCTRL[TCF2] is set)
- TCF1EN=0 (do not allow interrupt to be issued when CSCTRL[TCF1] is set)
- TCF1=0 (clear timer compare 1 interrupt source flag. This is set when counter register equals compare register 1 value and OFLAG is low)
- TCF2=0 (clear timer compare 2 interrupt source flag. This is set when counter register equals compare register 2 value and OFLAG is high)
- CL1=10 (load compare register when CSCTRL[TCF2] is asserted)
- CL2=01 (load compare register when CSCTRL[TCF1] is asserted)

Interrupt Service Routines

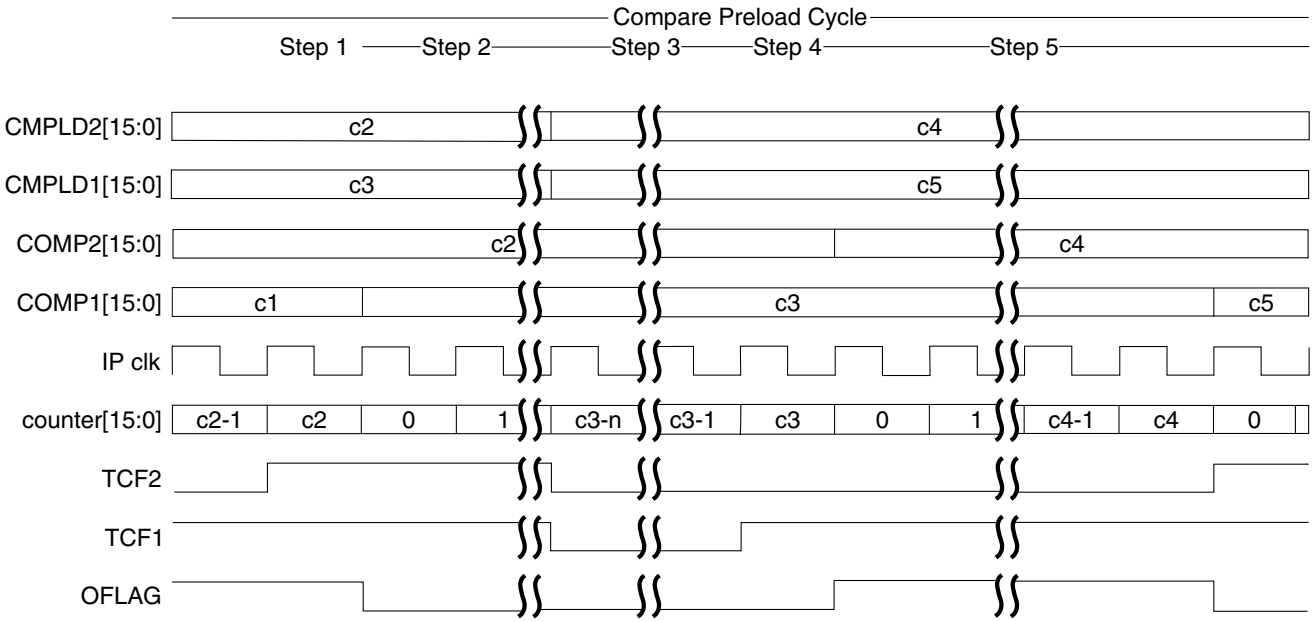
To service the CSCTRL[TCF2] interrupts generated by the timer, the interrupt controller must be configured to enable the interrupts for the particular timer being used. Additionally the user will need to write an interrupt service routine to do at a minimum the following:

- Clear CSCTRL[TCF2] and CSCTRL[TCF1] flags.
- Calculate and write new values for both CMPLD1 and CMPLD2.

Timing

This figure contains the timing for using the compare preload feature. The compare preload cycle begins with a compare event on COMP2 causing CSCTRL[TCF2] to be asserted. COMP1 is loaded with the value in the CMPLD1 (c3) one IP bus clock later. In addition an interrupt is asserted by the timer and the interrupt service routine is executed during which both comparator load registers are updated with new values (c4 and c5). When CSCTRL[TCF1] is asserted, COMP2 is loaded with the value in CMPLD2 (c4). And on the subsequent CSCTRL[TCF2] event, COMP1 is loaded with the value in

CMPLD1 (c5). The cycle starts over again as an interrupt is asserted and the interrupt service routine clears CSCTRL[TCF1] and CSCTRL[TCF2] and calculates new values for CMPLD1 and CMPLD2.



Step 1-- CNTR matches COMP2 value. CSCTRL[TCF2] is asserted and an interrupt request is generated.

Step 2-- One clock later, OFLAG toggles, CMPLD1 is copied to COMP1, LOAD is copied to CNTR, the counter starts counting.

Step 3-- The interrupt service routine clears CSCTRL[TCF1] and CSCTRL[TCF2] and the ISR loads CMPLD1 and CMPLD2 with the values for the next cycle. The counter continues counting until CNTR matches COMP1.

Step 4-- CSCTRL[TCF1] is asserted. One clock later, OFLAG toggles, CMPLD2 is copied to COMP2, LOAD is copied to CNTR and the counter starts counting.

Step 5--The counter continues counting until CNTR matches COMP2.

Figure 41-8. Compare Load Timing

Example: 41.7.5.14.1 Variable Frequency PWM Mode

```
//    (See Processor Expert PPG [Programmable Pulse Generator] bean.)
// This example starts with an 11 msec with a 31 msec cycle.
// Assuming the chip is operating at 60 MHz, the timer use IP_bus_clk/32 as its
//   clock source.
//
// Initial pulse period: 60e6/32 clocks/sec * 31 ms = 58125 total clocks in period
// Initial pulse width: 60e6/32 clocks/sec * 11 ms = 20625 clocks in pulse
//
//
```

```
// Once the initial values of COMP1/CMPLD1 and COMP2/CMPLD2 are set the pulse width
//   can be varied by load new values of CMPLD1 and CMPLD2 on each compare interrupt.
//     (See Usage of Compare Load Registers.)
//
void PPG1_Init(void)
{
  setReg(TMR0_LOAD,0);                /* Clear load register */
  setReg(TMR0_CNTR,0);                /* Clear counter */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=1,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,5);                /* Set Status and Control Register */
// Set compare preload operation and enable an interrupt on compare2 events.
  /* TMR0_CSCTRL: TCF2EN=1,TCF1EN=0,TCF2=0,TCF1=0,CL21=0,CL20=1,CL11=1,CL10=0 */
  setReg(TMR0_CSCTRL,0x86);          /* Set Comparator Status and Control Register */

  setReg(TMR0_COMP1,20625);            /* Set the pulse width of the off time */
  setReg(TMR0_CMPLD1,20625);          /* Set the pulse width of the off time */
  setReg(TMR0_COMP2,58125-20625);      /* Set the pulse width of the on time */
  setReg(TMR0_CMPLD2,58125-20625);    /* Set the pulse width of the on time */
  /* TMR0_CTRL: CM=1,PCS=0xD,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=4 */
  setRegBits(TMR0_CTRL,0x3A24);        /* Set variable PWM mode and run counter */
}
```

41.8 Resets

41.8.1 General

The TMR module can be reset only by the RST_B signal. This forces all registers to their reset state and clears the OFLAG signal if it is asserted. The counter will be turned off until the settings in the control register are changed.

Table 41-1. Reset Summary

Reset	Priority	Source	Characteristics
RST_B	n/a	Hardware Reset	Full System Reset

41.9 Clocks

41.9.1 General

The timer only receives the IP bus clock (system clock).

41.10 Interrupts

41.10.1 General

The TMR module can generate 20 interrupts, Five for each of the four counters/channels.

Table 41-2. Interrupt Summary

Core Interrupt	Interrupt	Description
TMR Channel 0	TMR0_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 0
	TMR0_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 0
	TMR0_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 0
	TMR0_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 0
	TMR0_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 0
TMR Channel 1	TMR1_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 1
	TMR1_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 1
	TMR1_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 1
	TMR1_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 1
	TMR1_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 1
TMR Channel 2	TMR2_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 2
	TMR2_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 2
	TMR2_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 2
	TMR2_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 2
	TMR2_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 2
TMR Channel 3	TMR3_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 3
	TMR3_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 3
	TMR3_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 3
	TMR3_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 3
	TMR3_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 3

41.10.2 Description of Interrupt Operation

41.10.2.1 Timer Compare Interrupts

These interrupts are generated when a successful compare occurs between a counter and it's compare registers while SCTRL[TCFIE] is set. These interrupts are cleared by writing a zero to the appropriate SCTRL[TCF].

When a timer compare interrupt is set in TMR_SCTRL and the Compare Load registers are available, one of the following two interrupts will also be asserted.

41.10.2.1.1 Timer Compare 1 Interrupts (Available with Compare Load Feature)

These interrupts are generated when a successful compare occurs between a counter and its COMP1 register while CSCTRL[TCF1EN] is set. These interrupts are cleared by writing a zero to the appropriate CSCTRL[TCF1].

41.10.2.1.2 Timer Compare 2 Interrupts (Available with Compare Load Feature)

These interrupts are generated when a successful compare occurs between a counter and its COMP2 register while CSCTRL[TCF2EN] is set. These interrupts are cleared by writing a zero to the appropriate CSCTRL[TCF2].

41.10.2.2 Timer Overflow Interrupts

These interrupts are generated when a counter rolls over its maximum value while SCTRL[TOFIE] is set. These interrupts are cleared by writing zero to the appropriate SCTRL[TOF].

41.10.2.3 Timer Input Edge Interrupts

These interrupts are generated by a transition of the input signal (either positive or negative depending on IPS setting) while SCTRL[IEFIE] is set. These interrupts are cleared by writing a zero to the appropriate SCTRL[IEF].

Chapter 42
Programmable Delay Block (PDB)

42.1 Chip-specific PDB information

42.1.1 Overview

This device contains one Programmable Delay Block (PDB). It provides controllable delays from either an internal or an external trigger, or a programmable interval tick, to the hardware trigger inputs of ADC.

42.1.2 Instantiation Information

This device contains one PDB modules.

NOTE

Before configuring PDB as the ADC hardware trigger source, user should set SIM_MISC_CTL[PDBADCTRG] bit to 1.

The PDB module clock can be switch from bus clock to MCGPLLCLK by setting SIM_CTRL_REG[PDBCLKSRC] bit. When MCGPLLCLK is used, the PDB register access will be limited. It is recommended that user finish the PDB registers configuration before PDB clock being switched to MCGPLLCLK.

Table 42-1. PDB output triggers

Number of PDB channels for ADC trigger	1
Number of per-triggers per PDB channel	4
Number of DAC triggers	0
Number of PulseOut	1

Table 42-2. PDB Input Trigger Options

PDB Trigger	PDB Input
0000	External Trigger (PDB0_EXTRG)
0001	CMP 0
0010	CMP 1
0011	CMP 2
0100	PIT0 Ch 0 Output
0101	PIT0 Ch 1 Output
0110	PIT1 Ch 0 Output
0111	PIT1 Ch 1 Output
1000	—
1001	—
1010	—
1011	LPTMR Output
1100	RTC Alarm
1101	RTC Clock Out
1110	XBAR Out 39
1111	Software Trigger

42.1.3 PDB Module Interconnections

PDB trigger outputs	Connection
Channel 0 Pre-trigger 0	XBAR IN 46
Channel 0 Pre-trigger 1	XBAR IN 47
Channel 0 Pre-trigger 2	XBAR IN 48
Channel 0 Pre-trigger 3	XBAR IN 49
Channel 0 Trigger	XBAR IN 50
Pulse-Out 0	XBAR IN 51

42.1.4 Back-to-back acknowledgement connections

Back-to-back operation enables the ADC conversions complete to trigger the next PDB channel pre-trigger and trigger output.

In this MCU, PDB back-to-back operation acknowledgment connections are implemented as follows:

- PDB channel 0 trigger/pre-trigger 0 acknowledgement input: XBAR_OUT35

- PDB channel 0 trigger/pre-trigger 1 acknowledgement input: XBAR_OUT36
- PDB channel 0 trigger/pre-trigger 2 acknowledgement input: XBAR_OUT37
- PDB channel 0 trigger/pre-trigger 3 acknowledgement input: XBAR_OUT38

42.2 Introduction

The Programmable Delay Block (PDB) provides controllable delays from either an internal or an external trigger, or a programmable interval tick, to the hardware trigger inputs of ADCs and/or generates the interval triggers to DACs, so that the precise timing between ADC conversions and/or DAC updates can be achieved. The PDB can optionally provide pulse outputs (Pulse-Out's) that are used as the sample window in the CMP block.

42.2.1 Features

- Up to 15 trigger input sources and one software trigger source
- Up to 8 configurable PDB channels for ADC hardware trigger
  - One PDB channel is associated with one ADC
  - One trigger output for ADC hardware trigger and up to 8 pre-trigger outputs for ADC trigger select per PDB channel
  - Trigger outputs can be enabled or disabled independently
  - One 16-bit delay register per pre-trigger output
  - Optional bypass of the delay registers of the pre-trigger outputs
  - Operation in One-Shot or Continuous modes
  - Optional back-to-back mode operation, which enables the ADC conversions complete to trigger the next PDB channel
  - One programmable delay interrupt
  - One sequence error interrupt
  - One channel flag and one sequence error flag per pre-trigger
  - DMA support
- Up to 8 DAC interval triggers
  - One interval trigger output per DAC

- One 16-bit delay interval register per DAC trigger output
- Optional bypass of the delay interval trigger registers
- Optional external triggers

- Up to 8 pulse outputs (pulse-out's)
  - Pulse-out's can be enabled or disabled independently
  - Programmable pulse width

NOTE

The number of PDB input and output triggers are chip-specific. See the chip-specific PDB information for details.

42.2.2 Implementation

In this section, the following letters refer to the number of output triggers:

- N—Total available number of PDB channels.
- n—PDB channel number, valid from 0 to *N*-1.
- M—Total available pre-trigger per PDB channel.
- m—Pre-trigger number, valid from 0 to *M*-1.
- X—Total number of DAC interval triggers.
- x—DAC interval trigger output number, valid from 0 to *X*-1.
- Y—Total number of Pulse-Out's.
- y—Pulse-Out number, valid value is from 0 to *Y*-1.

NOTE

The number of module output triggers to core is chip-specific. For module to core output triggers implementation, see the chip configuration information.

42.2.3 Back-to-back acknowledgment connections

PDB back-to-back operation acknowledgment connections are chip-specific. For implementation, see the chip configuration information.

42.2.4 DAC External Trigger Input Connections

The implementation of DAC external trigger inputs is chip-specific. See the chip configuration information for details.

42.2.5 Block diagram

This diagram illustrates the major components of the PDB.

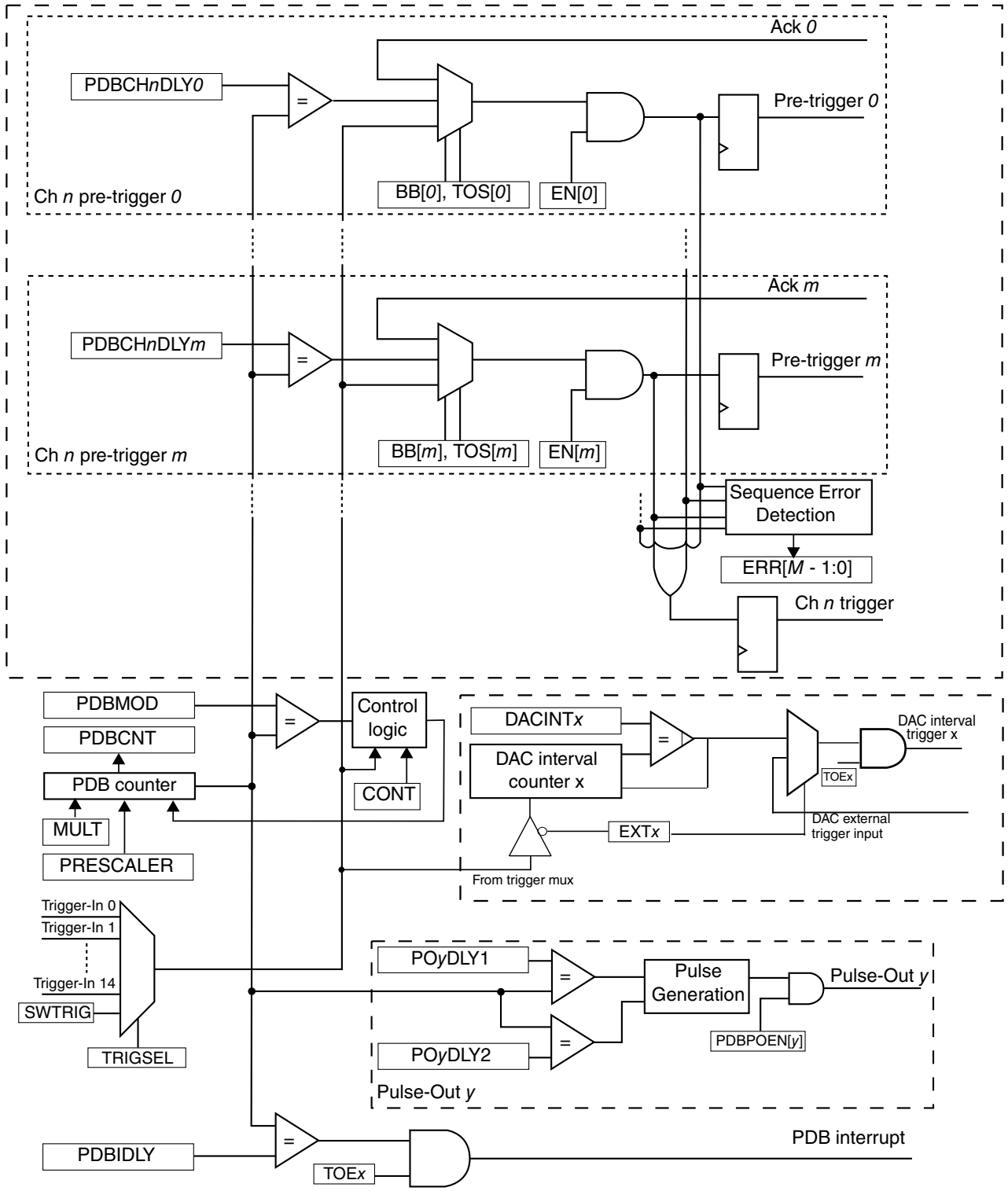


Figure 42-1. PDB block diagram

In this diagram, only one PDB channel *n*, one DAC interval trigger *x*, and one Pulse-Out *y* are shown. The PDB-enabled control logic and the sequence error interrupt logic are not shown.

42.2.6 Modes of operation

PDB ADC trigger operates in the following modes:

- Disabled—Counter is off, all pre-trigger and trigger outputs are low if PDB is not in back-to-back operation of Bypass mode.
- Debug—Counter is paused when processor is in Debug mode, and the counter for the DAC trigger is also paused in Debug mode.
- Enabled One-Shot—Counter is enabled and restarted at count zero upon receiving a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1. In each PDB channel, an enabled pre-trigger asserts once per trigger input event. The trigger output asserts whenever any of the pre-triggers is asserted.
- Enabled Continuous—Counter is enabled and restarted at count zero. The counter is rolled over to zero again when the count reaches the value specified in the modulus register, and the counting is restarted. This enables a continuous stream of pre-triggers/trigger outputs as a result of a single trigger input event.
- Enabled Bypassed—The pre-trigger and trigger outputs assert immediately after a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1, that is the delay registers are bypassed. It is possible to bypass any one or more of the delay registers; therefore, this mode can be used in conjunction with One-Shot or Continuous mode.

42.3 PDB signal descriptions

This table shows the detailed description of the external signal.

Table 42-3. PDB signal descriptions

Signal	Description	I/O
EXTRG	External Trigger Input Source If the PDB is enabled and external trigger input source is selected, a positive edge on the EXTRG signal resets and starts the counter.	I

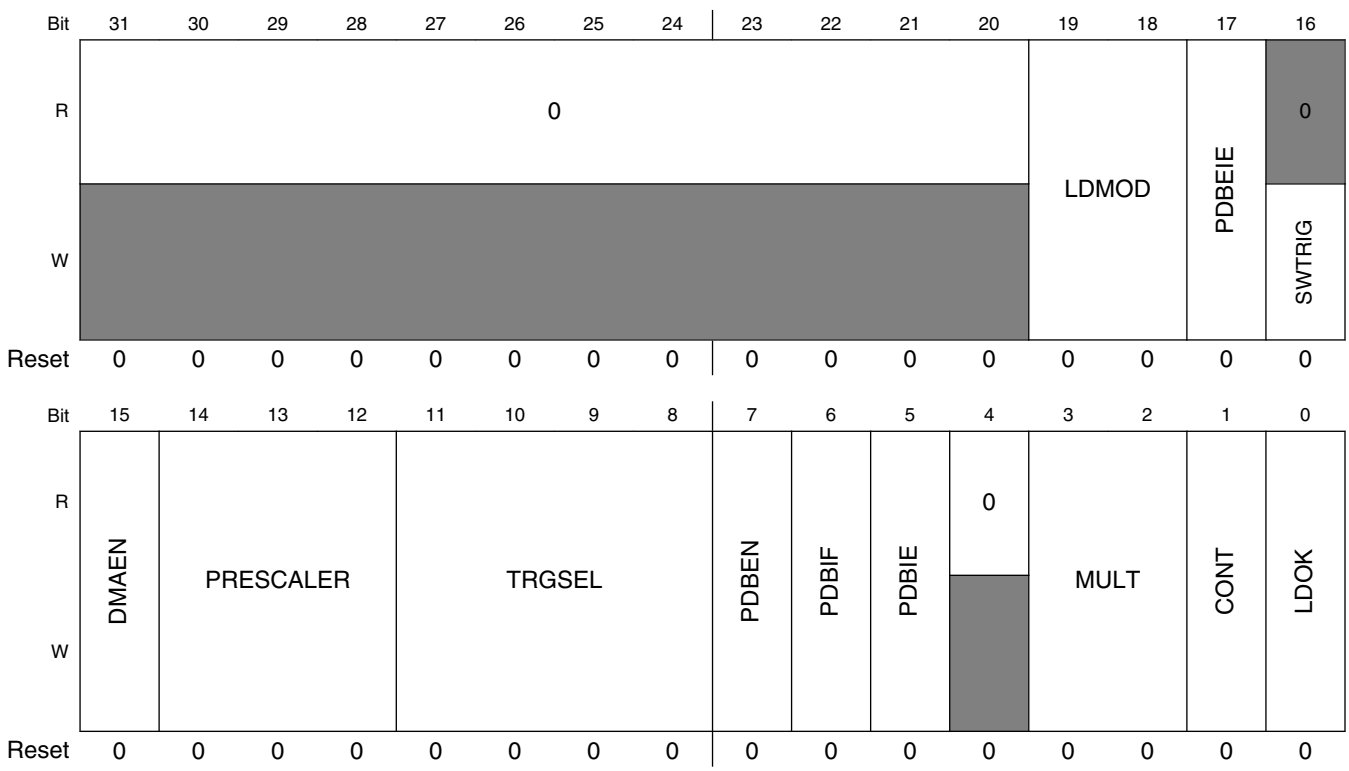
42.4 Memory map and register definition

PDB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_6000	Status and Control register (PDB0_SC)	32	R/W	0000_0000h	42.4.1/828
4003_6004	Modulus register (PDB0_MOD)	32	R/W	0000_FFFFh	42.4.2/831
4003_6008	Counter register (PDB0_CNT)	32	R	0000_0000h	42.4.3/831
4003_600C	Interrupt Delay register (PDB0_IDLY)	32	R/W	0000_FFFFh	42.4.4/832
4003_6010	Channel n Control register 1 (PDB0_CH0C1)	32	R/W	0000_0000h	42.4.5/832
4003_6014	Channel n Status register (PDB0_CH0S)	32	R/W	0000_0000h	42.4.6/833
4003_6018	Channel n Delay 0 register (PDB0_CH0DLY0)	32	R/W	0000_0000h	42.4.7/834
4003_601C	Channel n Delay 1 register (PDB0_CH0DLY1)	32	R/W	0000_0000h	42.4.8/834
4003_6020	Channel n Delay 2 register (PDB0_CH0DLY2)	32	R/W	0000_0000h	42.4.9/835
4003_6024	Channel n Delay 3 register (PDB0_CH0DLY3)	32	R/W	0000_0000h	42.4.10/ 835
4003_6190	Pulse-Out n Enable register (PDB0_POEN)	32	R/W	0000_0000h	42.4.11/ 836
4003_6194	Pulse-Out n Delay register (PDB0_PO0DLY)	32	R/W	0000_0000h	42.4.12/ 836

42.4.1 Status and Control register (PDBx_SC)

Address: 4003_6000h base + 0h offset = 4003_6000h



PDBx_SC field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 LDMOD	Load Mode Select Selects the mode to load the MOD, IDLY, CH*n*DLY*m*, INT*x*, and PO*y*DLY registers, after 1 is written to LDOK. 00 The internal registers are loaded with the values from their buffers immediately after 1 is written to LDOK. 01 The internal registers are loaded with the values from their buffers when the PDB counter reaches the MOD register value after 1 is written to LDOK. 10 The internal registers are loaded with the values from their buffers when a trigger input event is detected after 1 is written to LDOK. 11 The internal registers are loaded with the values from their buffers when either the PDB counter reaches the MOD register value or a trigger input event is detected, after 1 is written to LDOK.
17 PDBEIE	PDB Sequence Error Interrupt Enable Enables the PDB sequence error interrupt. When this field is set, any of the PDB channel sequence error flags generates a PDB sequence error interrupt. 0 PDB sequence error interrupt disabled. 1 PDB sequence error interrupt enabled.
16 SWTRIG	Software Trigger When PDB is enabled and the software trigger is selected as the trigger input source, writing 1 to this field resets and restarts the counter. Writing 0 to this field has no effect. Reading this field results 0.
15 DMAEN	DMA Enable When DMA is enabled, the PDBIF flag generates a DMA request instead of an interrupt. 0 DMA disabled. 1 DMA enabled.
14–12 PRESCALER	Prescaler Divider Select 000 Counting uses the peripheral clock divided by multiplication factor selected by MULT. 001 Counting uses the peripheral clock divided by twice of the multiplication factor selected by MULT. 010 Counting uses the peripheral clock divided by four times of the multiplication factor selected by MULT. 011 Counting uses the peripheral clock divided by eight times of the multiplication factor selected by MULT. 100 Counting uses the peripheral clock divided by 16 times of the multiplication factor selected by MULT. 101 Counting uses the peripheral clock divided by 32 times of the multiplication factor selected by MULT. 110 Counting uses the peripheral clock divided by 64 times of the multiplication factor selected by MULT. 111 Counting uses the peripheral clock divided by 128 times of the multiplication factor selected by MULT.
11–8 TRGSEL	Trigger Input Source Select

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	Selects the trigger input source for the PDB. The trigger input source can be internal or external (EXTRG pin), or the software trigger. Refer to chip configuration details for the actual PDB input trigger connections. 0000 Trigger-In 0 is selected. 0001 Trigger-In 1 is selected. 0010 Trigger-In 2 is selected. 0011 Trigger-In 3 is selected. 0100 Trigger-In 4 is selected. 0101 Trigger-In 5 is selected. 0110 Trigger-In 6 is selected. 0111 Trigger-In 7 is selected. 1000 Trigger-In 8 is selected. 1001 Trigger-In 9 is selected. 1010 Trigger-In 10 is selected. 1011 Trigger-In 11 is selected. 1100 Trigger-In 12 is selected. 1101 Trigger-In 13 is selected. 1110 Trigger-In 14 is selected. 1111 Software trigger is selected.
7 PDBEN	PDB Enable 0 PDB disabled. Counter is off. 1 PDB enabled.
6 PDBIF	PDB Interrupt Flag This field is set when the counter value is equal to the IDLY register. Writing zero clears this field.
5 PDBIE	PDB Interrupt Enable Enables the PDB interrupt. When this field is set and DMAEN is cleared, PDBIF generates a PDB interrupt. 0 PDB interrupt disabled. 1 PDB interrupt enabled.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 MULT	Multiplication Factor Select for Prescaler Selects the multiplication factor of the prescaler divider for the counter clock. 00 Multiplication factor is 1. 01 Multiplication factor is 10. 10 Multiplication factor is 20. 11 Multiplication factor is 40.
1 CONT	Continuous Mode Enable Enables the PDB operation in Continuous mode. 0 PDB operation in One-Shot mode 1 PDB operation in Continuous mode

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
0 LDOK	Load OK Writing 1 to this bit updates the internal registers of MOD, IDLY, CHnDLYm, DACINTx,and POyDLY with the values written to their buffers. The MOD, IDLY, CHnDLYm, DACINTx, and POyDLY will take effect according to the LDMOD. After 1 is written to the LDOK field, the values in the buffers of above registers are not effective and the buffers cannot be written until the values in buffers are loaded into their internal registers. LDOK can be written only when PDBEN is set or it can be written at the same time with PDBEN being written to 1. It is automatically cleared when the values in buffers are loaded into the internal registers or the PDBEN is cleared. Writing 0 to it has no effect.

42.4.2 Modulus register (PDBx_MOD)

Address: 4003_6000h base + 4h offset = 4003_6004h

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

PDBx_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MOD	PDB Modulus Specifies the period of the counter. When the counter reaches this value, it will be reset back to zero. If the PDB is in Continuous mode, the count begins anew. Reading this field returns the value of the internal register that is effective for the current cycle of PDB.

42.4.3 Counter register (PDBx_CNT)

Address: 4003_6000h base + 8h offset = 4003_6008h

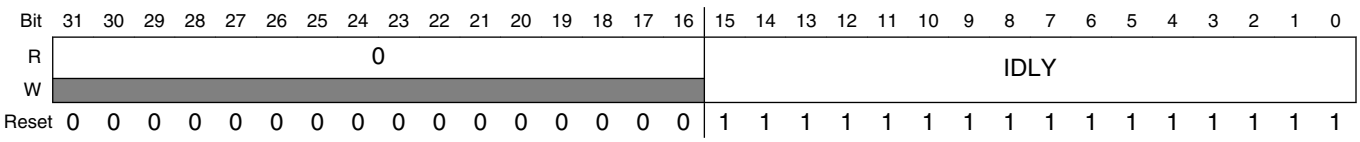
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDBx_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CNT	PDB Counter Contains the current value of the counter.

42.4.4 Interrupt Delay register (PDBx_IDLY)

Address: 4003_6000h base + Ch offset = 4003_600Ch



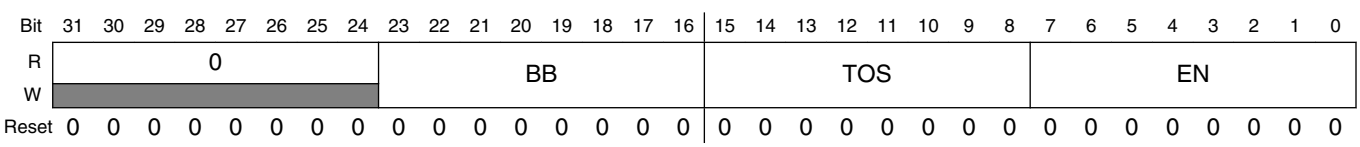
PDBx_IDLY field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IDLY	PDB Interrupt Delay Specifies the delay value to schedule the PDB interrupt. It can be used to schedule an independent interrupt at some point in the PDB cycle. If enabled, a PDB interrupt is generated, when the counter is equal to the IDLY. Reading this field returns the value of internal register that is effective for the current cycle of the PDB.

42.4.5 Channel n Control register 1 (PDBx_CH*n*C1)

Each PDB channel has one control register, CHnC1. The fields in this register control the functionality of each PDB channel operation.

Address: 4003_6000h base + 10h offset + (40d × i), where i=0d to 0d



PDBx_CH*n*C1 field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PDBx_CHnC1 field descriptions (continued)

Field	Description
23–16 BB	PDB Channel Pre-Trigger Back-to-Back Operation Enable Enables the PDB ADC pre-trigger operation as back-to-back mode. Only lower M pre-trigger bits are implemented in this MCU. Back-to-back operation enables the ADC conversions complete to trigger the next PDB channel pre-trigger and trigger output, so that the ADC conversions can be triggered on the next set of configuration and results registers. Application code must enable only the back-to-back operation of the PDB pre-triggers at the leading of the back-to-back connection chain. 0 PDB channel's corresponding pre-trigger back-to-back operation disabled. 1 PDB channel's corresponding pre-trigger back-to-back operation enabled.
15–8 TOS	PDB Channel Pre-Trigger Output Select These bits select the PDB ADC pre-trigger outputs. Only lower M pre-trigger bits are implemented in this MCU. 0 PDB channel's corresponding pre-trigger is in bypassed mode. The pre-trigger asserts one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 PDB channel's corresponding pre-trigger asserts when the counter reaches the channel delay register plus one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SETRIG is written with 1.
EN	PDB Channel Pre-Trigger Enable Enables the PDB ADC pre-trigger outputs. Only lower M pre-trigger fields are implemented in this MCU. 0 PDB channel's corresponding pre-trigger disabled. 1 PDB channel's corresponding pre-trigger enabled.

42.4.6 Channel n Status register (PDBx_CHnS)

Address: 4003_6000h base + 14h offset + (40d × i), where i=0d to 0d

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

PDBx_CHnS field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 CF	PDB Channel Flags The CF[*m*] field is set when the PDB counter matches the CH*n*DLY*m*. Write 0 to clear these bits.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ERR	PDB Channel Sequence Error Flags

Table continues on the next page...

PDBx_CHnS field descriptions (continued)

Field	Description
	Only the lower M bits are implemented in this MCU. 0 Sequence error not detected on PDB channel's corresponding pre-trigger. 1 Sequence error detected on PDB channel's corresponding pre-trigger. ADCn block can be triggered for a conversion by one pre-trigger from PDB channel *n*. When one conversion, which is triggered by one of the pre-triggers from PDB channel *n*, is in progress, new trigger from PDB channel's corresponding pre-trigger m cannot be accepted by ADCn, and ERR[m] is set. Writing 0's to clear the sequence error flags.

42.4.7 Channel n Delay 0 register (PDBx_CHnDLY0)

Address: 4003_6000h base + 18h offset + (40d × i), where i=0d to 0d

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

PDBx_CHnDLY0 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay Specifies the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading this field returns the value of internal register that is effective for the current PDB cycle.

42.4.8 Channel n Delay 1 register (PDBx_CHnDLY1)

Address: 4003_6000h base + 1Ch offset + (40d × i), where i=0d to 0d

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

PDBx_CHnDLY1 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay

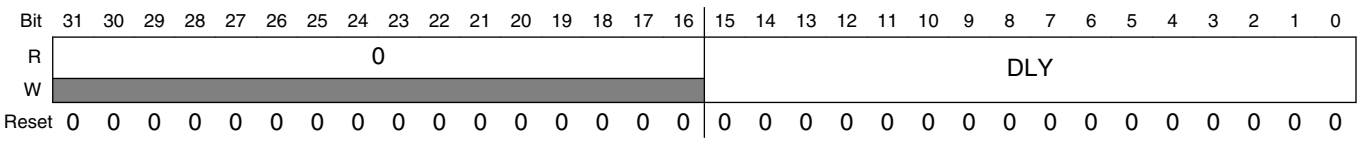
Table continues on the next page...

PDBx_CHnDLY1 field descriptions (continued)

Field	Description
	These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

42.4.9 Channel n Delay 2 register (PDBx_CHnDLY2)

Address: 4003_6000h base + 20h offset + (40d × i), where i=0d to 0d

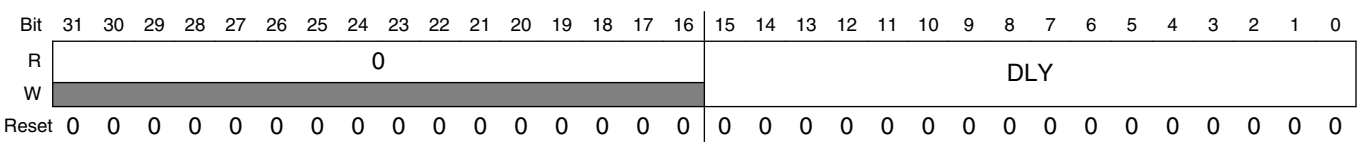


PDBx_CHnDLY2 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

42.4.10 Channel n Delay 3 register (PDBx_CHnDLY3)

Address: 4003_6000h base + 24h offset + (40d × i), where i=0d to 0d



PDBx_CHnDLY3 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

42.4.11 Pulse-Out n Enable register (PDB*x*_POEN)

Address: 4003_6000h base + 190h offset = 4003_6190h

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

PDB*x*_POEN field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
POEN	PDB Pulse-Out Enable Enables the pulse output. Only lower Y bits are implemented in this MCU. 0 PDB Pulse-Out disabled 1 PDB Pulse-Out enabled

42.4.12 Pulse-Out n Delay register (PDB*x*_PO*n*DLY)

Address: 4003_6000h base + 194h offset + (4d × i), where i=0d to 0d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY1																DLY2															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDB*x*_PO*n*DLY field descriptions

Field	Description
31–16 DLY1	PDB Pulse-Out Delay 1 These bits specify the delay 1 value for the PDB Pulse-Out. Pulse-Out goes high when the PDB counter is equal to the DLY1. Reading these bits returns the value of internal register that is effective for the current PDB cycle.
DLY2	PDB Pulse-Out Delay 2 These bits specify the delay 2 value for the PDB Pulse-Out. Pulse-Out goes low when the PDB counter is equal to the DLY2. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

42.5 Functional description

42.5.1 PDB pre-trigger and trigger outputs

The PDB contains a counter whose output is compared to several different digital values. If the PDB is enabled, then a trigger input event will reset the counter and make it start to count. A trigger input event is defined as a rising edge being detected on a selected trigger input source, or if a software trigger is selected and SC[SWTRIG] is written with 1. For each channel, a delay *m* determines the time between assertion of the trigger input event to the time at which changes in the pre-trigger *m* output signal are started. The time is defined as:

- Trigger input event to pre-trigger *m* = (prescaler × multiplication factor × delay *m*) + 2 peripheral clock cycles
- Add 1 additional peripheral clock cycle to determine the time when the channel trigger output changes.

Each channel is associated with 1 ADC block. PDB channel n pre-trigger outputs 0 to *M*; each pre-trigger output is connected to ADC hardware trigger select and hardware trigger inputs. The pre-triggers are used to precondition the ADC block before the actual trigger occurs. When the ADC receives the rising edge of the trigger, the ADC will start the conversion according to the precondition determined by the pre-triggers. The ADC contains *M* sets of configuration and result registers, allowing it to alternate conversions between *M* different analog sources (like a ping-pong game). The pre-trigger outputs are used to specify which signal will be sampled next. When a pre-trigger *m* is asserted, the ADC conversion is triggered with set *m* of the configuration and result registers.

The waveforms shown in the following diagram show the pre-trigger and trigger outputs of PDB channel *n*. The delays can be independently set using the CH*n*DLY*m* registers, and the pre-triggers can be enabled or disabled in CH*n*C1[EN[*m*]].

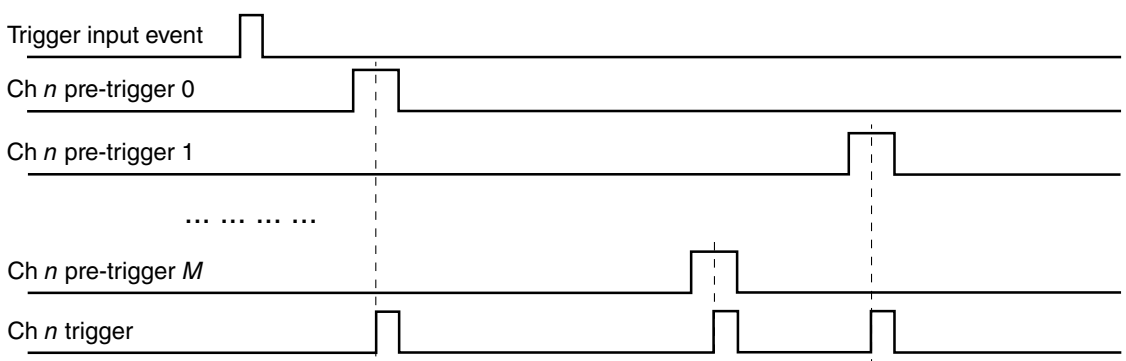


Figure 42-2. Pre-trigger and trigger outputs

The delay in CH*n*DLY*m* register can be optionally bypassed, if CH*n*C1[TOS[*m*]] is cleared. In this case, when the trigger input event occurs, the pre-trigger *m* is asserted after 2 peripheral clock cycles.

The PDB can be configured for back-to-back operation. Back-to-back operation enables the ADC conversion completions to trigger the next PDB channel pre-trigger and trigger outputs, so that the ADC conversions can be triggered on the next set of configuration and results registers. When back-to-back operation is enabled by setting CH*n*C1[BB[*m*]], then the delay *m* is ignored and the pre-trigger *m* is asserted 2 peripheral cycles after the acknowledgment *m* is received. The acknowledgment connections in this MCU are described in Back-to-back acknowledgment connections.

When a pre-trigger from a PDB channel *n* is asserted, the associated lock of the pre-trigger becomes active. The associated lock is released by the rising edge of the corresponding ADC*n*SC1[COCO]; the ADC*n*SC1[COCO] should be cleared after the conversion result is read, so that the next rising edge of ADC*n*SC1[COCO] can be generated to clear the lock later. The lock becomes inactive when:

- the rising edge of corresponding ADC*n*SC1[COCO] occurs,
- or the corresponding PDB pre-trigger is disabled,
- or the PDB is disabled

The channel *n* trigger output is suppressed when any of the locks of the pre-triggers in channel *n* is active. If a new pre-trigger *m* asserts when there is active lock in the PDB channel *n*, then a register flag bit CH*n*S[ERR[*m*]] (associated with the pre-trigger *m*) is set. If SC[PDBEIE] is set, then the sequence error interrupt is generated. A sequence error typically happens because the delay *m* is set too short and the pre-trigger *m* asserts before the previously triggered ADC conversion finishes.

When the PDB counter reaches the value set in IDLY register, the SC[PDBIF] flag is set. A PDB interrupt can be generated if SC[PDBIE] is set and SC[DMAEN] is cleared. If SC[DMAEN] is set, then the PDB requests a DMA transfer when the SC[PDBIF] flag is set.

The modulus value in the MOD register is used to reset the counter back to zero at the end of the count. If SC[CONT] is set, then the counter will then resume a new count; otherwise, the counter operation will stop until the next trigger input event occurs.

42.5.2 PDB trigger input source selection

The PDB has up to 15 trigger input sources, namely Trigger-In 0 to Trigger-In 14. They are connected to on-chip or off-chip event sources. The PDB can be triggered by software through SC[SWTRIG]. SC[TRIGSEL] selects the active trigger input source or software trigger.

For the trigger input sources implemented in this MCU, see chip configuration information.

42.5.3 DAC interval trigger outputs

PDB can generate the interval triggers for DACs to update their outputs periodically. DAC interval counter *x* is reset and started when a trigger input event occurs if DACINTC*x*[EXT] is cleared. When the interval counter *x* is equal to the value set in DACINT*x* register, the DAC interval trigger *x* output generates a pulse of one peripheral clock cycle width to update the DAC*x*. If DACINTC*x*[EXT] is set, the DAC interval counter is bypassed and the interval trigger output *x* generates a pulse following the detection of a rising edge on the DAC external trigger input. The counter and interval trigger can be disabled by clearing the DACINTC*x*[TOE].

DAC interval counters are also reset when the PDB counter reaches the MOD register value; therefore, when the PDB counter rolls over to zero, the DAC interval counters starts anew.

The DAC interval trigger pulse and the ADC pre-trigger/trigger pulses together allow precise timing of DAC updates and ADC measurements. This is outlined in the typical use case described in the following diagram.

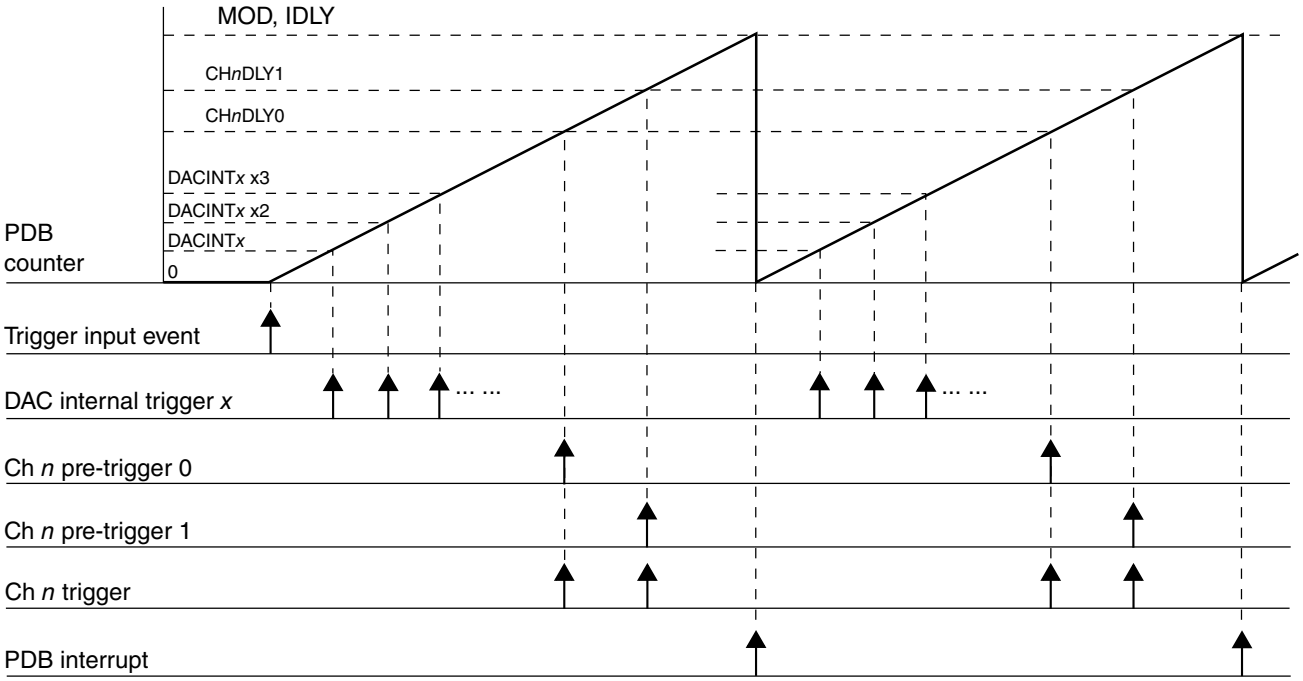


Figure 42-3. PDB ADC triggers and DAC interval triggers use case

NOTE

Because the DAC interval counters share the prescaler with PDB counter, PDB must be enabled if the DAC interval trigger outputs are used in the applications.

42.5.4 Pulse-Out's

PDB can generate pulse outputs of configurable width. When PDB counter reaches the value set in PO*y*DLY[DLY1], the Pulse-Out goes high; when the counter reaches PO*y*DLY[DLY2], it goes low. PO*y*DLY[DLY2] can be set either greater or less than PO*y*DLY[DLY1].

ADC pre-trigger/trigger outputs and Pulse-Out generation have the same time base, because they both share the PDB counter.

The pulse-out connections implemented in this MCU are described in the device's chip configuration details.

42.5.5 Updating the delay registers

The following registers control the timing of the PDB operation; and in some of the applications, they may need to become effective at the same time.

- PDB Modulus register (MOD)
- PDB Interrupt Delay register (IDLY)
- PDB Channel *n* Delay *m* register (CH*n*DLY*m*)
- DAC Interval *x* register (DACINT*x*)
- PDB Pulse-Out *y* Delay register (PO*y*DLY)

The internal registers of them are buffered and any values written to them are written first to their buffers. The circumstances that cause their internal registers to be updated with the values from the buffers are summarized as shown in the table below.

Table 42-4. Circumstances of update to the delay registers

SC[LDMOD]	Update to the delay registers
00	The internal registers are loaded with the values from their buffers immediately after 1 is written to SC[LDOK].
01	The PDB counter reaches the MOD register value after 1 is written to SC[LDOK].
10	A trigger input event is detected after 1 is written to SC[LDOK].
11	Either the PDB counter reaches the MOD register value, or a trigger input event is detected, after 1 is written to SC[LDOK].

After 1 is written to SC[LDOK], the buffers cannot be written until the values in buffers are loaded into their internal registers. SC[LDOK] is self-cleared when the internal registers are loaded, so the application code can read it to determine the updates to the internal registers.

The following diagrams show the cases of the internal registers being updated with SC[LDMOD] is 00 and x1.

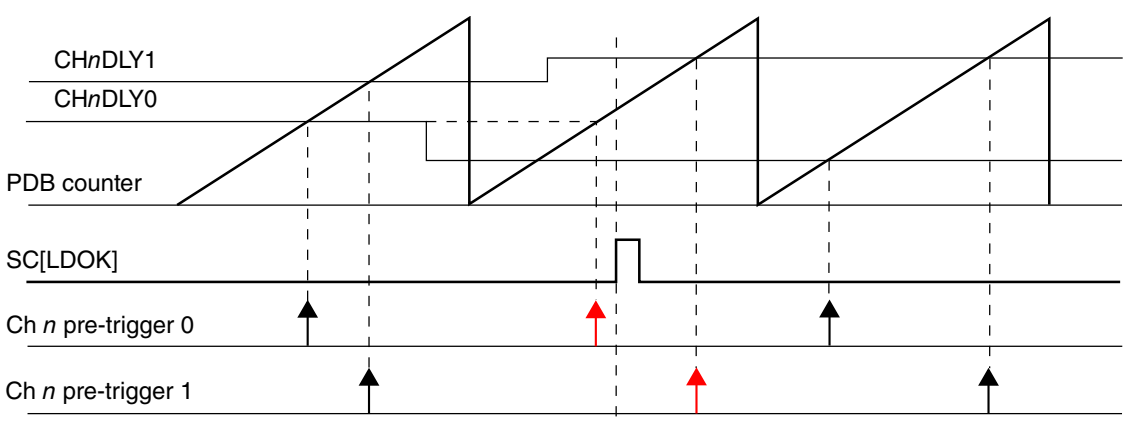


Figure 42-4. Registers update with SC[LDMOD] = 00

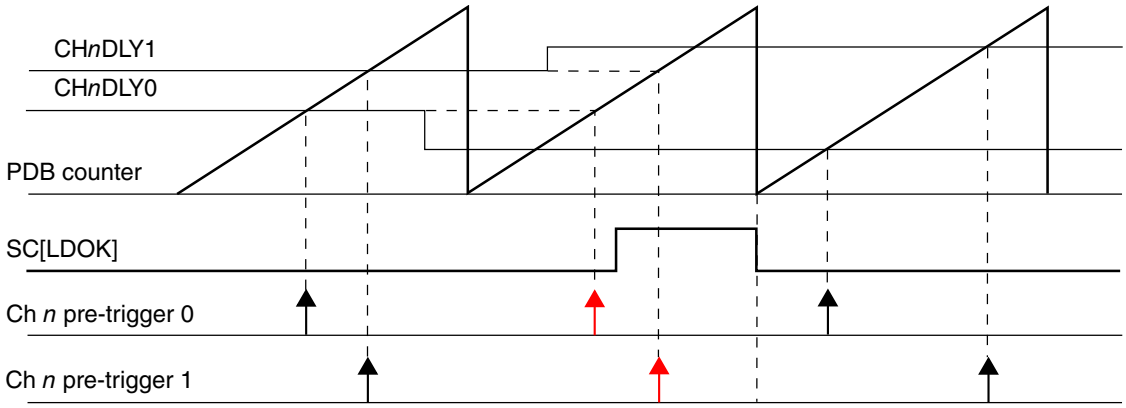


Figure 42-5. Registers update with SC[LDMOD] = x1

42.5.6 Interrupts

PDB can generate two interrupts: PDB interrupt and PDB sequence error interrupt. The following table summarizes the interrupts.

Table 42-5. PDB interrupt summary

Interrupt	Flags	Enable bit
PDB Interrupt	SC[PDBIF]	SC[PDBIE] = 1 and SC[DMAEN] = 0
PDB Sequence Error Interrupt	CH*n*S[ERR*m*]	SC[PDBEIE] = 1

42.5.7 DMA

If SC[DMAEN] is set, PDB can generate a DMA transfer request when SC[PDBIF] is set. When DMA is enabled, the PDB interrupt is not issued.

42.6 Application information

42.6.1 Impact of using the prescaler and multiplication factor on timing resolution

Use of prescaler and multiplication factor greater than 1 limits the count/delay accuracy in terms of peripheral clock cycles (to the modulus of the prescaler X multiplication factor). If the multiplication factor is set to 1 and the prescaler is set to 2 then the only

values of total peripheral clocks that can be detected are even values; if prescaler is set to 4 then the only values of total peripheral clocks that can be decoded as detected are mod(4) and so forth. If the applications need a really long delay value and use a prescaler set to 128, then the resolution would be limited to 128 peripheral clock cycles.

Therefore, use the lowest possible prescaler and multiplication factor for a given application.

Chapter 43
Inter-Integrated Circuit (I2C)

43.1 Chip-specific I2C information

43.1.1 Instantiation Information

This device contains two I2C modules: I2C0 and I2C1. These I2C modules support the SMBUS protocol along with DMA support.

I2C modules are clocked by the bus clock.

43.2 Introduction

The inter-integrated circuit (I^2C, I2C, or IIC) module provides a method of communication between a number of devices.

The interface is designed to operate up to at least 400 kbit/s with maximum bus loading and timing. The I2C device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF. The I2C module also complies with the *System Management Bus (SMBus) Specification, version 2*.

43.2.1 Features

The I2C module has the following features:

- Compatible with *The I^2C-Bus Specification*
- Multimaster operation

- Software programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- START and STOP signal generation and detection
- Repeated START signal generation and detection
- Acknowledge bit generation and detection
- Bus busy detection
- General call recognition
- 10-bit address extension
- Support for *System Management Bus (SMBus) Specification, version 2*
- Programmable input glitch filter
- Low power mode wakeup on slave address match
- Range slave address support
- DMA support
- Double buffering support to achieve higher baud rate

43.2.2 Modes of operation

The I2C module's operation in various low power modes is as follows:

- Run mode: This is the basic mode of operation. To conserve power in this mode, disable the module.
- Wait mode: The module continues to operate when the core is in Wait mode and can provide a wakeup interrupt.
- Stop mode: The module is inactive in Stop mode for reduced power consumption, except that address matching is enabled in Stop mode. The STOP instruction does not affect the I2C module's register states.

43.2.3 Block diagram

The following figure is a functional block diagram of the I2C module.

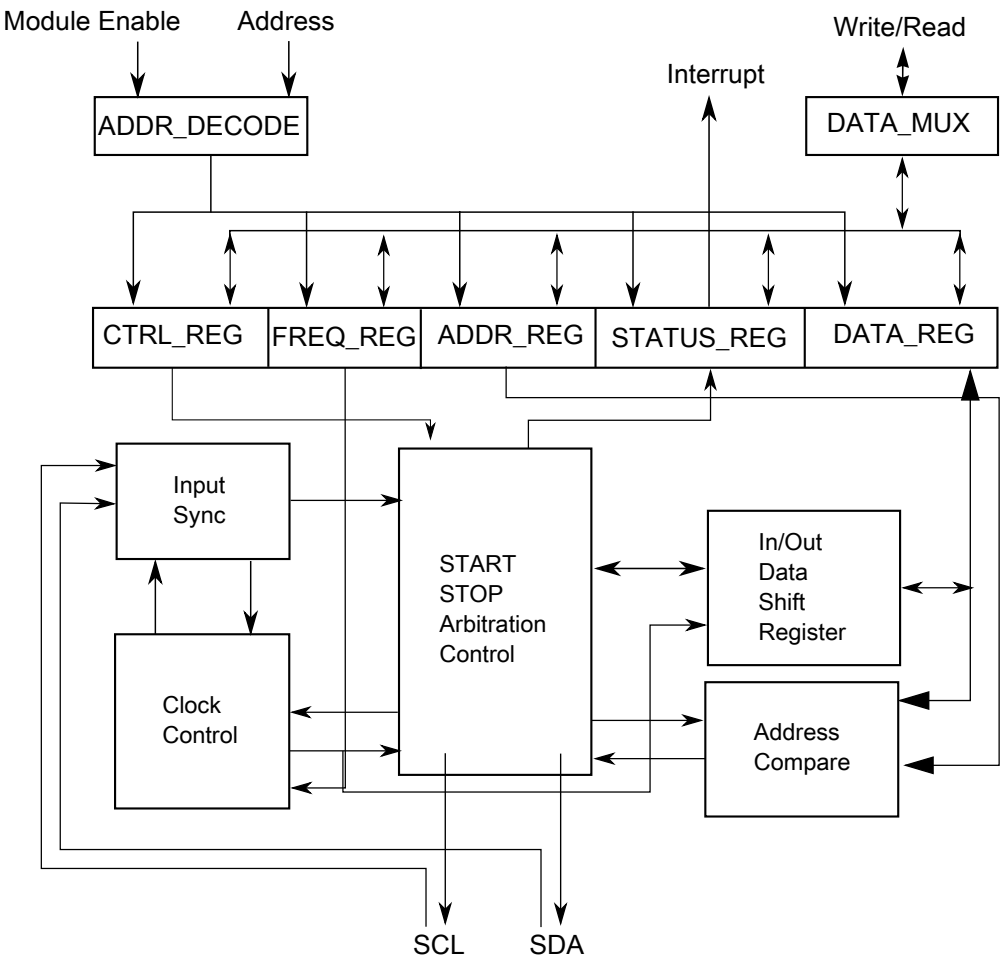


Figure 43-1. I2C Functional block diagram

43.3 I²C signal descriptions

The signal properties of I²C are shown in the table found here.

Table 43-1. I²C signal descriptions

Signal	Description	I/O
SCL	Bidirectional serial clock line of the I²C system.	I/O
SDA	Bidirectional serial data line of the I²C system.	I/O

43.4 Memory map/register definition

This section describes in detail all I2C registers accessible to the end user.

I2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_7000	I2C Address Register 1 (I2C0_A1)	8	R/W	00h	43.4.1/849
4006_7001	I2C Frequency Divider register (I2C0_F)	8	R/W	00h	43.4.2/849
4006_7002	I2C Control Register 1 (I2C0_C1)	8	R/W	00h	43.4.3/850
4006_7003	I2C Status register (I2C0_S)	8	R/W	80h	43.4.4/852
4006_7004	I2C Data I/O register (I2C0_D)	8	R/W	00h	43.4.5/854
4006_7005	I2C Control Register 2 (I2C0_C2)	8	R/W	00h	43.4.6/854
4006_7006	I2C Programmable Input Glitch Filter Register (I2C0_FLT)	8	R/W	00h	43.4.7/855
4006_7007	I2C Range Address register (I2C0_RA)	8	R/W	00h	43.4.8/857
4006_7008	I2C SMBus Control and Status register (I2C0_SMB)	8	R/W	00h	43.4.9/857
4006_7009	I2C Address Register 2 (I2C0_A2)	8	R/W	C2h	43.4.10/ 859
4006_700A	I2C SCL Low Timeout Register High (I2C0_SLTH)	8	R/W	00h	43.4.11/ 859
4006_700B	I2C SCL Low Timeout Register Low (I2C0_SLTL)	8	R/W	00h	43.4.12/ 860
4006_700C	I2C Status register 2 (I2C0_S2)	8	R/W	01h	43.4.13/ 860
4006_8000	I2C Address Register 1 (I2C1_A1)	8	R/W	00h	43.4.1/849
4006_8001	I2C Frequency Divider register (I2C1_F)	8	R/W	00h	43.4.2/849
4006_8002	I2C Control Register 1 (I2C1_C1)	8	R/W	00h	43.4.3/850
4006_8003	I2C Status register (I2C1_S)	8	R/W	80h	43.4.4/852
4006_8004	I2C Data I/O register (I2C1_D)	8	R/W	00h	43.4.5/854
4006_8005	I2C Control Register 2 (I2C1_C2)	8	R/W	00h	43.4.6/854
4006_8006	I2C Programmable Input Glitch Filter Register (I2C1_FLT)	8	R/W	00h	43.4.7/855
4006_8007	I2C Range Address register (I2C1_RA)	8	R/W	00h	43.4.8/857
4006_8008	I2C SMBus Control and Status register (I2C1_SMB)	8	R/W	00h	43.4.9/857
4006_8009	I2C Address Register 2 (I2C1_A2)	8	R/W	C2h	43.4.10/ 859
4006_800A	I2C SCL Low Timeout Register High (I2C1_SLTH)	8	R/W	00h	43.4.11/ 859
4006_800B	I2C SCL Low Timeout Register Low (I2C1_SLTL)	8	R/W	00h	43.4.12/ 860
4006_800C	I2C Status register 2 (I2C1_S2)	8	R/W	01h	43.4.13/ 860

43.4.1 I2C Address Register 1 (I2Cx_A1)

This register contains the slave address to be used by the I2C module.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read Write	AD[7:1]							0
Reset	0	0	0	0	0	0	0	0

I2Cx_A1 field descriptions

Field	Description
7–1 AD[7:1]	Address Contains the primary slave address used by the I2C module when it is addressed as a slave. This field is used in the 7-bit address scheme and the lower seven bits in the 10-bit address scheme.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.4.2 I2C Frequency Divider register (I2Cx_F)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	MULT		ICR					
Reset	0	0	0	0	0	0	0	0

I2Cx_F field descriptions

Field	Description
7–6 MULT	Multiplier Factor Defines the multiplier factor (mul). This factor is used along with the SCL divider to generate the I2C baud rate. 00 mul = 1 01 mul = 2 10 mul = 4 11 Reserved
ICR	ClockRate Prescales the I2C module clock for bit rate selection. This field and the MULT field determine the I2C baud rate, the SDA hold time, the SCL start hold time, and the SCL stop hold time. For a list of values corresponding to each ICR setting, see I2C divider and hold values. The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate. `I2C baud rate = I2C module clock speed (Hz)/(mul × SCL divider)`

Table continues on the next page...

I2Cx_F field descriptions (continued)

Field	Description

	The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data).
	
	`SDA hold time = I2C module clock period (s) × mul × SDA hold value`
	
	The SCL start hold time is the delay from the falling edge of SDA (I2C data) while SCL is high (start condition) to the falling edge of SCL (I2C clock).
	
	`SCL start hold time = I2C module clock period (s) × mul × SCL start hold value`
	
	The SCL stop hold time is the delay from the rising edge of SCL (I2C clock) to the rising edge of SDA (I2C data) while SCL is high (stop condition).
	
	`SCL stop hold time = I2C module clock period (s) × mul × SCL stop hold value`
	
	For example, if the I2C module clock speed is 8 MHz, the following table shows the possible hold time values with different ICR and MULT selections to achieve an I^2C baud rate of 100 kbit/s.

MULT	ICR	Hold times (µs)		
		SDA	SCL Start	SCL Stop
2h	00h	3.500	3.000	5.500
1h	07h	2.500	4.000	5.250
1h	0Bh	2.250	4.000	5.250
0h	14h	2.125	4.250	5.125
0h	18h	1.125	4.750	5.125

43.4.3 I2C Control Register 1 (I2Cx_C1)

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read	IICEN	IICIE	MST	TX	TXAK	0	WUEN	DMAEN
Write						RSTA		
Reset	0	0	0	0	0	0	0	0

I2Cx_C1 field descriptions

Field	Description
7 IICEN	I2C Enable Enables I2C module operation. 0 Disabled 1 Enabled
6 IICIE	I2C Interrupt Enable Enables I2C interrupt requests.

Table continues on the next page...

I2Cx_C1 field descriptions (continued)

Field	Description
	0 Disabled 1 Enabled
5 MST	Master Mode Select When MST is changed from 0 to 1, a START signal is generated on the bus and master mode is selected. When this bit changes from 1 to 0, a STOP signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	Transmit Mode Select Selects the direction of master and slave transfers. In master mode this bit must be set according to the type of transfer required. Therefore, for address cycles, this bit is always set. When addressed as a slave this bit must be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit
3 TXAK	Transmit Acknowledge Enable Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of SMB[FACK] affects NACK/ACK generation. **NOTE:** SCL is held low until TXAK is written. 0 An acknowledge signal is sent to the bus on the following receiving byte (if FACK is cleared) or the current receiving byte (if FACK is set). 1 No acknowledge signal is sent to the bus on the following receiving data byte (if FACK is cleared) or the current receiving data byte (if FACK is set).
2 RSTA	Repeat START Writing 1 to this bit generates a repeated START condition provided it is the current master. This bit will always be read as 0. Attempting a repeat at the wrong time results in loss of arbitration.
1 WUEN	Wakeup Enable The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs. 0 Normal operation. No interrupt generated when address matching in low power mode. 1 Enables the wakeup function in low power mode.
0 DMAEN	DMA Enable Enables or disables the DMA function. 0 All DMA signalling disabled. 1 DMA transfer is enabled. While SMB[FACK] = 0, the following conditions trigger the DMA request: • a data byte is received, and either address or data is transmitted. (ACK/NACK is automatic) • the first byte received matches the A1 register or is a general call address.

Table continues on the next page...

I2Cx_C1 field descriptions (continued)

Field	Description
	If any address matching occurs, S[IAAS] and S[TCF] are set. If the direction of transfer is known from master to slave, then it is not required to check S[SRW]. With this assumption, DMA can also be used in this case. In other cases, if the master reads data from the slave, then it is required to rewrite the C1 register operation. With this assumption, DMA cannot be used. When FACK = 1, an address or a data byte is transmitted.

43.4.4 I2C Status register (I2Cx_S)

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read	TCF	IAAS	BUSY	ARBL	RAM	SRW	IICIF	RXAK
Write				w1c			w1c	
Reset	1	0	0	0	0	0	0	0

I2Cx_S field descriptions

Field	Description
7 TCF	Transfer Complete Flag Acknowledges a byte transfer; TCF is set on the completion of a byte transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. TCF is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode. **NOTE:** In the buffer mode, TCF is cleared automatically by internal reading or writing the data register I2C_D, with no need waiting for manually reading/writing the I2C data register in Rx/Tx mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	Addressed As A Slave This bit is set by one of the following conditions: • The calling address matches the programmed primary slave address in the A1 register, or matches the range address in the RA register (which must be set to a nonzero value and under the condition I2C_C2[RMEN] = 1). • C2[GCAEN] is set and a general call is received. • SMB[SIICAEN] is set and the calling address matches the second programmed slave address. • ALERTEN is set and an SMBus alert response address is received • RMEN is set and an address is received that is within the range between the values of the A1 and RA registers. IAAS sets before the ACK bit. The CPU must check the SRW bit and set TX/RX accordingly. Writing the C1 register with any value clears this bit. 0 Not addressed 1 Addressed as a slave
5 BUSY	Bus Busy

Table continues on the next page...

I2C*x*_S field descriptions (continued)

Field	Description
	Indicates the status of the bus regardless of slave or master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	Arbitration Lost This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software, by writing 1 to it. 0 Standard bus operation. 1 Loss of arbitration.
3 RAM	Range Address Match This bit is set to 1 by any of the following conditions, if I2C_C2[RMEN] = 1: • Any nonzero calling address is received that matches the address in the RA register. • The calling address is within the range of values of the A1 and RA registers. **NOTE:** For the RAM bit to be set to 1 correctly, C1[IICIE] must be set to 1. Writing the C1 register with any value clears this bit to 0. 0 Not addressed 1 Addressed as a slave
2 SRW	Slave Read/Write When addressed as a slave, SRW indicates the value of the R/$\overline{W}$ command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	Interrupt Flag This bit sets when an interrupt is pending. This bit must be cleared by software by writing 1 to it, such as in the interrupt routine. One of the following events can set this bit: • One byte transfer, including ACK/NACK bit, completes if FACK is 0. An ACK or NACK is sent on the bus by writing 0 or 1 to TXAK after this bit is set in receive mode. • One byte transfer, excluding ACK/NACK bit, completes if FACK is 1. • Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address. • Arbitration lost • In SMBus mode, any timeouts except SCL and SDA high timeouts • I2C bus stop or start detection if the SSIE bit in the Input Glitch Filter register is 1 **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit in the Input Glitch Filter register by writing 1 to it, and then clear the IICIF bit. If this sequence is reversed, the IICIF bit is asserted again. 0 No interrupt pending 1 Interrupt pending
0 RXAK	Receive Acknowledge

Table continues on the next page...

I2Cx_S field descriptions (continued)

Field	Description
	0 Acknowledge signal was received after the completion of one byte of data transmission on the bus 1 No acknowledge signal detected

43.4.5 I2C Data I/O register (I2Cx_D)

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read Write	DATA							
Reset	0	0	0	0	0	0	0	0

I2Cx_D field descriptions

Field	Description
DATA	Data In master transmit mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data. **NOTE:** When making the transition out of master receive mode, switch the I2C mode before reading the Data register to prevent an inadvertent initiation of a master receive data transfer. In slave mode, the same functions are available after an address match occurs. The C1[TX] bit must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For example, if the I2C module is configured for master transmit but a master receive is desired, reading the Data register does not initiate the receive. Reading the Data register returns the last byte received while the I2C module is configured in master receive or slave receive mode. The Data register does not reflect every byte that is transmitted on the I2C bus, and neither can software verify that a byte has been written to the Data register correctly by reading it back. In master transmit mode, the first byte of data written to the Data register following assertion of MST (start bit) or assertion of RSTA (repeated start bit) is used for the address transfer and must consist of the calling address (in bits 7-1) concatenated with the required R/$\overline{W}$ bit (in position bit 0).

43.4.6 I2C Control Register 2 (I2Cx_C2)

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read Write	GCAEN	ADEXT	HDRS	SBRC	RMEN	AD[10:8]		
Reset	0	0	0	0	0	0	0	0

I2Cx_C2 field descriptions

Field	Description
7 GCAEN	General Call Address Enable Enables general call address. 0 Disabled 1 Enabled
6 ADEXT	Address Extension Controls the number of bits used for the slave address. 0 7-bit address scheme 1 10-bit address scheme
5 HDRS	High Drive Select Controls the drive capability of the I2C pads. 0 Normal drive mode 1 High drive mode
4 SBRC	Slave Baud Rate Control Enables independent slave mode baud rate at maximum frequency, which forces clock stretching on SCL in very fast I2C modes. To a slave, an example of a "very fast" mode is when the master transfers at 40 kbit/s but the slave can capture the master's data at only 10 kbit/s. 0 The slave baud rate follows the master baud rate and clock stretching may occur 1 Slave baud rate is independent of the master baud rate
3 RMEN	Range Address Matching Enable This bit controls the slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address matching occurs for any address greater than the value of the A1 register and less than or equal to the value of the RA register. 0 Range mode disabled. No address matching occurs for an address within the range of values of the A1 and RA registers. 1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers.
AD[10:8]	Slave Address Contains the upper three bits of the slave address in the 10-bit address scheme. This field is valid only while the ADEXT bit is set.

43.4.7 I2C Programmable Input Glitch Filter Register (I2Cx_FLT)

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read	SHEN	STOPF	SSIE	STARTF	FLT			
Write		w1c		w1c				
Reset	0	0	0	0	0	0	0	0

I2Cx_FLT field descriptions

Field	Description
7 SHEN	Stop Hold Enable Set this bit to hold off entry to stop mode when any data transmission or reception is occurring. The following scenario explains the holdoff functionality: 1. The I2C module is configured for a basic transfer, and the SHEN bit is set to 1. 2. A transfer begins. 3. The MCU signals the I2C module to enter stop mode. 4. The byte currently being transferred, including both address and data, completes its transfer. 5. The I2C slave or master acknowledges that the in-transfer byte completed its transfer and acknowledges the request to enter stop mode. 6. After receiving the I2C module's acknowledgment of the request to enter stop mode, the MCU determines whether to shut off the I2C module's clock. If the SHEN bit is set to 1 and the I2C module is in an idle or disabled state when the MCU signals to enter stop mode, the module immediately acknowledges the request to enter stop mode. If SHEN is cleared to 0 and the overall data transmission or reception that was suspended by stop mode entry was incomplete: To resume the overall transmission or reception after the MCU exits stop mode, software must reinitialize the transfer by resending the address of the slave. If the I2C Control Register 1's IICIE bit was set to 1 before the MCU entered stop mode, system software will receive the interrupt triggered by the I2C Status Register's TCF bit after the MCU wakes from the stop mode. 0 Stop holdoff is disabled. The MCU's entry to stop mode is not gated. 1 Stop holdoff is enabled.
6 STOPF	I2C Bus Stop Detect Flag Hardware sets this bit when the I2C bus's stop status is detected. The STOPF bit must be cleared by writing 1 to it. 0 No stop happens on I2C bus 1 Stop detected on I2C bus
5 SSIE	I2C Bus Stop or Start Interrupt Enable This bit enables the interrupt for I2C bus stop or start detection. **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit by writing 1 to it, and then clear the IICIF bit in the status register. If this sequence is reversed, the IICIF bit is asserted again. 0 Stop or start detection interrupt is disabled 1 Stop or start detection interrupt is enabled
4 STARTF	I2C Bus Start Detect Flag Hardware sets this bit when the I2C bus's start status is detected. The STARTF bit must be cleared by writing 1 to it. 0 No start happens on I2C bus 1 Start detected on I2C bus
FLT	I2C Programmable Filter Factor Controls the width of the glitch, in terms of I2C module clock cycles, that the filter must absorb. For any glitch whose size is less than or equal to this width setting, the filter does not allow the glitch to pass. 0h No filter/bypass 1-Fh Filter glitches up to width of *n* I2C module clock cycles, where *n*=1-15d

43.4.8 I2C Range Address register (I2C*x*_RA)

Address: Base address + 7h offset

Bit	7	6	5	4	3	2	1	0
Read	RAD							0
Write								
Reset	0	0	0	0	0	0	0	0

I2C*x*_RA field descriptions

Field	Description
7–1 RAD	Range Slave Address This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. If I2C_C2[RMEN] is set to 1, any nonzero value write enables this register. This register value can be considered as a maximum boundary in the range matching mode.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.4.9 I2C SMBus Control and Status register (I2C*x*_SMB)

NOTE

When the SCL and SDA signals are held high for a length of time greater than the high timeout period, the SHTF1 flag sets. Before reaching this threshold, while the system is detecting how long these signals are being held high, a master assumes that the bus is free. However, the SHTF1 bit is set to 1 in the bus transmission process with the idle bus state.

NOTE

When the TCKSEL bit is set, there is no need to monitor the SHTF1 bit because the bus speed is too high to match the protocol of SMBus.

Address: Base address + 8h offset

Bit	7	6	5	4	3	2	1	0
Read	FACK	ALERTEN	SIICAEN	TCKSEL	SLTF	SHTF1	SHTF2	SHTF2IE
Write					w1c		w1c	
Reset	0	0	0	0	0	0	0	0

I2Cx_SMB field descriptions

Field	Description
7 FACK	Fast NACK/ACK Enable For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte. 0 An ACK or NACK is sent on the following receiving data byte 1 Writing 0 to TXAK after receiving a data byte generates an ACK. Writing 1 to TXAK after receiving a data byte generates a NACK.
6 ALERTEN	SMBus Alert Response Address Enable Enables or disables SMBus alert response address matching. **NOTE:** After the host responds to a device that used the alert response address, you must use software to put the device's address on the bus. The alert protocol is described in the SMBus specification. 0 SMBus alert response address matching is disabled 1 SMBus alert response address matching is enabled
5 SIICAEN	Second I2C Address Enable Enables or disables SMBus device default address. 0 I2C address register 2 matching is disabled 1 I2C address register 2 matching is enabled
4 TCKSEL	Timeout Counter Clock Select Selects the clock source of the timeout counter. 0 Timeout counter counts at the frequency of the I2C module clock / 64 1 Timeout counter counts at the frequency of the I2C module clock
3 SLTF	SCL Low Timeout Flag This bit is set when the SLT register (consisting of the SLTH and SLTL registers) is loaded with a non-zero value (LoValue) and an SCL low timeout occurs. Software clears this bit by writing a logic 1 to it. **NOTE:** The low timeout function is disabled when the SLT register's value is 0. 0 No low timeout occurs 1 Low timeout occurs
2 SHTF1	SCL High Timeout Flag 1 This read-only bit sets when SCL and SDA are held high more than clock × LoValue / 512, which indicates the bus is free. This bit is cleared automatically. 0 No SCL high and SDA high timeout occurs 1 SCL high and SDA high timeout occurs
1 SHTF2	SCL High Timeout Flag 2 This bit sets when SCL is held high and SDA is held low more than clock × LoValue / 512. Software clears this bit by writing 1 to it. 0 No SCL high and SDA low timeout occurs 1 SCL high and SDA low timeout occurs

Table continues on the next page...

I2Cx_SMB field descriptions (continued)

Field	Description
0 SHTF2IE	SHTF2 Interrupt Enable Enables SCL high and SDA low timeout interrupt. 0 SHTF2 interrupt is disabled 1 SHTF2 interrupt is enabled

43.4.10 I2C Address Register 2 (I2Cx_A2)

Address: Base address + 9h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SAD							0
Reset	1	1	0	0	0	0	1	0

I2Cx_A2 field descriptions

Field	Description
7–1 SAD	SMBus Address Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.4.11 I2C SCL Low Timeout Register High (I2Cx_SLTH)

Address: Base address + Ah offset

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[15:8]							
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTH field descriptions

Field	Description
SSLT[15:8]	SSLT[15:8] Most significant byte of SCL low timeout value that determines the timeout period of SCL low.

43.4.12 I2C SCL Low Timeout Register Low (I2Cx_SLTL)

Address: Base address + Bh offset

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[7:0]							
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTL field descriptions

Field	Description
SSLT[7:0]	SSLT[7:0] Least significant byte of SCL low timeout value that determines the timeout period of SCL low.

43.4.13 I2C Status register 2 (I2Cx_S2)

Address: Base address + Ch offset

Bit	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	ERROR	EMPTY
Write							w1c	
Reset	0	0	0	0	0	0	0	1

I2Cx_S2 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 ERROR	Error flag Indicates if there are read or write errors with the Tx and Rx buffers. 0 The buffer is not full and all write/read operations have no errors. 1 There are 3 or more write/read errors during the data transfer phase (when the Empty flag is not set and the buffer is busy).

Table continues on the next page...

I2Cx_S2 field descriptions (continued)

Field	Description
0 EMPTY	Empty flag Indicates if the Tx or Rx buffer is empty. 0 Tx or Rx buffer is not empty and cannot be written to, that is new data cannot be loaded into the buffer. 1 Tx or Rx buffer is empty and can be written to, that is new data can be loaded into the buffer.

43.5 Functional description

This section provides a comprehensive functional description of the I2C module.

43.5.1 I2C protocol

The I2C bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers.

All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors depends on the system.

Normally, a standard instance of communication is composed of four parts:

1. START signal
2. Slave address transmission
3. Data transfer
4. STOP signal

The STOP signal should not be confused with the CPU STOP instruction. The following figure illustrates I2C bus system communication.

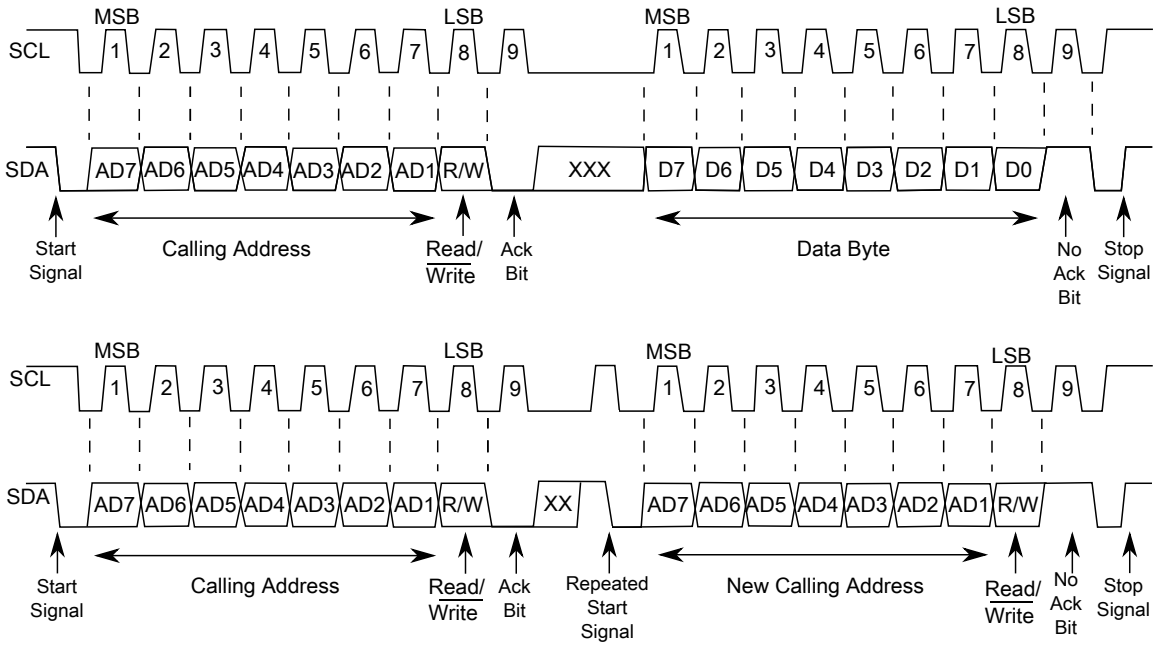


Figure 43-2. I2C bus transmission signals

43.5.1.1 START signal

The bus is free when no master device is engaging the bus (both SCL and SDA are high). When the bus is free, a master may initiate communication by sending a START signal. A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer—each data transfer might contain several bytes of data—and brings all slaves out of their idle states.

43.5.1.2 Slave address transmission

Immediately after the START signal, the first byte of a data transfer is the slave address transmitted by the master. This address is a 7-bit calling address followed by an R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave the desired direction of data transfer.

- 1 = Read transfer: The slave transmits data to the master
- 0 = Write transfer: The master transmits data to the slave

Only the slave with a calling address that matches the one transmitted by the master responds by sending an acknowledge bit. The slave sends the acknowledge bit by pulling SDA low at the ninth clock.

No two slaves in the system can have the same address. If the I2C module is the master, it must not transmit an address that is equal to its own slave address. The I2C module cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I2C module reverts to slave mode and operates correctly even if it is being addressed by another master.

43.5.1.3 Data transfers

When successful slave addressing is achieved, data transfer can proceed on a byte-by-byte basis in the direction specified by the R/$\overline{W}$ bit sent by the calling master.

All transfers that follow an address cycle are referred to as data transfers, even if they carry subaddress information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low. Data must be held stable while SCL is high. There is one clock pulse on SCL for each data bit, and the MSB is transferred first. Each data byte is followed by a ninth (acknowledge) bit, which is signaled from the receiving device by pulling SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit, the slave must leave SDA high. The master interprets the failed acknowledgement as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets it as an end to data transfer and releases the SDA line.

In the case of a failed acknowledgement by either the slave or master, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a STOP signal.
- Commences a new call by generating a repeated START signal.

43.5.1.4 STOP signal

The master can terminate the communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is asserted.

43.5.1.5 Repeated START signal

The master may generate a START signal followed by a calling command without generating a STOP signal first. This action is called a repeated START. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.The master needs to send a NACK signal before sending repeated-START in the buffering mode.

43.5.1.6 Arbitration procedure

The I2C bus is a true multimaster bus that allows more than one master to be connected on it.

If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The bus clock's low period is equal to the longest clock low period, and the high period is equal to the shortest one among the masters.

The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic level 1 while another master transmits logic level 0. The losing masters immediately switch to slave receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate the loss of arbitration.

43.5.1.7 Clock synchronization

Because wire AND logic is performed on SCL, a high-to-low transition on SCL affects all devices connected on the bus. The devices start counting their low period and, after a device's clock has gone low, that device holds SCL low until the clock reaches its high state. However, the change of low to high in this device clock might not change the state of SCL if another device clock is still within its low period. Therefore, the synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time; see the following diagram. When all applicable devices have counted off their low period, the synchronized clock SCL is released and pulled high. Afterward there is no difference between the device clocks and the state of SCL, and all devices start counting their high periods. The first device to complete its high period pulls SCL low again.

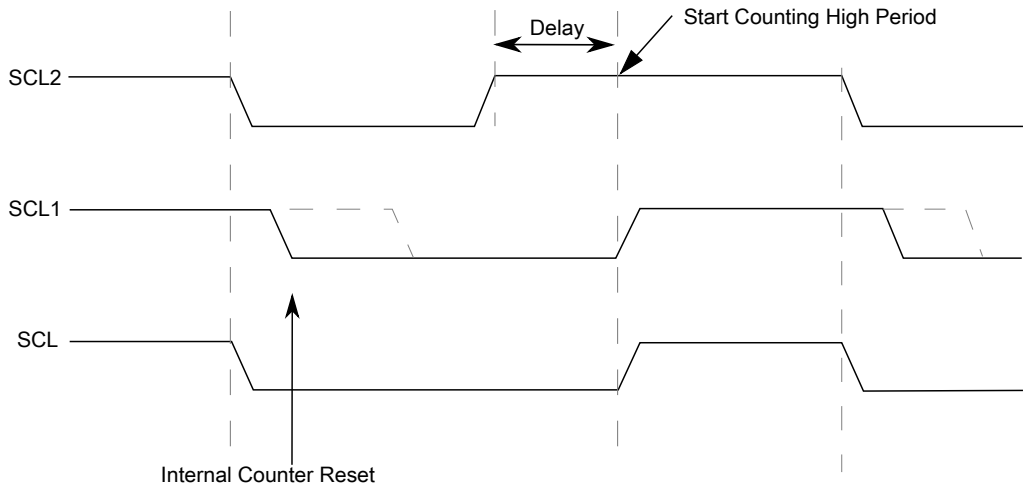


Figure 43-3. I2C clock synchronization

43.5.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. A slave device may hold SCL low after completing a single byte transfer (9 bits). In this case, it halts the bus clock and forces the master clock into wait states until the slave releases SCL.

43.5.1.9 Clock stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master drives SCL low, a slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal's low period is stretched. In other words, the SCL bus signal's low period is increased to be the same length as the slave's SCL low period.

43.5.1.10 I2C divider and hold values

NOTE

For some cases on some devices, the SCL divider value may vary by ±2 or ±4 when ICR's value ranges from 00h to 0Fh. These potentially varying SCL divider values are highlighted in the following table. For the actual SCL divider values for your device, see the chip-specific details about the I2C module.

Table 43-2. I2C divider and hold values

ICR (hex)	SCL divider	SDA hold value	SCL hold (start) value	SCL hold (stop) value	ICR (hex)	SCL divider (clocks)	SDA hold (clocks)	SCL hold (start) value	SCL hold (stop) value
00	20	7	6	11	20	160	17	78	81
01	22	7	7	12	21	192	17	94	97
02	24	8	8	13	22	224	33	110	113
03	26	8	9	14	23	256	33	126	129
04	28	9	10	15	24	288	49	142	145
05	30	9	11	16	25	320	49	158	161
06	34	10	13	18	26	384	65	190	193
07	40	10	16	21	27	480	65	238	241
08	28	7	10	15	28	320	33	158	161
09	32	7	12	17	29	384	33	190	193
0A	36	9	14	19	2A	448	65	222	225
0B	40	9	16	21	2B	512	65	254	257
0C	44	11	18	23	2C	576	97	286	289
0D	48	11	20	25	2D	640	97	318	321
0E	56	13	24	29	2E	768	129	382	385
0F	68	13	30	35	2F	960	129	478	481
10	48	9	18	25	30	640	65	318	321
11	56	9	22	29	31	768	65	382	385
12	64	13	26	33	32	896	129	446	449
13	72	13	30	37	33	1024	129	510	513
14	80	17	34	41	34	1152	193	574	577
15	88	17	38	45	35	1280	193	638	641
16	104	21	46	53	36	1536	257	766	769
17	128	21	58	65	37	1920	257	958	961
18	80	9	38	41	38	1280	129	638	641
19	96	9	46	49	39	1536	129	766	769
1A	112	17	54	57	3A	1792	257	894	897
1B	128	17	62	65	3B	2048	257	1022	1025
1C	144	25	70	73	3C	2304	385	1150	1153
1D	160	25	78	81	3D	2560	385	1278	1281
1E	192	33	94	97	3E	3072	513	1534	1537
1F	240	33	118	121	3F	3840	513	1918	1921

43.5.2 10-bit address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

43.5.2.1 Master-transmitter addresses a slave-receiver

The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first 7 bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit (R/$\overline{W}$ direction bit) is 0. It is possible that more than one device finds a match and generates an acknowledge (A1). Each slave that finds a match compares the 8 bits of the second byte of the slave address with its own address, but only one slave finds a match and generates an acknowledge (A2). The matching slave remains addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

Table 43-3. Master-transmitter addresses slave-receiver with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Data	A	...	Data	A/A	P

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

43.5.2.2 Master-receiver addresses a slave-transmitter

The transfer direction is changed after the second R/$\overline{W}$ bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and it tests whether the eighth (R/$\overline{W}$) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

Table 43-4. Master-receiver addresses a slave-transmitter with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Sr	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 1	A3	Data	A	...	Data	A	P

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

43.5.3 Address matching

All received addresses can be requested in 7-bit or 10-bit address format.

- AD[7:1] in Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. It provides a 7-bit address.
- If the ADEXT bit is set, AD[10:8] in Control Register 2 participates in the address matching process. It extends the I2C primary slave address to a 10-bit address.

Additional conditions that affect address matching include:

- If the GCAEN bit is set, general call participates the address matching process.
- If the ALERTEN bit is set, alert response participates the address matching process.
- If the SIICAEN bit is set, Address Register 2 participates in the address matching process.
- If the RMEN bit is set, when the Range Address register is programmed to a nonzero value, any address within the range of values of Address Register 1 (excluded) and the Range Address register (included) participates in the address matching process. The Range Address register must be programmed to a value greater than the value of Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

43.5.4 System management bus specification

SMBus provides a control bus for system and power management related tasks. A system can use SMBus to pass messages to and from devices instead of tripping individual control lines.

Removing the individual control lines reduces pin count. Accepting messages ensures future expandability. With the system management bus, a device can provide manufacturer information, tell the system what its model/part number is, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

43.5.4.1 Timeouts

The $T_{TIMEOUT,MIN}$ parameter allows a master or slave to conclude that a defective device is holding the clock low indefinitely or a master is intentionally trying to drive devices off the bus. The slave device must release the bus (stop driving the bus and let SCL and SDA float high) when it detects any single clock held low longer than $T_{TIMEOUT,MIN}$. Devices that have detected this condition must reset their communication and be able to receive a new START condition within the timeframe of $T_{TIMEOUT,MAX}$.

SMBus defines a clock low timeout, $T_{TIMEOUT}$, of 35 ms, specifies $T_{LOW:SEXT}$ as the cumulative clock low extend time for a slave device, and specifies $T_{LOW:MEXT}$ as the cumulative clock low extend time for a master device.

43.5.4.1.1 SCL low timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than a timeout value condition. Devices that have detected the timeout condition must reset the communication. When the I2C module is an active master, if it detects that SMBCLK low has exceeded the value of $T_{TIMEOUT,MIN}$, it must generate a stop condition within or after the current data byte in the transfer process. When the I2C module is a slave, if it detects the $T_{TIMEOUT,MIN}$ condition, it resets its communication and is then able to receive a new START condition.

43.5.4.1.2 SCL high timeout

When the I2C module has determined that the SMBCLK and SMBDAT signals have been high for at least $T_{HIGH:MAX}$, it assumes that the bus is idle.

A HIGH timeout occurs after a START condition appears on the bus but before a STOP condition appears on the bus. Any master detecting this scenario can assume the bus is free when either of the following occurs:

- SHTF1 rises.
- The BUSY bit is high and SHTF1 is high.

When the SMBDAT signal is low and the SMBCLK signal is high for a period of time, another kind of timeout occurs. The time period must be defined in software. SHTF2 is used as the flag when the time limit is reached. This flag is also an interrupt resource, so it triggers IICIF.

43.5.4.1.3 CSMBCLK TIMEOUT MEXT and CSMBCLK TIMEOUT SEXT

The following figure illustrates the definition of the timeout intervals $T_{LOW:SEXT}$ and $T_{LOW:MEXT}$. When in master mode, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:MEXT}$ within a byte, where each byte is defined as START-to-ACK, ACK-to-ACK, or ACK-to-STOP. When CSMBCLK TIMEOUT MEXT occurs, SMBus MEXT rises and also triggers the SLTF.

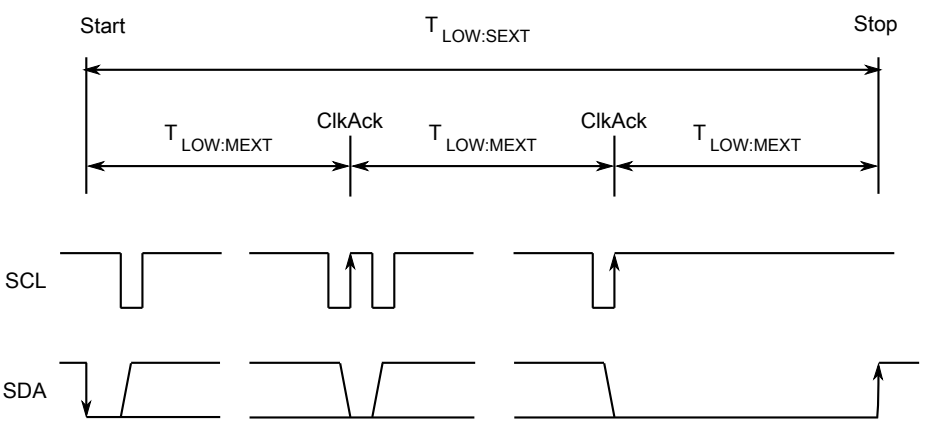


Figure 43-4. Timeout measurement intervals

A master is allowed to abort the transaction in progress to any slave that violates the $T_{LOW:SEXT}$ or $T_{TIMEOUT,MIN}$ specifications. To abort the transaction, the master issues a STOP condition at the conclusion of the byte transfer in progress. When a slave, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:SEXT}$ during any message from the initial START to the STOP. When CSMBCLK TIMEOUT SEXT occurs, SEXT rises and also triggers SLTF.

NOTE

CSMBCLK TIMEOUT SEXT and CSMBCLK TIMEOUT MEXT are optional functions that are implemented in the second step.

43.5.4.2 FAST ACK and NACK

To improve reliability and communication robustness, implementation of packet error checking (PEC) by SMBus devices is optional for SMBus devices but required for devices participating in and only during the address resolution protocol (ARP) process. The PEC is a CRC-8 error checking byte, calculated on all the message bytes. The PEC is appended to the message by the device that supplied the last data byte. If the PEC is present but not correct, a NACK is issued by the receiver. Otherwise an ACK is issued. To calculate the CRC-8 by software, this module can hold the SCL line low after receiving the eighth SCL (8th bit) if this byte is a data byte. So software can determine whether an ACK or NACK should be sent to the bus by setting or clearing the TXAK bit if the FACK (fast ACK/NACK enable) bit is enabled.

SMBus requires a device always to acknowledge its own address, as a mechanism to detect the presence of a removable device (such as a battery or docking station) on the bus. In addition to indicating a slave device busy condition, SMBus uses the NACK mechanism to indicate the reception of an invalid command or invalid data. Because such a condition may occur on the last byte of the transfer, SMBus devices are required to have the ability to generate the not acknowledge after the transfer of each byte and before the completion of the transaction. This requirement is important because SMBus does not provide any other resend signaling. This difference in the use of the NACK signaling has implications on the specific implementation of the SMBus port, especially in devices that handle critical system data such as the SMBus host and the SBS components.

NOTE

In the last byte of master receive slave transmit mode, the master must send a NACK to the bus, so FACK must be switched off before the last byte transmits.

43.5.5 Resets

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

43.5.6 Interrupts

The I2C module generates an interrupt when any of the events in the table found here occur, provided that the IICIE bit is set.

The interrupt is driven by the IICIF bit (of the I2C Status Register) and masked with the IICIE bit (of the I2C Control Register 1). The IICIF bit must be cleared (by software) by writing 1 to it in the interrupt routine. The SMBus timeouts interrupt is driven by SLTF and masked with the IICIE bit. The SLTF bit must be cleared by software by writing 1 to it in the interrupt routine. You can determine the interrupt type by reading the Status Register.

NOTE

In master receive mode, the FACK bit must be set to zero before the last byte transfer.

Table 43-5. Interrupt summary

Interrupt source	Status	Flag	Local enable
Complete 1-byte transfer	TCF	IICIF	IICIE
Match of received calling address	IAAS	IICIF	IICIE
Arbitration lost	ARBL	IICIF	IICIE
I^2C bus stop detection	STOPF	IICIF	IICIE & SSIE
I^2C bus start detection	STARTF	IICIF	IICIE & SSIE
SMBus SCL low timeout	SLTF	IICIF	IICIE
SMBus SCL high SDA low timeout	SHTF2	IICIF	IICIE & SHTF2IE
Wakeup from stop or wait mode	IAAS	IICIF	IICIE & WUEN

43.5.6.1 Byte transfer interrupt

The Transfer Complete Flag (TCF) bit is set at the falling edge of the ninth clock to indicate the completion of a byte and acknowledgement transfer. When FACK is enabled, TCF is then set at the falling edge of eighth clock to indicate the completion of byte.

43.5.6.2 Address detect interrupt

When the calling address matches the programmed slave address (I2C Address Register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the Status Register is set. The CPU is interrupted, provided the IICIE bit is set. The CPU must check the SRW bit and set its Tx mode accordingly.

43.5.6.3 Stop Detect Interrupt

When the stop status is detected on the I^2C bus, the STOPF bit is set to 1. The CPU is interrupted, provided the IICIE and SSIE bits are both set to 1.

43.5.6.4 Exit from low-power/stop modes

The slave receive input detect circuit and address matching feature are still active on low power modes (wait and stop). An asynchronous input matching slave address or general call address brings the CPU out of low power/stop mode if the interrupt is not masked. Therefore, TCF and IAAS both can trigger this interrupt.

43.5.6.5 Arbitration lost interrupt

The I2C is a true multimaster bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The I2C module asserts the arbitration-lost interrupt when it loses the data arbitration process and the ARBL bit in the Status Register is set.

Arbitration is lost in the following circumstances:

1. SDA is sampled as low when the master drives high during an address or data transmit cycle.
2. SDA is sampled as low when the master drives high during the acknowledge bit of a data receive cycle.
3. A START cycle is attempted when the bus is busy.
4. A repeated START cycle is requested in slave mode.
5. A STOP condition is detected when the master did not request it.

The ARBL bit must be cleared (by software) by writing 1 to it.

43.5.6.6 Timeout interrupt in SMBus

When the IICIE bit is set, the I2C module asserts a timeout interrupt (outputs SLTF and SHTF2) upon detection of any of the mentioned timeout conditions, with one exception. The SCL high and SDA high TIMEOUT mechanism must not be used to influence the timeout interrupt output, because this timeout indicates an idle condition on the bus. SHTF1 rises when it matches the SCL high and SDA high TIMEOUT and falls automatically just to indicate the bus status. The SHTF2's timeout period is the same as that of SHTF1, which is short compared to that of SLTF, so another control bit, SHTF2IE, is added to enable or disable it.

43.5.7 Programmable input glitch filter

An I2C glitch filter has been added outside legacy I2C modules but within the I2C package. This filter can absorb glitches on the I2C clock and data lines for the I2C module.

The width of the glitch to absorb can be specified in terms of the number of (half) I2C module clock cycles. A single Programmable Input Glitch Filter control register is provided. Effectively, any down-up-down or up-down-up transition on the data line that occurs within the number of clock cycles programmed in this register is ignored by the I2C module. The programmer must specify the size of the glitch (in terms of I2C module clock cycles) for the filter to absorb and not pass.

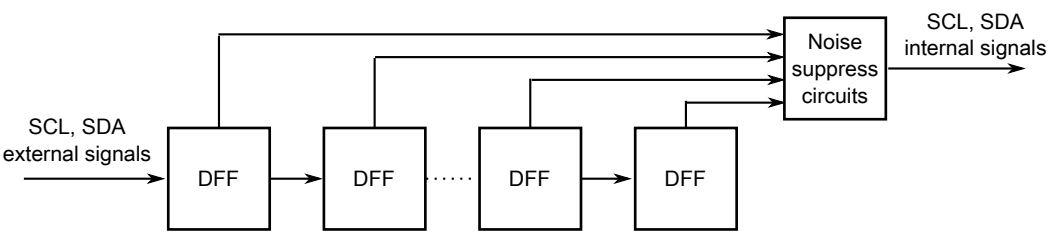


Figure 43-5. Programmable input glitch filter diagram

43.5.8 Address matching wake-up

When a primary, range, or general call address match occurs when the I2C module is in slave receive mode, the MCU wakes from a low power mode where no peripheral bus is running.

Data sent on the bus that is the same as a target device address might also wake the target MCU.

After the address matching IAAS bit is set, an interrupt is sent at the end of address matching to wake the core. The IAAS bit must be cleared after the clock recovery.

NOTE

After the system recovers and is in Run mode, restart the I2C module if it is needed to transfer packets. To avoid I2C transfer problems resulting from the situation, firmware should prevent the MCU execution of a STOP instruction when the I2C module is in the middle of a transfer unless the Stop mode holdoff feature is used during this period (set FLT[SHEN] to 1).

NOTE

After I2C address matching wake-up, the master must wait a time long enough for the slave ISR to finish running and resend start or repeat start signals.

For the SRW bit to function properly, it only supports Address +Write to wake up by I2C address matching. Before entering the next low power mode, Address+Write must be sent to change the SRW status.

43.5.9 DMA support

If the DMAEN bit is cleared and the IICIE bit is set, an interrupt condition generates an interrupt request.

If the DMAEN bit is set and the IICIE bit is set, an interrupt condition generates a DMA request instead. DMA requests are generated by the transfer complete flag (TCF).

If the DMAEN bit is set, only the TCF initiates a DMA request. All other events generate CPU interrupts.

NOTE

Before the last byte of master receive mode, TXAK must be set to send a NACK after the last byte's transfer. Therefore, the DMA must be disabled before the last byte's transfer.

NOTE

In 10-bit address mode transmission, the addresses to send occupy 2–3 bytes. During this transfer period, the DMA must be disabled because the C1 register is written to send a repeat start or to change the transfer direction.

43.5.10 Double buffering mode

In the double buffering mode, the data transfer is processed byte by byte. However, the data can be transferred without waiting for the interrupt or the polling to finish. This means the write/read I2C_D operation will not block the data transfer, as the hardware has already finished the internal write or read. The benefit is that the baud rate is able to achieve higher speed.

There are several items to consider as follows:

- When initiating a double buffering transfer at Tx side, the user can write 2 values to the I2C_D buffer before transfer. However, that is allowed only at one time per package frame (due to the buffer depth, and because two-times writes in each ISR are not allowed). The second write to the I2C_D buffer must wait for the Empty flag. On the other hand, at Rx side the user can read twice in a one-byte transfer (if needed).

NOTE

Check Empty flag before write to I2C_D.

Write twice to the I2C_D buffer ONLY after the address matching byte. Do not write twice (Address+Data) before START or at the beginning of I2C transfer, especially when the baud rate is very slow.

- To write twice in one frame, during the next-to-last ISR, do a dummy read from the I2C_D buffer at Tx side (or the TCF will stay high, because the TCF is cleared by write/read operation). In the next-to-last ISR, do not send data again (the buffer data will be under running).
- To keep new ISRs software-compatible with previous ISRs, the write/read I2C_D operation will not block the internal-hardware-released SCL/SDA signals. At the ACK phase, the bus is released to accept the next byte if the master can send the clock immediately.
- On the slave side, two-times writes to the I2C_D buffer may be limited by the master's clock and START/repeated-START signal. This is not currently supported, and the master's START/repeated-START signal will break data transfers. To release the bus, do a dummy read or write to the I2C_D buffer again. It is suggested to send repeated-START/START during intervals as before.
- The master receive should send a NACK in the next-to-last ISR, if it wants to do the STOP or the repeated-START work. The transmitting slave which receives the NACK, will switch to receive mode, and do a dummy read to release SCL and SDA signals.

43.6 Initialization/application information

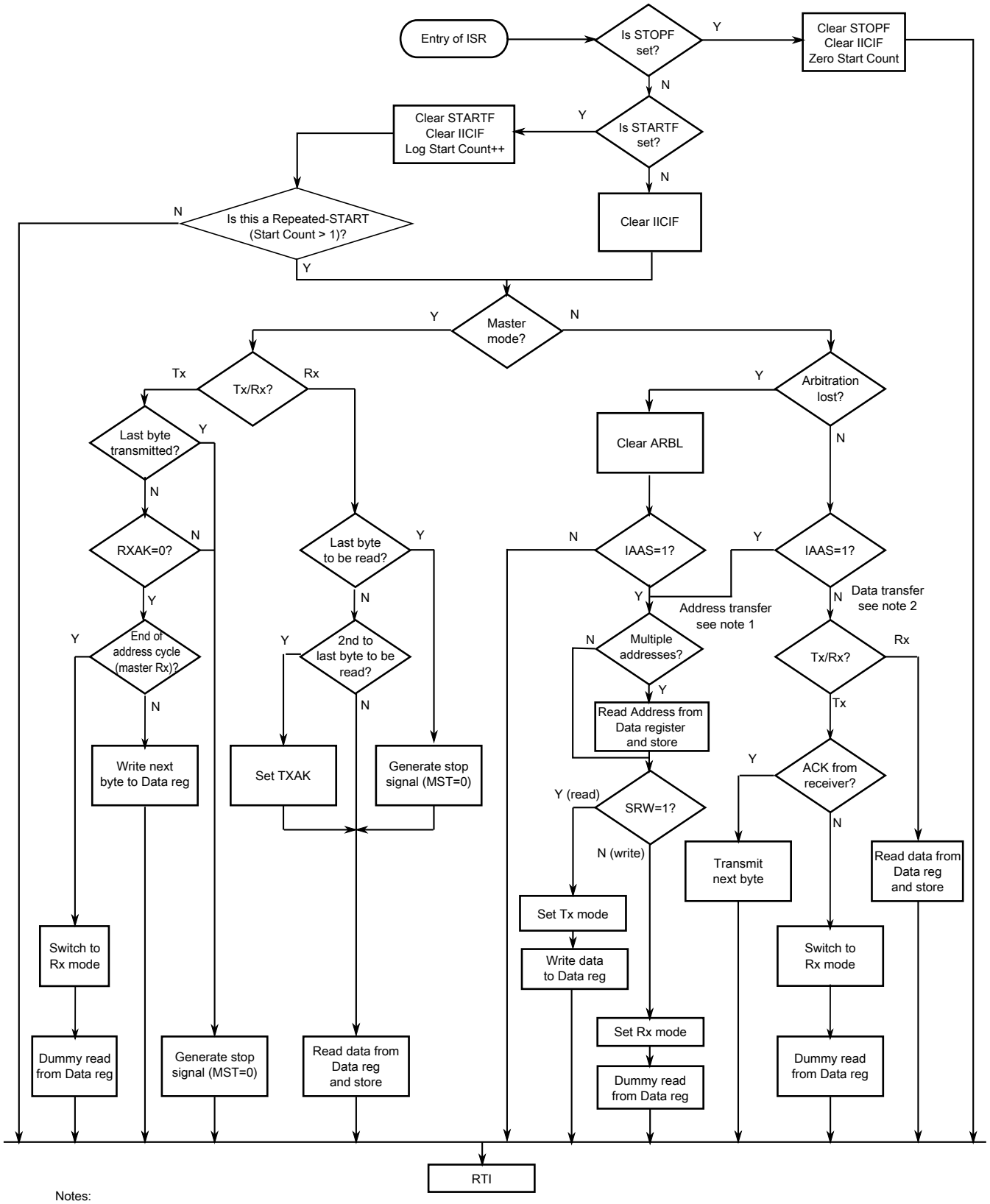
Module Initialization (Slave)

1. Write: Control Register 2
   - to enable or disable general call
   - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address
3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
5. Initialize RAM variables used to achieve the routine shown in the following figure

Module Initialization (Master)

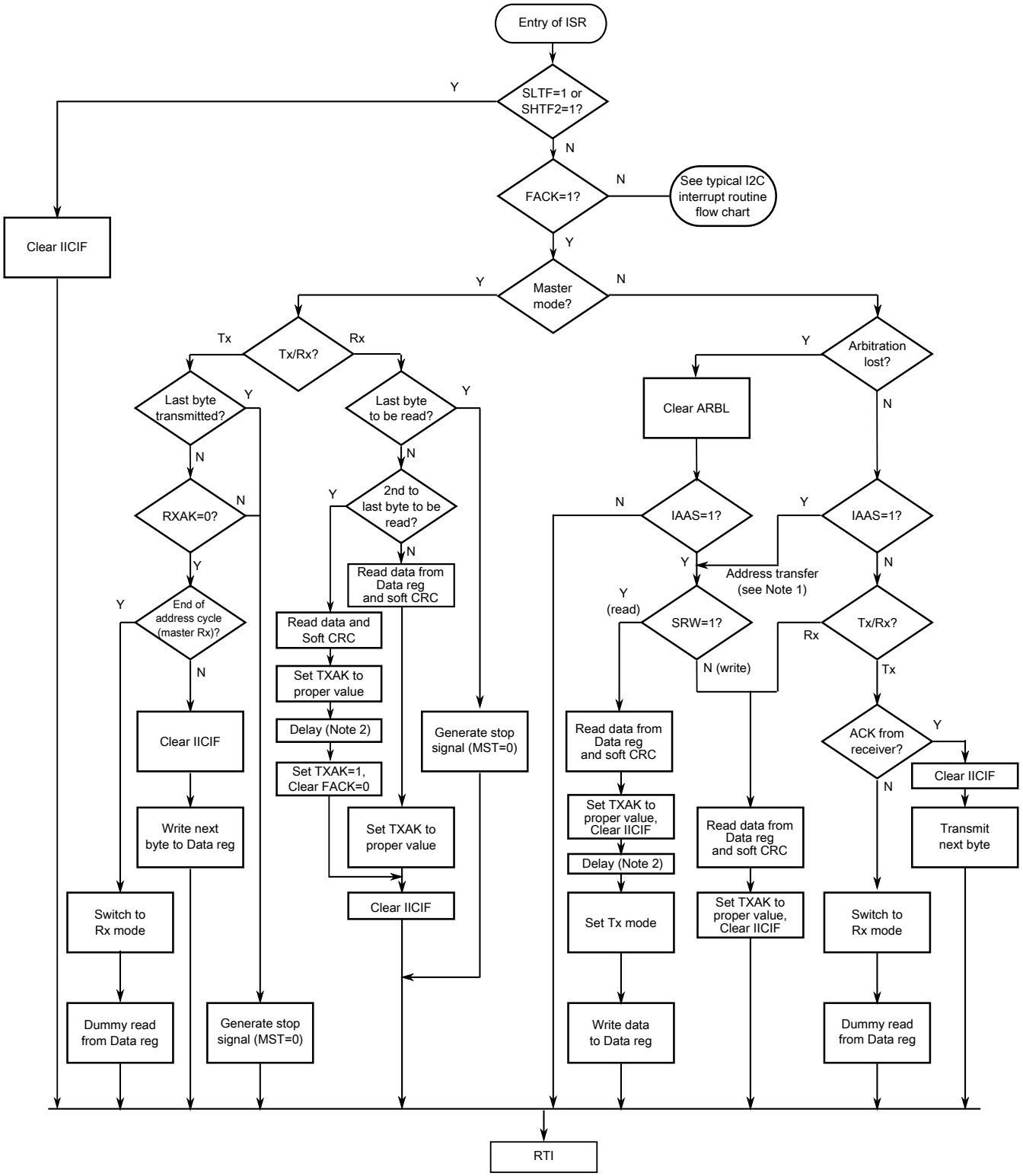
1. Write: Frequency Divider register to set the I2C baud rate (see example in description of ICR)
2. Write: Control Register 1 to enable the I2C module and interrupts
3. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
4. Initialize RAM variables used to achieve the routine shown in the following figure
5. Write: Control Register 1 to enable TX
6. Write: Control Register 1 to enable MST (master mode)
7. Write: Data register with the address of the target slave (the LSB of this byte determines whether the communication is master receive or transmit)

The routine shown in the following figure encompasses both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be initiated by writing the Data register. An example of an I2C driver which implements many of the steps described here is available in AN4342: Using the Inter-Integrated Circuit on ColdFire+ and Kinetis .



Notes:
1. If general call is enabled, check to determine if the received address is a general call address (0x00).
   If the received address is a general call address, the general call must be handled by user software.
2. When 10-bit addressing addresses a slave, the slave sees an interrupt following the first byte of the extended address.
   Ensure that for this interrupt, the contents of the Data register are ignored and not treated as a valid data transfer.

Figure 43-6. Typical I2C interrupt routine



Notes:
1. If general call or SIICAEN is enabled, check to determine if the received address is a general call address (0x00) or an SMBus device default address. In either case, they must be handled by user software.
2. In receive mode, one bit time delay may be needed before the first and second data reading, to wait for the possible longest time period (in worst case) of the 9th SCL cycle.

Figure 43-7. Typical I2C SMBus interrupt routine

Chapter 44
Serial Peripheral Interface (SPI)

44.1 Chip-specific SPI information

The following registers are not available in this device:

Table 44-1. SPI register

Absolute address	Register	Instance
0x4007_500A	SPI clear interrupt register (SPI0_CI)	SPI0
0x4007_500B	SPI control register 3 (SPI0_C3)	SPI0

NOTE

SPI0 does not have SPI0_CI, SPI0_C3, and relative register bits in SPI0_S, as there is no FIFO in SPI0.

44.1.1 Instantiation Information

KM3x_256 family of devices have two SPI modules. The SPI instances support the following features:

Table 44-2. SPI instantiation

Features	SPI0	SPI1
FIFO Depth (in Bytes)	—	8 for Tx 8 for Rx
AMR support	3 V only	3 V only
DMA support	Yes	Yes
Module Clock	75 MHz	75 MHz

SPI supports a baud rate up to the module clock divided by two in master mode and up to the module clock divided by 4 in slave mode.

The SPI modules can be configured to work in WAIT, VLPR, VLPW modes. In STOP or VLPS modes, if SPI is configured to be slave, it can receive up to 16bit data before generating interrupt to wake up MCU.

44.1.2 Data Output Invertor Control

The external 3V-to-5V level shifter circuit is needed to interface with external 5V SPI device. To built this simple external level shift with 1 triode, the SPI output can to be inverted before output from IO.

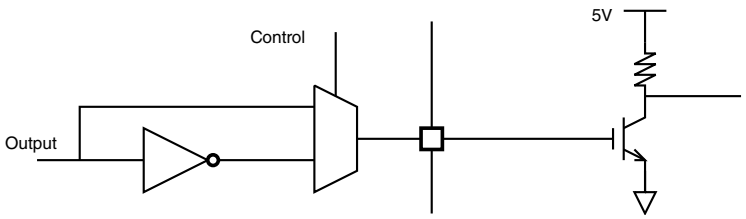


Figure 44-1. SPI Output Invert Control

To output inverted or non-inverted data can be controlled by control bits in SIM module:

- SPI0 signals, configurable through SIM_CTRL_REG[SPI0_INV] bits
- SPI1 signals, configurable through SIM_CTRL_REG[SPI1_INV] bits

44.2 Introduction

The serial peripheral interface (SPI) module provides for full-duplex, synchronous, serial communication between the MCU and peripheral devices. These peripheral devices can include other microcontrollers, analog-to-digital converters, shift registers, sensors, and memories, among others.

The SPI runs at a baud rate up to the SPI module clock divided by two in master mode and up to the SPI module clock divided by four in slave mode. Software can poll the status flags, or SPI operation can be interrupt driven.

NOTE

For the actual maximum SPI baud rate, refer to the Chip Configuration details and to the device's Data Sheet.

The SPI also supports a data length of 8 or 16 bits and includes a hardware match feature for the receive data buffer.

The SPI includes an internal DMA interface to support continuous SPI transmission through an on-chip DMA controller instead of through the CPU. This feature decreases CPU loading, allowing CPU time to be used for other work.

44.2.1 Features

The SPI includes these distinctive features:

- Master mode or slave mode operation
- Full-duplex or single-wire bidirectional mode
- Programmable transmit bit rate
- Double-buffered transmit and receive data register
- Serial clock phase and polarity options
- Slave select output
- Mode fault error flag with CPU interrupt capability
- Control of SPI operation during wait mode
- Selectable MSB-first or LSB-first shifting
- Programmable 8- or 16-bit data transmission length
- Receive data buffer hardware match feature
- 64-bit FIFO mode for high speed/large amounts of data transfers
- Support transmission of both Transmit and Receive by DMA

44.2.2 Modes of operation

The SPI functions in the following three modes.

- Run mode

  This is the basic mode of operation.

- Wait mode

  SPI operation in Wait mode is a configurable low power mode, controlled by the SPISWAI bit located in the SPIx_C2 register. In Wait mode, if C2[SPISWAI] is clear, the SPI operates like in Run mode. If C2[SPISWAI] is set, the SPI goes into a

power conservative state, with the SPI clock generation turned off. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU enters Run mode. If the SPI is configured as a slave, reception and transmission of a byte continues, so that the slave stays synchronized to the master.

- Stop mode

  To reduce power consumption, the SPI is inactive in stop modes where the peripheral bus clock is stopped but internal logic states are retained. If the SPI is configured as a master, any transmission in progress stops, but is resumed after the CPU enters run mode. If the SPI is configured as a slave, reception and transmission of a data continues, so that the slave stays synchronized to the master.

  The SPI is completely disabled in Stop modes where the peripheral bus clock is stopped and internal logic states are not retained. When the CPU wakes from these Stop modes, all SPI register content is reset.

Detailed descriptions of operating modes appear in Low-power mode options.

44.2.3 Block diagrams

This section includes block diagrams showing SPI system connections, the internal organization of the SPI module, and the SPI clock dividers that control the master mode bit rate.

44.2.3.1 SPI system block diagram

The following figure shows the SPI modules of two MCUs connected in a master-slave arrangement. The master device initiates all SPI data transfers. During a transfer, the master shifts data out (on the MOSI pin) to the slave while simultaneously shifting data in (on the MISO pin) from the slave. The transfer effectively exchanges the data that was in the SPI shift registers of the two SPI systems. The SPSCK signal is a clock output from the master and an input to the slave. The slave device must be selected by a low level on the slave select input (SS pin). In this system, the master device has configured its SS pin as an optional slave select output.

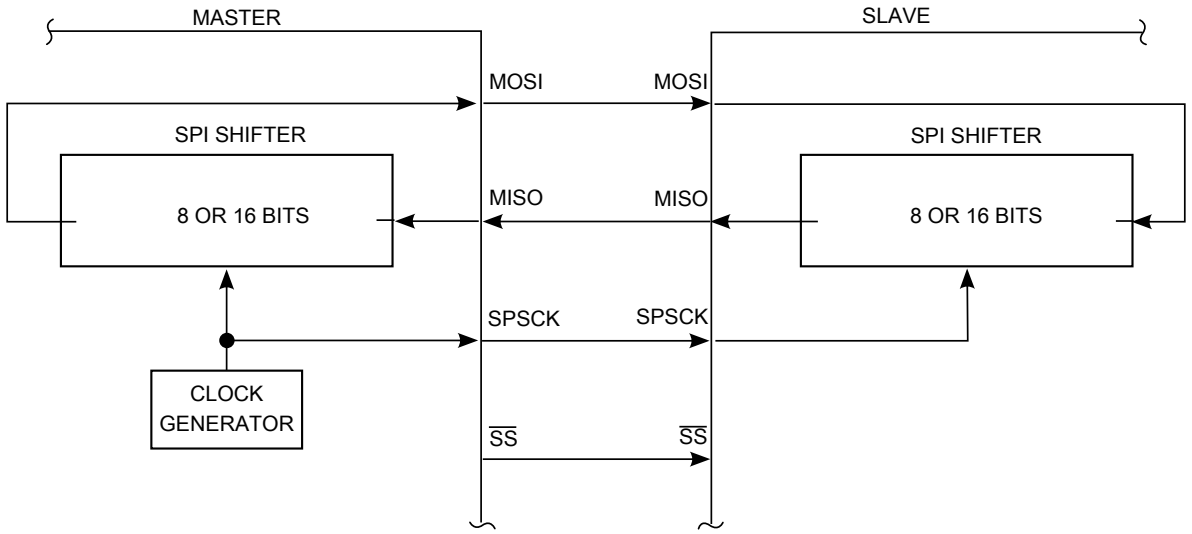


Figure 44-2. SPI system connections

44.2.3.2 SPI module block diagram

The following is a block diagram of the SPI module. The central element of the SPI is the SPI shift register. Data is written to the double-buffered transmitter (write to SPIx_DH:SPIx_DL) and gets transferred to the SPI Shift Register at the start of a data transfer. After shifting in 8 bits or 16 bits (as determined by the SPIMODE bit) of data, the data is transferred into the double-buffered receiver where it can be read from SPIx_DH:SPIx_DL. Pin multiplexing logic controls connections between MCU pins and the SPI module.

When the FIFO feature is supported: Additionally there is an 8-byte receive FIFO and an 8-byte transmit FIFO that (once enabled) provide features to allow fewer CPU interrupts to occur when transmitting/receiving high volume/high speed data. When FIFO mode is enabled, the SPI can still function in either 8-bit or 16-bit mode (as per SPIMODE bit) and three additional flags help monitor the FIFO status. Two of these flags can provide CPU interrupts.

When the SPI is configured as a master, the clock output is routed to the SPSCK pin, the shifter output is routed to MOSI, and the shifter input is routed from the MISO pin.

When the SPI is configured as a slave, the SPSCK pin is routed to the clock input of the SPI, the shifter output is routed to MISO, and the shifter input is routed from the MOSI pin.

In the external SPI system, simply connect all SPSCK pins to each other, all MISO pins together, and all MOSI pins together. Peripheral devices often use slightly different names for these pins.

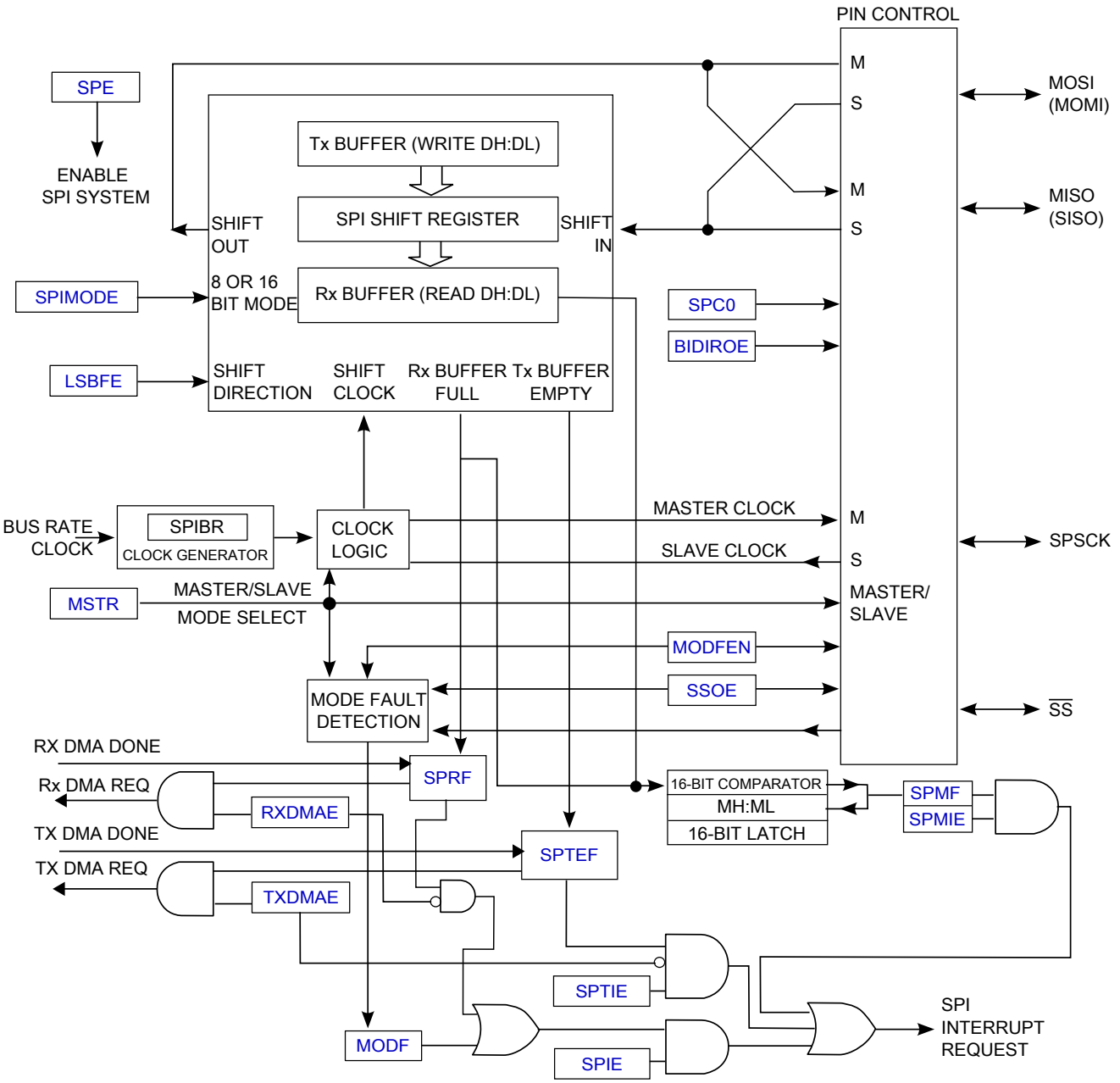


Figure 44-3. SPI module block diagram without FIFO

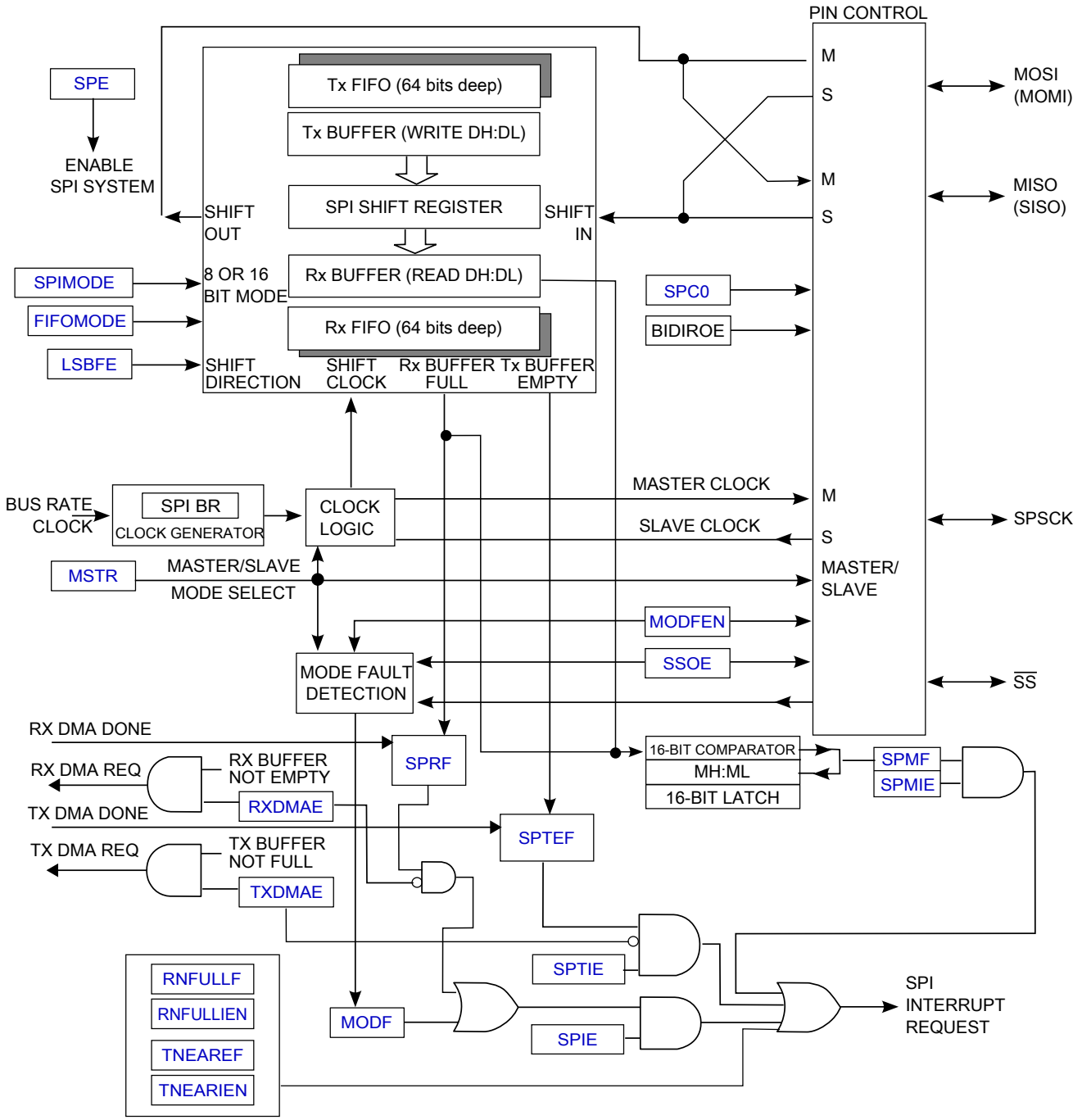


Figure 44-4. SPI Module Block Diagram with FIFO

44.3 External signal description

The SPI optionally shares four port pins. The function of these pins depends on the settings of SPI control bits. When the SPI is disabled (SPE = 0), these four pins revert to other functions that are not controlled by the SPI (based on chip configuration).

44.3.1 SPSCK — SPI Serial Clock

When the SPI is enabled as a slave, this pin is the serial clock input. When the SPI is enabled as a master, this pin is the serial clock output.

44.3.2 MOSI — Master Data Out, Slave Data In

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data output. When the SPI is enabled as a slave and SPC0 is 0, this pin is the serial data input. If SPC0 is 1 to select single-wire bidirectional mode, and master mode is selected, this pin becomes the bidirectional data I/O pin (MOMI). Also, the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE is 0) or an output (BIDIROE is 1). If SPC0 is 1 and slave mode is selected, this pin is not used by the SPI and reverts to other functions (based on chip configuration).

44.3.3 MISO — Master Data In, Slave Data Out

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data input. When the SPI is enabled as a slave and SPC0 is 0, this pin is the serial data output. If SPC0 is 1 to select single-wire bidirectional mode, and slave mode is selected, this pin becomes the bidirectional data I/O pin (SISO), and the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE is 0) or an output (BIDIROE is 1). If SPC0 is 1 and master mode is selected, this pin is not used by the SPI and reverts to other functions (based on chip configuration).

44.3.4 $\overline{\text{SS}}$ — Slave Select

When the SPI is enabled as a slave, this pin is the low-true slave select input. When the SPI is enabled as a master and mode fault enable is off (MODFEN is 0), this pin is not used by the SPI and reverts to other functions (based on chip configuration). When the SPI is enabled as a master and MODFEN is 1, the slave select output enable bit determines whether this pin acts as the mode fault input (SSOE is 0) or as the slave select output (SSOE is 1).

44.4 Memory map/register definition

The SPI has 8-bit registers to select SPI options, to control baud rate, to report SPI status, to hold an SPI data match value, and for transmit/receive data.

SPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_5000	SPI Status Register (SPI0_S)	8	R	20h	44.4.1/889
4007_5001	SPI Baud Rate Register (SPI0_BR)	8	R/W	00h	44.4.2/893
4007_5002	SPI Control Register 2 (SPI0_C2)	8	R/W	00h	44.4.3/894
4007_5003	SPI Control Register 1 (SPI0_C1)	8	R/W	04h	44.4.4/896
4007_5004	SPI Match Register low (SPI0_ML)	8	R/W	00h	44.4.5/897
4007_5005	SPI match register high (SPI0_MH)	8	R/W	00h	44.4.6/898
4007_5006	SPI Data Register low (SPI0_DL)	8	R/W	00h	44.4.7/898
4007_5007	SPI data register high (SPI0_DH)	8	R/W	00h	44.4.8/899
4007_500A	SPI clear interrupt register (SPI0_CI)	8	R/W	00h	44.4.9/900
4007_500B	SPI control register 3 (SPI0_C3)	8	R/W	00h	44.4.10/ 901
4007_6000	SPI Status Register (SPI1_S)	8	R	20h	44.4.1/889
4007_6001	SPI Baud Rate Register (SPI1_BR)	8	R/W	00h	44.4.2/893
4007_6002	SPI Control Register 2 (SPI1_C2)	8	R/W	00h	44.4.3/894
4007_6003	SPI Control Register 1 (SPI1_C1)	8	R/W	04h	44.4.4/896
4007_6004	SPI Match Register low (SPI1_ML)	8	R/W	00h	44.4.5/897
4007_6005	SPI match register high (SPI1_MH)	8	R/W	00h	44.4.6/898
4007_6006	SPI Data Register low (SPI1_DL)	8	R/W	00h	44.4.7/898
4007_6007	SPI data register high (SPI1_DH)	8	R/W	00h	44.4.8/899
4007_600A	SPI clear interrupt register (SPI1_CI)	8	R/W	00h	44.4.9/900
4007_600B	SPI control register 3 (SPI1_C3)	8	R/W	00h	44.4.10/ 901

44.4.1 SPI Status Register (SPIx_S)

This register contains read-only status bits.

NOTE

When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): Bits 3 through 0 are not implemented and always read 0.

When the FIFO is supported and enabled (FIFOMODE is 1): This register has four flags that provide mechanisms to support an 8-byte FIFO mode: RNFULLF, TNEARF, TXFULLF, and RFIFOEF. When the SPI is in 8-byte FIFO mode, the function of SPRF and SPTEF differs slightly from their function in the normal buffered modes, mainly regarding how these flags are cleared by the amount available in the transmit and receive FIFOs.

- The RNFULLF and TNEAREF help improve the efficiency of FIFO operation when transfering large amounts of data. These flags provide a "watermark" feature of the FIFOs to allow continuous transmissions of data when running at high speed.
- The RNFULLF can generate an interrupt if the RNFULLIEN bit in the C3 register is set, which allows the CPU to start emptying the receive FIFO without delaying the reception of subsequent bytes. The user can also determine if all data in the receive FIFO has been read by monitoring the RFIFOEF.
- The TNEAREF can generate an interrupt if the TNEARIEN bit in the C3 register is set, which allows the CPU to start filling the transmit FIFO before it is empty and thus to prevent breaks in SPI transmission.

NOTE

At an initial POR, the values of TNEAREF and RFIFOEF are 0. However, the status (S) register and both TX and RX FIFOs are reset due to a change of SPIMODE, FIFOMODE or SPE. If this type of reset occurs and FIFOMODE is 0, TNEAREF and RFIFOEF continue to reset to 0. If this type of reset occurs and FIFOMODE is 1, TNEAREF and RFIFOEF reset to 1.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	SPRF	SPMF	SPTEF	MODF	RNFULLF	TNEAREF	TXFULLF	RFIFOEF
Write		w1c						
Reset	0	0	1	0	0	0	0	0

SPIx_S field descriptions

Field	Description
7 SPRF	SPI Read Buffer Full Flag (when FIFO is not supported or not enabled) or SPI read FIFO FULL flag (when FIFO is supported and enabled) When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): SPRF is set at the completion of an SPI transfer to indicate that received data may be read from the SPI data (DH:DL) register. When the receive DMA request is disabled (RXDMAE is 0), SPRF is cleared by reading SPRF while it is set and then reading the SPI data register. When the receive DMA request is enabled (RXDMAE

Table continues on the next page...

SPIx_S field descriptions (continued)

Field	Description
	is 1), SPRF is automatically cleared when the DMA transfer for the receive DMA request is completed (RX DMA Done is asserted). When FIFOMODE is 1: This bit indicates the status of the read FIFO when FIFOMODE is enabled. The SPRF is set when the read FIFO has received 64 bits (4 words or 8 bytes) of data from the shifter and there have been no CPU reads of the SPI data (DH:DL) register. When the receive DMA request is disabled (RXDMAE is 0), SPRF is cleared by reading the SPI data register, resulting in the FIFO no longer being full, assuming another SPI message is not received. When the receive DMA request is enabled (RXDMAE is 1), SPRF is automatically cleared when the first DMA transfer for the receive DMA request is completed (RX DMA Done is asserted). 0 No data available in the receive data buffer (when FIFOMODE is not present or is 0) or Read FIFO is not full (when FIFOMODE is 1) 1 Data available in the receive data buffer (when FIFOMODE is not present or is 0) or Read FIFO is full (when FIFOMODE is 1)
6 SPMF	SPI Match Flag SPMF is set after SPRF is 1 when the value in the receive data buffer matches the value in the MH:ML registers. To clear the flag, read SPMF when it is set and then write a 1 to it. 0 Value in the receive data buffer does not match the value in the MH:ML registers 1 Value in the receive data buffer matches the value in the MH:ML registers
5 SPTEF	SPI Transmit Buffer Empty Flag (when FIFO is not supported or not enabled) or SPI transmit FIFO empty flag (when FIFO is supported and enabled) When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): This bit is set when the transmit data buffer is empty. When the transmit DMA request is disabled (TXDMAE is 0), SPTEF is cleared by reading the S register with SPTEF set and then writing a data value to the transmit buffer at DH:DL. The S register must be read with SPTEF set to 1 before writing data to the DH:DL register; otherwise, the DH:DL write is ignored. When the transmit DMA request is enabled (TXDMAE is 1), SPTEF is automatically cleared when the DMA transfer for the transmit DMA request is completed (TX DMA Done is asserted). SPTEF is automatically set when all data from the transmit buffer transfers into the transmit shift register. For an idle SPI, data written to DH:DL is transferred to the shifter almost immediately so that SPTEF is set within two bus cycles, allowing a second set of data to be queued into the transmit buffer. After completion of the transfer of the data in the shift register, the queued data from the transmit buffer automatically moves to the shifter, and SPTEF is set to indicate that room exists for new data in the transmit buffer. If no new data is waiting in the transmit buffer, SPTEF simply remains set and no data moves from the buffer to the shifter. When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): If a transfer does not stop, the last data that was transmitted is sent out again. When the FIFO is supported and enabled (FIFOMODE is 1): This bit provides the status of the FIFO rather than an 8-bit or a 16-bit buffer. This bit is set when the transmit FIFO is empty. When the transmit DMA request is disabled (TXDMAE is 0), SPTEF is cleared by writing a data value to the transmit FIFO at DH:DL. When the transmit DMA request is enabled (TXDMAE is 1), SPTEF is automatically cleared when the DMA transfer for the transmit DMA request is completed (TX DMA Done is asserted). SPTEF is automatically set when all data from the transmit FIFO transfers into the transmit shift register. For an idle SPI, data written to the DH:DL register is transferred to the shifter almost immediately, so that SPTEF is set within two bus cycles. A second write of data to the DH:DL register clears this SPTEF flag. After completion of the transfer of the data in the shift register, the queued data from the transmit FIFO automatically moves to the shifter, and SPTEF will be set only when all data written to the transmit FIFO has been transfered to the shifter. If no new data is waiting in the transmit FIFO, SPTEF simply remains set and no data moves from the buffer to the shifter.

Table continues on the next page...

SPIx_S field descriptions (continued)

Field	Description
	0 SPI transmit buffer not empty (when FIFOMODE is not present or is 0) or SPI FIFO not empty (when FIFOMODE is 1) 1 SPI transmit buffer empty (when FIFOMODE is not present or is 0) or SPI FIFO empty (when FIFOMODE is 1)
4 MODF	Master Mode Fault Flag MODF is set if the SPI is configured as a master and the slave select input goes low, indicating some other SPI device is also configured as a master. The $\overline{SS}$ pin acts as a mode fault error input only when C1[MSTR] is 1, C2[MODFEN] is 1, and C1[SSOE] is 0; otherwise, MODF will never be set. MODF is cleared by reading MODF while it is 1 and then writing to the SPI Control Register 1 (C1). 0 No mode fault error 1 Mode fault error detected
3 RNFULLF	Receive FIFO nearly full flag This flag is set when more than three 16-bit words or six 8-bit bytes of data remain in the receive FIFO, provided C3[RNFULLF_MARK] is 0, or when more than two 16-bit words or four 8-bit bytes of data remain in the receive FIFO, provided C3[RNFULLF_MARK] is 1. It has no function if FIFOMODE is not present or is 0. 0 Receive FIFO has received less than 48 bits (when C3[RNFULLF_MARK] is 0) or less than 32 bits (when C3[RNFULLF_MARK] is 1) 1 Receive FIFO has received data of an amount equal to or greater than 48 bits (when C3[RNFULLF_MARK] is 0) or 32 bits (when C3[RNFULLF_MARK] is 1)
2 TNEAREF	Transmit FIFO nearly empty flag This flag is set when only one 16-bit word or two 8-bit bytes of data remain in the transmit FIFO, provided C3[TNEAREF_MARK] is 0, or when only two 16-bit words or four 8-bit bytes of data remain in the transmit FIFO, provided C3[TNEAREF_MARK] is 1. If FIFOMODE is not enabled, ignore this bit. **NOTE:** At an initial POR, the values of TNEAREF and RFIFOEF are 0. However, the status (S) register and both TX and RX FIFOs are reset due to a change of SPIMODE, FIFOMODE or SPE. If this type of reset occurs and FIFOMODE is 0, TNEAREF and RFIFOEF continue to reset to 0. If this type of reset occurs and FIFOMODE is 1, TNEAREF and RFIFOEF reset to 1. 0 Transmit FIFO has more than 16 bits (when C3[TNEAREF_MARK] is 0) or more than 32 bits (when C3[TNEAREF_MARK] is 1) remaining to transmit 1 Transmit FIFO has an amount of data equal to or less than 16 bits (when C3[TNEAREF_MARK] is 0) or 32 bits (when C3[TNEAREF_MARK] is 1) remaining to transmit
1 TXFULLF	Transmit FIFO full flag This bit indicates the status of the transmit FIFO when FIFOMODE is enabled. This flag is set when there are 8 bytes in the transmit FIFO. If FIFOMODE is not enabled, ignore this bit. When FIFOMODE and DMA are both enabled, the inverted TXFULLF is used to trigger a DMA transfer. So when the transmit FIFO is not full, the DMA request is active, and remains active until the FIFO is full. 0 Transmit FIFO has less than 8 bytes 1 Transmit FIFO has 8 bytes of data
0 RFIFOEF	SPI read FIFO empty flag This bit indicates the status of the read FIFO when FIFOMODE is enabled. If FIFOMODE is not enabled, ignore this bit.

Table continues on the next page...

SPIx_S field descriptions (continued)

Field	Description
	When FIFOMODE and DMA are both enabled, the inverted RXIFOEF is used to trigger a DMA transfer. So when the receive FIFO is not empty, the DMA request is active, and remains active until the FIFO is empty. **NOTE:** At an initial POR, the values of TNEAREF and RFIFOEF are 0. However, the status (S) register and both TX and RX FIFOs are reset due to a change of SPIMODE, FIFOMODE or SPE. If this type of reset occurs and FIFOMODE is 0, TNEAREF and RFIFOEF continue to reset to 0. If this type of reset occurs and FIFOMODE is 1, TNEAREF and RFIFOEF reset to 1. 0 Read FIFO has data. Reads of the DH:DL registers in 16-bit mode or the DL register in 8-bit mode will empty the read FIFO. 1 Read FIFO is empty.

44.4.2 SPI Baud Rate Register (SPIx_BR)

Use this register to set the prescaler and bit rate divisor for an SPI master. This register may be read or written at any time.

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read	0	SPPR[2:0]			SPR[3:0]			
Write								
Reset	0	0	0	0	0	0	0	0

SPIx_BR field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 SPPR[2:0]	SPI Baud Rate Prescale Divisor This 3-bit field selects one of eight divisors for the SPI baud rate prescaler. The input to this prescaler is the SPI module clock. The output of this prescaler drives the input of the SPI baud rate divider. Refer to the description of “SPI Baud Rate Generation” for details. 000 Baud rate prescaler divisor is 1. 001 Baud rate prescaler divisor is 2. 010 Baud rate prescaler divisor is 3. 011 Baud rate prescaler divisor is 4. 100 Baud rate prescaler divisor is 5. 101 Baud rate prescaler divisor is 6. 110 Baud rate prescaler divisor is 7. 111 Baud rate prescaler divisor is 8.
SPR[3:0]	SPI Baud Rate Divisor

Table continues on the next page...

SPIx_BR field descriptions (continued)

Field	Description
	This 4-bit field selects one of nine divisors for the SPI baud rate divider. The input to this divider comes from the SPI baud rate prescaler. Refer to the description of “SPI Baud Rate Generation” for details. 0000 Baud rate divisor is 2. 0001 Baud rate divisor is 4. 0010 Baud rate divisor is 8. 0011 Baud rate divisor is 16. 0100 Baud rate divisor is 32. 0101 Baud rate divisor is 64. 0110 Baud rate divisor is 128. 0111 Baud rate divisor is 256. 1000 Baud rate divisor is 512. All others Reserved

44.4.3 SPI Control Register 2 (SPIx_C2)

This read/write register is used to control optional features of the SPI system.

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SPMIE	SPIMODE	TXDMAE	MODFEN	BIDIROE	RXDMAE	SPISWAI	SPC0
Reset	0	0	0	0	0	0	0	0

SPIx_C2 field descriptions

Field	Description
7 SPMIE	SPI Match Interrupt Enable This is the interrupt enable bit for the SPI receive data buffer hardware match (SPMF) function. 0 Interrupts from SPMF inhibited (use polling) 1 When SPMF is 1, requests a hardware interrupt
6 SPIMODE	SPI 8-bit or 16-bit mode This bit allows the user to select either an 8-bit or 16-bit SPI data transmission length. In master mode, a change of this bit aborts a transmission in progress, forces the SPI system into an idle state, and resets all status bits in the S register. Refer to the description of “Data Transmission Length” for details. 0 8-bit SPI shift register, match register, and buffers 1 16-bit SPI shift register, match register, and buffers
5 TXDMAE	Transmit DMA enable This bit enables a transmit DMA request. When this bit is set to 1, a transmit DMA request is asserted when both SPTEF and SPE are set, and the interrupt from SPTEF is disabled.

Table continues on the next page...

SPIx_C2 field descriptions (continued)

Field	Description
	0 DMA request for transmit is disabled and interrupt from SPTEF is allowed 1 DMA request for transmit is enabled and interrupt from SPTEF is disabled
4 MODFEN	Master Mode-Fault Function Enable When the SPI is configured for slave mode, this bit has no meaning or effect. (The $\overline{SS}$ pin is the slave select input.) In master mode, this bit determines how the $\overline{SS}$ pin is used. For details, refer to the description of the SSOE bit in the C1 register. 0 Mode fault function disabled, master $\overline{SS}$ pin reverts to general-purpose I/O not controlled by SPI 1 Mode fault function enabled, master $\overline{SS}$ pin acts as the mode fault input or the slave select output
3 BIDIROE	Bidirectional Mode Output Enable When bidirectional mode is enabled because SPI pin control 0 (SPC0) is set to 1, BIDIROE determines whether the SPI data output driver is enabled to the single bidirectional SPI I/O pin. Depending on whether the SPI is configured as a master or a slave, it uses the MOSI (MOMI) or MISO (SISO) pin, respectively, as the single SPI data I/O pin. When SPC0 is 0, BIDIROE has no meaning or effect. 0 Output driver disabled so SPI data I/O pin acts as an input 1 SPI I/O pin enabled as an output
2 RXDMAE	Receive DMA enable This is the enable bit for a receive DMA request. When this bit is set to 1, a receive DMA request is asserted when both SPRF and SPE are set, and the interrupt from SPRF is disabled. 0 DMA request for receive is disabled and interrupt from SPRF is allowed 1 DMA request for receive is enabled and interrupt from SPRF is disabled
1 SPISWAI	SPI Stop in Wait Mode This bit is used for power conservation while the device is in Wait mode. 0 SPI clocks continue to operate in Wait mode. 1 SPI clocks stop when the MCU enters Wait mode.
0 SPC0	SPI Pin Control 0 Enables bidirectional pin configurations. 0 SPI uses separate pins for data input and data output (pin mode is normal). In master mode of operation: MISO is master in and MOSI is master out. In slave mode of operation: MISO is slave out and MOSI is slave in. 1 SPI configured for single-wire bidirectional operation (pin mode is bidirectional). In master mode of operation: MISO is not used by SPI; MOSI is master in when BIDIROE is 0 or master I/O when BIDIROE is 1. In slave mode of operation: MISO is slave in when BIDIROE is 0 or slave I/O when BIDIROE is 1; MOSI is not used by SPI.

44.4.4 SPI Control Register 1 (SPIx_C1)

This read/write register includes the SPI enable control, interrupt enables, and configuration options.

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset	0	0	0	0	0	1	0	0

SPIx_C1 field descriptions

Field	Description
7 SPIE	SPI Interrupt Enable: for SPRF and MODF (when FIFO is not supported or not enabled) or for read FIFO (when FIFO is supported and enabled) When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): Enables the interrupt for SPI receive buffer full (SPRF) and mode fault (MODF) events. When the FIFO is supported and enabled (FIFOMODE is 1): This bit enables the SPI to interrupt the CPU when the receive FIFO is full. An interrupt occurs when the SPRF bit is set or the MODF bit is set. 0 Interrupts from SPRF and MODF are inhibited—use polling (when FIFOMODE is not present or is 0) or Read FIFO Full Interrupts are disabled (when FIFOMODE is 1) 1 Request a hardware interrupt when SPRF or MODF is 1 (when FIFOMODE is not present or is 0) or Read FIFO Full Interrupts are enabled (when FIFOMODE is 1)
6 SPE	SPI System Enable Enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, the SPI is disabled and forced into an idle state, and all status bits in the S register are reset. 0 SPI system inactive 1 SPI system enabled
5 SPTIE	SPI Transmit Interrupt Enable When the FIFO is not supported or not enabled (FIFOMODE is not present or is 0): This is the interrupt enable bit for SPI transmit buffer empty (SPTEF). An interrupt occurs when the SPI transmit buffer is empty (SPTEF is set). When the FIFO is supported and enabled (FIFOMODE is 1): This is the interrupt enable bit for SPI transmit FIFO empty (SPTEF). An interrupt occurs when the SPI transmit FIFO is empty (SPTEF is set). 0 Interrupts from SPTEF inhibited (use polling) 1 When SPTEF is 1, hardware interrupt requested
4 MSTR	Master/Slave Mode Select Selects master or slave mode operation. 0 SPI module configured as a slave SPI device 1 SPI module configured as a master SPI device

Table continues on the next page...

SPIx_C1 field descriptions (continued)

Field	Description
3 CPOL	Clock Polarity Selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. This bit effectively places an inverter in series with the clock signal either from a master SPI device or to a slave SPI device. Refer to the description of “SPI Clock Formats” for details. 0 Active-high SPI clock (idles low) 1 Active-low SPI clock (idles high)
2 CPHA	Clock Phase Selects one of two clock formats for different kinds of synchronous serial peripheral devices. Refer to the description of “SPI Clock Formats” for details. 0 First edge on SPSCK occurs at the middle of the first cycle of a data transfer. 1 First edge on SPSCK occurs at the start of the first cycle of a data transfer.
1 SSOE	Slave Select Output Enable This bit is used in combination with the Mode Fault Enable (MODFEN) field in the C2 register and the Master/Slave (MSTR) control bit to determine the function of the $\overline{SS}$ pin. 0 When C2[MODFEN] is 0: In master mode, $\overline{SS}$ pin function is general-purpose I/O (not SPI). In slave mode, $\overline{SS}$ pin function is slave select input. When C2[MODFEN] is 1: In master mode, $\overline{SS}$ pin function is $\overline{SS}$ input for mode fault. In slave mode, $\overline{SS}$ pin function is slave select input. 1 When C2[MODFEN] is 0: In master mode, $\overline{SS}$ pin function is general-purpose I/O (not SPI). In slave mode, $\overline{SS}$ pin function is slave select input. When C2[MODFEN] is 1: In master mode, $\overline{SS}$ pin function is automatic $\overline{SS}$ output. In slave mode: $\overline{SS}$ pin function is slave select input.
0 LSBFE	LSB First (shifter direction) This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7 (or bit 15 in 16-bit mode). 0 SPI serial data transfers start with the most significant bit. 1 SPI serial data transfers start with the least significant bit.

44.4.5 SPI Match Register low (SPIx_ML)

This register, together with the MH register, contains the hardware compare value. When the value received in the SPI receive data buffer equals this hardware compare value, the SPI Match Flag in the S register (S[SPMF]) sets.

In 8-bit mode, only the ML register is available. Reads of the MH register return all zeros. Writes to the MH register are ignored.

In 16-bit mode, reading either byte (the MH or ML register) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. Writing to either byte (the MH or ML register) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent value into the SPI match registers.

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read Write	Bits[7:0]							
Reset	0	0	0	0	0	0	0	0

SPIx_ML field descriptions

Field	Description
Bits[7:0]	Hardware compare value (low byte)

44.4.6 SPI match register high (SPIx_MH)

Refer to the description of the ML register.

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read Write	Bits[15:8]							
Reset	0	0	0	0	0	0	0	0

SPIx_MH field descriptions

Field	Description
Bits[15:8]	Hardware compare value (high byte)

44.4.7 SPI Data Register low (SPIx_DL)

This register, together with the DH register, is both the input and output register for SPI data. A write to the registers writes to the transmit data buffer, allowing data to be queued and transmitted.

When the SPI is configured as a master, data queued in the transmit data buffer is transmitted immediately after the previous transmission has completed.

The SPTEF bit in the S register indicates when the transmit data buffer is ready to accept new data. When the transmit DMA request is disabled (TXDMAE is 0): The S register must be read when S[SPTEF] is set before writing to the SPI data registers; otherwise, the

write is ignored. When the transmit DMA request is enabled (TXDMAE is 1) when S[SPTEF] is set, the SPI data registers can be written automatically by DMA without reading the S register first.

Data may be read from the SPI data registers any time after S[SPRF] is set and before another transfer is finished. Failure to read the data out of the receive data buffer before a new transfer ends causes a receive overrun condition, and the data from the new transfer is lost. The new data is lost because the receive buffer still held the previous character and was not ready to accept the new data. There is no indication for a receive overrun condition, so the application system designer must ensure that previous data has been read from the receive buffer before a new transfer is initiated.

In 8-bit mode, only the DL register is available. Reads of the DH register return all zeros. Writes to the DH register are ignored.

In 16-bit mode, reading either byte (the DH or DL register) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. Writing to either byte (the DH or DL register) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the transmit data buffer.

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read Write	Bits[7:0]							
Reset	0	0	0	0	0	0	0	0

SPIx_DL field descriptions

Field	Description
Bits[7:0]	Data (low byte)

44.4.8 SPI data register high (SPIx_DH)

Refer to the description of the DL register.

Address: Base address + 7h offset

Bit	7	6	5	4	3	2	1	0
Read Write	Bits[15:8]							
Reset	0	0	0	0	0	0	0	0

SPIx_DH field descriptions

Field	Description
Bits[15:8]	Data (high byte)

44.4.9 SPI clear interrupt register (SPIx_CI)

This register applies only for an instance of the SPI module that supports the FIFO feature.

The register has four bits dedicated to clearing the interrupts. Writing 1 to these bits clears the corresponding interrupts if the INTCLR bit in the C3 register is 1. Reading these bits always returns 0.

This register also has two read-only bits to indicate the transmit FIFO and receive FIFO overrun conditions. When the receive FIFO is full and data is received, RXFOF is set. Similarily, when the transmit FIFO is full and a write to the data register occurs, TXFOF is set. These flags are cleared when the CI register is read while the flags are set.

The register has two more read-only bits to indicate the error flags. These flags are set when, due to some spurious reason, entries in the FIFO total more than 64 bits of data. At this point, all the flags in the status register are reset, and entries in the FIFO are flushed with the corresponding error flags set. These flags are cleared when the CI register is read while the flags are set.

Address: Base address + Ah offset

Bit	7	6	5	4	3	2	1	0
Read	TXFERR	RXFERR	TXFOF	RXFOF	0	0	0	0
Write					TNEAREFCI	RNFULLFCI	SPTEFCI	SPRFCI
Reset	0	0	0	0	0	0	0	0

SPIx_CI field descriptions

Field	Description
7 TXFERR	Transmit FIFO error flag This flag indicates that a transmit FIFO error occurred because entries in the FIFO total more than 64 bits of data. 0 No transmit FIFO error occurred 1 A transmit FIFO error occurred
6 RXFERR	Receive FIFO error flag This flag indicates that a receive FIFO error occurred because entries in the FIFO total more than 64 bits of data. 0 No receive FIFO error occurred 1 A receive FIFO error occurred
5 TXFOF	Transmit FIFO overflow flag This flag indicates that a transmit FIFO overflow condition has occurred.

Table continues on the next page...

SPIx_CI field descriptions (continued)

Field	Description
	0 Transmit FIFO overflow condition has not occurred 1 Transmit FIFO overflow condition occurred
4 RXFOF	Receive FIFO overflow flag This flag indicates that a receive FIFO overflow condition has occurred. 0 Receive FIFO overflow condition has not occurred 1 Receive FIFO overflow condition occurred
3 TNEAREFCI	Transmit FIFO nearly empty flag clear interrupt Writing 1 to this bit clears the TNEAREF interrupt provided that C3[3] is set.
2 RNFULLFCI	Receive FIFO nearly full flag clear interrupt Writing 1 to this bit clears the RNFULLF interrupt provided that C3[3] is set.
1 SPTEFCI	Transmit FIFO empty flag clear interrupt Writing 1 to this bit clears the SPTEF interrupt provided that C3[3] is set.
0 SPRFCI	Receive FIFO full flag clear interrupt Writing 1 to this bit clears the SPRF interrupt provided that C3[3] is set.

44.4.10 SPI control register 3 (SPIx_C3)

This register introduces a 64-bit FIFO function on both transmit and receive buffers. It applies only for an instance of the SPI module that supports the FIFO feature.

FIFO mode is enabled by setting the FIFOMODE bit to 1. A write to this register occurs only when it sets the FIFOMODE bit to 1.

Using this FIFO feature allows the SPI to provide high speed transfers of large amounts of data without consuming large amounts of the CPU bandwidth.

Enabling this FIFO function affects the behavior of some of the read/write buffer flags in the S register as follows:

- When the receive FIFO has data in it, S[RFIFOEF] is 0. As a result:
  - If C2[RXDMAE] is 1, RFIFOEF_b generates a receive DMA request. The DMA request remains active until RFIFOEF is set to 1, indicating the receive buffer is empty.
- If C2[RXDMAE] is 0 and C1[SPIE] is 1, SPRF interrupts the CPU.
- When the transmit FIFO is not full, S[TXFULLF] is 0. As a result:
  - If C2[TXDMAE] is 1, TXFULLF_b generates a transmit DMA request. The DMA request remains active until TXFULLF is set to 1, indicating the transmit FIFO is full.
- If C2[TXDMAE] is 0 and C1[SPTIE] is 1, SPTEF interrupts the CPU.

Two interrupt enable bits, TNEARIEN and RNFULLIEN, provide CPU interrupts based on the "watermark" feature of the TNEARF and RNFULLF flags of the S register.

Address: Base address + Bh offset

Bit	7	6	5	4	3	2	1	0
Read	0		TNEAREF_MARK	RNFULLF_MARK	INTCLR	TNEARIEN	RNFULLIEN	FIFOMODE
Write								
Reset	0	0	0	0	0	0	0	0

SPIx_C3 field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 TNEAREF_MARK	Transmit FIFO nearly empty watermark This bit selects the mark after which the TNEAREF flag is asserted. 0 TNEAREF is set when the transmit FIFO has 16 bits or less 1 TNEAREF is set when the transmit FIFO has 32 bits or less
4 RNFULLF_MARK	Receive FIFO nearly full watermark This bit selects the mark after which the RNFULLF flag is asserted. 0 RNFULLF is set when the receive FIFO has 48 bits or more 1 RNFULLF is set when the receive FIFO has 32 bits or more
3 INTCLR	Interrupt clearing mechanism select This bit selects the mechanism by which the SPRF, SPTEF, TNEAREF, and RNFULLF interrupts are cleared. 0 These interrupts are cleared when the corresponding flags are cleared depending on the state of the FIFOs 1 These interrupts are cleared by writing the corresponding bits in the CI register
2 TNEARIEN	Transmit FIFO nearly empty interrupt enable Writing 1 to this bit enables the SPI to interrupt the CPU when the TNEAREF flag is set. This bit is ignored and has no function if the FIFOMODE bit is 0. 0 No interrupt upon TNEAREF being set 1 Enable interrupts upon TNEAREF being set
1 RNFULLIEN	Receive FIFO nearly full interrupt enable Writing 1 to this bit enables the SPI to interrupt the CPU when the RNFULLF flag is set. This bit is ignored and has no function if the FIFOMODE bit is 0. 0 No interrupt upon RNFULLF being set 1 Enable interrupts upon RNFULLF being set
0 FIFOMODE	FIFO mode enable This bit enables the SPI to use a 64-bit FIFO (8 bytes or four 16-bit words) for both transmit and receive buffers. 0 FIFO mode disabled 1 FIFO mode enabled

44.5 Functional description

This section provides the functional description of the module.

44.5.1 General

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI Control Register 1. While C1[SPE] is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select (SS)
- Serial clock (SPSCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

An SPI transfer is initiated in the master SPI device by reading the SPI status register (SPIx_S) when S[SPTEF] = 1 and then writing data to the transmit data buffer (write to SPIx_DH:SPIx_DL). When a transfer is complete, received data is moved into the receive data buffer. The SPIx_DH:SPIx_DL registers act as the SPI receive data buffer for reads and as the SPI transmit data buffer for writes.

The Clock Phase Control (CPHA) and Clock Polarity Control (CPOL) bits in the SPI Control Register 1 (SPIx_C1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. C1[CPHA] is used to accommodate two fundamentally different protocols by sampling data on odd numbered SPSCK edges or on even numbered SPSCK edges.

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI Control Register 1 is set, master mode is selected; when C1[MSTR] is clear, slave mode is selected.

44.5.2 Master mode

The SPI operates in master mode when C1[MSTR] is set. Only a master SPI module can initiate transmissions. A transmission begins by reading the SPIx_S register while S[SPTEF] = 1 and writing to the master SPI data registers. If the shift register is empty, the byte immediately transfers to the shift register. The data begins shifting out on the MOSI pin under the control of the serial clock.

- SPSCK
  - The SPR3, SPR2, SPR1, and SPR0 baud rate selection bits in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI Baud Rate register control the baud rate generator and determine the speed of the transmission. The SPSCK pin is the SPI clock output. Through the SPSCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.
- MOSI, MISO pin
  - In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{SS}$ pin
  - If C2[MODFEN] and C1[SSOE] are set, the SS pin is configured as slave select output. The SS output becomes low during each transmission and is high when the SPI is in idle state. If C2[MODFEN] is set and C1[SSOE] is cleared, the $\overline{SS}$ pin is configured as input for detecting mode fault error. If the SS input becomes low this indicates a mode fault error where another master tries to drive the MOSI and SPSCK lines. In this case, the SPI immediately switches to slave mode by clearing C1[MSTR] and also disables the slave output buffer MISO (or SISO in bidirectional mode). As a result, all outputs are disabled, and SPSCK, MOSI and MISO are inputs. If a transmission is in progress when the mode fault occurs, the transmission is aborted and the SPI is forced into idle state. This mode fault error also sets the Mode Fault (MODF) flag in the SPI Status Register (SPIx_S). If the SPI Interrupt Enable bit (SPIE) is set when S[MODF] gets set, then an SPI interrupt sequence is also requested. When a write to the SPI Data Register in the master occurs, there is a half SPSCK-cycle delay. After the delay, SPSCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI Control Register 1 (see SPI clock formats).

Note

A change of C1[CPOL], C1[CPHA], C1[SSOE], C1[LSBFE], C2[MODFEN], C2[SPC0], C2[BIDIROE] with C2[SPC0] set, SPIMODE, FIFOMODE, SPPR2-SPPR0 and SPR3-SPR0 in master mode abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master has to ensure that the remote slave is set back to idle state.

44.5.3 Slave mode

The SPI operates in slave mode when the MSTR bit in SPI Control Register 1 is clear.

- SPSCK

  In slave mode, SPSCK is the SPI clock input from the master.

- MISO, MOSI pin

  In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI Control Register 2.

- SS pin

  The SS pin is the slave select input. Before a data transmission occurs, the SS pin of the slave SPI must be low. SS must remain low until the transmission is complete. If SS goes high, the SPI is forced into an idle state.

  The SS input also controls the serial data output pin. If SS is high (not selected), the serial data output pin is high impedance. If SS is low, the first bit in the SPI Data Register is driven out of the serial data output pin. Also, if the slave is not selected (SS is high), then the SPSCK input is ignored and no internal shifting of the SPI shift register occurs.

  Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

Note

When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI Control Register 1 is clear, odd numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

If C1[CPHA] is set, even numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on C1[LSBFE].

When C1[CPHA] is set, the first edge is used to get the first data bit onto the serial data output pin. When C1[CPHA] is clear and the SS input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth (SPIMODE = 0) or sixteenth (SPIMODE = 1) shift, the transfer is considered complete and the received data is transferred into the SPI Data register. To indicate transfer is complete, the SPRF flag in the SPI Status Register is set.

Note

A change of the bits FIFOMODE,SPIMODE, C2[BIDIROE] with C2[SPC0] set, C1[CPOL], C1[CPHA], C1[SSOE], C1[LSBFE], C2[MODFEN], and C2[SPC0] in slave mode will corrupt a transmission in progress and must be avoided.

44.5.4 SPI FIFO Mode

When the FIFO feature is supported: The SPI works in FIFO mode when the C3[FIFOMODE] bit is set. When the module is in FIFO mode, the SPI RX buffer and SPI TX buffer are replaced by an 8-byte-deep FIFO, as the following figures show.

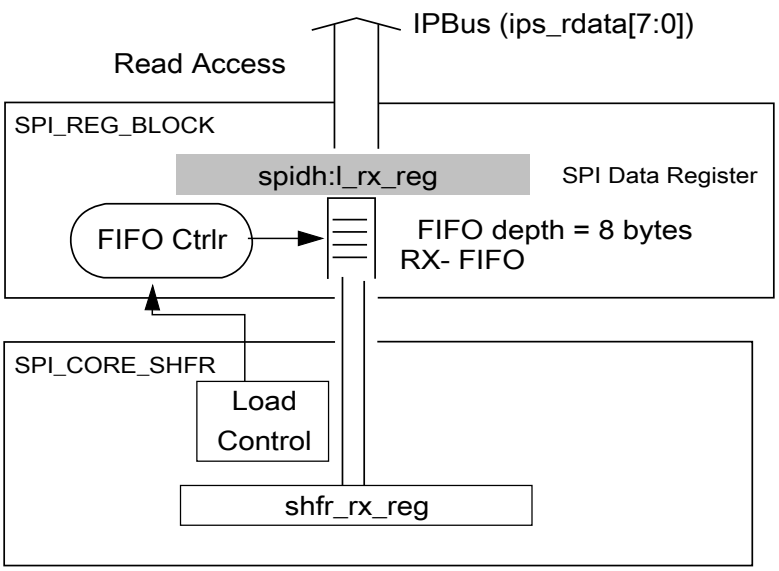


Figure 44-5. SPIH:L read side structural overview in FIFO mode

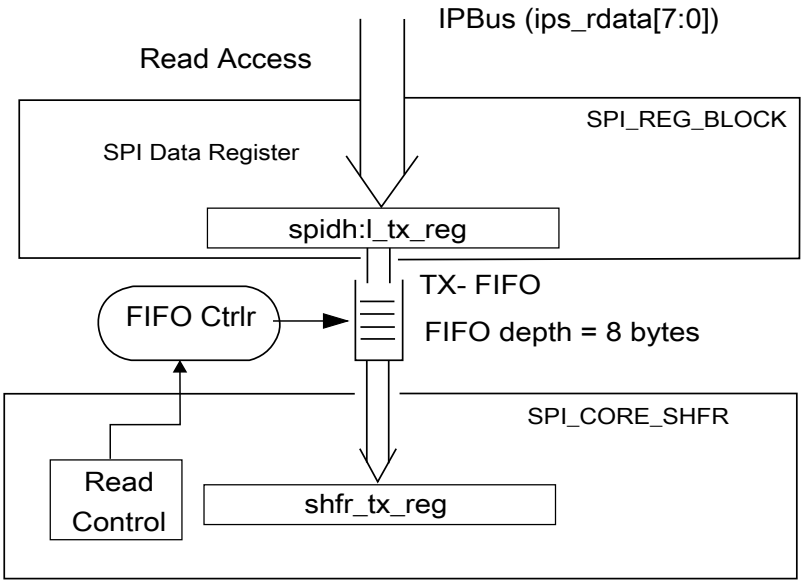


Figure 44-6. SPIH:L write side structural overview in FIFO mode

44.5.5 SPI Transmission by DMA

SPI supports both Transmit and Receive by DMA. The basic flow of SPI transmission by DMA is as below.

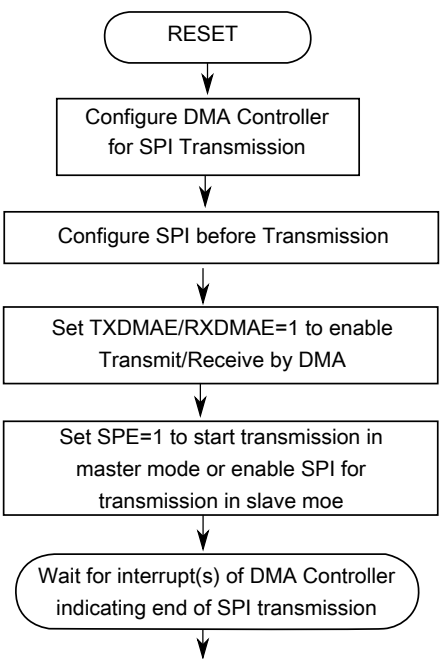


Figure 44-7. Basic Flow of SPI Transmission by DMA

44.5.5.1 Transmit by DMA

Transmit by DMA is supported only when TXDMAE is set. A transmit DMA request is asserted when both SPE and SPTEF are set. Then the on-chip DMA controller detects this request and transfers data from memory into the SPI data register. After that, TX DMA DONE is asserted to clear SPTEF automatically. This process repeats until all data for transmission (the number is decided by the configuration register[s] of the DMA controller) is sent.

When the FIFO feature is supported: In FIFO mode (FIFOMODE=1) and when a data length of 8 bits is selected (SPIMODE=0), the DMA transfer for one transmit DMA request can write more than 1 byte (up to 8 bytes) to the DL register because the TX FIFO can store 8 bytes of transmit data. In FIFO mode (FIFOMODE=1) and when a data length of 16 bits is selected (SPIMODE=1), the DMA transfer for one transmit DMA request can write more than 1 word (up to 4 words) to the DH:DL registers because the TX FIFO can store 4 words of transmit data. A larger number of bytes or words transferred from memory to the SPI data register for each transmit DMA request results in a lower total number of transmit DMA requests.

When FIFOMODE is 0: Cycle Steal (DMA_DCRn[CS] = 1) should be enabled when using the DMA controller to transfer data from memory to the SPI data register. The DMA performs a single data transfer per DMA request in cycle steal mode. Therefore, a single byte/word is written to the SPI data register from memory and transmitted by the SPI module for each DMA request, as long as the BCR value is greater than zero (DMA_DSR_BCRn[BCR] > 0). Once the BCR has reached zero, software must reconfigure the DMA controller if more data is to be transmitted. If a configuration error occurs (DMA_DSR_BCRn[CE] = 1) when the BCR is equal to 0, software must:

- disable peripheral requests when the BCR is equal to 0,
- perform 16-bit transfers (SPIMODE = 1), or
- decrease the SPI baud rate.

Software can disable peripheral requests by setting DMA_DCRn[D_REQ] = 1 when initializing the DMA controller, or by clearing DMA_DCRn[ERQ] once the BCR is equal to zero. Also, to continue transmitting data software must re-enable peripheral requests (DMA_DCRn[ERQ] = 1) after reconfiguring the DMA controller.

44.5.5.2 Receive by DMA

Receive by DMA is supported only when RXDMAE is set. A receive DMA request is asserted when both SPE and SPRF are set. Then the on-chip DMA controller detects this request and transfers data from the SPI data register into memory. After that, RX DMA DONE is asserted to clear SPRF automatically. This process repeats until all data to be

received (the number is decided by configuration register[s] of the DMA controller) is received or no receive DMA request is generated again because the SPI transmission is finished.

When the FIFO feature is supported: In FIFO mode (FIFOMODE=1) and when a data length of 8 bits is selected (SPIMODE=0), the DMA transfer for one receive DMA request can read more than 1 byte (up to 8 bytes) from the SPI data register because the RX FIFO can hold 8 bytes. In FIFO mode (FIFOMODE=1) and when a data length of 16 bits is selected (SPIMODE=1), the DMA transfer for one receive DMA request can read more than 1 word (up to 4 words) from the DH:DL registers because the RX FIFO can hold 4 words. A larger number of bytes or words transferred from the SPI data register to memory for one receive DMA request results in a lower total number of receive DMA requests.

44.5.6 Data Transmission Length

The SPI can support data lengths of 8 or 16 bits. The length can be configured with the SPIMODE bit in the SPIx_C2 register.

In 8-bit mode (SPIMODE = 0), the SPI Data Register is comprised of one byte: SPIx_DL. The SPI Match Register is also comprised of only one byte: SPIx_ML. Reads of SPIx_DH and SPIx_MH will return zero. Writes to SPIx_DH and SPIx_MH will be ignored.

In 16-bit mode (SPIMODE = 1), the SPI Data Register is comprised of two bytes: SPIx_DH and SPIx_DL. Reading either byte (SPIx_DH or SPIx_DL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. Writing to either byte (SPIx_DH or SPIx_DL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the transmit data buffer.

In 16-bit mode, the SPI Match Register is also comprised of two bytes: SPIx_MH and SPIx_ML. There is no buffer mechanism for the reading of SPIxMH and SPIxML since they can only be changed by writing at CPU side. Writing to either byte (SPIx_MH or SPIx_ML) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the SPI Match Register.

Any switching between 8- and 16-bit data transmission length (controlled by SPIMODE bit) in master mode will abort a transmission in progress, force the SPI system into idle state, and reset all status bits in the SPIx_S register. To initiate a transfer after writing to SPIMODE, the SPIx_S register must be read with SPTEF = 1, and data must be written to SPIx_DH:SPIx_DL in 16-bit mode (SPIMODE = 1) or SPIx_DL in 8-bit mode (SPIMODE = 0).

In slave mode, user software should write to SPIMODE only once to prevent corrupting a transmission in progress.

Note

Data can be lost if the data length is not the same for both master and slave devices.

44.5.7 SPI clock formats

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the SPI system has a Clock Polarity (CPOL) bit and a Clock Phase (CPHA) control bit in the Control Register 1 to select one of four clock formats for data transfers. C1[CPOL] selectively inserts an inverter in series with the clock. C1[CPHA] chooses between two different clock phase relationships between the clock and data.

The following figure shows the clock formats when SPIMODE = 0 (8-bit mode) and CPHA = 1. At the top of the figure, the eight bit times are shown for reference with bit 1 starting at the first SPSCK edge and bit 8 ending one-half SPSCK cycle after the eighth SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in C1[CPOL]. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The $\overline{SS}$ OUT waveform applies to the slave select output from a master (provided C2[MODFEN] and C1[SSOE] = 1). The master $\overline{SS}$ output goes to active low one-half SPSCK cycle before the start of the transfer and goes back high at the end of the eighth bit time of the transfer. The $\overline{SS}$ IN waveform applies to the slave select input of a slave.

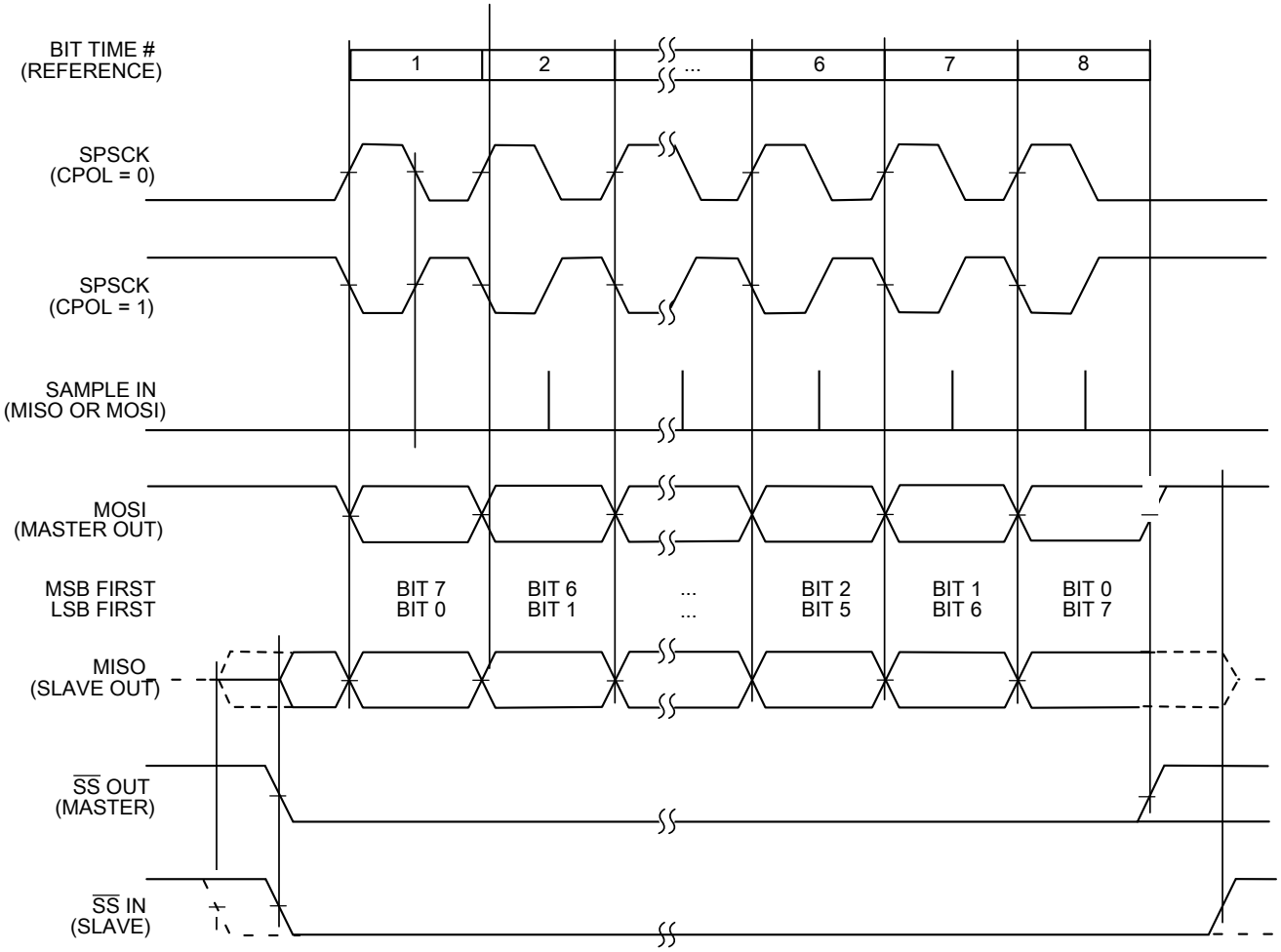


Figure 44-8. SPI clock formats (CPHA = 1)

When C1[CPHA] = 1, the slave begins to drive its MISO output when $\overline{SS}$ goes to active low, but the data is not defined until the first SPSCK edge. The first SPSCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output of the slave. The next SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the third SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled, and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively.

When C1[CPHA] = 1, the slave's $\overline{SS}$ input is not required to go to its inactive high level between transfers. In this clock format, a back-to-back transmission can occur, as follows:

1. A transmission is in progress.
2. A new data byte is written to the transmit buffer before the in-progress transmission is complete.
3. When the in-progress transmission is complete, the new, ready data byte is transmitted immediately.

Between these two successive transmissions, no pause is inserted; the $\overline{SS}$ pin remains low.

The following figure shows the clock formats when SPIMODE = 0 and C1[CPHA] = 0. At the top of the figure, the eight bit times are shown for reference with bit 1 starting as the slave is selected (SS IN goes low), and bit 8 ends at the last SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The SS OUT waveform applies to the slave select output from a master (provided C2[MODFEN] and C1[SSOE] = 1). The master SS output goes to active low at the start of the first bit time of the transfer and goes back high one-half SPSCK cycle after the end of the eighth bit time of the transfer. The SS IN waveform applies to the slave select input of a slave.

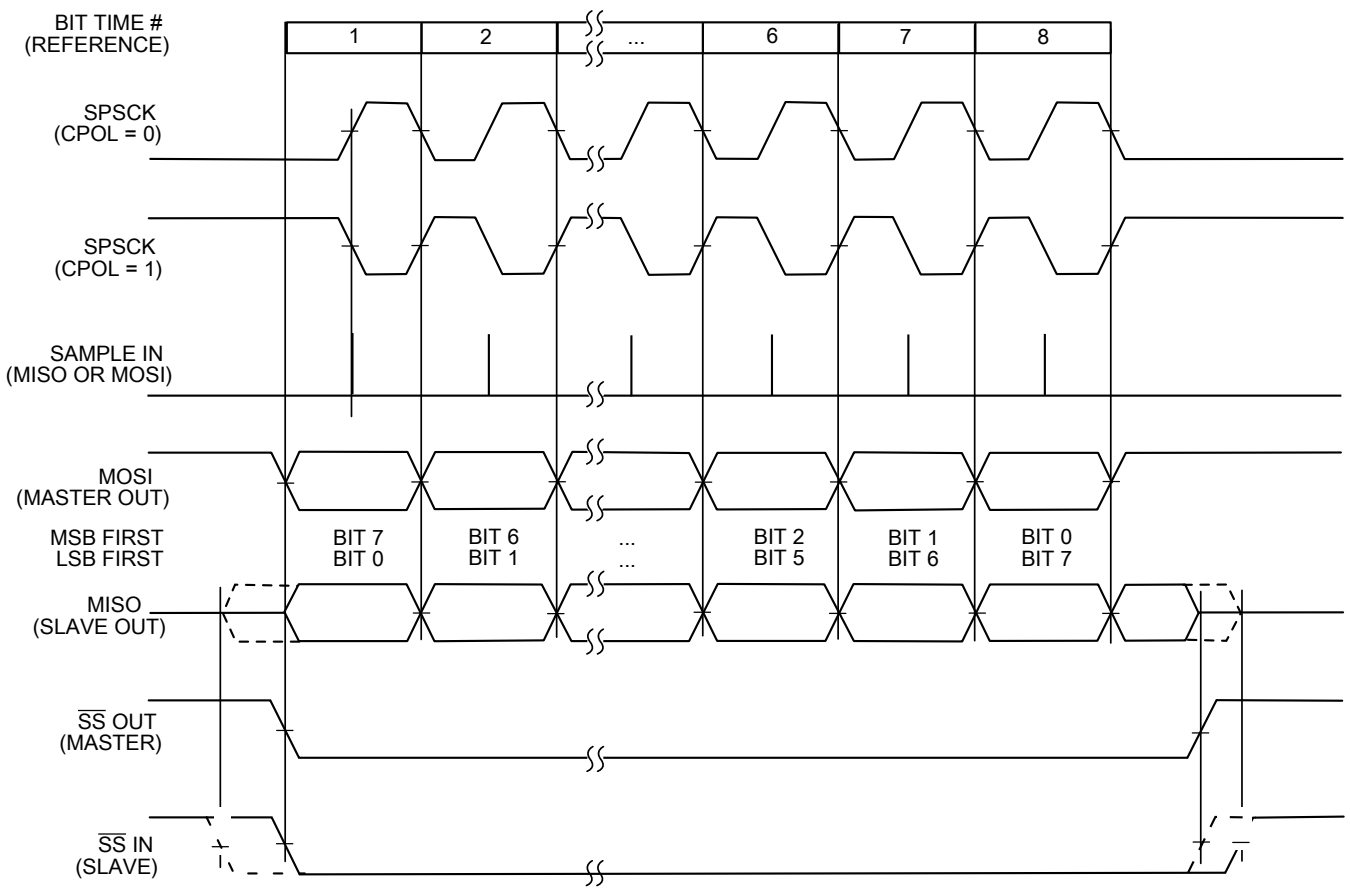


Figure 44-9. SPI clock formats (CPHA = 0)

When C1[CPHA] = 0, the slave begins to drive its MISO output with the first data bit value (MSB or LSB depending on LSBFE) when SS goes to active low. The first SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the second SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively. When C1[CPHA] = 0, the slave's SS input must go to its inactive high level between transfers.

44.5.8 SPI baud rate generation

As shown in the following figure, the clock source for the SPI baud rate generator is the SPI module clock. The prescale bits (SPPR2:SPPR1:SPPR0) choose a prescale divisor of 1, 2, 3, 4, 5, 6, 7, or 8. The rate-select bits (SPR3:SPR2:SPR1:SPR0) divide the output of the prescaler stage by 2, 4, 8, 16, 32, 64, 128, 256, or 512 to get the internal SPI master mode bit-rate clock.

The baud rate generator is activated only when the SPI is in the master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease I_{DD} current.

The baud rate divisor equation is as follows (except those reserved combinations in the SPI Baud Rate Divisor table).

$$BaudRateDivisor = (SPPR + 1) \times 2^{(SPR + 1)}$$

The baud rate can be calculated with the following equation:

$$BaudRate = SPI\ Module\ Clock / BaudRateDivisor$$

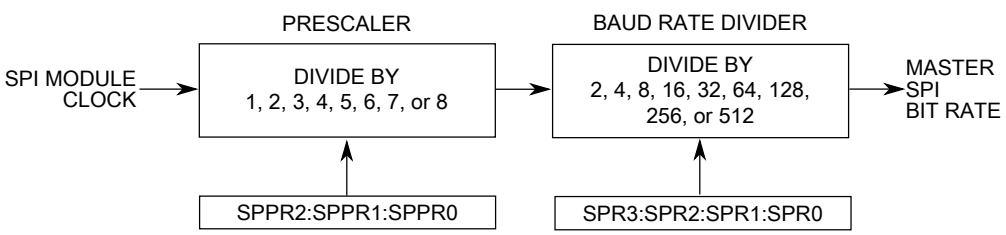


Figure 44-10. SPI baud rate generation

44.5.9 Special features

The following section describes the special features of SPI module.

44.5.9.1 $\overline{SS}$ Output

The $\overline{SS}$ output feature automatically drives the $\overline{SS}$ pin low during transmission to select external devices and drives the $\overline{SS}$ pin high during idle to deselect external devices. When the $\overline{SS}$ output is selected, the $\overline{SS}$ output pin is connected to the $\overline{SS}$ input pin of the external device.

The $\overline{SS}$ output is available only in master mode during normal SPI operation by asserting C1[SSOE] and C2[MODFEN] as shown in the description of C1[SSOE].

The mode fault feature is disabled while $\overline{SS}$ output is enabled.

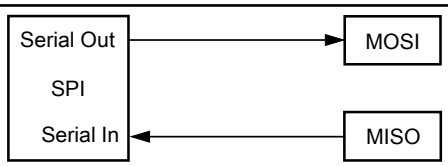
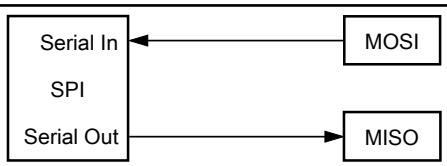
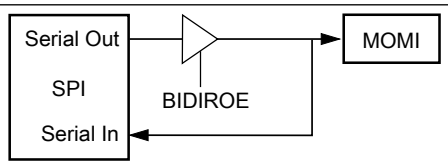
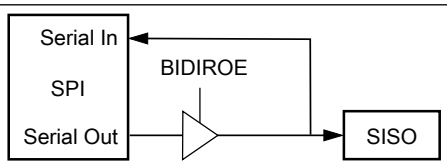
Note

Be careful when using the $\overline{SS}$ output feature in a multimaster system because the mode fault feature is not available for detecting system errors between masters.

44.5.9.2 Bidirectional mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI Control Register 2 (see the following table). In this mode, the SPI uses only one serial data pin for the interface with one or more external devices. C1[MSTR] decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 44-3. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on C2[BIDIROE]. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

The SPSCK is an output for the master mode and an input for the slave mode.

$\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.

The bidirectional mode does not affect SPSCK and $\overline{SS}$ functions.

Note

In bidirectional master mode, with the mode fault feature enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode. In this case, MISO becomes occupied by the SPI and MOSI is not used. Consider this scenario if the MISO pin is used for another purpose.

44.5.10 Error conditions

The SPI module has one error condition: the mode fault error.

44.5.10.1 Mode fault error

If the $\overline{SS}$ input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SPSCK lines simultaneously. This condition is not permitted in normal operation, and it sets the MODF bit in the SPI status register automatically provided that C2[MODFEN] is set.

In the special case where the SPI is in master mode and C2[MODFEN] is cleared, the $\overline{SS}$ pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. If the SPI system is configured as a slave, the $\overline{SS}$ pin is a dedicated input pin. A mode fault error does not occur in slave mode.

If a mode fault error occurs, the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So the SPSCK, MISO and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for an SPI system configured in master mode, the output enable of MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for the SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI Status Register (with MODF set) followed by a write to SPI Control Register 1. If the mode fault flag is cleared, the SPI becomes a normal master or slave again.

44.5.11 Low-power mode options

This section describes the low-power mode options.

44.5.11.1 SPI in Run mode

In Run mode, with the SPI system enable (SPE) bit in the SPI Control Register 1 clear, the SPI system is in a low-power, disabled state. SPI registers can still be accessed, but clocks to the core of this module are disabled.

44.5.11.2 SPI in Wait mode

SPI operation in Wait mode depends upon the state of the SPISWAI bit in SPI Control Register 2.

- If C2[SPISWAI] is clear, the SPI operates normally when the CPU is in Wait mode.
- If C2[SPISWAI] is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
  - If C2[SPISWAI] is set and the SPI is configured for master, any transmission and reception in progress stops at Wait mode entry. The transmission and reception resumes when the SPI exits Wait mode.
  - If C2[SPISWAI] is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SPSCK continues to be driven from the master. This keeps the slave synchronized to the master and the SPSCK.

    If the master transmits data while the slave is in wait mode, the slave continues to send data consistent with the operation mode at the start of wait mode (that is, if the slave is currently sending its SPIx_DH:SPIx_DL to the master, it continues to send the same byte. Otherwise, if the slave is currently sending the last data received byte from the master, it continues to send each previously received data from the master byte).

Note

Care must be taken when expecting data from a master while the slave is in a Wait mode or a Stop mode where the peripheral bus clock is stopped but internal logic states are retained. Even though the shift register continues to operate, the rest of the SPI is shut down (that is, an SPRF interrupt is not generated until an exit from Stop or Wait mode). Also, the data from the shift register is not copied into the SPIx_DH:SPIx_DL registers until after the slave SPI has exited Wait or Stop mode. An SPRF flag and SPIx_DH:SPIx_DL copy is only generated if Wait mode is entered or exited during a transmission. If the slave enters Wait mode in idle mode and exits Wait mode in idle mode, neither an SPRF nor a SPIx_DH:SPIx_DL copy occurs.

44.5.11.3 SPI in Stop mode

Operation in a Stop mode where the peripheral bus clock is stopped but internal logic states are retained depends on the SPI system. The Stop mode does not depend on C2[SPISWAI]. Upon entry to this type of stop mode, the SPI module clock is disabled (held high or low).

- If the SPI is in master mode and exchanging data when the CPU enters the Stop mode, the transmission is frozen until the CPU exits stop mode. After the exit from stop mode, data to and from the external SPI is exchanged correctly.
- In slave mode, the SPI remains synchronized with the master.

The SPI is completely disabled in a stop mode where the peripheral bus clock is stopped and internal logic states are not retained. After an exit from this type of stop mode, all registers are reset to their default values, and the SPI module must be reinitialized.

44.5.12 Reset

The reset values of registers and signals are described in the Memory Map and Register Descriptions content, which details the registers and their bitfields.

- If a data transmission occurs in slave mode after a reset without a write to SPIx_DH:SPIx_DL, the transmission consists of "garbage" or the data last received from the master before the reset.
- Reading from SPIx_DH:SPIx_DL after reset always returns zeros.

44.5.13 Interrupts

The SPI originates interrupt requests only when the SPI is enabled (the SPE bit in the SPIx_C1 register is set). The following is a description of how the SPI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

Four flag bits, three interrupt mask bits, and one interrupt vector are associated with the SPI system. The SPI interrupt enable mask (SPIE) enables interrupts from the SPI receiver full flag (SPRF) and mode fault flag (MODF). The SPI transmit interrupt enable mask (SPTIE) enables interrupts from the SPI transmit buffer empty flag (SPTEF). The SPI match interrupt enable mask bit (SPIMIE) enables interrupts from the SPI match flag (SPMF). When one of the flag bits is set, and the associated interrupt mask bit is set, a hardware interrupt request is sent to the CPU. If the interrupt mask bits are cleared, software can poll the associated flag bits instead of using interrupts. The SPI interrupt service routine (ISR) should check the flag bits to determine which event caused the interrupt. The service routine should also clear the flag bit(s) before returning from the ISR (usually near the beginning of the ISR).

44.5.13.1 MODF

MODF occurs when the master detects an error on the $\overline{SS}$ pin. The master SPI must be configured for the MODF feature (see the description of the C1[SSOE] bit). Once MODF is set, the current transfer is aborted and the master (MSTR) bit in the SPIx_C1 register resets to 0.

The MODF interrupt is reflected in the status register's MODF flag. Clearing the flag also clears the interrupt. This interrupt stays active while the MODF flag is set. MODF has an automatic clearing process that is described in the SPI Status Register.

44.5.13.2 SPRF

SPRF occurs when new data has been received and copied to the SPI receive data buffer. In 8-bit mode, SPRF is set only after all 8 bits have been shifted out of the shift register and into SPIx_DL. In 16-bit mode, SPRF is set only after all 16 bits have been shifted out of the shift register and into SPIx_DH:SPIx_DL.

After SPRF is set, it does not clear until it is serviced. SPRF has an automatic clearing process that is described in the SPI Status Register details. If the SPRF is not serviced before the end of the next transfer (that is, SPRF remains active throughout another transfer), the subsequent transfers are ignored and no new data is copied into the Data register.

44.5.13.3 SPTEF

SPTEF occurs when the SPI transmit buffer is ready to accept new data. In 8-bit mode, SPTEF is set only after all 8 bits have been moved from SPIx_DL into the shifter. In 16-bit mode, SPTEF is set only after all 16 bits have been moved from SPIx_DH:SPIx_DL into the shifter.

After SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process that is described in the SPI Status Register details.

44.5.13.4 SPMF

SPMF occurs when the data in the receive data buffer is equal to the data in the SPI Match Register. In 8-bit mode, SPMF is set only after bits 7–0 in the receive data buffer are determined to be equivalent to the value in SPIx_ML. In 16-bit mode, SPMF is set after bits 15–0 in the receive data buffer are determined to be equivalent to the value in SPIx_MH:SPIx_ML.

44.5.13.5 TNEAREF

The TNEAREF bit applies when the FIFO feature is supported.

The TNEAREF flag is set when only one 16-bit word or two 8-bit bytes of data remain in the transmit FIFO provided C3[5] = 0 or when only two 16-bit words or four 8-bit bytes of data remain in the transmit FIFO provided C3[5] =1. If FIFOMODE is not enabled, ignore this bit.

Clearing this interrupt depends on the state of C3[3] and the status of TNEAREF. Refer to the description of the SPI status (S) register.

44.5.13.6 RNFULLF

The RNFULLF bit applies when the FIFO feature is supported.

RNFULLF is set when more than three 16-bit words or six 8-bit bytes of data remain in the receive FIFO provided C3[4] = 0 or when more than two 16-bit words or four 8-bit bytes of data remain in the receive FIFO provided C3[4] = 1.

Clearing this interrupt depends on the state of C3[3] and the status of RNFULLF. Refer to the description of the SPI status (S) register.

44.5.13.7 Asynchronous interrupt in low-power modes

When the CPU is in Wait mode or Stop mode and the SPI module receives a transmission, the SPI module can generate an asynchronous interrupt to wake the CPU from the low power mode. The module generates the asynchronous interrupt only when all of the following conditions apply:

1. C1[SPIE] is set to 1.
2. The CPU is in Wait mode—in which case C2[SPISWAI] must be 1—or in Stop mode where the peripheral bus clock is stopped but internal logic states are retained.
3. The SPI module is in slave mode.
4. The received transmission ends.
5. When the FIFO feature is supported, FIFO mode is disabled: C3[FIFOMODE] is 0.

After the interrupt wakes the CPU and the peripheral bus clock is active again, the SPI module copies the received data from the shifter into the Data register and generates flags or DMA request signals. During the wakeup phase, a continuous transmission from a master would destroy the first received data.

44.6 Initialization/application information

This section discusses an example of how to initialize and use the SPI.

NOTE

When operating the SPI at the maximum baud rate it must be configured for 16 bit operation.

44.6.1 Initialization sequence

Before the SPI module can be used for communication, an initialization procedure must be carried out, as follows:

1. Update the Control Register 1 (SPIx_C1) to enable the SPI and to control interrupt enables. This register also sets the SPI as master or slave, determines clock phase and polarity, and configures the main SPI options.
2. Update the Control Register 2 (SPIx_C2) to enable additional SPI functions such as the SPI match interrupt feature, the master mode-fault function, and bidirectional mode output as well as to control 8- or 16-bit mode selection and other optional features.
3. Update the Baud Rate Register (SPIx_BR) to set the prescaler and bit rate divisor for an SPI master.
4. Update the Hardware Match Register (SPIx_MH:SPIx_ML) with the value to be compared to the receive data register for triggering an interrupt if hardware match interrupts are enabled.
5. In the master, read SPIx_S while S[SPTEF] = 1, and then write to the transmit data register (SPIx_DH:SPIx_DL) to begin transfer.

44.6.2 Pseudo-Code Example

In this example, the SPI module is set up for master mode with only hardware match interrupts enabled. The SPI runs in 16-bit mode at a maximum baud rate of SPI module clock divided by 2. Clock phase and polarity are set for an active-high SPI clock where the first edge on SPSCK occurs at the start of the first cycle of a data transfer.

SPIx_C1=0x54(%01010100)				
Bit 7	SPIE	=	0	Disables receive and mode fault interrupts
Bit 6	SPE	=	1	Enables the SPI system
Bit 5	SPTIE	=	0	Disables SPI transmit interrupts
Bit 4	MSTR	=	1	Sets the SPI module as a master SPI device
Bit 3	CPOL	=	0	Configures SPI clock as active-high
Bit 2	CPHA	=	1	First edge on SPSCK at start of first data transfer cycle
Bit 1	SSOE	=	0	Determines SS pin function when mode fault enabled
Bit 0	LSBFE	=	0	SPI serial data transfers start with most significant bit

SPIx_C2 = 0xC0(%11000000)				
Bit 7	SPMIE	=	1	SPI hardware match interrupt enabled
Bit 6	SPIMODE	=	1	Configures SPI for 16-bit mode
Bit 5	TXDMAE	=	0	DMA request disabled
Bit 4	MODFEN	=	0	Disables mode fault function

Table continues on the next page...

SPIx_C2 = 0xC0(%11000000)				
Bit 3	BIDIROE	=	0	SPI data I/O pin acts as input
Bit 2	RXDMAE	=	0	DMA request disabled
Bit 1	SPISWAI	=	0	SPI clocks operate in wait mode
Bit 0	SPC0	=	0	uses separate pins for data input and output

SPIx_BR = 0x00(%00000000)			
Bit 7	=	0	Reserved
Bit 6:4	=	000	Sets prescale divisor to 1
Bit 3:0	=	0000	Sets baud rate divisor to 2

SPIx_S = 0x00(%00000000)				
Bit 7	SPRF	=	0	Flag is set when receive data buffer is full
Bit 6	SPMF	=	0	Flag is set when SPIx_MH/ML = receive data buffer
Bit 5	SPTEF	=	0	Flag is set when transmit data buffer is empty
Bit 4	MODF	=	0	Mode fault flag for master mode
Bit 3:0	RNFULLF, TNEARF, TXFULLF, and RFIFOEF	=	0	Reserved (when FIFOMODE is not present or is 0) or FIFO flags (when FIFOMODE is 1)
Bit 3:0		=	0	FIFOMODE is not enabled

SPIx_MH = 0xXX
In 16-bit mode, this register holds bits 8–15 of the hardware match buffer. In 8-bit mode, writes to this register will be ignored.

SPIx_ML = 0xXX
Holds bits 0–7 of the hardware match buffer.

SPIx_DH = 0xxx
In 16-bit mode, this register holds bits 8–15 of the data to be transmitted by the transmit buffer and received by the receive buffer.

SPIx_DL = 0xxx
Holds bits 0–7 of the data to be transmitted by the transmit buffer and received by the receive buffer.

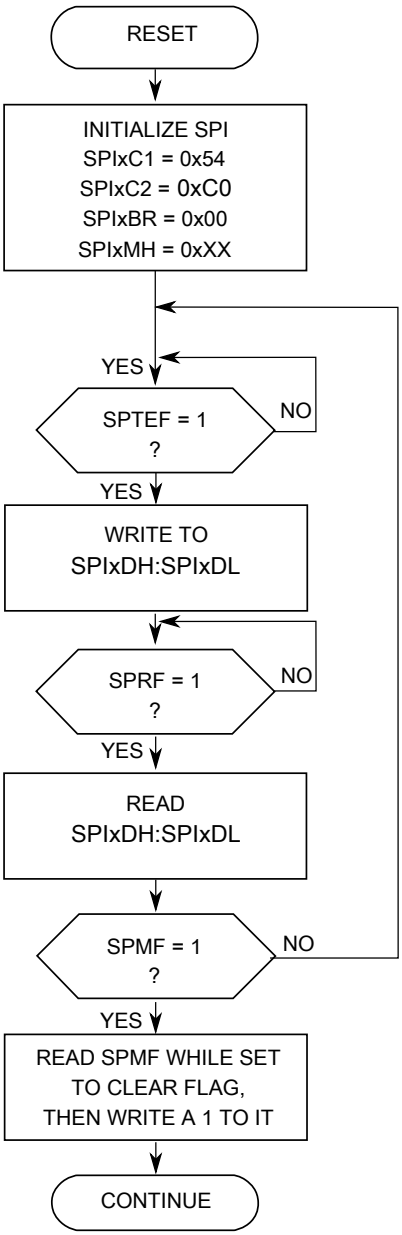


Figure 44-11. Initialization Flowchart Example for SPI Master Device in 16-bit Mode for FIFOMODE = 0

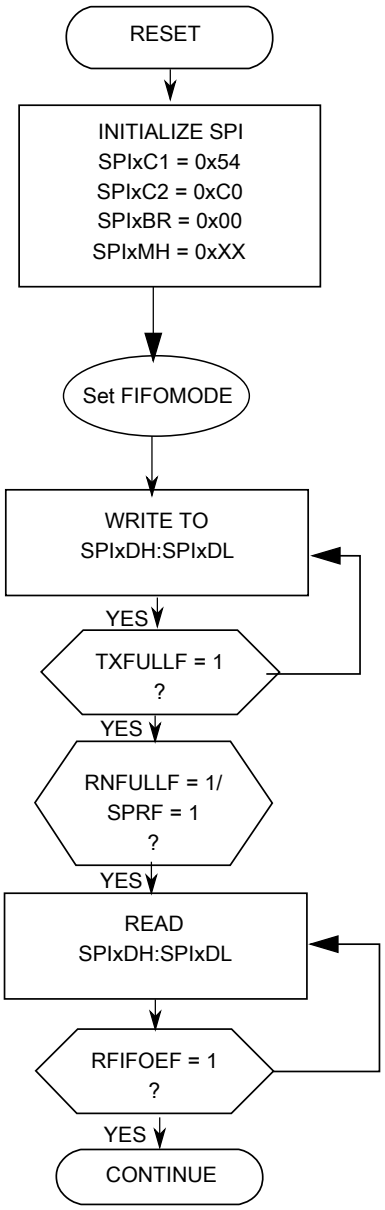


Figure 44-12. Initialization Flowchart Example for SPI Master Device in 16-bit Mode for FIFOMODE = 1

Chapter 45
Universal Asynchronous Receiver/Transmitter (UART)

45.1 Chip-specific UART information

45.1.1 Overview

The UART allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The UART transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the UART, writes the data to be transmitted, and processes received data.

45.1.2 Instantiation Information

This section describes how each UART module is instantiated.

Table 45-1. UART instantiation information

Feature	UART0	UART1	UART2	UART3
UART modules with standard features	Yes	Yes	Yes	Yes
UART modules with ISO7816	No [1]	Yes	No [1]	Yes
IrDA [2]	Yes (via CMP0 & XBAR)	Yes (via CMP0 & XBAR)	Yes (via CMP0 & XBAR)	Yes (via CMP0 & XBAR)
Flow Control	Yes	Yes	Yes	Yes
FIFO Depth (number of entries)	8 for Tx 8 for Rx	8 for Tx 8 for Rx	8 for Tx 8 for Rx	8 for Tx 8 for Rx
Module Clock (max.)	25 MHz [3]	75 MHz [4]	25 MHz[3]	75 MHz[4]

Table continues on the next page...

Table 45-1. UART instantiation information (continued)

Feature	UART0	UART1	UART2	UART3
Buad Rate	1.5625 Mbps	4.6875 Mbps [5]	1.5625 Mbps	4.6875 Mbps
AMR Support	3 V only	3 V only	3 V only	3 V only

1. **Note**: In the "Memory map and registers" section, ISO7816 related registers are **not** available for this UART instance.
2. **Note**: The UART standalone cannot support IrDA (*whereas the LPUART module supports IrDA directly*). The interconnection of UART, CMP and XBAR will implement the IrDA operation. See "IR Communication Support" section for more information.
3. Clocked on bus clock.
4. Clocked on core clock.
5. Baud rate might be less for UART1 pads which are VPP pad (PTI1).

1. Standard features of all UARTs:
   - RS-485 support
   - Hardware flow control (RTS/CTS)
   - 9-bit UART to support address mark with parity
   - MSB/LSB configuration on data
2. IrDA is supported on all UART by connecting the corresponding Comparator output to Rx input of UART via the peripheral crossbar. However, one UART can be used at a time. Whereas for the LPUART module, IrDA function is directly supported.
3. UART0, UART1 and UART3 are muxed on multiple pins. See Signal multiplexing section for pin details.

IR Communication Support: In this device, modulated output IR pulse can be performed using any Quad Timer channel operating at 38 kHz pulse output mode. The modulated serial data stream will be created using the logic AND of SCI_Tx (i.e. UART_Tx on this device) and Quad Timer channel output (routed via the PXBAR). This will be supported on all SCIs (i.e. UART on this device) (one SCI used at a time).

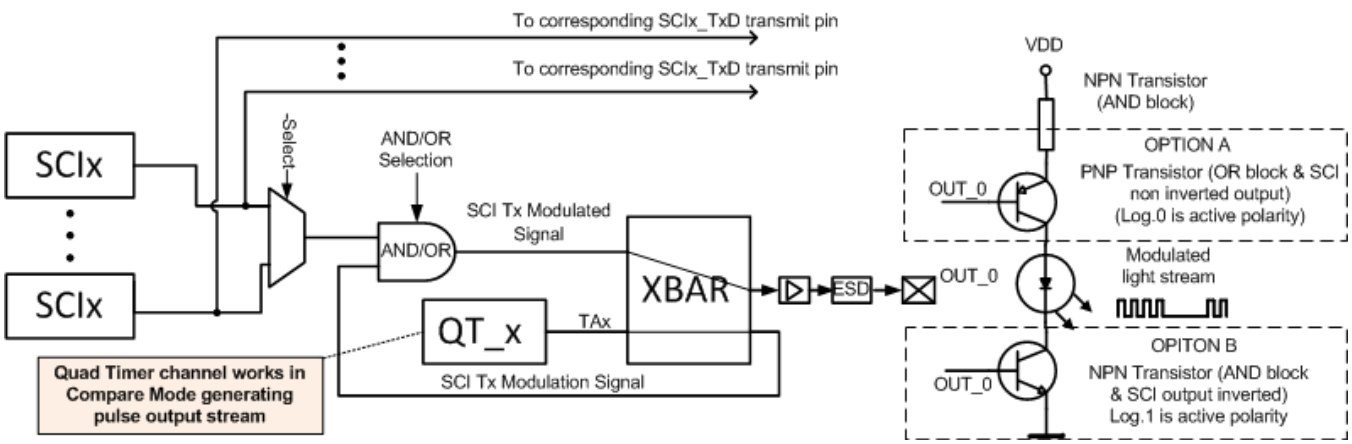


Figure 45-1. SCI (UART) Tx Modulation for IR Communications

The received modulated IR serial data stream to be decoded is filtered using external RC filter as shown in the figure below. The RC filter forms envelope of the modulated signal. The signal envelope propagates further to High Speed Comparator (HSCMP) for edge recognition. Comparator output routes via PXBAR to input signal of respective SCI module (i.e. UART on this device) (supported on all SCIs, but one SCI used at a time).

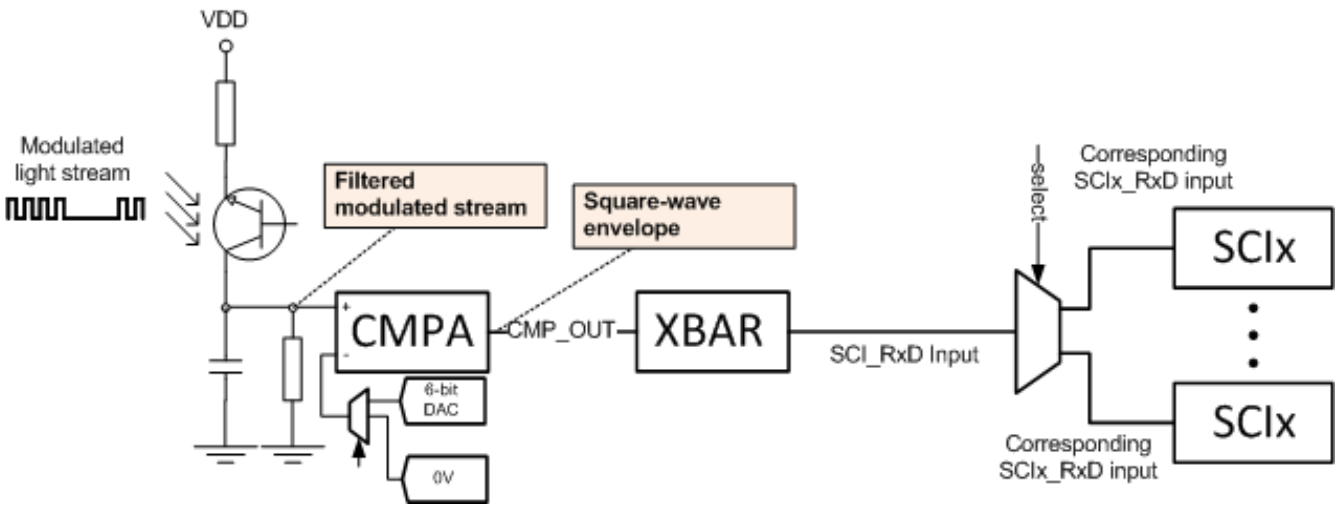


Figure 45-2. SCI (UART) Rx Connection for IR Communications

45.1.3 Wakeup

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

45.2 Introduction

The UART allows asynchronous serial communication with peripheral devices and CPUs.

45.2.1 Features

The UART includes the following features:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format

- 13-bit baud rate selection with /32 fractional divide, based on the module clock frequency
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable transmitter output polarity
- Programmable receive input polarity
- Up to 14-bit break character transmission.
- 11-bit break character detection option
- Independent FIFO structure for transmit and receive
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Address match feature in the receiver to reduce address mark wakeup ISR overhead
- Ability to select MSB or LSB to be first bit on wire
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Support for ISO 7816 protocol to interface with SIM cards and smart cards
  - Support for T=0 and T=1 protocols
  - Automatic retransmission of NACK'd packets with programmable retry threshold
  - Support for 11 and 12 ETU transfers
  - Detection of initial packet and automated transfer parameter programming
  - Interrupt-driven operation with seven ISO-7816 specific interrupts:
    - Wait time violated
    - Character wait time violated
    - Block wait time violated
    - Initial frame detected
    - Transmit error threshold exceeded

        - Receive error threshold exceeded
        - Guard time violated
- Interrupt-driven operation with flags, not specific to ISO-7816 support
    - Transmitter data buffer at or below watermark
    - Transmission complete
    - Receiver data buffer at or above watermark
    - Idle receiver input
    - Receiver data buffer overrun
    - Receiver data buffer underflow
    - Transmit data buffer overflow
    - Noise error
    - Framing error
    - Parity error
    - Active edge on receive pin
- Receiver framing error detection
- Hardware parity generation and checking
- 1/16 bit-time noise detection
- DMA interface

45.2.2 Modes of operation

The UART functions in the same way in all the normal modes.

It has the following low power mode:

- Stop mode

45.2.2.1 Run mode

This is the normal mode of operation.

45.2.2.2 Stop mode

The UART is inactive during Stop mode for reduced power consumption. The STOP instruction does not affect the UART register states, but the UART module clock is disabled. The UART operation resumes after an external interrupt brings the CPU out of Stop mode. Bringing the CPU out of Stop mode by reset aborts any ongoing transmission or reception and resets the UART. Entering or leaving Stop mode does not initiate any power down or power up procedures for the ISO-7816 smartcard interface.

45.3 UART signal descriptions

The UART signals are shown in the following table.

Table 45-2. UART signal descriptions

Signal	Description	I/O
CTS	Clear to send	I
$\overline{\text{RTS}}$	Request to send	O
RXD	Receive data	I
TXD	Transmit data	O

45.3.1 Detailed signal descriptions

The detailed signal descriptions of the UART are shown in the following table.

Table 45-3. UART—Detailed signal descriptions

Signal	I/O	Description	
CTS	I	Clear to send. Indicates whether the UART can start transmitting data when flow control is enabled.	
		State meaning	Asserted—Data transmission can start. Negated—Data transmission cannot start.
		Timing	Assertion—When transmitting device's RTS asserts. Negation—When transmitting device's RTS deasserts.
RTS	O	Request to send. When driven by the receiver, indicates whether the UART is ready to receive data. When driven by the transmitter, can enable an external transceiver during transmission.	
		State meaning	Asserted—When driven by the receiver, ready to receive data. When driven by the transmitter, enable the external transmitter.

Table continues on the next page...

Table 45-3. UART—Detailed signal descriptions (continued)

Signal	I/O	Description	
			Negated—When driven by the receiver, not ready to receive data. When driven by the transmitter, disable the external transmitter.
		Timing	Assertion—Can occur at any time; can assert asynchronously to the other input signals. Negation—Can occur at any time; can deassert asynchronously to the other input signals.
RXD	I	Receive data. Serial data input to receiver.	
		State meaning	Whether RXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Sampled at a frequency determined by the module clock divided by the baud rate.
TXD	O	Transmit data. Serial data output from transmitter.	
		State meaning	Whether TXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Driven at the beginning or within a bit time according to the bit encoding method along with other configuration settings. Otherwise, transmissions are independent of reception timing.

45.4 Memory map and registers

This section provides a detailed description of all memory and registers.

Only byte accesses are supported.

UART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A000	UART Baud Rate Registers: High (UART0_BDH)	8	R/W	00h	45.4.1/937
4006_A001	UART Baud Rate Registers: Low (UART0_BDL)	8	R/W	04h	45.4.2/938
4006_A002	UART Control Register 1 (UART0_C1)	8	R/W	00h	45.4.3/939
4006_A003	UART Control Register 2 (UART0_C2)	8	R/W	00h	45.4.4/940
4006_A004	UART Status Register 1 (UART0_S1)	8	R	C0h	45.4.5/942
4006_A005	UART Status Register 2 (UART0_S2)	8	R/W	00h	45.4.6/945
4006_A006	UART Control Register 3 (UART0_C3)	8	R/W	00h	45.4.7/946
4006_A007	UART Data Register (UART0_D)	8	R/W	00h	45.4.8/948
4006_A008	UART Match Address Registers 1 (UART0_MA1)	8	R/W	00h	45.4.9/949
4006_A009	UART Match Address Registers 2 (UART0_MA2)	8	R/W	00h	45.4.10/ 949
4006_A00A	UART Control Register 4 (UART0_C4)	8	R/W	00h	45.4.11/ 950

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A00B	UART Control Register 5 (UART0_C5)	8	R/W	00h	45.4.12/ 950
4006_A00C	UART Extended Data Register (UART0_ED)	8	R	00h	45.4.13/ 951
4006_A00D	UART Modem Register (UART0_MODEM)	8	R/W	00h	45.4.14/ 952
4006_A010	UART FIFO Parameters (UART0_PFIFO)	8	R/W	See section	45.4.15/ 954
4006_A011	UART FIFO Control Register (UART0_CFIFO)	8	R/W	00h	45.4.16/ 955
4006_A012	UART FIFO Status Register (UART0_SFIFO)	8	R/W	C0h	45.4.17/ 956
4006_A013	UART FIFO Transmit Watermark (UART0_TWFIFO)	8	R/W	00h	45.4.18/ 957
4006_A014	UART FIFO Transmit Count (UART0_TCFIFO)	8	R	00h	45.4.19/ 958
4006_A015	UART FIFO Receive Watermark (UART0_RWFIFO)	8	R/W	01h	45.4.20/ 958
4006_A016	UART FIFO Receive Count (UART0_RCFIFO)	8	R	00h	45.4.21/ 959
4006_A018	UART 7816 Control Register (UART0_C7816)	8	R/W	00h	45.4.22/ 959
4006_A019	UART 7816 Interrupt Enable Register (UART0_IE7816)	8	R/W	00h	45.4.23/ 961
4006_A01A	UART 7816 Interrupt Status Register (UART0_IS7816)	8	R/W	00h	45.4.24/ 962
4006_A01B	UART 7816 Wait Parameter Register (UART0_WP7816)	8	R/W	00h	45.4.25/ 964
4006_A01C	UART 7816 Wait N Register (UART0_WN7816)	8	R/W	00h	45.4.26/ 964
4006_A01D	UART 7816 Wait FD Register (UART0_WF7816)	8	R/W	01h	45.4.27/ 965
4006_A01E	UART 7816 Error Threshold Register (UART0_ET7816)	8	R/W	00h	45.4.28/ 965
4006_A01F	UART 7816 Transmit Length Register (UART0_TL7816)	8	R/W	00h	45.4.29/ 966
4006_A03A	UART 7816 ATR Duration Timer Register A (UART0_AP7816A_T0)	8	R/W	00h	45.4.30/ 966
4006_A03B	UART 7816 ATR Duration Timer Register B (UART0_AP7816B_T0)	8	R/W	00h	45.4.31/ 967
4006_A03C	UART 7816 Wait Parameter Register A (UART0_WP7816A_T0)	8	R/W	00h	45.4.32/ 968
4006_A03C	UART 7816 Wait Parameter Register A (UART0_WP7816A_T1)	8	R/W	00h	45.4.33/ 968

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A03D	UART 7816 Wait Parameter Register B (UART0_WP7816B_T0)	8	R/W	14h	45.4.34/ 969
4006_A03D	UART 7816 Wait Parameter Register B (UART0_WP7816B_T1)	8	R/W	14h	45.4.35/ 969
4006_A03E	UART 7816 Wait and Guard Parameter Register (UART0_WGP7816_T1)	8	R/W	06h	45.4.36/ 970
4006_A03F	UART 7816 Wait Parameter Register C (UART0_WP7816C_T1)	8	R/W	0Bh	45.4.37/ 970
4006_B000	UART Baud Rate Registers: High (UART1_BDH)	8	R/W	00h	45.4.1/937
4006_B001	UART Baud Rate Registers: Low (UART1_BDL)	8	R/W	04h	45.4.2/938
4006_B002	UART Control Register 1 (UART1_C1)	8	R/W	00h	45.4.3/939
4006_B003	UART Control Register 2 (UART1_C2)	8	R/W	00h	45.4.4/940
4006_B004	UART Status Register 1 (UART1_S1)	8	R	C0h	45.4.5/942
4006_B005	UART Status Register 2 (UART1_S2)	8	R/W	00h	45.4.6/945
4006_B006	UART Control Register 3 (UART1_C3)	8	R/W	00h	45.4.7/946
4006_B007	UART Data Register (UART1_D)	8	R/W	00h	45.4.8/948
4006_B008	UART Match Address Registers 1 (UART1_MA1)	8	R/W	00h	45.4.9/949
4006_B009	UART Match Address Registers 2 (UART1_MA2)	8	R/W	00h	45.4.10/ 949
4006_B00A	UART Control Register 4 (UART1_C4)	8	R/W	00h	45.4.11/ 950
4006_B00B	UART Control Register 5 (UART1_C5)	8	R/W	00h	45.4.12/ 950
4006_B00C	UART Extended Data Register (UART1_ED)	8	R	00h	45.4.13/ 951
4006_B00D	UART Modem Register (UART1_MODEM)	8	R/W	00h	45.4.14/ 952
4006_B010	UART FIFO Parameters (UART1_PFIFO)	8	R/W	See section	45.4.15/ 954
4006_B011	UART FIFO Control Register (UART1_CFIFO)	8	R/W	00h	45.4.16/ 955
4006_B012	UART FIFO Status Register (UART1_SFIFO)	8	R/W	C0h	45.4.17/ 956
4006_B013	UART FIFO Transmit Watermark (UART1_TWFIFO)	8	R/W	00h	45.4.18/ 957
4006_B014	UART FIFO Transmit Count (UART1_TCFIFO)	8	R	00h	45.4.19/ 958
4006_B015	UART FIFO Receive Watermark (UART1_RWFIFO)	8	R/W	01h	45.4.20/ 958
4006_B016	UART FIFO Receive Count (UART1_RCFIFO)	8	R	00h	45.4.21/ 959
4006_B018	UART 7816 Control Register (UART1_C7816)	8	R/W	00h	45.4.22/ 959

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_B019	UART 7816 Interrupt Enable Register (UART1_IE7816)	8	R/W	00h	45.4.23/ 961
4006_B01A	UART 7816 Interrupt Status Register (UART1_IS7816)	8	R/W	00h	45.4.24/ 962
4006_B01B	UART 7816 Wait Parameter Register (UART1_WP7816)	8	R/W	00h	45.4.25/ 964
4006_B01C	UART 7816 Wait N Register (UART1_WN7816)	8	R/W	00h	45.4.26/ 964
4006_B01D	UART 7816 Wait FD Register (UART1_WF7816)	8	R/W	01h	45.4.27/ 965
4006_B01E	UART 7816 Error Threshold Register (UART1_ET7816)	8	R/W	00h	45.4.28/ 965
4006_B01F	UART 7816 Transmit Length Register (UART1_TL7816)	8	R/W	00h	45.4.29/ 966
4006_B03A	UART 7816 ATR Duration Timer Register A (UART1_AP7816A_T0)	8	R/W	00h	45.4.30/ 966
4006_B03B	UART 7816 ATR Duration Timer Register B (UART1_AP7816B_T0)	8	R/W	00h	45.4.31/ 967
4006_B03C	UART 7816 Wait Parameter Register A (UART1_WP7816A_T0)	8	R/W	00h	45.4.32/ 968
4006_B03C	UART 7816 Wait Parameter Register A (UART1_WP7816A_T1)	8	R/W	00h	45.4.33/ 968
4006_B03D	UART 7816 Wait Parameter Register B (UART1_WP7816B_T0)	8	R/W	14h	45.4.34/ 969
4006_B03D	UART 7816 Wait Parameter Register B (UART1_WP7816B_T1)	8	R/W	14h	45.4.35/ 969
4006_B03E	UART 7816 Wait and Guard Parameter Register (UART1_WGP7816_T1)	8	R/W	06h	45.4.36/ 970
4006_B03F	UART 7816 Wait Parameter Register C (UART1_WP7816C_T1)	8	R/W	0Bh	45.4.37/ 970
4006_C000	UART Baud Rate Registers: High (UART2_BDH)	8	R/W	00h	45.4.1/937
4006_C001	UART Baud Rate Registers: Low (UART2_BDL)	8	R/W	04h	45.4.2/938
4006_C002	UART Control Register 1 (UART2_C1)	8	R/W	00h	45.4.3/939
4006_C003	UART Control Register 2 (UART2_C2)	8	R/W	00h	45.4.4/940
4006_C004	UART Status Register 1 (UART2_S1)	8	R	C0h	45.4.5/942
4006_C005	UART Status Register 2 (UART2_S2)	8	R/W	00h	45.4.6/945
4006_C006	UART Control Register 3 (UART2_C3)	8	R/W	00h	45.4.7/946
4006_C007	UART Data Register (UART2_D)	8	R/W	00h	45.4.8/948
4006_C008	UART Match Address Registers 1 (UART2_MA1)	8	R/W	00h	45.4.9/949
4006_C009	UART Match Address Registers 2 (UART2_MA2)	8	R/W	00h	45.4.10/ 949
4006_C00A	UART Control Register 4 (UART2_C4)	8	R/W	00h	45.4.11/ 950

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_C00B	UART Control Register 5 (UART2_C5)	8	R/W	00h	45.4.12/ 950
4006_C00C	UART Extended Data Register (UART2_ED)	8	R	00h	45.4.13/ 951
4006_C00D	UART Modem Register (UART2_MODEM)	8	R/W	00h	45.4.14/ 952
4006_C010	UART FIFO Parameters (UART2_PFIFO)	8	R/W	See section	45.4.15/ 954
4006_C011	UART FIFO Control Register (UART2_CFIFO)	8	R/W	00h	45.4.16/ 955
4006_C012	UART FIFO Status Register (UART2_SFIFO)	8	R/W	C0h	45.4.17/ 956
4006_C013	UART FIFO Transmit Watermark (UART2_TWFIFO)	8	R/W	00h	45.4.18/ 957
4006_C014	UART FIFO Transmit Count (UART2_TCFIFO)	8	R	00h	45.4.19/ 958
4006_C015	UART FIFO Receive Watermark (UART2_RWFIFO)	8	R/W	01h	45.4.20/ 958
4006_C016	UART FIFO Receive Count (UART2_RCFIFO)	8	R	00h	45.4.21/ 959
4006_C018	UART 7816 Control Register (UART2_C7816)	8	R/W	00h	45.4.22/ 959
4006_C019	UART 7816 Interrupt Enable Register (UART2_IE7816)	8	R/W	00h	45.4.23/ 961
4006_C01A	UART 7816 Interrupt Status Register (UART2_IS7816)	8	R/W	00h	45.4.24/ 962
4006_C01B	UART 7816 Wait Parameter Register (UART2_WP7816)	8	R/W	00h	45.4.25/ 964
4006_C01C	UART 7816 Wait N Register (UART2_WN7816)	8	R/W	00h	45.4.26/ 964
4006_C01D	UART 7816 Wait FD Register (UART2_WF7816)	8	R/W	01h	45.4.27/ 965
4006_C01E	UART 7816 Error Threshold Register (UART2_ET7816)	8	R/W	00h	45.4.28/ 965
4006_C01F	UART 7816 Transmit Length Register (UART2_TL7816)	8	R/W	00h	45.4.29/ 966
4006_C03A	UART 7816 ATR Duration Timer Register A (UART2_AP7816A_T0)	8	R/W	00h	45.4.30/ 966
4006_C03B	UART 7816 ATR Duration Timer Register B (UART2_AP7816B_T0)	8	R/W	00h	45.4.31/ 967
4006_C03C	UART 7816 Wait Parameter Register A (UART2_WP7816A_T0)	8	R/W	00h	45.4.32/ 968
4006_C03C	UART 7816 Wait Parameter Register A (UART2_WP7816A_T1)	8	R/W	00h	45.4.33/ 968

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_C03D	UART 7816 Wait Parameter Register B (UART2_WP7816B_T0)	8	R/W	14h	45.4.34/ 969
4006_C03D	UART 7816 Wait Parameter Register B (UART2_WP7816B_T1)	8	R/W	14h	45.4.35/ 969
4006_C03E	UART 7816 Wait and Guard Parameter Register (UART2_WGP7816_T1)	8	R/W	06h	45.4.36/ 970
4006_C03F	UART 7816 Wait Parameter Register C (UART2_WP7816C_T1)	8	R/W	0Bh	45.4.37/ 970
4006_D000	UART Baud Rate Registers: High (UART3_BDH)	8	R/W	00h	45.4.1/937
4006_D001	UART Baud Rate Registers: Low (UART3_BDL)	8	R/W	04h	45.4.2/938
4006_D002	UART Control Register 1 (UART3_C1)	8	R/W	00h	45.4.3/939
4006_D003	UART Control Register 2 (UART3_C2)	8	R/W	00h	45.4.4/940
4006_D004	UART Status Register 1 (UART3_S1)	8	R	C0h	45.4.5/942
4006_D005	UART Status Register 2 (UART3_S2)	8	R/W	00h	45.4.6/945
4006_D006	UART Control Register 3 (UART3_C3)	8	R/W	00h	45.4.7/946
4006_D007	UART Data Register (UART3_D)	8	R/W	00h	45.4.8/948
4006_D008	UART Match Address Registers 1 (UART3_MA1)	8	R/W	00h	45.4.9/949
4006_D009	UART Match Address Registers 2 (UART3_MA2)	8	R/W	00h	45.4.10/ 949
4006_D00A	UART Control Register 4 (UART3_C4)	8	R/W	00h	45.4.11/ 950
4006_D00B	UART Control Register 5 (UART3_C5)	8	R/W	00h	45.4.12/ 950
4006_D00C	UART Extended Data Register (UART3_ED)	8	R	00h	45.4.13/ 951
4006_D00D	UART Modem Register (UART3_MODEM)	8	R/W	00h	45.4.14/ 952
4006_D010	UART FIFO Parameters (UART3_PFIFO)	8	R/W	See section	45.4.15/ 954
4006_D011	UART FIFO Control Register (UART3_CFIFO)	8	R/W	00h	45.4.16/ 955
4006_D012	UART FIFO Status Register (UART3_SFIFO)	8	R/W	C0h	45.4.17/ 956
4006_D013	UART FIFO Transmit Watermark (UART3_TWFIFO)	8	R/W	00h	45.4.18/ 957
4006_D014	UART FIFO Transmit Count (UART3_TCFIFO)	8	R	00h	45.4.19/ 958
4006_D015	UART FIFO Receive Watermark (UART3_RWFIFO)	8	R/W	01h	45.4.20/ 958
4006_D016	UART FIFO Receive Count (UART3_RCFIFO)	8	R	00h	45.4.21/ 959
4006_D018	UART 7816 Control Register (UART3_C7816)	8	R/W	00h	45.4.22/ 959

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_D019	UART 7816 Interrupt Enable Register (UART3_IE7816)	8	R/W	00h	45.4.23/ 961
4006_D01A	UART 7816 Interrupt Status Register (UART3_IS7816)	8	R/W	00h	45.4.24/ 962
4006_D01B	UART 7816 Wait Parameter Register (UART3_WP7816)	8	R/W	00h	45.4.25/ 964
4006_D01C	UART 7816 Wait N Register (UART3_WN7816)	8	R/W	00h	45.4.26/ 964
4006_D01D	UART 7816 Wait FD Register (UART3_WF7816)	8	R/W	01h	45.4.27/ 965
4006_D01E	UART 7816 Error Threshold Register (UART3_ET7816)	8	R/W	00h	45.4.28/ 965
4006_D01F	UART 7816 Transmit Length Register (UART3_TL7816)	8	R/W	00h	45.4.29/ 966
4006_D03A	UART 7816 ATR Duration Timer Register A (UART3_AP7816A_T0)	8	R/W	00h	45.4.30/ 966
4006_D03B	UART 7816 ATR Duration Timer Register B (UART3_AP7816B_T0)	8	R/W	00h	45.4.31/ 967
4006_D03C	UART 7816 Wait Parameter Register A (UART3_WP7816A_T0)	8	R/W	00h	45.4.32/ 968
4006_D03C	UART 7816 Wait Parameter Register A (UART3_WP7816A_T1)	8	R/W	00h	45.4.33/ 968
4006_D03D	UART 7816 Wait Parameter Register B (UART3_WP7816B_T0)	8	R/W	14h	45.4.34/ 969
4006_D03D	UART 7816 Wait Parameter Register B (UART3_WP7816B_T1)	8	R/W	14h	45.4.35/ 969
4006_D03E	UART 7816 Wait and Guard Parameter Register (UART3_WGP7816_T1)	8	R/W	06h	45.4.36/ 970
4006_D03F	UART 7816 Wait Parameter Register C (UART3_WP7816C_T1)	8	R/W	0Bh	45.4.37/ 970

45.4.1 UART Baud Rate Registers: High (UARTx_BDH)

This register, along with the BDL register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting (SBR[12:0]), first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written.

BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	Reserved	RXEDGIE	0	SBR				
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_BDH field descriptions

Field	Description
7 Reserved	Reserved. This field is reserved.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. 0 Hardware interrupts from RXEDGIF disabled using polling. 1 RXEDGIF interrupt request enabled.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SBR	UART Baud Rate Bits The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details. **NOTE:** • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

45.4.2 UART Baud Rate Registers: Low (UARTx_BDL)

This register, along with the BDH register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting, SBR[12:0], first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written. BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read	SBR							
Write								
Reset	0	0	0	0	0	1	0	0

UARTx_BDL field descriptions

Field	Description
SBR	UART Baud Rate Bits The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details.

UARTx_BDL field descriptions (continued)

Field	Description
	NOTE: • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

45.4.3 UART Control Register 1 (UARTx_C1)

This read/write register controls various optional features of the UART system.

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	LOOPS	Reserved	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

UARTx_C1 field descriptions

Field	Description
7 LOOPS	Loop Mode Select When LOOPS is set, the RxD pin is disconnected from the UART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation. 1 Loop mode where transmitter output is internally connected to receiver input. The receiver input is determined by RSRC.
6 Reserved	Reserved. This field is reserved.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0 Selects internal loop back mode. The receiver input is internally connected to transmitter output. 1 Single wire UART mode where the receiver input is connected to the transmit pin input signal.
4 M	9-bit or 8-bit Mode Select This field must be set when C7816[ISO_7816E] is set/enabled. 0 Normal—start + 8 data bits (MSB/LSB first as determined by MSBF) + stop. 1 Use—start + 9 data bits (MSB/LSB first as determined by MSBF) + stop.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the UART: • Address mark in the most significant bit position of a received data character, or • An idle condition on the receive pin input signal.

Table continues on the next page...

UARTx_C1 field descriptions (continued)

Field	Description
	0 Idle line wakeup. 1 Address mark wakeup.
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** • In case the UART is programmed with ILT = 1, a logic of 1'b0 is automatically shifted after a received stop bit, therefore resetting the idle count. • In case the UART is programmed for IDLE line wakeup (RWU = 1 and WAKE = 0), ILT has no effect on when the receiver starts counting logic 1s as idle character bits. In idle line wakeup, an idle character is recognized at anytime the receiver sees 10, 11, or 12 1s depending on the M, PE, and C4[M10] fields. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables the parity function. When parity is enabled, parity function inserts a parity bit in the bit position immediately preceding the stop bit. This field must be set when C7816[ISO_7816E] is set/enabled. 0 Parity function disabled. 1 Parity function enabled.
0 PT	Parity Type Determines whether the UART generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. This field must be cleared when C7816[ISO_7816E] is set/enabled. 0 Even parity. 1 Odd parity.

45.4.4 UART Control Register 2 (UARTx_C2)

This register can be read or written at any time.

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read Write	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

UARTx_C2 field descriptions

Field	Description
7 TIE	Transmitter Interrupt or DMA Transfer Enable.

Table continues on the next page...

UARTx_C2 field descriptions (continued)

Field	Description
	Enables S1[TDRE] to generate interrupt requests or DMA transfer requests, based on the state of C5[TDMAS]. **NOTE:** If C2[TIE] and C5[TDMAS] are both set, then TCIE must be cleared, and D[D] must not be written unless servicing a DMA request. 0 TDRE interrupt and DMA transfer requests disabled. 1 TDRE interrupt or DMA transfer requests enabled.
6 TCIE	Transmission Complete Interrupt Enable Enables the transmission complete flag, S1[TC], to generate interrupt requests . 0 TC interrupt requests disabled. 1 TC interrupt requests enabled.
5 RIE	Receiver Full Interrupt or DMA Transfer Enable Enables S1[RDRF] to generate interrupt requests or DMA transfer requests, based on the state of C5[RDMAS]. 0 RDRF interrupt and DMA transfer requests disabled. 1 RDRF interrupt or DMA transfer requests enabled.
4 ILIE	Idle Line Interrupt DMA Transfer Enable Enables the idle line flag, S1[IDLE], to generate interrupt requestsor DMA transfer requests based on the state of C5[ILDMAS]. 0 IDLE interrupt requests disabled. and DMA transfer 1 IDLE interrupt requests enabled. or DMA transfer
3 TE	Transmitter Enable Enables the UART transmitter. TE can be used to queue an idle preamble by clearing and then setting TE. When C7816[ISO_7816E] is set/enabled and C7816[TTYPE] = 1, this field is automatically cleared after the requested block has been transmitted. This condition is detected when TL7816[TLEN] = 0 and four additional characters are transmitted. 0 Transmitter off. 1 Transmitter on.
2 RE	Receiver Enable Enables the UART receiver. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control This field can be set to place the UART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when C1[WAKE] is clear or an address match when C1[WAKE] is set. This field must be cleared when C7816[ISO_7816E] is set. **NOTE:** RWU must be set only with C1[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by S2[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the UART will discard data. This is because the data must be received after an IDLE is detected before IDLE is allowed to reasserted.

Table continues on the next page...

UARTx_C2 field descriptions (continued)

Field	Description
	0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	Send Break Toggling SBK sends one break character from the following: See Transmitting break characters for the number of logic 0s for the different configurations. Toggling implies clearing the SBK field before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10, 11, or 12 bits, or 13 or 14 bits). Ensure that C2[TE] is asserted atleast 1 clock before assertion of this bit. • 10, 11, or 12 logic 0s if S2[BRK13] is cleared • 13 or 14 logic 0s if S2[BRK13] is set. This field must be cleared when C7816[ISO_7816E] is set. 0 Normal transmitter operation. 1 Queue break characters to be sent.

45.4.5 UART Status Register 1 (UARTx_S1)

The S1 register provides inputs to the MCU for generation of UART interrupts or DMA requests. This register can also be polled by the MCU to check the status of its fields. To clear a flag, the status register should be read followed by a read or write to D register, depending on the interrupt flag type. Other instructions can be executed between the two steps as long the handling of I/O is not compromised, but the order of operations is important for flag clearing. When a flag is configured to trigger a DMA request, assertion of the associated DMA done signal from the DMA controller clears the flag.

NOTE

- If the condition that results in the assertion of the flag, interrupt, or DMA request is not resolved prior to clearing the flag, the flag, and interrupt/DMA request, reasserts. For example, if the DMA or interrupt service routine fails to write sufficient data to the transmit buffer to raise it above the watermark level, the flag reasserts and generates another interrupt or DMA request.
- Reading an empty data register to clear one of the flags of the S1 register causes the FIFO pointers to become misaligned. A receive FIFO flush reinitializes the pointers. A better way to prevent this situation is to always leave one byte in FIFO and this byte will be read eventually in clearing the flag bit.

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write								
Reset	1	1	0	0	0	0	0	0

UARTx_S1 field descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag TDRE will set when the number of datawords in the transmit buffer (D and C3[T8])is equal to or less than the number indicated by TWFIFO[TXWATER]. A character that is in the process of being transmitted is not included in the count. To clear TDRE, read S1 when TDRE is set and then write to the UART data register (D). For more efficient interrupt servicing, all data except the final value to be written to the buffer must be written to D/C3[T8]. Then S1 can be read before writing the final data value, resulting in the clearing of the TRDE flag. This is more efficient because the TDRE reasserts until the watermark has been exceeded. So, attempting to clear the TDRE with every write will be ineffective until sufficient data has been written. 0 The amount of data in the transmit buffer is greater than the value indicated by TWFIFO[TXWATER]. 1 The amount of data in the transmit buffer is less than or equal to the value indicated by TWFIFO[TXWATER] at some point in time since the flag has been cleared.
6 TC	Transmit Complete Flag TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by reading S1 with TC set and then doing one of the following: When C7816[ISO_7816E] is set/enabled, this field is set after any NACK signal has been received, but prior to any corresponding guard times expiring. • Writing to D to transmit new data. • Queuing a preamble by clearing and then setting C2[TE]. • Queuing a break character by writing 1 to SBK in C2. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
5 RDRF	Receive Data Register Full Flag RDRF is set when the number of datawords in the receive buffer is equal to or more than the number indicated by RWFIFO[RXWATER]. A dataword that is in the process of being received is not included in the count. To clear RDRF, read S1 when RDRF is set and then read D. For more efficient interrupt and DMA operation, read all data except the final value from the buffer, using D/C3[T8]/ED. Then read S1 and the final data value, resulting in the clearing of the RDRF flag. Even if RDRF is set, data will continue to be received until an overrun condition occurs. 0 The number of datawords in the receive buffer is less than the number indicated by RXWATER. 1 The number of datawords in the receive buffer is equal to or greater than the number indicated by RXWATER at some point in time since this flag was last cleared.
4 IDLE	Idle Line Flag After the IDLE flag is cleared, a frame must be received (although not necessarily stored in the data buffer, for example if C2[RWU] is set). To clear IDLE, read UART status S1 with IDLE set and then read D. IDLE is set when either of the following appear on the receiver input: • 10 consecutive logic 1s if C1[M] = 0 • 11 consecutive logic 1s if C1[M] = 1 and C4[M10] = 0 • 12 consecutive logic 1s if C1[M] = 1, C4[M10] = 1, and C1[PE] = 1 Idle detection is not supported when7816Eis set/enabled and hence this flag is ignored.

Table continues on the next page...

UARTx_S1 field descriptions (continued)

Field	Description
	NOTE: When RWU is set and WAKE is cleared, an idle line condition sets the IDLE flag if RWUID is set, else the IDLE flag does not become set. 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared. 1 Receiver input has become idle or the flag has not been cleared since it last asserted.
3 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the UART data registers is not affected. If the OR flag is set, no data is stored in the data buffer even if sufficient room exists. Additionally, while the OR flag is set, the RDRF and IDLE flags are blocked from asserting, that is, transition from an inactive to an active state. To clear OR, read S1 when OR is set and then read D. See functional description for more details regarding the operation of the OR bit. In 7816 mode, it is possible to configure a NACK to be returned by programing C7816[ONACK]. 0 No overrun has occurred since the last time the flag was cleared. 1 Overrun has occurred or the overrun flag has not been cleared since the last overrun occured.
2 NF	Noise Flag NF is set when the UART detects noise on the receiver input. NF does not become set in the case of an overrun. When NF is set, it indicates only that a dataword has been received with noise since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has noise or that there is only one dataword in the buffer that was received with noise unless the receive buffer has a depth of one. To clear NF, read S1 and then read D. 0 No noise detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1 then there may be data in the receiver buffer that was received with noise. 1 At least one dataword was received with noise detected since the last time the flag was cleared.
1 FE	Framing Error Flag FE is set when a logic 0 is accepted as the stop bit. FE does not set in the case of an overrun. FE inhibits further data reception until it is cleared. To clear FE, read S1 with FE set and then read D. The last data in the receive buffer represents the data that was received with the frame error enabled. Framing errors are not supported when 7816E is set/enabled. However, if this flag is set, data is still not received in 7816 mode. 0 No framing error detected. 1 Framing error.
0 PF	Parity Error Flag PF is set when PE is set and the parity of the received data does not match its parity bit. The PF is not set in the case of an overrun condition. When PF is set, it indicates only that a dataword was received with parity error since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has a parity error or that there is only one dataword in the buffer that was received with a parity error, unless the receive buffer has a depth of one. To clear PF, read S1 and then read D. Within the receive buffer structure the received dataword is tagged if it is received with a parity error. This information is available by reading the ED register prior to reading the D register. 0 No parity error detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1, then there may be data in the receive buffer what was received with a parity error. 1 At least one dataword was received with a parity error since the last time this flag was cleared.

45.4.6 UART Status Register 2 (UARTx_S2)

The S2 register provides inputs to the MCU for generation of UART interrupts or DMA requests. Also, this register can be polled by the MCU to check the status of these bits. This register can be read or written at any time, with the exception of the MSBF and RXINV bits, which should be changed by the user only between transmit and receive packets.

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	0	RXEDGIF	MSBF	RXINV	RWUID	BRK13	Reserved	RAF
Write		w1c						
Reset	0	0	0	0	0	0	0	0

UARTx_S2 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge occurs on the RxD pin. The active edge is falling if RXINV = 0, and rising if RXINV=1. RXEDGIF is cleared by writing a 1 to it. See for additional details. RXEDGIF description **NOTE:** The active edge is detected only in two wire mode and on receiving data coming from the RxD pin. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
5 MSBF	Most Significant Bit First Setting this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. This field is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. 0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1 MSB (bit8, bit7 or bit6) is the first bit that is transmitted following the start bit, depending on the setting of C1[M] and C1[PE]. Further, the first bit received after the start bit is identified as bit8, bit7, or bit6, depending on the setting of C1[M] and C1[PE].
4 RXINV	Receive Data Inversion Setting this field reverses the polarity of the received data input. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. This field is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. **NOTE:** Setting RXINV inverts the RxD input for data bits, start and stop bits, break, and idle. When C7816[ISO7816E] is set/enabled, only the data bits and the parity bit are inverted.

Table continues on the next page...

UARTx_S2 field descriptions (continued)

Field	Description
	0 Receive data is not inverted. 1 Receive data is inverted.
3 RWUID	Receive Wakeup Idle Detect When RWU is set and WAKE is cleared, this field controls whether the idle character that wakes the receiver sets S1[IDLE]. This field must be cleared when C7816[ISO7816E] is set/enabled. 0 S1[IDLE] is not set upon detection of an idle character. 1 S1[IDLE] is set upon detection of an idle character.
2 BRK13	Break Transmit Character Length Determines whether the transmit break character is 10, 11, or 12 bits long, or 13 or 14 bits long. See for the length of the break character for the different configurations. The detection of a framing error is not affected by this field. Transmitting break characters 0 Break character is 10, 11, or 12 bits long. 1 Break character is 13 or 14 bits long.
1 Reserved	Reserved. This field is reserved.
0 RAF	Receiver Active Flag RAF is set when the UART receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character when C7816[ISO7816E] is cleared/disabled. When C7816[ISO7816E] is enabled, the RAF is cleared if the C7816[TTYPE] = 0 expires or the C7816[TTYPE] = 1 expires. **NOTE:** In case C7816[ISO7816E] is set and C7816[TTYPE] = 0, it is possible to configure the guard time to 12. However, if a NACK is required to be transmitted, the data transfer actually takes 13 ETU with the 13th ETU slot being a inactive buffer. Therefore, in this situation, the RAF may deassert one ETU prior to actually being inactive. 0 UART receiver idle/inactive waiting for a start bit. 1 UART receiver active, RxD input not idle.

45.4.7 UART Control Register 3 (UARTx_C3)

Writing R8 does not have any effect. TXDIR and TXINV can be changed only between transmit and receive packets.

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C3 field descriptions

Field	Description
7 R8	Received Bit 8 R8 is the ninth data bit received when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. The R8 value corresponds to the current data value in the UARTx_D register. To read the 9th bit, read the value of UARTx_C3[R8], then read the UARTx_D register.
6 T8	Transmit Bit 8 T8 is the ninth data bit transmitted when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. **NOTE:** If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten. To correctly transmit the 9th bit, write UARTx_C3[T8] to the desired value, then write the UARTx_D register with the remaining data.
5 TXDIR	Transmitter Pin Data Direction in Single-Wire mode Determines whether the TXD pin is used as an input or output in the single-wire mode of operation. This field is relevant only to the single wire mode. When C7816[ISO7816E] is set/enabled and C7816[TTYPE] = 1, this field is automatically cleared after the requested block is transmitted. This condition is detected when TL7816[TLEN] = 0 and 4 additional characters are transmitted. Additionally, if C7816[ISO7816E] is set/enabled and C7816[TTYPE] = 0 and a NACK is being transmitted, the hardware automatically overrides this field as needed. In this situation, TXDIR does not reflect the temporary state associated with the NACK. 0 TXD pin is an input in single wire mode. 1 TXD pin is an output in single wire mode.
4 TXINV	Transmit Data Inversion. Setting this field reverses the polarity of the transmitted data output. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. This field is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. **NOTE:** Setting TXINV inverts all transmitted values, including idle, break, start, and stop bits. In loop mode, if TXINV is set, the receiver gets the transmit inversion bit when RXINV is disabled. When C7816[ISO7816E] is set/enabled then only the transmitted data bits and parity bit are inverted. 0 Transmit data is not inverted. 1 Transmit data is inverted.
3 ORIE	Overrun Error Interrupt Enable Enables the overrun error flag, S1[OR], to generate interrupt requests. 0 OR interrupts are disabled. 1 OR interrupt requests are enabled.
2 NEIE	Noise Error Interrupt Enable Enables the noise flag, S1[NF], to generate interrupt requests. 0 NF interrupt requests are disabled. 1 NF interrupt requests are enabled.
1 FEIE	Framing Error Interrupt Enable

Table continues on the next page...

UARTx_C3 field descriptions (continued)

Field	Description
	Enables the framing error flag, S1[FE], to generate interrupt requests. 0 FE interrupt requests are disabled. 1 FE interrupt requests are enabled.
0 PEIE	Parity Error Interrupt Enable Enables the parity error flag, S1[PF], to generate interrupt requests. 0 PF interrupt requests are disabled. 1 PF interrupt requests are enabled.

45.4.8 UART Data Register (UARTx_D)

This register is actually two separate registers. Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

NOTE

- In 8-bit or 9-bit data format, only UART data register (D) needs to be accessed to clear the S1[RDRF] bit (assuming receiver buffer level is less than RWFIFO[RXWATER]). The C3 register needs to be read, prior to the D register, only if the ninth bit of data needs to be captured. Similarly, the ED register needs to be read, prior to the D register, only if the additional flag data for the dataword needs to be captured.
- In the normal 8-bit mode (M bit cleared) if the parity is enabled, you get seven data bits and one parity bit. That one parity bit is loaded into the D register. So, for the data bits, mask off the parity bit from the value you read out of this register.
- When transmitting in 9-bit data format and using 8-bit write instructions, write first to transmit bit 8 in UART control register 3 (C3[T8]), then D. A write to C3[T8] stores the data in a temporary register. If D register is written first, and then the new data on data bus is stored in D, the temporary value written by the last write to C3[T8] gets stored in the C3[T8] register.

Address: Base address + 7h offset

Bit	7	6	5	4	3	2	1	0
Read Write	RT							
Reset	0	0	0	0	0	0	0	0

UARTx_D field descriptions

Field	Description
RT	Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

45.4.9 UART Match Address Registers 1 (UARTx_MA1)

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. These registers can be read and written at anytime.

Address: Base address + 8h offset

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

UARTx_MA1 field descriptions

Field	Description
MA	Match Address

45.4.10 UART Match Address Registers 2 (UARTx_MA2)

These registers can be read and written at anytime. The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded.

Address: Base address + 9h offset

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

UARTx_MA2 field descriptions

Field	Description
MA	Match Address

45.4.11 UART Control Register 4 (UARTx_C4)

Address: Base address + Ah offset

Bit	7	6	5	4	3	2	1	0
Read Write	MAEN1	MAEN2	M10	BRFA				
Reset	0	0	0	0	0	0	0	0

UARTx_C4 field descriptions

Field	Description
7 MAEN1	Match Address Mode Enable 1 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN2 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA1 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer. This field must be cleared when C7816[ISO7816E] is set/enabled.
6 MAEN2	Match Address Mode Enable 2 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN1 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA2 register. If no match occurs, the data is discarded. If a match occurs, data is transferred to the data buffer. This field must be cleared when C7816[ISO7816E] is set/enabled.
5 M10	10-bit Mode select Causes a tenth, non-memory mapped bit to be part of the serial transmission. This tenth bit is generated and interpreted as a parity bit. If M10 is set, then both C1[M] and C1[PE] must also be set. This field must be cleared when C7816[ISO7816E] is set/enabled. See Data format (non ISO-7816) for more information. 0 The parity bit is the ninth bit in the serial transmission. 1 The parity bit is the tenth bit in the serial transmission.
BRFA	Baud Rate Fine Adjust This bit field is used to add more timing resolution to the average baud frequency, in increments of 1/32. See Baud rate generation for more information.

45.4.12 UART Control Register 5 (UARTx_C5)

Address: Base address + Bh offset

Bit	7	6	5	4	3	2	1	0
Read Write	TDMAS	0	RDMAS	ILDMAS	0			
Reset	0	0	0	0	0	0	0	0

UARTx_C5 field descriptions

Field	Description
7 TDMAS	Transmitter DMA Select Configures the transmit data register empty flag, S1[TDRE], to generate interrupt or DMA requests if C2[TIE] is set. **NOTE:** • If C2[TIE] is cleared, TDRE DMA and TDRE interrupt request signals are not asserted when the TDRE flag is set, regardless of the state of TDMAS. • If C2[TIE] and TDMAS are both set, then C2[TCIE] must be cleared, and D must not be written unless a DMA request is being serviced. 0 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE interrupt request signal is asserted to request interrupt service. 1 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE DMA request signal is asserted to request a DMA transfer.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 RDMAS	Receiver Full DMA Select Configures the receiver data register full flag, S1[RDRF], to generate interrupt or DMA requests if C2[RIE] is set. **NOTE:** If C2[RIE] is cleared, and S1[RDRF] is set, the RDRF DMA and RDFR interrupt request signals are not asserted, regardless of the state of RDMAS. 0 If C2[RIE] and S1[RDRF] are set, the RDFR interrupt request signal is asserted to request an interrupt service. 1 If C2[RIE] and S1[RDRF] are set, the RDRF DMA request signal is asserted to request a DMA transfer.
4 ILDMAS	Idle Line DMA Select Configures the idle line flag, S1[IDLE], to generate interrupt or DMA requests if C2[ILIE] is set. **NOTE:** If C2[ILIE] is cleared, and S1[IDLE] is set, the IDLE DMA and IDLE interrupt request signals are not asserted, regardless of the state of ILDMAS. 0 If C2[ILIE] and S1[IDLE] are set, the IDLE interrupt request signal is asserted to request an interrupt service. 1 If C2[ILIE] and S1[IDLE] are set, the IDLE DMA request signal is asserted to request a DMA transfer.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

45.4.13 UART Extended Data Register (UARTx_ED)

This register contains additional information flags that are stored with a received dataword. This register may be read at any time but contains valid data only if there is a dataword in the receive FIFO.

NOTE

- The data contained in this register represents additional information regarding the conditions on which a dataword

was received. The importance of this data varies with the application, and in some cases maybe completely optional. These fields automatically update to reflect the conditions of the next dataword whenever D is read.

- If S1[NF] and S1[PF] have not been set since the last time the receive buffer was empty, the NOISY and PARITYE fields will be zero.

Address: Base address + Ch offset

Bit	7	6	5	4	3	2	1	0
Read	NOISY	PARITYE	0					
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_ED field descriptions

Field	Description
7 NOISY	The current received dataword contained in D and C3[R8] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.
6 PARITYE	The current received dataword contained in D and C3[R8] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

45.4.14 UART Modem Register (UARTx_MODEM)

The MODEM register controls options for setting the modem configuration.

NOTE

RXRTSE, TXRTSPOL, TXRTSE, and TXCTSE must all be cleared when C7816[ISO7816EN] is enabled. This will cause the RTS to deassert during ISO-7816 wait times. The ISO-7816 protocol does not use the RTS and CTS signals.

Address: Base address + Dh offset

Bit	7	6	5	4	3	2	1	0
Read	0				RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
Write								
Reset	0	0	0	0	0	0	0	0

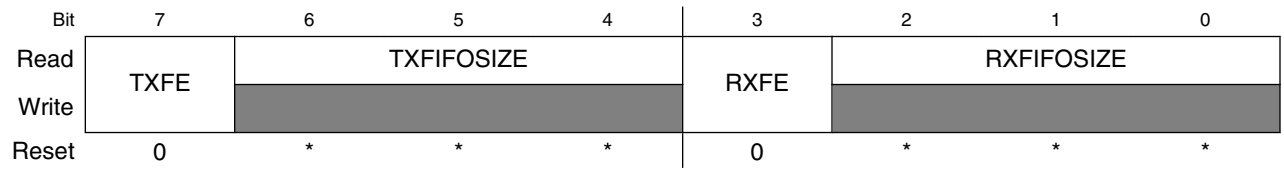
UARTx_MODEM field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. **NOTE:** Do not set both RXRTSE and TXRTSE. 0 The receiver has no effect on RTS. 1 RTS is deasserted if the number of characters in the receiver data register (FIFO) is equal to or greater than RWFIFO[RXWATER]. RTS is asserted when the number of characters in the receiver data register (FIFO) is less than RWFIFO[RXWATER]. See Hardware flow control
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set. 0 Transmitter RTS is active low. 1 Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. 0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. (FIFO) **NOTE:** Ensure that C2[TE] is asserted before assertion of this bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0 CTS has no effect on the transmitter. 1 Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

45.4.15 UART FIFO Parameters (UARTx_PFIFO)

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when C2[RE] and C2[TE] are cleared/not set and when the data buffer/FIFO is empty.

Address: Base address + 10h offset



* Notes:
- TXFIFOSIZE field: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.
- RXFIFOSIZE field: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.

UARTx_PFIFO field descriptions

Field	Description
7 TXFE	Transmit FIFO Enable When this field is set, the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this field. Additionally, TXFLUSH and RXFLUSH commands must be issued immediately after changing this field. 0 Transmit FIFO is not enabled. Buffer is depth 1. (Legacy support). 1 Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
6–4 TXFIFOSIZE	Transmit FIFO. Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only. 000 Transmit FIFO/Buffer depth = 1 dataword. 001 Transmit FIFO/Buffer depth = 4 datawords. 010 Transmit FIFO/Buffer depth = 8 datawords. 011 Transmit FIFO/Buffer depth = 16 datawords. 100 Transmit FIFO/Buffer depth = 32 datawords. 101 Transmit FIFO/Buffer depth = 64 datawords. 110 Transmit FIFO/Buffer depth = 128 datawords. 111 Reserved.
3 RXFE	Receive FIFO Enable

Table continues on the next page...

UARTx_PFIFO field descriptions (continued)

Field	Description
	When this field is set, the built in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this field. Additionally, TXFLUSH and RXFLUSH commands must be issued immediately after changing this field. 0 Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
RXFIFOSIZE	Receive FIFO. Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000 Receive FIFO/Buffer depth = 1 dataword. 001 Receive FIFO/Buffer depth = 4 datawords. 010 Receive FIFO/Buffer depth = 8 datawords. 011 Receive FIFO/Buffer depth = 16 datawords. 100 Receive FIFO/Buffer depth = 32 datawords. 101 Receive FIFO/Buffer depth = 64 datawords. 110 Receive FIFO/Buffer depth = 128 datawords. 111 Reserved.

45.4.16 UART FIFO Control Register (UARTx_CFIFO)

This register provides the ability to program various control fields for FIFO operation. This register may be read or written at any time. Note that writing to TXFLUSH and RXFLUSH may result in data loss and requires careful action to prevent unintended/ unpredictable behavior. Therefore, it is recommended that TE and RE be cleared prior to flushing the corresponding FIFO.

Address: Base address + 11h offset

Bit	7	6	5	4	3	2	1	0
Read	0	0	0			RXOFE	TXOFE	RXUFE
Write	TXFLUSH	RXFLUSH						
Reset	0	0	0	0	0	0	0	0

UARTx_CFIFO field descriptions

Field	Description
7 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register. 0 No flush operation occurs. 1 All data in the transmit FIFO/Buffer is cleared out.

Table continues on the next page...

UARTx_CFIFO field descriptions (continued)

Field	Description
6 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0 No flush operation occurs. 1 All data in the receive FIFO/buffer is cleared out.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RXOFE	Receive FIFO Overflow Interrupt Enable When this field is set, the RXOF flag generates an interrupt to the host. 0 RXOF flag does not generate an interrupt to the host. 1 RXOF flag generates an interrupt to the host.
1 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0 TXOF flag does not generate an interrupt to the host. 1 TXOF flag generates an interrupt to the host.
0 RXUFE	Receive FIFO Underflow Interrupt Enable When this field is set, the RXUF flag generates an interrupt to the host. 0 RXUF flag does not generate an interrupt to the host. 1 RXUF flag generates an interrupt to the host.

45.4.17 UART FIFO Status Register (UARTx_SFIFO)

This register provides status information regarding the transmit and receiver buffers/ FIFOs, including interrupt information. This register may be written to or read at any time.

Address: Base address + 12h offset

Bit	7	6	5	4	3	2	1	0
Read	TXEMPT	RXEMPT	0			RXOF	TXOF	RXUF
Write						w1c	w1c	w1c
Reset	1	1	0	0	0	0	0	0

UARTx_SFIFO field descriptions

Field	Description
7 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register.

Table continues on the next page...

UARTx_SFIFO field descriptions (continued)

Field	Description
	0 Transmit buffer is not empty. 1 Transmit buffer is empty.
6 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer is not empty. 1 Receive buffer is empty.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RXOF	Receiver Buffer Overflow Flag Indicates that more data has been written to the receive buffer than it can hold. This field will assert regardless of the value of CFIFO[RXOFE]. However, an interrupt will be issued to the host only if CFIFO[RXOFE] is set. This flag is cleared by writing a 1. 0 No receive buffer overflow has occurred since the last time the flag was cleared. 1 At least one receive buffer overflow has occurred since the last time the flag was cleared.
1 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of CFIFO[TXOFE]. However, an interrupt will be issued to the host only if CFIFO[TXOFE] is set. This flag is cleared by writing a 1. 0 No transmit buffer overflow has occurred since the last time the flag was cleared. 1 At least one transmit buffer overflow has occurred since the last time the flag was cleared.
0 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of CFIFO[RXUFE]. However, an interrupt will be issued to the host only if CFIFO[RXUFE] is set. This flag is cleared by writing a 1. 0 No receive buffer underflow has occurred since the last time the flag was cleared. 1 At least one receive buffer underflow has occurred since the last time the flag was cleared.

45.4.18 UART FIFO Transmit Watermark (UARTx_TWFIFO)

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when C2[TE] is not set. Changing the value of the watermark will not clear the S1[TDRE] flag.

Address: Base address + 13h offset

Bit	7	6	5	4	3	2	1	0
Read Write	TXWATER							
Reset	0	0	0	0	0	0	0	0

UARTx_TWFIFO field descriptions

Field	Description
TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt via S1[TDRE] or a DMA request via C5[TDMAS] is generated as determined by C5[TDMAS] and C2[TIE]. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by PFIFO[TXFIFOSIZE] and PFIFO[TXFE].

45.4.19 UART FIFO Transmit Count (UARTx_TCFIFO)

This is a read only register that indicates how many datawords are currently in the transmit buffer/FIFO. It may be read at any time.

Address: Base address + 14h offset

Bit	7	6	5	4	3	2	1	0
Read	TXCOUNT							
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_TCFIFO field descriptions

Field	Description
TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with PFIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.

45.4.20 UART FIFO Receive Watermark (UARTx_RWFIFO)

This register provides the ability to set a programmable threshold for notification of the need to remove data from the receiver FIFO/buffer. This register may be read at any time but must be written only when C2[RE] is not asserted. Changing the value in this register will not clear S1[RDRF].

Address: Base address + 15h offset

Bit	7	6	5	4	3	2	1	0
Read Write	RXWATER							
Reset	0	0	0	0	0	0	0	1

UARTx_RWFIFO field descriptions

Field	Description
RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is equal to or greater than the value in this register field, an interrupt via S1[RDRF] or a DMA request via C5[RDMAS] is generated as determined by C5[RDMAS] and C2[RIE]. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by PFIFO[RXFIFOSIZE] and PFIFO[RXFE] and must be greater than 0.

45.4.21 UART FIFO Receive Count (UARTx_RCFIFO)

This is a read only register that indicates how many datawords are currently in the receive FIFO/buffer. It may be read at any time.

Address: Base address + 16h offset

Bit	7	6	5	4	3	2	1	0
Read	RXCOUNT							
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_RCFIFO field descriptions

Field	Description
RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with PFIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.

45.4.22 UART 7816 Control Register (UARTx_C7816)

The C7816 register is the primary control register for ISO-7816 specific functionality. This register is specific to 7816 functionality and the values in this register have no effect on UART operation and should be ignored if ISO_7816E is not set/enabled. This register may be read at any time but values must be changed only when ISO_7816E is not set.

Address: Base address + 18h offset

Bit	7	6	5	4	3	2	1	0
Read	0			ONACK	ANACK	INIT	TTYPE	ISO_7816E
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C7816 field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 ONACK	Generate NACK on Overflow When this field is set, the receiver automatically generates a NACK response if a receive buffer overrun occurs, as indicated by S1[OR]. In many systems, this results in the transmitter resending the packet that overflowed until the retransmit threshold for that transmitter is reached. A NACK is generated only if TTYPE=0. This field operates independently of ANACK. See . Overrun NACK considerations 0 The received data does not generate a NACK when the receipt of the data results in an overflow event. 1 If the receiver buffer overflows, a NACK is automatically sent on a received character.
3 ANACK	Generate NACK on Error When this field is set, the receiver automatically generates a NACK response if a parity error occurs or if INIT is set and an invalid initial character is detected. A NACK is generated only if TTYPE = 0. If ANACK is set, the UART attempts to retransmit the data indefinitely. To stop retransmission attempts, clear C2[TE] or ISO_7816E and do not set until S1[TC] sets C2[TE] again. 0 No NACK is automatically generated. 1 A NACK is automatically generated if a parity error is detected or if an invalid initial character is detected.
2 INIT	Detect Initial Character When this field is set, all received characters are searched for a valid initial character. If an invalid initial character is identified, and ANACK is set, a NACK is sent. All received data is discarded and error flags blocked (S1[NF], S1[OR], S1[FE], S1[PF], IS7816[WT], IS7816[CWT], IS7816[BWT], IS7816[ADT], IS7816[GTV]) until a valid initial character is detected. Upon detecting a valid initial character, the configuration values S2[MSBF], C3[TXINV], and S2[RXINV] are automatically updated to reflect the initial character that was received. The actual INIT data value is not stored in the receive buffer. Additionally, upon detection of a valid initial character, IS7816[INITD] is set and an interrupt issued as programmed by IE7816[INITDE]. When a valid initial character is detected, INIT is automatically cleared. This Initial Character Detect feature is supported only in T = 0 protocol mode. 0 Normal operating mode. Receiver does not seek to identify initial character. 1 Receiver searches for initial character.
1 TTYPE	Transfer Type Indicates the transfer protocol being used. See ISO-7816 / smartcard support for more details. 0 T = 0 per the ISO-7816 specification. 1 T = 1 per the ISO-7816 specification.
0 ISO_7816E	ISO-7816 Functionality Enabled Indicates that the UART is operating according to the ISO-7816 protocol. **NOTE:** This field must be modified only when no transmit or receive is occurring. If this field is changed during a data transfer, the data being transmitted or received may be transferred incorrectly. 0 ISO-7816 functionality is turned off/not enabled. 1 ISO-7816 functionality is turned on/enabled.

45.4.23 UART 7816 Interrupt Enable Register (UARTx_IE7816)

The IE7816 register controls which flags result in an interrupt being issued. This register is specific to 7816 functionality, the corresponding flags that drive the interrupts are not asserted when 7816E is not set/enabled. However, these flags may remain set if they are asserted while 7816E was set and not subsequently cleared. This register may be read or written to at any time.

Address: Base address + 19h offset

Bit	7	6	5	4	3	2	1	0
Read Write	WTE	CWTE	BWTE	INITDE	ADTE	GTVE	TXTE	RXTE
Reset	0	0	0	0	0	0	0	0

UARTx_IE7816 field descriptions

Field	Description
7 WTE	Wait Timer Interrupt Enable 0 The assertion of IS7816[WT] does not result in the generation of an interrupt. 1 The assertion of IS7816[WT] results in the generation of an interrupt.
6 CWTE	Character Wait Timer Interrupt Enable 0 The assertion of IS7816[CWT] does not result in the generation of an interrupt. 1 The assertion of IS7816[CWT] results in the generation of an interrupt.
5 BWTE	Block Wait Timer Interrupt Enable 0 The assertion of IS7816[BWT] does not result in the generation of an interrupt. 1 The assertion of IS7816[BWT] results in the generation of an interrupt.
4 INITDE	Initial Character Detected Interrupt Enable 0 The assertion of IS7816[INITD] does not result in the generation of an interrupt. 1 The assertion of IS7816[INITD] results in the generation of an interrupt.
3 ADTE	ATR Duration Timer Interrupt Enable 0 The assertion of IS7816[ADT] does not result in the generation of an interrupt. 1 The assertion of IS7816[ADT] results in the generation of an interrupt.
2 GTVE	Guard Timer Violated Interrupt Enable 0 The assertion of IS7816[GTV] does not result in the generation of an interrupt. 1 The assertion of IS7816[GTV] results in the generation of an interrupt.
1 TXTE	Transmit Threshold Exceeded Interrupt Enable 0 The assertion of IS7816[TXT] does not result in the generation of an interrupt. 1 The assertion of IS7816[TXT] results in the generation of an interrupt.
0 RXTE	Receive Threshold Exceeded Interrupt Enable 0 The assertion of IS7816[RXT] does not result in the generation of an interrupt. 1 The assertion of IS7816[RXT] results in the generation of an interrupt.

45.4.24 UART 7816 Interrupt Status Register (UARTx_IS7816)

The IS7816 register provides a mechanism to read and clear the interrupt flags. All flags/ interrupts are cleared by writing a 1 to the field location. Writing a 0 has no effect. All bits are "sticky", meaning they indicate that only the flag condition that occurred since the last time the bit was cleared, not that the condition currently exists. The status flags are set regardless of whether the corresponding field in the IE7816 is set or cleared. The IE7816 controls only if an interrupt is issued to the host processor. This register is specific to 7816 functionality and the values in this register have no affect on UART operation and should be ignored if 7816E is not set/enabled. This register may be read or written at anytime.

Address: Base address + 1Ah offset

Bit	7	6	5	4	3	2	1	0
Read	WT	CWT	BWT	INITD	ADT	GTV	TXT	RXT
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

UARTx_IS7816 field descriptions

Field	Description
7 WT	Wait Timer Interrupt Indicates that the wait time, the time between the leading edge of a character being transmitted and the leading edge of the next response character, has exceeded the programmed value. This flag asserts only when C7816[TTYPE] = 0. This interrupt is cleared by writing 1. 0 Wait time (WT) has not been violated. 1 Wait time (WT) has been violated.
6 CWT	Character Wait Timer Interrupt Indicates that the character wait time, the time between the leading edges of two consecutive characters in a block, has exceeded the programmed value. This flag asserts only when C7816[TTYPE] = 1. This interrupt is cleared by writing 1. 0 Character wait time (CWT) has not been violated. 1 Character wait time (CWT) has been violated.
5 BWT	Block Wait Timer Interrupt Indicates that the block wait time, the time between the leading edge of first received character of a block and the leading edge of the last character the previously transmitted block, has exceeded the programmed value. This flag asserts only when C7816[TTYPE] = 1.This interrupt is cleared by writing 1. 0 Block wait time (BWT) has not been violated. 1 Block wait time (BWT) has been violated.
4 INITD	Initial Character Detected Interrupt Indicates that a valid initial character is received. This interrupt is cleared by writing 1.

Table continues on the next page...

UARTx_IS7816 field descriptions (continued)

Field	Description
	0 A valid initial character has not been received. 1 A valid initial character has been received.
3 ADT	ATR Duration Time Interrupt Indicates that the ATR duration time, the time between the leading edge of the TS character being received and the leading edge of the next response character, has exceeded the programmed value. This flag asserts only when C7816[TTYPE] = 0. This interrupt is cleared by writing 1. 0 ATR Duration time (ADT) has not been violated. 1 ATR Duration time (ADT) has been violated.
2 GTV	Guard Timer Violated Interrupt Indicates that one or more of the character guard time, block guard time, or guard time are violated. This interrupt is cleared by writing 1. 0 A guard time (GT, CGT, or BGT) has not been violated. 1 A guard time (GT, CGT, or BGT) has been violated.
1 TXT	Transmit Threshold Exceeded Interrupt Indicates that the transmit NACK threshold has been exceeded as indicated by ET7816[TXTHRESHOLD]. Regardless of whether this flag is set, the UART continues to retransmit indefinitely. This flag asserts only when C7816[TTYPE] = 0. If 7816E is cleared/disabled, ANACK is cleared/disabled, C2[TE] is cleared/disabled, C7816[TTYPE] = 1, or packet is transferred without receiving a NACK, the internal NACK detection counter is cleared and the count restarts from zero on the next received NACK. This interrupt is cleared by writing 1. 0 The number of retries and corresponding NACKS does not exceed the value in ET7816[TXTHRESHOLD]. 1 The number of retries and corresponding NACKS exceeds the value in ET7816[TXTHRESHOLD].
0 RXT	Receive Threshold Exceeded Interrupt Indicates that there are more than ET7816[RXTHRESHOLD] consecutive NACKS generated in response to parity errors on received data. This flag requires ANACK to be set. Additionally, this flag asserts only when C7816[TTYPE] = 0. Clearing this field also resets the counter keeping track of consecutive NACKS. The UART will continue to attempt to receive data regardless of whether this flag is set. If 7816E is cleared/disabled, RE is cleared/disabled, C7816[TTYPE] = 1, or packet is received without needing to issue a NACK, the internal NACK detection counter is cleared and the count restarts from zero on the next transmitted NACK. This interrupt is cleared by writing 1. 0 The number of consecutive NACKS generated as a result of parity errors and buffer overruns is less than or equal to the value in ET7816[RXTHRESHOLD]. 1 The number of consecutive NACKS generated as a result of parity errors and buffer overruns is greater than the value in ET7816[RXTHRESHOLD].

45.4.25 UART 7816 Wait Parameter Register (UARTx_WP7816)

The WP7816 register contains the WTX variable used in the generation of the block wait timer. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 1Bh offset

Bit	7	6	5	4	3	2	1	0
Read Write	WTX							
Reset	0	0	0	0	0	0	0	0

UARTx_WP7816 field descriptions

Field	Description
WTX	Wait Time Multiplier (C7816[TTYPE] = 1) Used to calculate the value used for the BWT counter. It represents a value between 0 and 255. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters.

45.4.26 UART 7816 Wait N Register (UARTx_WN7816)

The WN7816 register contains a parameter that is used in the calculation of the guard time counter. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 1Ch offset

Bit	7	6	5	4	3	2	1	0
Read Write	GTN							
Reset	0	0	0	0	0	0	0	0

UARTx_WN7816 field descriptions

Field	Description
GTN	Guard Band N Defines a parameter used in the calculation of GT, CGT, and BGT counters. The value represents an integer number between 0 and 255. See Wait time and guard time parameters .

45.4.27 UART 7816 Wait FD Register (UARTx_WF7816)

The WF7816 contains parameters that are used in the generation of various counters including GT, CGT, BGT, WT, and BWT. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 1Dh offset

Bit	7	6	5	4	3	2	1	0
Read Write	GTFD							
Reset	0	0	0	0	0	0	0	1

UARTx_WF7816 field descriptions

Field	Description
GTFD	FD Multiplier Used as another multiplier in the calculation of BWT. This value represents a number between 1 and 255. The value of 0 is invalid. This value is not used in baud rate generation. See Wait time and guard time parameters and Baud rate generation .

45.4.28 UART 7816 Error Threshold Register (UARTx_ET7816)

The ET7816 register contains fields that determine the number of NACKs that must be received or transmitted before the host processor is notified. This register may be read at anytime. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 1Eh offset

Bit	7	6	5	4	3	2	1	0
Read Write	TXTHRESHOLD				RXTHRESHOLD			
Reset	0	0	0	0	0	0	0	0

UARTx_ET7816 field descriptions

Field	Description
7–4 TXTHRESHOLD	Transmit NACK Threshold The value written to this field indicates the maximum number of failed attempts (NACKs) a transmitted character can have before the host processor is notified. This field is meaningful only when C7816[TTYPE] = 0 and C7816[ANACK] = 1. The value read from this field represents the number of consecutive NACKs that have been received since the last successful transmission. This counter saturates at 4'hF and does not wrap around. Regardless of how many NACKs that are received, the UART continues to retransmit indefinitely. This flag only asserts when C7816[TTYPE] = 0. For additional information see the IS7816[TXT] field description. 0 TXT asserts on the first NACK that is received. 1 TXT asserts on the second NACK that is received.

Table continues on the next page...

UARTx_ET7816 field descriptions (continued)

Field	Description
RXTHRESHOLD	Receive NACK Threshold The value written to this field indicates the maximum number of consecutive NACKs generated as a result of a parity error or receiver buffer overruns before the host processor is notified. After the counter exceeds that value in the field, the IS7816[RXT] is asserted. This field is meaningful only when C7816[TTYPE] = 0. The value read from this field represents the number of consecutive NACKs that have been transmitted since the last successful reception. This counter saturates at 4'hF and does not wrap around. Regardless of the number of NACKs sent, the UART continues to receive valid packets indefinitely. For additional information, see IS7816[RXT] field description.

45.4.29 UART 7816 Transmit Length Register (UARTx_TL7816)

The TL7816 register is used to indicate the number of characters contained in the block being transmitted. This register is used only when C7816[TTYPE] = 1. This register may be read at anytime. This register must be written only when C2[TE] is not enabled.

Address: Base address + 1Fh offset

Bit	7	6	5	4	3	2	1	0
Read Write	TLEN							
Reset	0	0	0	0	0	0	0	0

UARTx_TL7816 field descriptions

Field	Description
TLEN	Transmit Length This value plus four indicates the number of characters contained in the block being transmitted. This register is automatically decremented by 1 for each character in the information field portion of the block. Additionally, this register is automatically decremented by 1 for the first character of a CRC in the epilogue field. Therefore, this register must be programmed with the number of bytes in the data packet if an LRC is being transmitted, and the number of bytes + 1 if a CRC is being transmitted. This register is not decremented for characters that are assumed to be part of the Prologue field, that is, the first three characters transmitted in a block, or the LRC or last CRC character in the Epilogue field, that is, the last character transmitted. This field must be programed or adjusted only when C2[TE] is cleared.

45.4.30 UART 7816 ATR Duration Timer Register A (UARTx_AP7816A_T0)

The AP7816A_T0 register contains variables used in the generation of the ATR Duration Timer. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set, except when writing 0 to clear the ADT Counter.

NOTE

The ADT Counter starts counting on detection of the complete TS Character. It must be noted that by this time, exactly 10 ETUs have elapsed since the start bit of the TS character. The user must take this into account while programming this register.

Address: Base address + 3Ah offset

Bit	7	6	5	4	3	2	1	0
Read Write	ADTI_H							
Reset	0	0	0	0	0	0	0	0

UARTx_AP7816A_T0 field descriptions

Field	Description
ADTI_H	ATR Duration Time Integer High (C7816[TTYPE] = 0) Used to calculate the value used for the ADT Counter. This register field provides the most significant byte of the 16 bit ATR Duration Time Integer field ADTI formed by {AP7816A_T0[ADTI_H], AP7816B_T0[ADTI_L]}. Programming a value of ADTI = 0 disables the ADT counter. This value is used only when C7816[TTYPE] = 0. See ATR Duration Time Counter.

45.4.31 UART 7816 ATR Duration Timer Register B (UARTx_AP7816B_T0)

The AP7816B_T0 register contains variables used in the generation of the ATR Duration Timer. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set, except when writing 0 to clear the ADT Counter.

NOTE

The ADT Counter starts counting on detection of the complete TS Character. It must be noted that by this time, exactly 10 ETUs have elapsed since the start bit of the TS character. The user must take this into account while programming this register.

Address: Base address + 3Bh offset

Bit	7	6	5	4	3	2	1	0
Read Write	ADTI_L							
Reset	0	0	0	0	0	0	0	0

UARTx_AP7816B_T0 field descriptions

Field	Description
ADTI_L	ATR Duration Time Integer Low (C7816[TTYPE] = 0)

UARTx_AP7816B_T0 field descriptions (continued)

Field	Description
	Used to calculate the value used for the ADT counter. This register field provides the least significant byte of the 16 bit ATR Duration Time Integer field ADTI formed by {AP7816A_T0[ADTI_H], AP7816B_T0[ADTI_L]}. Programming a value of ADTI = 0 disables the ADT counter. This value is used only when C7816[TTYPE] = 0. See ATR Duration Time Counter.

45.4.32 UART 7816 Wait Parameter Register A (UARTx_WP7816A_T0)

The WP7816A_T0 register contains constants used in the generation of various wait time counters. To save register space, this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Ch offset

Bit	7	6	5	4	3	2	1	0
Read Write	WI_H							
Reset	0	0	0	0	0	0	0	0

UARTx_WP7816A_T0 field descriptions

Field	Description
WI_H	Wait Time Integer High (C7816[TTYPE] = 0) Used to calculate the value used for the WT counter. This register field provides the most significant byte of the 16 bit Wait Time Integer field WI formed by {WP7816A_T0[WI_H], WP7816B_T0[WI_L]}. The value of WI = 0 is invalid and must not be programmed. This value is used only when C7816[TTYPE] = 0. See Wait time and guard time parameters.

45.4.33 UART 7816 Wait Parameter Register A (UARTx_WP7816A_T1)

The WP7816A_T1 register contains constants used in the generation of various wait time counters. To save register space, this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Ch offset

Bit	7	6	5	4	3	2	1	0
Read Write	BWI_H							
Reset	0	0	0	0	0	0	0	0

UARTx_WP7816A_T1 field descriptions

Field	Description
BWI_H	Block Wait Time Integer High (C7816[TTYPE] = 1) Used to calculate the value used for the BWT counter. This register field provides the most significant byte of the 16 bit Block Wait Time Integer field BWI formed by {WP7816A_T1[BWI_H], WP7816B_T1[BWI_L]}. The value of BWI = 0 is invalid and should not be programmed. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters.

45.4.34 UART 7816 Wait Parameter Register B (UARTx_WP7816B_T0)

The WP7816B_T0 register contains constants used in the generation of various wait time counters. To save register space, this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Dh offset

Bit	7	6	5	4	3	2	1	0
Read Write	WI_L							
Reset	0	0	0	1	0	1	0	0

UARTx_WP7816B_T0 field descriptions

Field	Description
WI_L	Wait Time Integer Low (C7816[TTYPE] = 0) Used to calculate the value used for the WT counter. This register field provides the least significant byte of the 16 bit Wait Time Integer field WI formed by {WP7816A_T0[WI_H], WP7816B_T0[WI_L]} . The value of WI = 0 is invalid and must not be programmed. This value is used only when C7816[TTYPE] = 0. See Wait time and guard time parameters.

45.4.35 UART 7816 Wait Parameter Register B (UARTx_WP7816B_T1)

The WP7816B_T1 register contains constants used in the generation of various wait time counters. To save register space, this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Dh offset

Bit	7	6	5	4	3	2	1	0
Read Write	BWI_L							
Reset	0	0	0	1	0	1	0	0

UARTx_WP7816B_T1 field descriptions

Field	Description
BWI_L	Block Wait Time Integer Low (C7816[TTYPE] = 1) Used to calculate the value used for the BWT counter. This register field provides the least significant byte of the 16 bit Block Wait Time Integer field BWI formed by {WP7816A_T1[BWI_H], WP7816B_T1[BWI_L]}. The value of BWI = 0 is invalid and should not be programmed. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters.

45.4.36 UART 7816 Wait and Guard Parameter Register (UARTx_WGP7816_T1)

The WGP7816_T1 register contains constants used in the generation of various wait and guard timer counters. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Eh offset

Bit	7	6	5	4	3	2	1	0
Read Write	CWI1				BGI			
Reset	0	0	0	0	0	1	1	0

UARTx_WGP7816_T1 field descriptions

Field	Description
7–4 CWI1	Character Wait Time Integer 1 (C7816[TTYPE] = 1) Used to calculate the value used for the CWT counter. It represents a value between 0 and 15. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters .
BGI	Block Guard Time Integer (C7816[TTYPE] = 1) Used to calculate the value used for the BGT counter. It represent a value between 0 and 15. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters .

45.4.37 UART 7816 Wait Parameter Register C (UARTx_WP7816C_T1)

The WP7816C_T1 register contains constants used in the generation of various wait timer counters. This register may be read at any time. This register must be written to only when C7816[ISO_7816E] is not set.

Address: Base address + 3Fh offset

Bit	7	6	5	4	3	2	1	0
Read	0				CWI2			
Write								
Reset	0	0	0	0	1	0	1	1

UARTx_WP7816C_T1 field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CWI2	Character Wait Time Integer 2 (C7816[TTYPE] = 1) Used to calculate the value used for the CWT counter. It represents a value between 0 and 31. This value is used only when C7816[TTYPE] = 1. See Wait time and guard time parameters .

45.5 Functional description

This section provides a complete functional description of the UART block.

The UART allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The UART transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the UART, writes the data to be transmitted, and processes received data.

45.5.1 Transmitter

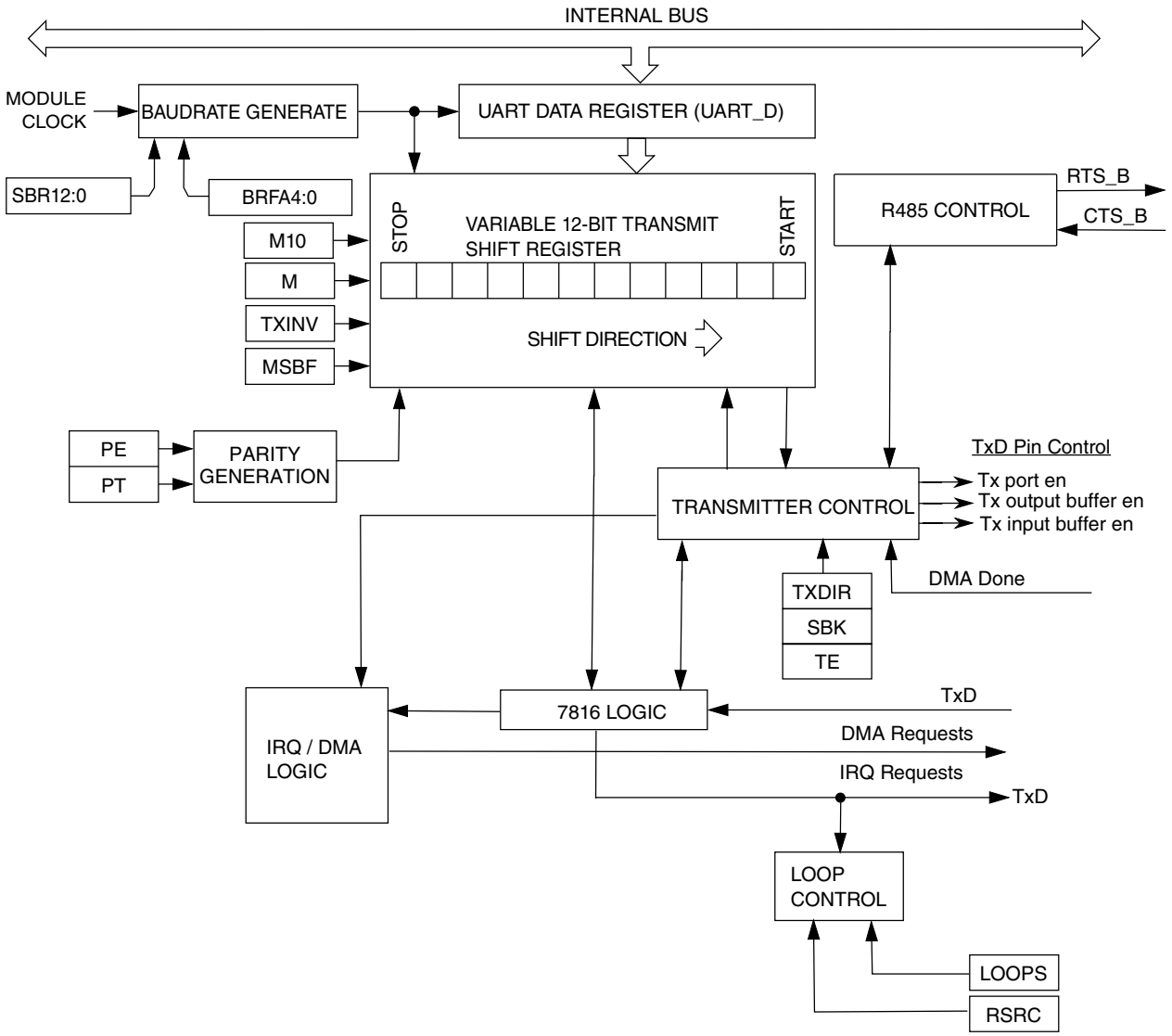


Figure 45-3. Transmitter Block Diagram

45.5.1.1 Transmitter character length

The UART transmitter can accommodate either 8, 9, or 10-bit data characters. The state of the C1[M] and C1[PE] bits and the C4[M10] bit determine the length of data characters. When transmitting 9-bit data, bit C3[T8] is the ninth bit (bit 8).

45.5.1.2 Transmission bit order

When S2[MSBF] is set, the UART automatically transmits the MSB of the data word as the first bit after the start bit. Similarly, the LSB of the data word is transmitted immediately preceding the parity bit, or the stop bit if parity is not enabled. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data written to D for transmission is completely independent of the S2[MSBF] setting.

45.5.1.3 Character transmission

To transmit data, the MCU writes the data bits to the UART transmit buffer using UART data registers C3[T8] and D. Data in the transmit buffer is then transferred to the transmitter shift register as needed. The transmit shift register then shifts a frame out through the transmit data output signal after it has prefaced it with any required start and stop bits. The UART data registers, C3[T8] and D, provide access to the transmit buffer structure.

The UART also sets a flag, the transmit data register empty flag S1[TDRE], and generates an interrupt or DMA request (C5[TDMAS]) whenever the number of datawords in the transmit buffer is equal to or less than the value indicated by TWFIFO[TXWATER]. The transmit driver routine may respond to this flag by writing additional datawords to the transmit buffer using C3[T8]/D as space permits.

See Application information for specific programing sequences.

Setting C2[TE] automatically loads the transmit shift register with the following preamble:

- 10 logic 1s if C1[M] = 0
- 11 logic 1s if C1[M] = 1 and C4[M10] = 0
- 12 logic 1s if C1[M] = 1, C4[M10] = 1, C1[PE] = 1

After the preamble shifts out, control logic transfers the data from the D register into the transmit shift register. The transmitter automatically transmits the correct start bit and stop bit before and after the dataword.

When C7816[ISO_7816E] = 1, setting C2[TE] does not result in a preamble being generated. The transmitter starts transmitting as soon as the corresponding guard time expires. When C7816[TTYPE] = 0, the value in GT is used. When C7816[TTYPE] = 1, the value in BGT is used, because C2[TE] will remain asserted until the end of the block transfer. C2[TE] is automatically cleared when C7816[TTYPE] = 1 and the block being transmitted has completed. When C7816[TTYPE] = 0, the transmitter listens for a NACK indication. If no NACK is received, it is assumed that the character was correctly

received. If a NACK is received, the transmitter resends the data, assuming that the number of retries for that character, that is, the number of NACKs received, is less than or equal to the value in ET7816[TXTHRESHOLD].

Hardware supports odd or even parity. When parity is enabled, the bit immediately preceding the stop bit is the parity bit.

When the transmit shift register is not transmitting a frame, the transmit data output signal goes to the idle condition, logic 1. If at any time software clears C2[TE], the transmitter enable signal goes low and the transmit signal goes idle.

If the software clears C2[TE] while a transmission is in progress, the character in the transmit shift register continues to shift out, provided S1[TC] was cleared during the data write sequence. To clear S1[TC], the S1 register must be read followed by a write to D register.

If S1[TC] is cleared during character transmission and C2[TE] is cleared, the transmission enable signal is deasserted at the completion of the current frame. Following this, the transmit data out signal enters the idle state even if there is data pending in the UART transmit data buffer. To ensure that all the data written in the FIFO is transmitted on the link before clearing C2[TE], wait for S1[TC] to set. Alternatively, the same can be achieved by setting TWFIFO[TXWATER] to 0x0 and waiting for S1[TDRE] to set.

45.5.1.4 Transmitting break characters

Setting C2[SBK] loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on C1[M], C1[PE], S2[BRK13] and C4[M10]. See the following table.

Table 45-4. Transmit break character length

S2[BRK13]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	—	—	10
0	1	1	0	11
0	1	1	1	12
1	0	—	—	13
1	1	—	—	14

As long as C2[SBK] is set, the transmitter logic continuously loads break characters into the transmit shift register. After the software clears C2[SBK], the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character. Break bits are not supported when C7816[ISO_7816E] is set/enabled.

NOTE

When queuing a break character, it will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that, if data is queued in the data buffer to be transmitted, the break character preempts that queued data. The queued data is then transmitted after the break character is complete.

45.5.1.5 Idle characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on C1[M], C1[PE] and C4[M10]. The preamble is a synchronizing idle character that begins the first transmission initiated after setting C2[TE]. When C7816[ISO_7816E] is set/enabled, idle characters are not sent or detected. When data is not being transmitted, the data I/O line is in an inactive state.

If C2[TE] is cleared during a transmission, the transmit data output signal becomes idle after completion of the transmission in progress. Clearing and then setting C2[TE] during a transmission queues an idle character to be sent after the dataword currently being transmitted.

Note

When queuing an idle character, the idle character will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that if data is queued in the data buffer to be transmitted, the idle character preempts that queued data. The queued data is then transmitted after the idle character is complete.

If C2[TE] is cleared and the transmission is completed, the UART is not the master of the TXD pin.

45.5.1.6 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the clear-to-send operation is enabled, the character is transmitted when CTS is asserted. If CTS is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and TXD remains in the mark state until CTS is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS. Also, if the transmitter is forced to send a continuous low condition because it is sending a break character, the transmitter ignores the state of CTS regardless of whether the clear-to-send operation is enabled.

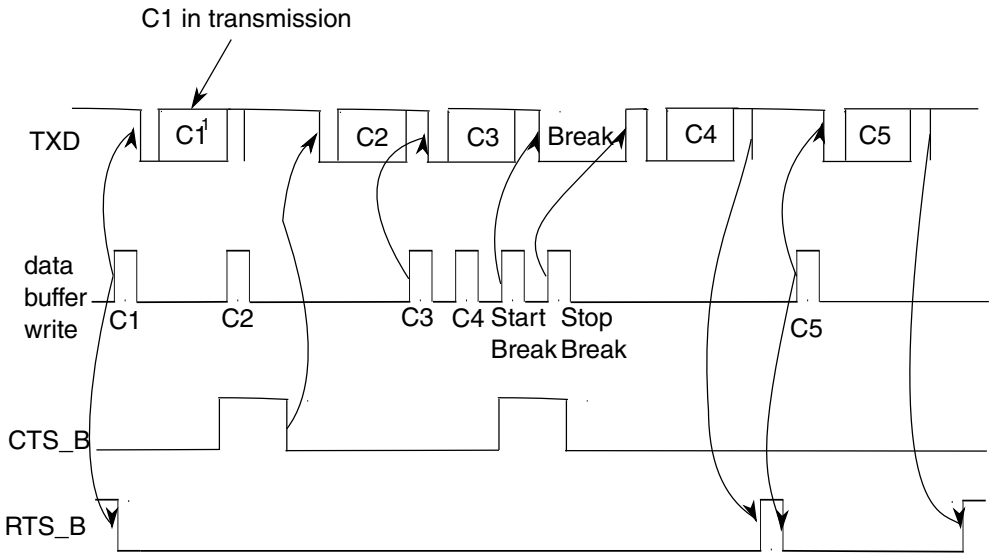
The transmitter's CTS signal can also be enabled even if the same UART receiver's RTS signal is disabled.

45.5.1.7 Transceiver driver enable

The transmitter can use RTS as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, RTS asserts one bit time before the start bit is transmitted. RTS remains asserted for the whole time that the transmitter data buffer has any characters. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts RTS, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's RTS signal asserts only when the transmitter is enabled. However, the transmitter's RTS signal is unaffected by its CTS signal. RTS will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

The following figure shows the functional timing information for the transmitter. Along with the actual character itself, TXD shows the start bit. The stop bit is also indicated, with a dashed line if necessary.



1. Cn = transmit characters

Figure 45-4. Transmitter RTS and CTS timing diagram

45.5.2 Receiver

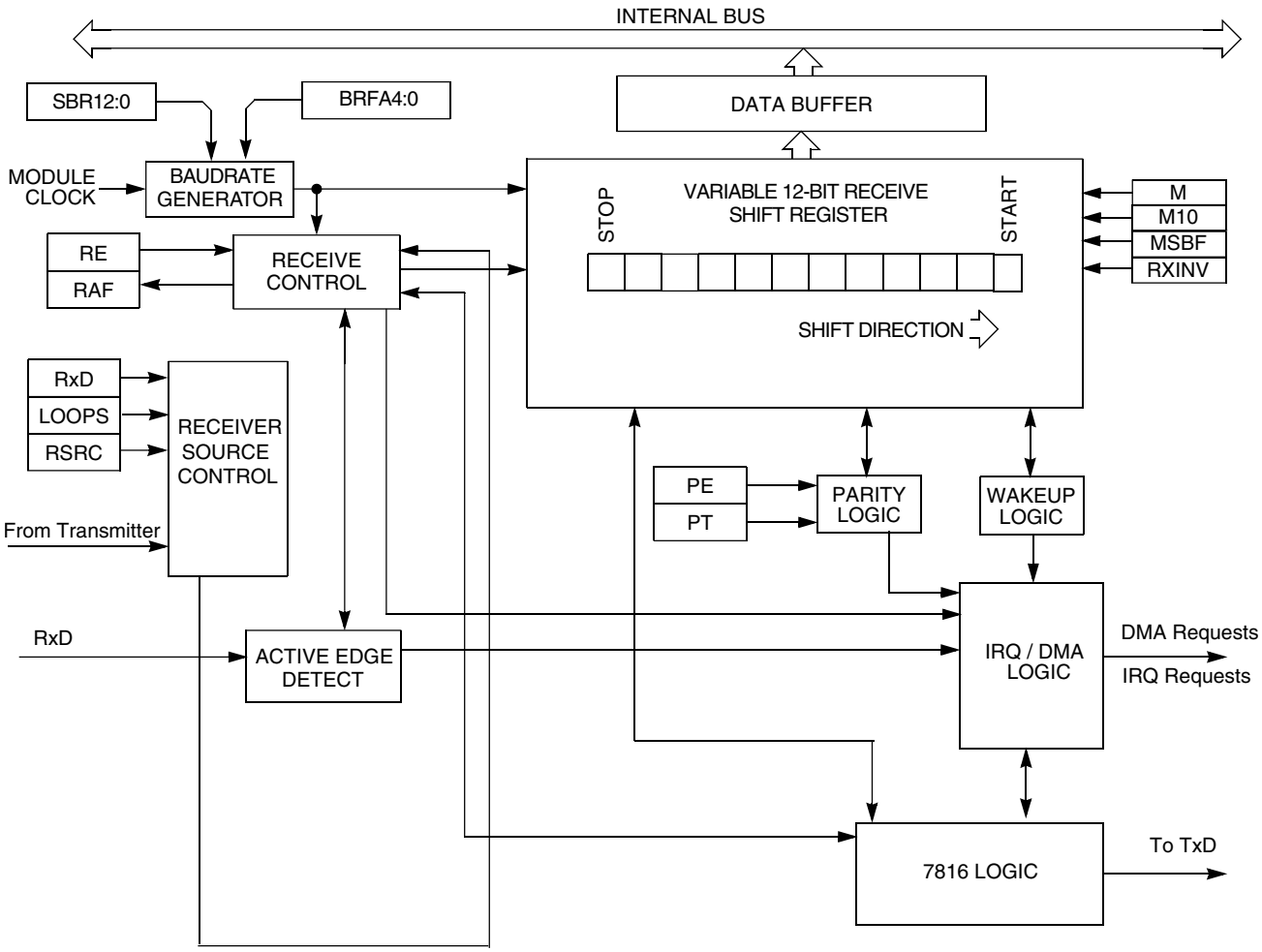


Figure 45-5. UART receiver block diagram

45.5.2.1 Receiver character length

The UART receiver can accommodate 8-, 9-, or 10-bit data characters. The states of C1[M], C1[PE] and C4[M10] determine the length of data characters. When receiving 9 or 10-bit data, C3[R8] is the ninth bit (bit 8).

45.5.2.2 Receiver bit ordering

When S2[MSBF] is set, the receiver operates such that the first bit received after the start bit is the MSB of the dataword. Similarly, the bit received immediately preceding the parity bit, or the stop bit if parity is not enabled, is treated as the LSB for the dataword. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data read from receive data buffer is completely independent of S2[MSBF].

45.5.2.3 Character reception

During UART reception, the receive shift register shifts a frame in from the unsynchronized receiver input signal. After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the UART receive buffer. Additionally, the noise and parity error flags that are calculated during the receive process are also captured in the UART receive buffer. The receive data buffer is accessible via the D and C3[T8] registers. Additional received information flags regarding the receive dataword can be read in ED register. S1[RDRF] is set if the number of resulting datawords in the receive buffer is equal to or greater than the number indicated by RWFIFO[RXWATER]. If the C2[RIE] is also set, RDRF generates an RDRF interrupt request. Alternatively, by programming C5[RDMAS], a DMA request can be generated.

When C7816[ISO_7816E] is set/enabled and C7816[TTYPE] = 0, character reception operates slightly differently. Upon receipt of the parity bit, the validity of the parity bit is checked. If C7816[ANACK] is set and the parity check fails, or if INIT and the received character is not a valid initial character, then a NACK is sent by the receiver. If the number of consecutive receive errors exceeds the threshold set by ET7816[RXTHRESHOLD], then IS7816[RXT] is set and an interrupt generated if IE7816[RXTE] is set. If an error is detected due to parity or an invalid initial character, the data is not transferred from the receive shift register to the receive buffer. Instead, the data is overwritten by the next incoming data.

When the C7816[ISO_7816E] is set/enabled, C7816[ONACK] is set/enabled, and the received character results in the receive buffer overflowing, a NACK is issued by the receiver. Additionally, S1[OR] is set and an interrupt is issued if required, and the data in the shift register is discarded.

45.5.2.4 Data sampling

The receiver samples the unsynchronized receiver input signal at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see the following figure) is re-synchronized:

- After every start bit.
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0).

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

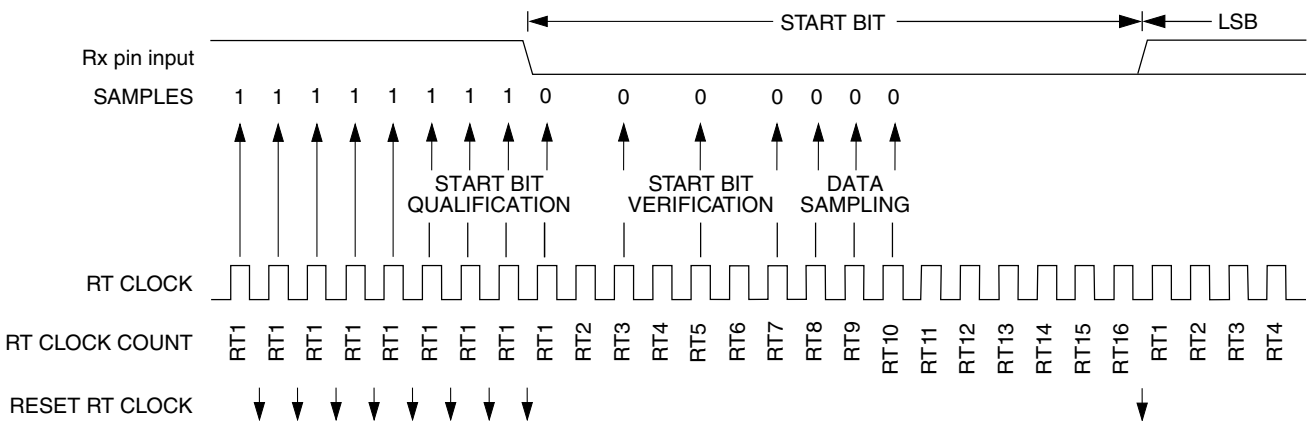


Figure 45-6. Receiver data sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7 when C7816[ISO_7816E] is cleared/disabled and RT8, RT9 and RT10 when C7816[ISO_7816E] is set/enabled. The following table summarizes the results of the start bit verification samples.

Table 45-5. Start bit verification

RT3, RT5, and RT7 samples RT8, RT9, RT10 samples when 7816E	Start bit verification	Noise flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the data bit samples.

Table 45-6. Data bit recovery

RT8, RT9, and RT10 samples	Data bit determination	Noise flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

Note

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (S1[NF]) is set and the receiver assumes that the bit is a start bit (logic 0). With the exception of when C7816[ISO_7816E] is set/enabled, where the values of RT8, RT9 and RT10 exclusively determine if a start bit exists.

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the stop bit samples. In the event that C7816[ISO_7816E] is set/enabled and C7816[TTYPE] = 0, verification of a stop bit does not take place. Rather, starting with RT8 the receiver transmits a NACK as programmed until time RT9 of the following time period. Framing Error detection is not supported when C7816[ISO_7816E] is set/enabled.

Table 45-7. Stop bit recovery

RT8, RT9, and RT10 samples	Framing error flag	Noise flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In the following figure, the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. In this example C7816[ISO_7816E] = 0. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

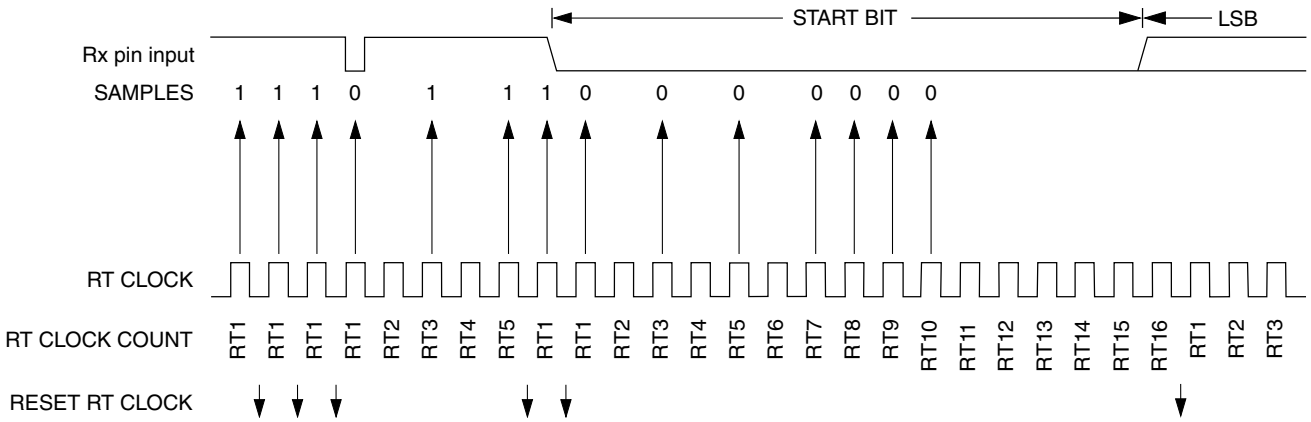


Figure 45-7. Start bit search example 1 (C7816[ISO_7816E] = 0)

In the following figure, verification sample at RT3 is high. In this example C7816[ISO_7816E] = 0. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

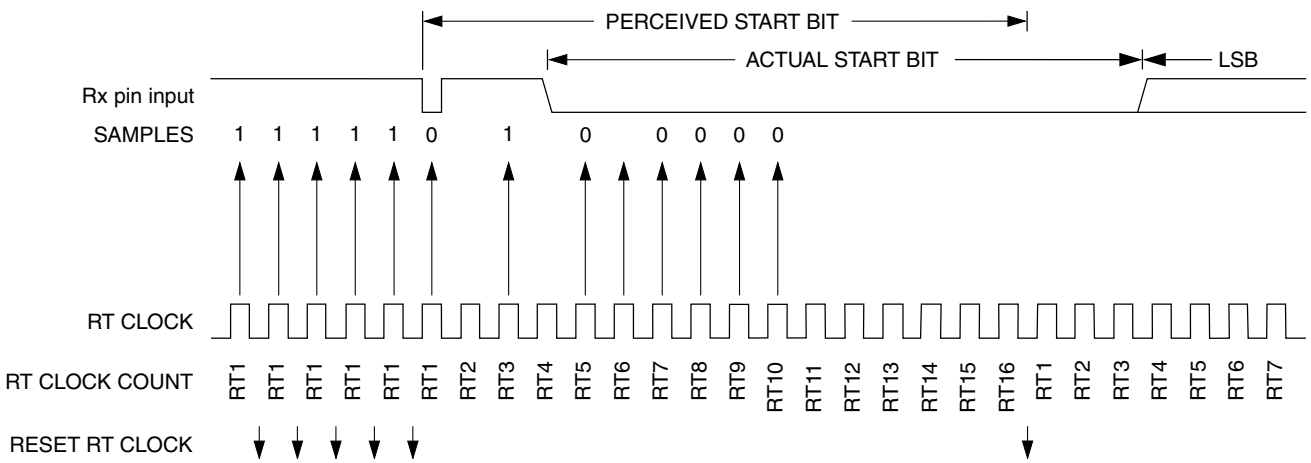


Figure 45-8. Start bit search example 2 (C7816[ISO_7816E] = 0)

In the following figure, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. In this example C7816[ISO_7816E] = 0. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

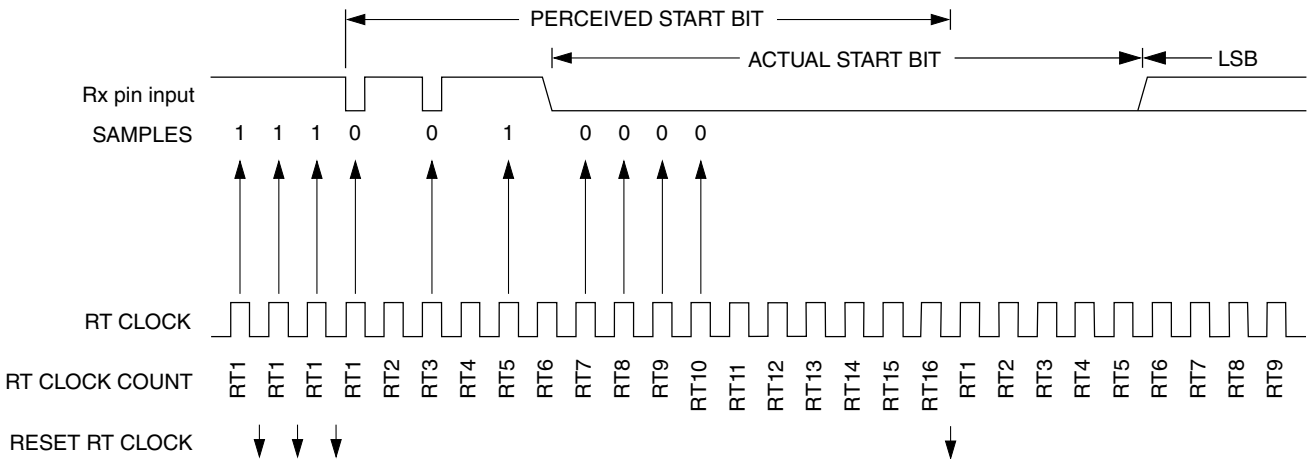


Figure 45-9. Start bit search example 3 (C7816[ISO_7816E] = 0)

The following figure shows the effect of noise early in the start bit time. In this example C7816[ISO_7816E] = 0. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

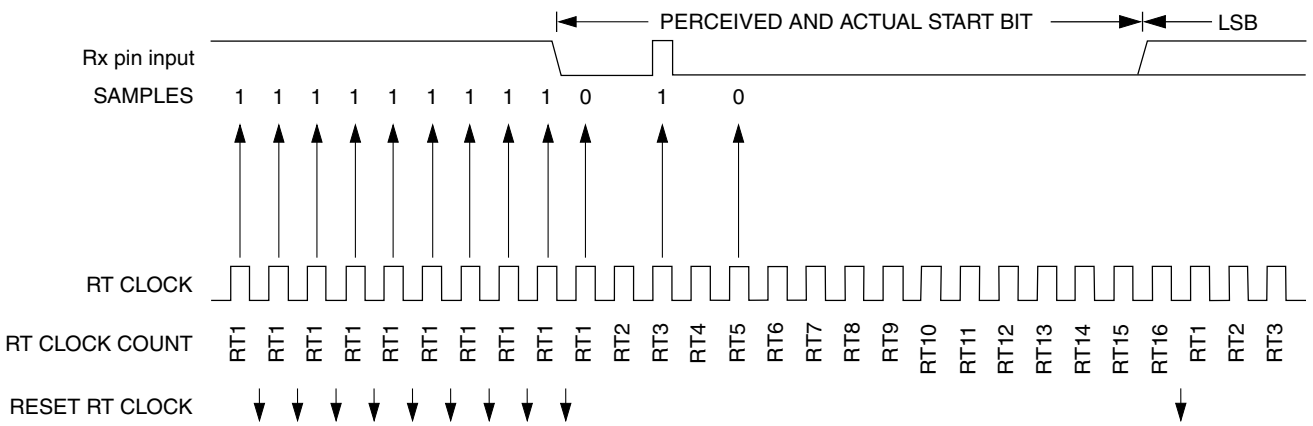


Figure 45-10. Start bit search example 4 (C7816[ISO_7816E] = 0)

The following figure shows a burst of noise near the beginning of the start bit that resets the RT clock. In this example C7816[ISO_7816E] = 0. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

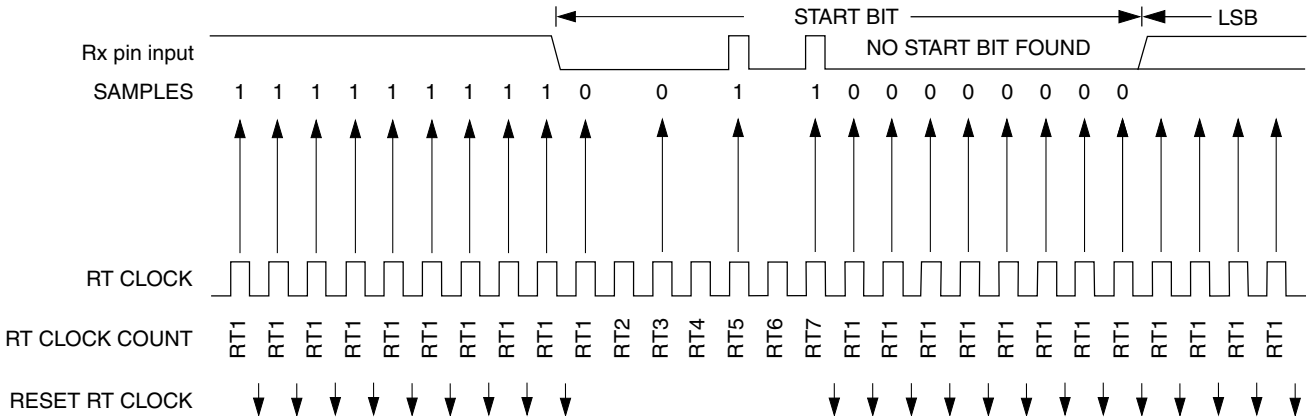


Figure 45-11. Start bit search example 5 (C7816[ISO_7816E] = 0)

In the following figure, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. In this example C7816[ISO_7816E] = 0. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored. In this example, if C7816[ISO_7816E] = 1 then a start bit would not have been detected at all since at least two of the three samples (RT8, RT9, RT10) were high.

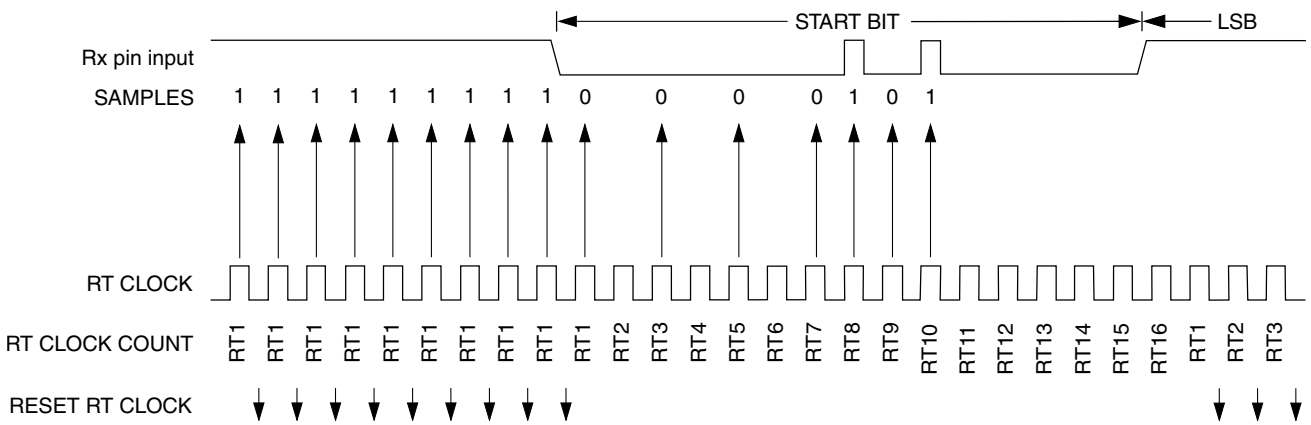


Figure 45-12. Start bit search example 6

45.5.2.5 Framing errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, S1[FE]. A break character also sets the S1[FE] because a break character has no stop bit. S1[FE] is set at the same time that received data is placed in the receive data buffer. Framing errors are not supported when C7816[ISO7816E] is set/enabled. However, if S1[FE] is set, data will not be received when C7816[ISO7816E] is set.

45.5.2.6 Receiving break characters

The UART recognizes a break character when a start bit is followed by eight, nine, or ten logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on UART registers:

- Sets the framing error flag, S1[FE].
- Writes an all 0 dataword to the data buffer, which may cause S1[RDRF] to set, depending on the watermark and number of values in the data buffer.
- May set the overrun flag, S1[OR], noise flag, S1[NF], parity error flag, S1[PE], or the receiver active flag, S2[RAF].

The UART break character detection threshold depends on C1[M], C1[PE] and C4[M10]. See the following table.

Table 45-8. Receive break character detection threshold

M	M10	PE	Threshold (bits)
0	—	—	10
1	0	—	11
1	1	1	12

Break characters are not detected or supported when C7816[ISO_7816E] is set/enabled.

45.5.2.7 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert RTS.

- RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts RTS if the number of characters in the receiver data register is equal to or greater than receiver data buffer's watermark, RWFIFO[RXWATER].
- The receiver asserts RTS when the number of characters in the receiver data register is less than the watermark. It is not affected if RDRF is asserted.
- Even if RTS is deasserted, the receiver continues to receive characters until the receiver data buffer is full or is overrun.
- If the receiver request-to-send functionality is disabled, the receiver RTS remains deasserted.

The following figure shows receiver hardware flow control functional timing. Along with the actual character itself, RXD shows the start bit. The stop bit can also indicated, with a dashed line, if necessary. The watermark is set to 2.

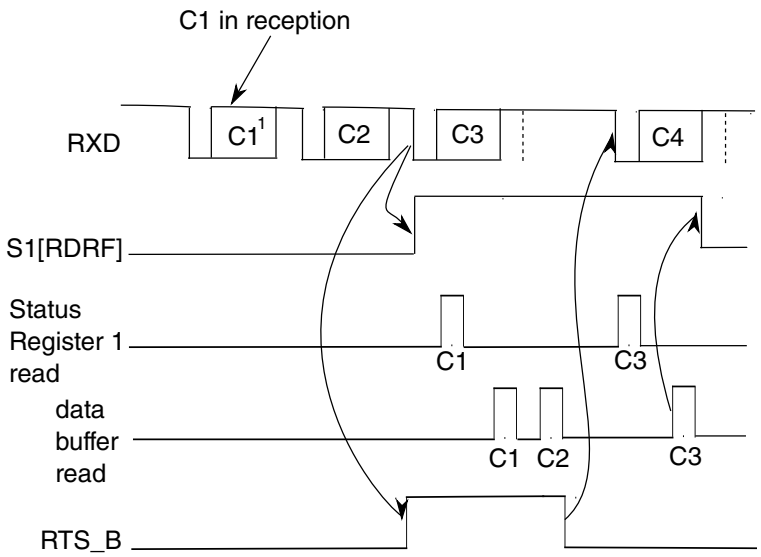


Figure 45-13. Receiver hardware flow control timing diagram

45.5.2.8 Baud rate tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic 0.

As the receiver samples an incoming frame, it resynchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames corrects a misalignment between transmitter bit times and receiver bit times.

45.5.2.8.1 Slow data tolerance

The following figure shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

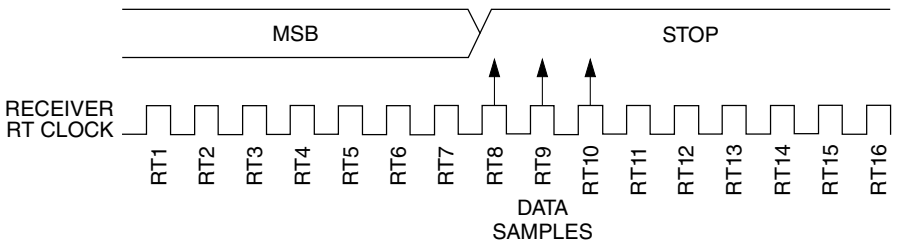


Figure 45-14. Slow data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 45-14, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 147 RT cycles (9 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((154 - 147) \div 154) \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 45-14, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 163 RT cycles (10 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((170 - 163) \div 170) \times 100 = 4.12\%$$

45.5.2.8.2 Fast data tolerance

The following figure shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

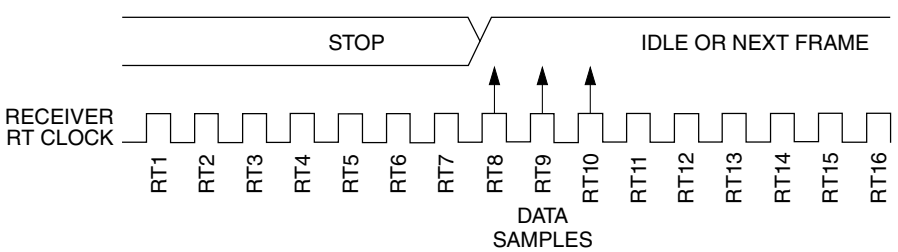


Figure 45-15. Fast data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 45-15, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 160 RT cycles (10 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$((154 - 160) \div 154) \times 100 = 3.90\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 45-15, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 176 RT cycles (11 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$((170 - 176) \div 170) \times 100 = 3.53\%$$

45.5.2.9 Receiver wakeup

C1[WAKE] determines how the UART is brought out of the standby state to process an incoming message. C1[WAKE] enables either idle line wakeup or address mark wakeup.

Receiver wakeup is not supported when C7816[ISO_7816E] is set/enabled because multi-receiver systems are not allowed.

45.5.2.9.1 Idle input line wakeup (C1[WAKE] = 0)

In this wakeup method, an idle condition on the unsynchronized receiver input signal clears C2[RWU] and wakes the UART. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another idle character appears on the unsynchronized receiver input signal.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

When C2[RWU] is 1 and S2[RWUID] is 0, the idle character that wakes the receiver does not set S1[IDLE] or the receive data register full flag, S1[RDRF]. The receiver wakes and waits for the first data character of the next message which is stored in the receive data buffer. When S2[RWUID] and C2[RWU] are set and C1[WAKE] is cleared, any idle condition sets S1[IDLE] and generates an interrupt if enabled.

Idle input line wakeup is not supported when C7816[ISO_7816E] is set/enabled.

45.5.2.9.2 Address mark wakeup (C1[WAKE] = 1)

In this wakeup method, a logic 1 in the bit position immediately preceding the stop bit of a frame clears C2[RWU] and wakes the UART. A logic 1 in the bit position immediately preceeding the stop bit marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another address frame appears on the unsynchronized receiver input signal.

A logic 1 in the bit position immediately preceding the stop bit clears the receiver's C2[RWU] after the stop bit is received and places the received data into the receiver data buffer. Note that if Match Address operation is enabled i.e. C4[MAEN1] or C4[MAEN2] is set, then received frame is transferred to receive buffer only if the comparison matches.

Address mark wakeup allows messages to contain idle characters but requires that the bit position immediately preceding the stop bit be reserved for use in address frames.

If module is in standby mode and nothing triggers to wake the UART, no error flag is set even if an invalid error condition is detected on the receiving data line.

Address mark wakeup is not supported when C7816[ISO_7816E] is set/enabled.

45.5.2.9.3 Match address operation

Match address operation is enabled when C4[MAEN1] or C4[MAEN2] is set. In this function, a frame received by the RX pin with a logic 1 in the bit position of the address mark is considered an address and is compared with the associated MA1 or MA2 register. The frame is transferred to the receive buffer, and S1[RDRF] is set, only if the comparison matches. All subsequent frames received with a logic 0 in the bit position of the address mark are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs, then no transfer is made to the receive data buffer, and all following frames with logic 0 in the bit position of the address mark are also discarded. If both C4[MAEN1] and C4[MAEN2] are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match address operation functions in the same way for both MA1 and MA2 registers. Note that the position of the address mark is the same as the Parity Bit when parity is enabled for 8 bit and 9 bit data formats.

- If only one of C4[MAEN1] and C4[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If C4[MAEN1] and C4[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

Address match operation is not supported when C7816[ISO_7816E] is set/enabled.

45.5.3 Baud rate generation

A 13-bit modulus counter and a 5-bit fractional fine-adjust counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the module clock divisor. The SBR bits are in the UART baud rate registers, BDH and BDL. The baud rate clock is synchronized with the module clock and drives the receiver. The fractional fine-adjust counter adds fractional delays to the baud rate clock to allow fine trimming of the baud rate to match the system baud rate. The transmitter is driven by the baud rate clock divided by 16. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency. This error can be reduced with the fine-adjust counter.
- Synchronization with the module clock can cause phase shift.

The Table 45-9 lists the available baud divisor fine adjust values.

UART baud rate = UART module clock / (16 × (SBR[12:0] + BRFD))

The following table lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz, with and without fractional fine adjustment.

Table 45-9. Baud rates (example: module clock = 10.2 MHz)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
17	00000	0	600,000.0	37,500.0	38,400	2.3

Table continues on the next page...

Table 45-9. Baud rates (example: module clock = 10.2 MHz) (continued)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
16	10011	19/32=0.59375	614,689.3	38,418.08	38,400	0.047
33	00000	0	309,090.9	19,318.2	19,200	0.62
33	00110	6/32=0.1875	307,344.6	19,209.04	19,200	0.047
66	00000	0	154,545.5	9659.1	9600	0.62
133	00000	0	76,691.7	4793.2	4800	0.14
266	00000	0	38,345.9	2396.6	2400	0.14
531	00000	0	19,209.0	1200.6	1200	0.11
1062	00000	0	9604.5	600.3	600	0.05
2125	00000	0	4800.0	300.0	300	0.00
4250	00000	0	2400.0	150.0	150	0.00
5795	00000	0	1760.1	110.0	110	0.00

Table 45-10. Baud rate fine adjust

BRFA	Baud Rate Fractional Divisor (BRFD)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875
1 0 0 0 0	16/32 = 0.5
1 0 0 0 1	17/32 = 0.53125
1 0 0 1 0	18/32 = 0.5625
1 0 0 1 1	19/32 = 0.59375
1 0 1 0 0	20/32 = 0.625
1 0 1 0 1	21/32 = 0.65625

Table continues on the next page...

Table 45-10. Baud rate fine adjust (continued)

BRFA	Baud Rate Fractional Divisor (BRFD)
1 0 1 1 0	22/32 = 0.6875
1 0 1 1 1	23/32 = 0.71875
1 1 0 0 0	24/32 = 0.75
1 1 0 0 1	25/32 = 0.78125
1 1 0 1 0	26/32 = 0.8125
1 1 0 1 1	27/32 = 0.84375
1 1 1 0 0	28/32 = 0.875
1 1 1 0 1	29/32 = 0.90625
1 1 1 1 0	30/32 = 0.9375
1 1 1 1 1	31/32 = 0.96875

45.5.4 Data format (non ISO-7816)

Each data character is contained in a frame that includes a start bit and a stop bit. The rest of the data format depends upon C1[M], C1[PE], S2[MSBF] and C4[M10].

45.5.4.1 Eight-bit configuration

Clearing C1[M] configures the UART for 8-bit data characters, that is, eight bits are memory mapped in D. A frame with eight data bits has a total of 10 bits. The most significant bit of the eight data bits can be used as an address mark to wake the receiver. If the most significant bit is used in this way, then it serves as an address or data indication, leaving the remaining seven bits as actual data. When C1[PE] is set, the eighth data bit is automatically calculated as the parity bit. See the following table.

Table 45-11. Configuration of 8-bit data format

UART_C1[PE]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	1	8	0	0	1
0	1	7	1 [-1]	0	1
1	1	7	0	1	1

1. The address bit identifies the frame as an address character. See Receiver wakeup.
2. The address bit identifies the frame as an address character. See Receiver wakeup.

45.5.4.2 Nine-bit configuration

When C1[M] is set and C4[M10] is cleared, the UART is configured for 9-bit data characters. If C1[PE] is enabled, the ninth bit is either C3[T8/R8] or the internally generated parity bit. This results in a frame consisting of a total of 11 bits. In the event that the ninth data bit is selected to be C3[T8], it will remain unchanged after transmission and can be used repeatedly without rewriting it, unless the value needs to be changed. This feature may be useful when the ninth data bit is being used as an address mark.

When C1[M] and C4[M10] are set, the UART is configured for 9-bit data characters, but the frame consists of a total of 12 bits. The 12 bits include the start and stop bits, the 9 data character bits, and a tenth internal data bit. Note that if C4[M10] is set, C1[PE] must also be set. In this case, the tenth bit is the internally generated parity bit. The ninth bit can either be used as an address mark or a ninth data bit.

See the following table.

Table 45-12. Configuration of 9-bit data formats

C1[PE]	UC1[M]	C1[M10]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	0	0	See Eight-bit configuration				
0	0	1	Invalid configuration				
0	1	0	1	9	0	0	1
0	1	0	1	8	1 [1]	0	1
0	1	1	Invalid Configuration				
1	0	0	See Eight-bit configuration				
1	0	1	Invalid Configuration				
1	1	0	1	8	0	1	1
1	1	1	1	9	0	1	1
1	1	1	1	8	1 [1]	1	1

1. The address bit identifies the frame as an address character.

Note

Unless in 9-bit mode with M10 set, do not use address mark wakeup with parity enabled.

45.5.4.3 Timing examples

Timing examples of these configurations in the NRZ mark/space data format are illustrated in the following figures. The timing examples show all of the configurations in the following sub-sections along with the LSB and MSB first variations.

45.5.4.3.1 Eight-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

Figure 45-16. Eight bits of data with LSB first

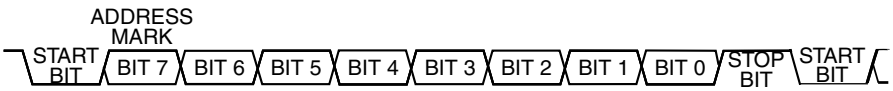


Figure 45-17. Eight bits of data with MSB first

45.5.4.3.2 Eight-bit format with parity enabled

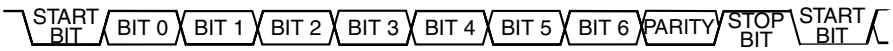


Figure 45-18. Seven bits of data with LSB first and parity

Figure 45-19. Seven bits of data with MSB first and parity

45.5.4.3.3 Nine-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

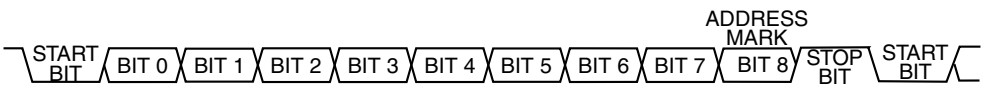


Figure 45-20. Nine bits of data with LSB first

Figure 45-21. Nine bits of data with MSB first

45.5.4.3.4 Nine-bit format with parity enabled

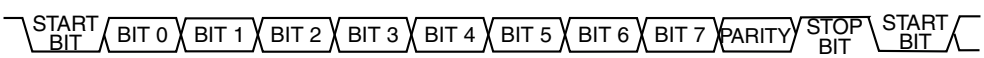


Figure 45-22. Eight bits of data with LSB first and parity

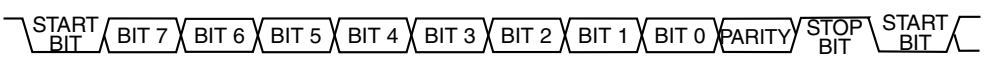


Figure 45-23. Eight bits of data with MSB first and parity

45.5.4.3.5 Non-memory mapped tenth bit for parity

The most significant memory-mapped bit can be used for address mark wakeup.

Figure 45-24. Nine bits of data with LSB first and parity

Figure 45-25. Nine bits of data with MSB first and parity

45.5.5 Single-wire operation

Normally, the UART uses two pins for transmitting and receiving. In single wire operation, the RXD pin is disconnected from the UART and the UART implements a half-duplex serial connection. The UART uses the TXD pin for both receiving and transmitting.

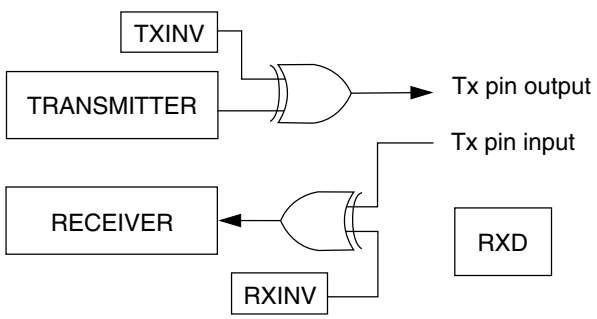


Figure 45-26. Single-wire operation (C1[LOOPS] = 1, C1[RSRC] = 1)

Enable single wire operation by setting C1[LOOPS] and the receiver source field, C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Setting C1[RSRC] connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1). When C7816[ISO_7816EN] is set, it is not required that both C2[TE] and C2[RE] are set.

45.5.6 Loop operation

In loop operation, the transmitter output goes to the receiver input. The unsynchronized receiver input signal is disconnected from the UART.

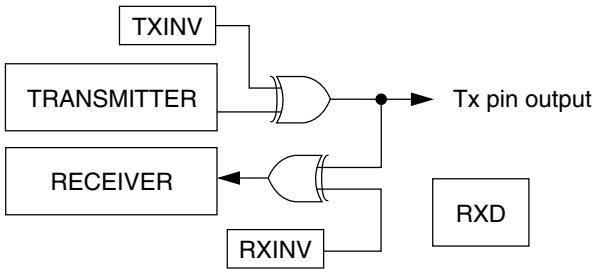


Figure 45-27. Loop operation (C1[LOOPS] = 1, C1[RSRC] = 0)

Enable loop operation by setting C1[LOOPS] and clearing C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Clearing C1[RSRC] connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1). When C7816[ISO_7816EN] is set, it is not required that both C2[TE] and C2[RE] are set.

45.5.7 ISO-7816/smartcard support

The UART provides mechanisms to support the ISO-7816 protocol that is commonly used to interface with smartcards. The ISO-7816 protocol is an NRZ, single wire, half-duplex interface. The TxD pin is used in open-drain mode because the data signal is used for both transmitting and receiving. There are multiple subprotocols within the ISO-7816 standard. The UART supports both T = 0 and T = 1 protocols. The module also provides for automated initial character detection and configuration, which allows for support of both direct convention and inverse convention data formats. A variety of interrupts specific to 7816 are provided in addition to the general interrupts to assist software. Additionally, the module is able to provide automated NACK responses and has programmed automated retransmission of failed packets. An assortment of programmable timeouts and guard band times are also supported.

The term elemental time unit (ETU) is frequently used in the context of ISO-7816. This concept is used to relate the frequency that the system (UART) is running at and the frequency that data is being transmitted and received. One ETU represents the time it takes to transmit or receive a single bit. For example, a standard 7816 packet, excluding any guard time or NACK elements is 10 ETUs (start bit, 8 data bits, and a parity bit). Guard times and wait times are also measured in ETUs.,

NOTE

The ISO-7816 specification may have certain configuration options that are reserved. To maintain maximum flexibility to support future 7816 enhancements or devices that may not strictly conform to the specification, the UART does not prevent those options being used. Further, the UART may

provide configuration options that exceed the flexibility of options explicitly allowed by the 7816 specification. Failure to correctly configure the UART may result in unexpected behavior or incompatibility with the ISO-7816 specification.

45.5.7.1 Initial characters

In ISO-7816 with T = 0 mode, the UART can be configured to use C7816[INIT] to detect the next valid initial character, referred to by the ISO-7816 specifically as a TS character. When the initial character is detected, the UART provides the host processor with an interrupt if IE7816[INITDE] is set. Additionally, the UART will alter S2[MSBF], C3[TXINV], and S2[RXINV] automatically, based on the initial character. The corresponding initial character and resulting register settings are listed in the following table.

Table 45-13. Initial character automated settings

Initial character (bit 1-10)	Initial character (hex)	MSBF	TXINV	RXINV
LHHL LLL LLH inverse convention	3F	1	1	1
LHHL HHH LLH direct convention	3B	0	0	0

S2[MSBF], C3[TXINV], and S2[RXINV] must be reset to their default values before C7816[INIT] is set. Once C7816[INIT] is set, the receiver searches all received data for the first valid initial character. Detecting a Direct Convention Initial Character will cause no change to S2[MSBF], C3[TXINV], and S2[RXINV], while detecting an Inverse Convention Initial Character will cause these fields to set automatically. All data received, which is not a valid initial character, is ignored and all flags resulting from the invalid data are blocked from asserting. If C7816[ANACK] is set, a NACK is returned for invalid received initial characters and an RXT interrupt is generated as programmed.

45.5.7.2 Protocol T = 0

When T = 0 protocol is selected, a relatively complex error detection scheme is used. Data characters are formatted as illustrated in the following figure. This scheme is also used for answer to reset and Peripheral Pin Select (PPS) formats.

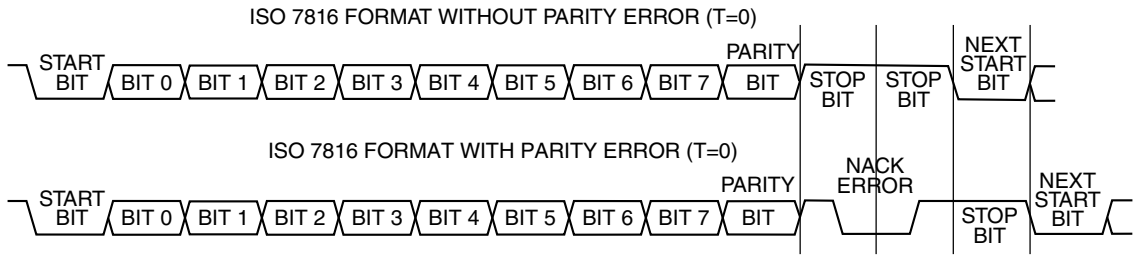


Figure 45-28. ISO-7816 T = 0 data format

As with other protocols supported by the UART, the data character includes a start bit. However, in this case, there are two stop bits rather than the typical single stop bit. In addition to a standard even parity check, the receiver has the ability to generate and return a NACK during the second half of the first stop bit period. The NACK must be at least one time period (ETU) in length and no more than two time periods (ETU) in length. The transmitter must wait for at least two time units (ETU) after detection of the error signal before attempting to retransmit the character.

It is assumed that the UART and the device (smartcard) know in advance which device is receiving and which is transmitting. No special mechanism is supplied by the UART to control receive and transmit in the mode other than C2[TE] and C2[RE]. Initial Character Detect feature is also supported in this mode.

45.5.7.3 Protocol T = 1

When T = 1 protocol is selected, the NACK error detection scheme is not used. Rather, the parity bit is used on a character basis and a CRC or LRC is used on the block basis, that is, for each group of characters. In this mode, the data format allows for a single stop bit although additional inactive bit periods may be present between the stop bit and the next start bit. Data characters are formatted as illustrated in the following figure.

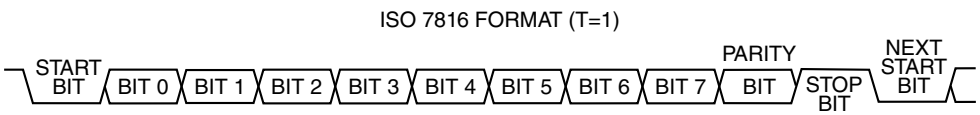


Figure 45-29. ISO 7816 T=1 data format

The smallest data unit that is transferred is a block. A block is made up of several data characters and may vary in size depending on the block type. The UART does not provide a mechanism to decode the block type. As part of the block, an LRC or CRC is included. The UART does not calculate the CRC or LRC for transmitted blocks, nor does it verify the validity of the CRC or LRC for received blocks. The 7816 protocol requires that the initiator and the smartcard (device) takes alternate turns in transmitting and receiving blocks. When the UART detects that the last character in a block has been transmitted it will automatically clear C2[TE], C3[TXDIR] and enter receive mode.

Therefore, the software must program the transmit buffer with the next data to be transmitted and then enable C2[TE] and set C3[TXDIR], once the software has determined that the last character of the received block has been received. The UART detects that the last character of the transmit block has been sent when TL7816[TLEN] = 0 and four additional characters have been sent. The four additional characters are made up of three prior to TL7816[TLEN] decrementing (prologue) and one after TL7816[TLEN] = 0, the final character of the epilogue.

45.5.7.4 Wait time and guard time parameters

The ISO-7816 specification defines several wait time and guard time parameters. The UART allows for flexible configuration and violation detection of these settings. On reset, the wait time (IS7816[WT]) defaults to 9600 ETUs and guard time (GT) to 12 ETUs. These values are controlled by parameters in the WP7816, WN7816, and WF7816 registers. Additionally, the value of C7816[TTYPE] also factors into the calculation. The formulae used to calculate the number ETUs for each wait time and guard time value are shown in Table 45-14.

Wait time (WT) is defined as the maximum allowable time between the leading edge of a character transmitted by the smartcard device and the leading edge of the previous character that was transmitted by the UART or the device. Similarly, character wait time (CWT) is defined as the maximum allowable time between the leading edge of two characters within the same block. Block wait time (BWT) is defined as the maximum time between the leading edge character of the last block received by the smartcard device and the leading edge of the first character transmitted by the smartcard device.

Guard time (GT) is defined as the minimum allowable time between the leading edge of two consecutive characters. Character guard time (CGT) is the minimum allowable time between the leading edges of two consecutive characters in the same direction, that is, transmission or reception. Block guard time (BGT) is the minimum allowable time between the leading edges of two consecutive characters in opposite directions, that is, transmission then reception or reception then transmission.

The GT and WT counters reset whenever C7816[TTYPE] = 1 or C7816[ISO_7816E] = 0 or a new dataword start bit has been received or transmitted as specified by the counter descriptions. The CWT, CGT, BWT, BGT counters reset whenever C7816[TTYPE] = 0 or C7816[ISO_7816E] = 0 or a new dataword start bit is received or transmitted as specified by the counter descriptions. When C7816[TTYPE] = 1, some of the counter values require an assumption regarding the first data transferred when the UART first starts. This assumption is required when the 7816E is disabled, when transition from C7816[TTYPE] = 0 to C7816[TTYPE] = 1 or when coming out of reset. In this case, it is assumed that the previous non-existent transfer was a received transfer.

The UART will automatically handle GT, CGT, and BGT such that the UART will not send a packet before the corresponding guard time expiring.

Table 45-14. Wait and guard time calculations

Parameter	Reset value [ETU]	C7816[TTYPE] = 0 [ETU]	C7816[TTYPE] = 1 [ETU]
Wait time (WT)	9600	WI × 480	Not used
Character wait time (CWT)	Not used	Not used	$2^{(CWI1)}$ + CWI2
Block wait time (BWT)	Not used	Not used	(11 + (BWI × 960 × GTFD)) * (WTX + 1)
Guard time (GT)	12	**GTN not equal to 255** 12 + GTN **GTN equal to 255** 12	Not used
Character guard time (CGT)	Not used	Not used	**GTN not equal to 255** 12 + GTN **GTN equal to 255** 11
Block guard time (BGT)	Not used	Not used	16 + BGI

NOTE

- User must ensure that the Character Wait time (CWT) programmed using the formula above is atleast 12. Values smaller than 12 are invalid and will lead to unexpected CWT interrupts.
- The 16 bit Wait Time integer WI is formed by concatenation of {WP7816A_T0[WI_H], WP7816B_T0[WI_L]}.
- The 16 bit Block Wait Time integer BWI is formed by concatenation of {WP7816A_T1[BWI_H], WP7816B_T1[BWI_L]}.

45.5.7.5 ATR Duration Time Counter

The ISO-7816 specification defines a specific time (in etus) within which the terminal must receive the ATR (Answer to Reset), failing which the terminal must abort the card session by initiating the deactivation sequence.

UART supports this in hardware via the ATR Duration Time (ATD) Counter which can be programmed using AP7816a_T0 and AP7816b_T0 registers. The value loaded into the ADT (ATR Duration Time) counter is given by the concatenation of the register fields as

shown; ADT = {AP7816a_T0[ADTI_H], AP7816a_T0[ADTI_L]}. This counter begins to count on detection of the TS character which is detected when IS7816[INITD] flag is set. Once the ATR process is completed, the ATD Counter must be disabled by writing 0 to AP7816x_T0 registers, in order to prevent the false occurrence of the ATD Duration Time interrupt IS7816[ATD]. Note that this feature is only supported in T = 0 mode.

NOTE

The ADT counter starts counting on detection of the complete TS Character. It must be noted that by this time, exactly 10 ETUs have elapsed since the start bit of the TS character. The user must take this into account while programming AP7816a_T0 and AP7816b_T0 registers.

45.5.7.6 Baud rate generation

The value in WF7816[GTFD] does not impact the clock frequency. SBR and BRFD are used to generate the clock frequency. This clock frequency is used by the UART only and is not seen by the smartcard device. The transmitter clocks operates at 1/16 the frequency of the receive clock so that the receiver is able to sample the received value 16 times during the ETU.

45.5.7.7 UART restrictions in ISO-7816 operation

Due to the flexibility of the UART module, there are several features and interrupts that are not supported while running in ISO-7816 mode. These restrictions are documented within the register field definitions.

45.6 Reset

All registers reset to a particular value are indicated in Memory map and registers.

45.7 System level interrupt sources

There are several interrupt signals that are sent from the UART. The following table lists the interrupt sources generated by the UART. The local enables for the UART interrupt sources are described in this table. Details regarding the individual operation of each interrupt are contained under various sub-sections of Memory map and registers.

However, RXEDGIF description also outlines additional details regarding the RXEDGIF interrupt because of its complexity of operation. Any of the UART interrupt requests listed in the table can be used to bring the CPU out of Wait mode.

Table 45-15. UART interrupt sources

Interrupt Source	Flag	Local enable	DMA select
Transmitter	TDRE	TIE	TDMAS = 0
Transmitter	TC	TCIE	-
Receiver	IDLE	ILIE	ILDMAS = 0
Receiver	RDRF	RIE	RDMAS = 0
Receiver	RXEDGIF	RXEDGIE	-
Receiver	OR	ORIE	-
Receiver	NF	NEIE	-
Receiver	FE	FEIE	-
Receiver	PF	PEIE	-
Receiver	RXUF	RXUFE	-
Transmitter	TXOF	TXOFE	-
Receiver	WT	WTWE	-
Receiver	CWT	CWTE	-
Receiver	BWT	BWTE	-
Receiver	INITD	INITDE	-
Receiver	TXT	TXTE	-
Receiver	RXT	RXTE	-
Receiver	GTV	GTVE	-

45.7.1 RXEDGIF description

S2[RXEDGIF] is set when an active edge is detected on the RxD pin. Therefore, the active edge can be detected only when in two wire mode. A RXEDGIF interrupt is generated only when S2[RXEDGIF] is set. If RXEDGIE is not enabled before S2[RXEDGIF] is set, an interrupt is not generated.

45.7.1.1 RxD edge detect sensitivity

Edge sensitivity can be software programmed to be either falling or rising. The polarity of the edge sensitivity is selected using S2[RXINV]. To detect the falling edge, S2[RXINV] is programmed to 0. To detect the rising edge, S2[RXINV] is programmed to 1.

Synchronizing logic is used prior to detect edges. Prior to detecting an edge, the receive data on RxD input must be at the deasserted logic level. A falling edge is detected when the RxD input signal is seen as a logic 1 (the deasserted level) during one module clock cycle, and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input is seen as a logic 0 during one module clock cycle and then a logic 1 during the next cycle.

45.7.1.2 Clearing RXEDGIF interrupt request

Writing a logic 1 to S2[RXEDGIF] immediately clears the RXEDGIF interrupt request even if the RxD input remains asserted. S2[RXEDGIF] remains set if another active edge is detected on RxD while attempting to clear S2[RXEDGIF] by writing a 1 to it.

45.7.1.3 Exit from low-power modes

The receive input active edge detect circuit is still active on low power modes (Wait and Stop). An active edge on the receive input brings the CPU out of low power mode if the interrupt is not masked (S2[RXEDGIF] = 1).

45.8 DMA operation

In the transmitter, S1[TDRE] can be configured to assert a DMA transfer request. In the receiver, S1[RDRF], S1[IDLE], can be configured to assert a DMA transfer request. The following table shows the configuration field settings required to configure each flag for DMA operation.

Table 45-16. DMA configuration

Flag	Request enable bit	DMA select bit
TDRE	TIE = 1	TDMAS = 1
RDRF	RIE = 1	RDMAS = 1
IDLE	ILIE = 1	ILDMAS = 1

When a flag is configured for a DMA request, its associated DMA request is asserted when the flag is set. When S1[RDRF] is configured as a DMA request, the clearing mechanism of reading S1, followed by reading D, does not clear the associated flag. The DMA request remains asserted until an indication is received that the DMA transactions

are done. When this indication is received, the flag bit and the associated DMA request is cleared. If the DMA operation failed to remove the situation that caused the DMA request, another request is issued.

45.9 Application information

This section describes the UART application information.

45.9.1 Transmit/receive data buffer operation

The UART has independent receive and transmit buffers. The size of these buffers may vary depending on the implementation of the module. The implemented size of the buffers is a fixed constant via PFIFO[TXFIFOSIZE] and PFIFO[RXFIFOSIZE]. Additionally, legacy support is provided that allows for the FIFO structure to operate as a depth of one. This is the default/reset behavior of the module and can be adjusted using the PFIFO[RXFE] and PFIFO[TXFE] bits. Individual watermark levels are also provided for transmit and receive.

There are multiple ways to ensure that a data block, which is a set of characters, has completed transmission. These methods include:

1. Set TXFIFO[TXWATER] to 0. TDRE asserts when there is no further data in the transmit buffer. Alternatively the S1[TC] flag can be used to indicate when the transmit shift register is also empty.
2. Poll TCFIFO[TXCOUNT]. Assuming that only data for a data block has been put into the data buffer, when TCFIFO[TXCOUNT] = 0, all data has been transmitted or is in the process of transmission.
3. S1[TC] can be monitored. When S1[TC] asserts, it indicates that all data has been transmitted and there is no data currently being transmitted in the shift register.

45.9.2 ISO-7816 initialization sequence

This section outlines how to program the UART for ISO-7816 operation. Elements such as procedures to power up or power down the smartcard, and when to take those actions, are beyond the scope of this description. To set up the UART for ISO-7816 operation:

1. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH has no effect without also writing to BDL. According to the 7816 specification the initial (default) baud rating setting should be Fi = 372 and Di = 1 and a maximum frequency of 5 MHz. In other words, the BDH, BDL, and C4 registers should be programmed such that the transmission frequency provided to the smartcard device must be 1/372th of the clock and must not exceed 5 MHz.
2. Write to C1 to configure word length, parity, and other configuration fields (LOOPS, RSRC) and set C1[M] = 1, C1[PE] = 1, and C1[PT] = 0.
3. Write to set S2[RWUID] = 0
4. Write to set MODEM[RXRTSE] = 0, MODEM[TXRTSPOL] = 0, MODEM[TXRTSE] = 0, and MODEM[TXCTSE] = 0.
5. Write to set up interrupt enable fields desired (C3[ORIE], C3[NEIE], C3[PEIE], and C3[FEIE])
6. Write to set C4[MAEN1] = 0 and C4[MAEN2] = 0.
7. Write to C5 register and configure DMA control register fields as desired for application.
8. Write to set C7816[INIT] = 1,C7816[TTYPE] = 0, and C7816[ISO_7816E] = 1. Program C7816[ONACK] and C7816[ANACK] as desired.
9. Write to IE7816 to set interrupt enable parameters as desired.
10. Write to ET7816 and set as desired.
11. Write to set C2[ILIE] = 0, C2[RE] = 1, C2[TE] = 1, C2[RWU] = 0, and C2[SBK] = 0. Set up interrupt enables C2[TIE], C2[TCIE], and C2[RIE] as desired.

At this time, the UART will start listening for an initial character. After being identified, it will automatically adjust S2[MSBF], C3[TXINV], and S2[RXINV]. The software must then receive and process an answer to reset. Upon processing the answer to reset, the software must write to set C2[RE] = 0 and C2[TE] = 0. The software should then adjust 7816 specific and UART generic parameters to match and configure data that was received during the answer on reset period. After the new settings have been programmed, including the new baud rate and C7816[TTYPE], C2[RE] and C2[TE] can be reenabled as required.

45.9.2.1 Transmission procedure for (C7816[TTYPE] = 0)

When the protocol selected is C7816[TTYPE] = 0, it is assumed that the software has a prior knowledge of who should be transmitting and receiving. Therefore, no mechanism is provided for automated transmission/receipt control. The software must monitor S1[TDRE], or configure for an interrupt, and provide additional data for transmission, as appropriate. Additionally, software should set C2[TE] = 1 and control TXDIR whenever it is the UART's turn to transmit information. For ease of monitoring, it is suggested that only data be transmitted until the next receiver/transmit switchover is loaded into the transmit FIFO/buffer.

45.9.2.2 Transmission procedure for (C7816[TTYPE] = 1)

When the protocol selected is C7816[TTYPE] = 1, data is transferred in blocks. Before starting a transmission, the software must write the size, in number of bytes, for the Information Field portion of the block into TLEN. If a CRC is being transmitted for the block, the value in TLEN must be one more than the size of the information field. The software must then set C2[TE] = 1 and C2[RE] = 1. The software must then monitor S1[TDRE]/interrupt and write the prologue, information, and epilogue field to the transmit buffer. TLEN automatically decrements, except for prologue bytes and the final epilogue byte. When the final epilogue byte has been transmitted, the UART automatically clears C2[TE] and C3[TXDIR] to 0, and the UART automatically starts capturing the response to the block that was transmitted. After the software has detected the receipt of the response, the transmission process must be repeated as needed with sufficient urgency to ensure that the block wait time and character wait times are not violated.

45.9.3 Initialization sequence (non ISO-7816)

To initiate a UART transmission:

1. Configure the UART.
    a. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH has no effect without also writing to BDL.
    b. Write to C1 to configure word length, parity, and other configuration bits (LOOPS, RSRC, M, WAKE, ILT, PE, and PT). Write to C4, MA1, and MA2 to configure.

c. Enable the transmitter, interrupts, receiver, and wakeup as required, by writing to C2 (TIE, TCIE, RIE, ILIE, TE, RE, RWU, and SBK), S2 (MSBF and BRK13), and C3 (ORIE, NEIE, PEIE, and FEIE). A preamble or idle character is then shifted out of the transmitter shift register.

2. Transmit procedure for each byte.

   a. Monitor S1[TDRE] by reading S1 or responding to the TDRE interrupt. The amount of free space in the transmit buffer directly using TCFIFO[TXCOUNT] can also be monitored.

   b. If the TDRE flag is set, or there is space in the transmit buffer, write the data to be transmitted to (C3[T8]/D). A new transmission will not result until data exists in the transmit buffer.

3. Repeat step 2 for each subsequent transmission.

Note

During normal operation, S1[TDRE] is set when the shift register is loaded with the next data to be transmitted from the transmit buffer and the number of datawords contained in the transmit buffer is less than or equal to the value in TWFIFO[TXWATER]. This occurs 9/16ths of a bit time after the start of the stop bit of the previous frame.

To separate messages with preambles with minimum idle line time, use this sequence between messages.

1. Write the last dataword of the first message to C3[T8]/D.
2. Wait for S1[TDRE] to go high with TWFIFO[TXWATER] = 0, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting C2[TE].
4. Write the first and subsequent datawords of the second message to C3[T8]/D.

45.9.4 Overrun (OR) flag implications

To be flexible, the overrun flag (OR) operates slight differently depending on the mode of operation. There may be implications that need to be carefully considered. This section clarifies the behavior and the resulting implications. Regardless of mode, if a dataword is received while S1[OR] is set, S1[RDRF] and S1[IDLE] are blocked from asserting. If S1[RDRF] or S1[IDLE] were previously asserted, they will remain asserted until cleared.

45.9.4.1 Overrun operation

The assertion of S1[OR] indicates that a significant event has occurred. The assertion indicates that received data has been lost because there was a lack of room to store it in the data buffer. Therefore, while S1[OR] is set, no further data is stored in the data buffer until S1[OR] is cleared. This ensures that the application will be able to handle the overrun condition.

In most applications, because the total amount of lost data is known, the application will attempt to return the system to a known state. Before S1[OR] is cleared, all received data will be dropped. For this, the software does the following.

1. Remove data from the receive data buffer. This could be done by reading data from the data buffer and processing it if the data in the FIFO was still valuable when the overrun event occurred, or using CFIFO[RXFLUSH] to clear the buffer.
2. Clear S1[OR]. Note that if data was cleared using CFIFO[RXFLUSH], then clearing S1[OR] will result in SFIFO[RXUF] asserting. This is because the only way to clear S1[OR] requires reading additional information from the FIFO. Care should be taken to disable the SFIFO[RXUF] interrupt prior to clearing the OR flag and then clearing SFIFO[RXUF] after the OR flag has been cleared.

Note that, in some applications, if an overrun event is responded to fast enough, the lost data can be recovered. For example, when C7816[ISO_7816E] is asserted, C7816[TTYPE]=1 and C7816[ONACK] = 1, the application may reasonably be able to determine whether the lost data will be resent by the device. In this scenario, flushing the receiver data buffer may not be required. Rather, if S1[OR] is cleared, the lost data may be resent and therefore may be recoverable.

45.9.5 Overrun NACK considerations

When C7816[ISO_7816E] is enabled and C7816[TTYPE] = 0, the retransmission feature of the 7816 protocol can be used to help avoid lost data when the data buffer overflows. Using C7816[ONACK], the module can be programmed to issue a NACK on an overflow event. Assuming that the smartcard device has implemented retransmission, the lost data will be retransmitted. While useful, there is a programming implication that may require special consideration. The need to transmit a NACK must be determined and committed to prior to the dataword being fully received. While the NACK is being received, it is possible that the application code will read the data buffer such that sufficient room will be made to store the dataword that is being NACKed. Even if room

has been made in the data buffer after the transmission of a NACK is completed, the received data will always be discarded as a result of an overflow and the ET7816[RXTHRESHOLD] value will be incremented by one. However, if sufficient space now exists to write the received data which was NACK'ed, S1[OR] will be blocked and kept from asserting.

45.9.6 Match address registers

The two match address registers allow a second match address function for a broadcast or general call address to the serial bus, as an example.

45.9.7 Modem feature

This section describes the modem features.

45.9.7.1 Ready-to-receive using RTS

To help to stop overrun of the receiver data buffer, the RTS signal can be used by the receiver to indicate to another UART that it is ready to receive data. The other UART can send the data when its CTS signal is asserted. This handshaking conforms to the TIA-232-E standard. A transceiver is necessary if the required voltage levels of the communication link do not match the voltage levels of the UART's RTS and CTS signals.

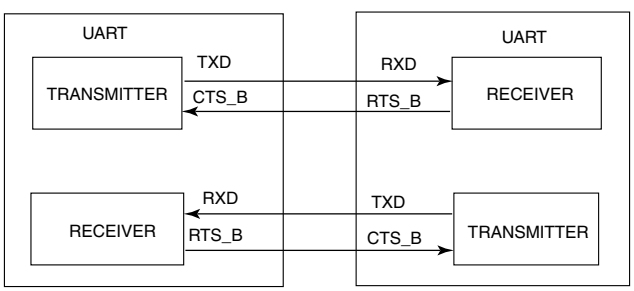


Figure 45-30. Ready-to-receive

The transmitter's CTS signal can be used for hardware flow control whether its RTS signal is used for hardware flow control, transceiver driver enable, or not at all.

45.9.7.2 Transceiver driver enable using RTS

RS-485 is a multiple drop communication protocol in which the UART transceiver's driver is 3-stated unless the UART is driving. The RTS signal can be used by the transmitter to enable the driver of a transceiver. The polarity of RTS can be matched to the polarity of the transceiver's driver enable signal. See the following figure.

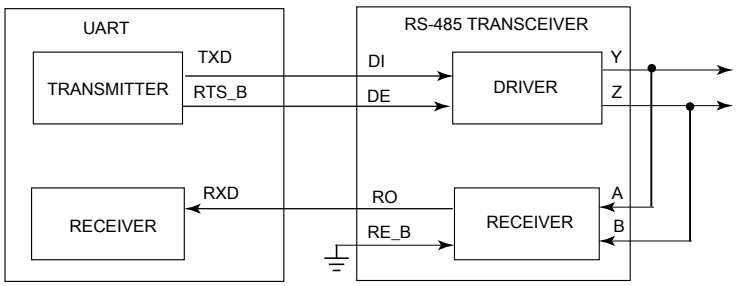


Figure 45-31. Transceiver driver enable using RTS

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. This option can be refined further by operating the UART in single wire mode, freeing the RXD pin for other uses.

45.9.8 Clearing 7816 wait timer (WT, BWT, CWT) interrupts

The 7816 wait timer interrupts associated with IS7816[WT], IS7816[BWT], and IS7816[CWT] will automatically reassert if they are cleared and the wait time is still violated. This behavior is similar to most of the other interrupts on the UART. In most cases, if the condition that caused the interrupt to trigger still exists when the interrupt is cleared, then the interrupt will reassert. For example, consider the following scenario:

1. IS7816[WT] is programmed to assert after 9600 cycles of unresponsiveness.
2. The 9600 cycles pass without a response resulting in the WT interrupt asserting.
3. The IS7816[WT] is cleared at cycle 9700 by the interrupt service routine.
4. After the WT interrupt has been cleared, the smartcard remains unresponsive. At cycle 9701 the WT interrupt will be reasserted.

If the intent of clearing the interrupt is such that it does not reassert, the interrupt service routine must remove or clear the condition that originally caused the interrupt to assert prior to clearing the interrupt. There are multiple ways that this can be accomplished, including ensuring that an event that results in the wait timer resetting occurs, such as, the transmission of another packet.

45.9.9 Legacy and reverse compatibility considerations

Recent versions of the UART have added several new features. Whenever reasonably possible, reverse compatibility was maintained. However, in some cases this was either not feasible or the behavior was deemed as not intended. This section describes several differences to legacy operation that resulted from these recent enhancements. If application code from previous versions is used, it must be reviewed and modified to take the following items into account. Depending on the application code, additional items that are not listed here may also need to be considered.

1. Various reserved registers and register bits are used, such as, MSFB and M10.
2. This module now generates an error when invalid address spaces are used.
3. While documentation indicated otherwise, in some cases it was possible for S1[IDLE] to assert even if S1[OR] was set.
4. S1[OR] will be set only if the data buffer (FIFO) does not have sufficient room. Previously, the data buffer was always a fixed size of one and the S1[OR] flag would set so long as S1[RDRF] was set even if there was room in the data buffer. While the clearing mechanism has remained the same for S1[RDRF], keeping the OR flag assertion tied to the RDRF event rather than the data buffer being full would have greatly reduced the usefulness of the buffer when its size is larger than one.
5. Previously, when C2[RWU] was set (and WAKE = 0), the IDLE flag could reassert up to every bit period causing an interrupt and requiring the host processor to reassert C2[RWU]. This behavior has been modified. Now, when C2[RWU] is set (and WAKE = 0), at least one non-idle bit must be detected before an idle can be detected.

Chapter 46
Low-Power Universal Asynchronous Receiver/ Transmitter (LPUART)

46.1 Chip-specific LPUART information

46.1.1 Overview

The LPUART module supports basic UART with DMA interface function and ×4 to ×32 oversampling of baud-rate.

The module can keep functional in VLPS mode provided the clock it is using remains enabled.

LPUART combines all standard UART interrupt sources into a single interrupt request. It supports MODEM and IrDA, while does not support LON or ISO7816.

This module supports LIN slave operation.

46.1.2 LPUART0 Clocking

LPUART0 can select its clock source from one of the following clocks:

- MCGFLLCLK
- MCGPLLCLK
- MCGIRCLK
- OSCERCLK
- OSC32KCLK (RTC Oscillator output)
- BUSCLK

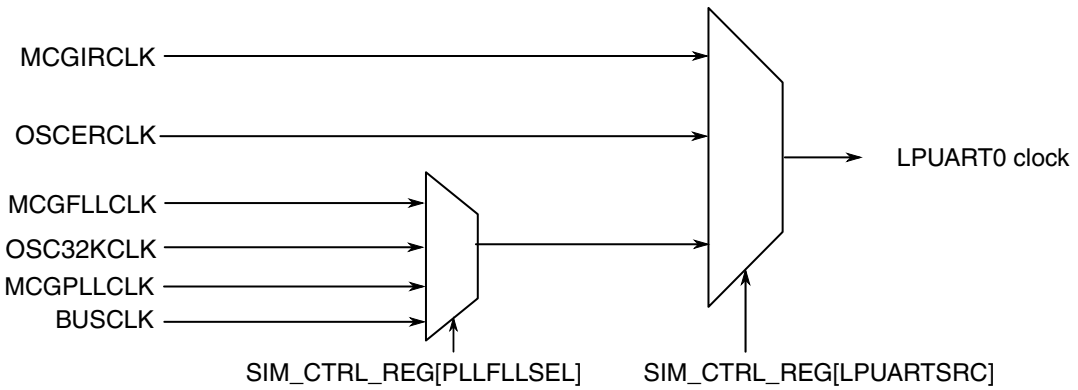


Figure 46-1. LPUART0 clock generation

46.2 Introduction

46.2.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
  - Receive data match

- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit break character generation / 11-bit break character detection
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64 or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width

46.2.2 Modes of operation

46.2.2.1 Stop mode

The LPUART will remain functional during Stop mode, provided the asynchronous transmit and receive clock remains enabled. The LPUART can generate an interrupt or DMA request to cause a wakeup from Stop mode.

46.2.2.2 Wait mode

The LPUART can be configured to Stop in Wait modes, when the DOZEEN bit is set. The transmitter and receiver will finish transmitting/receiving the current word.

46.2.2.3 Debug mode

The LPUART remains functional in debug mode.

46.2.3 Signal Descriptions

Signal	Description	I/O
LPUART_TX	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is	I/O

Table continues on the next page...

Signal	Description	I/O
	disabled or transmit direction is configured for receive data.	
LPUART_RX	Receive data.	I
LPUART_CTS	Clear to send.	I
LPUART_RTS	Request to send.	O

46.2.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

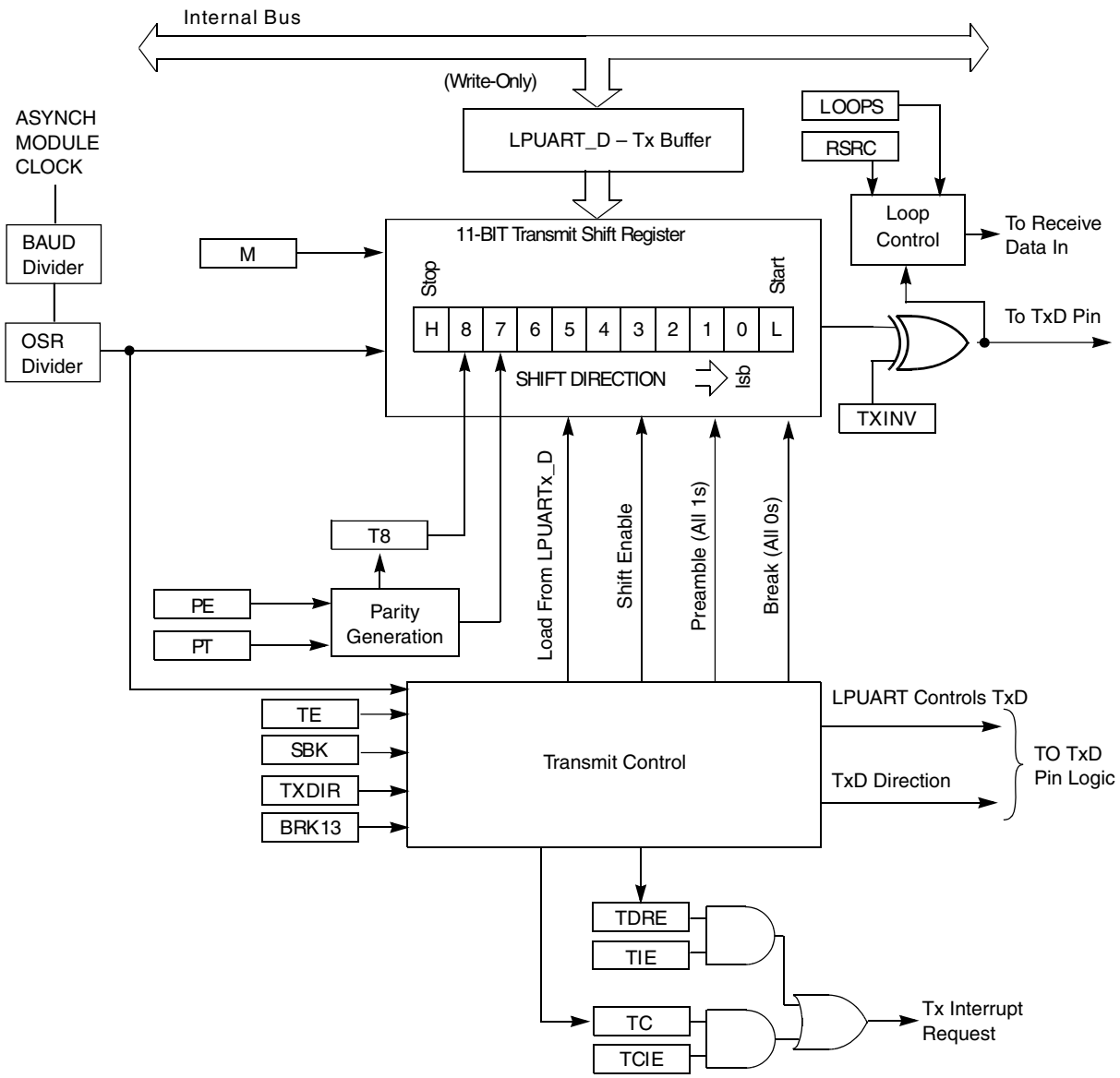


Figure 46-2. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

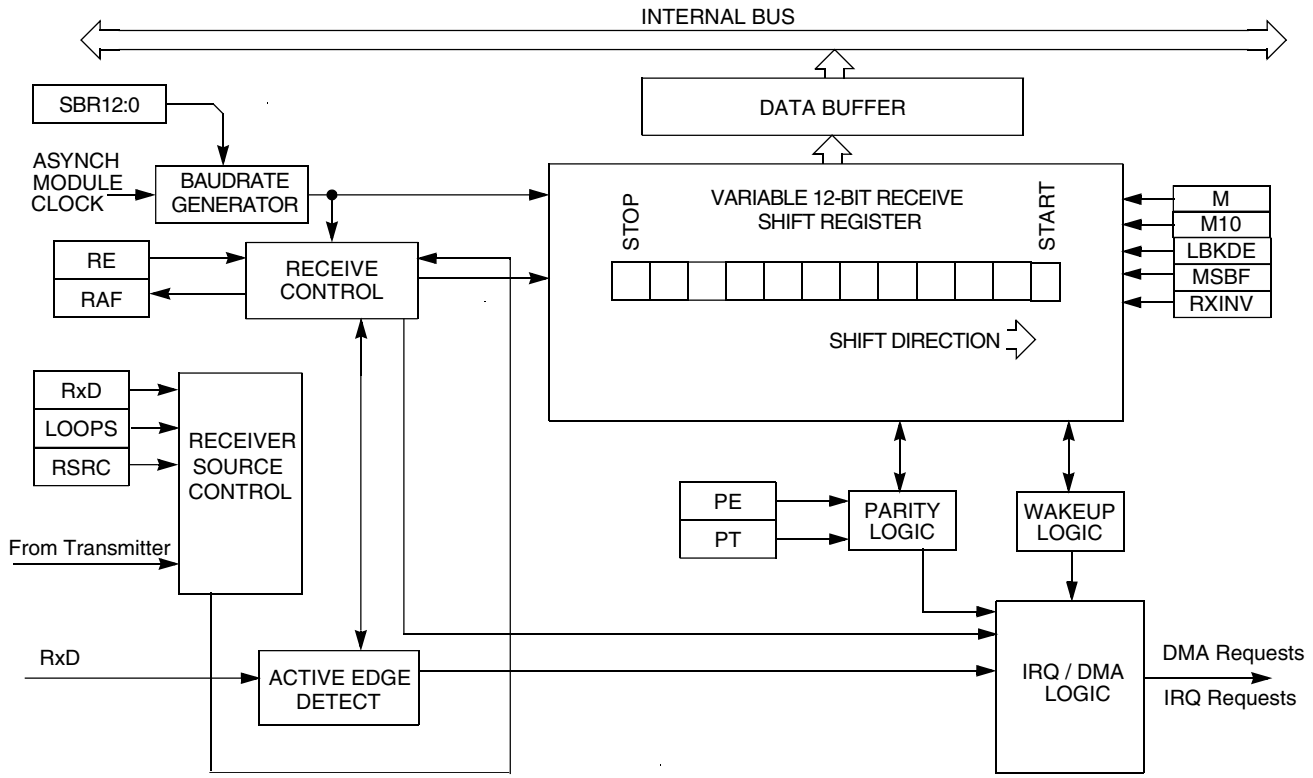


Figure 46-3. LPUART receiver block diagram

46.3 Register definition

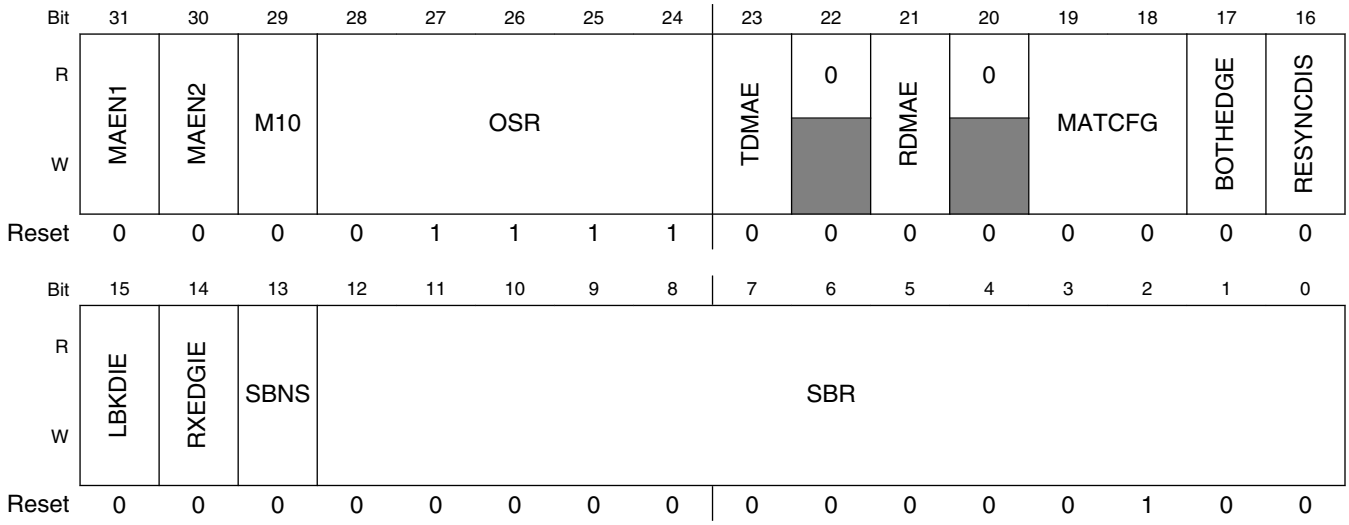
The LPUART includes registers to control baud rate, select LPUART options, report LPUART status, and for transmit/receive data. Access to an address outside the valid memory map will generate a bus error.

LPUART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_A000	LPUART Baud Rate Register (LPUART0_BAUD)	32	R/W	0F00_0004h	46.3.1/1018
4002_A004	LPUART Status Register (LPUART0_STAT)	32	R/W	00C0_0000h	46.3.2/1020
4002_A008	LPUART Control Register (LPUART0_CTRL)	32	R/W	0000_0000h	46.3.3/1024
4002_A00C	LPUART Data Register (LPUART0_DATA)	32	R/W	0000_1000h	46.3.4/1029
4002_A010	LPUART Match Address Register (LPUART0_MATCH)	32	R/W	0000_0000h	46.3.5/1031
4002_A014	LPUART Modem IrDA Register (LPUART0_MODIR)	32	R/W	0000_0000h	46.3.6/1031

46.3.1 LPUART Baud Rate Register (LPUARTx_BAUD)

Address: 4002_A000h base + 0h offset = 4002_A000h



LPUARTx_BAUD field descriptions

Field	Description
31 MAEN1	Match Address Mode Enable 1 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA1].
30 MAEN2	Match Address Mode Enable 2 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA2].
29 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled. 0 Receiver and transmitter use 8-bit or 9-bit data characters. 1 Receiver and transmitter use 10-bit data characters.
28–24 OSR	Over Sampling Ratio This field configures the oversampling ratio for the receiver between 4x (00011) and 32x (11111). Writing an invalid oversampling ratio will default to an oversampling ratio of 16 (01111). This field should only be changed when the transmitter and receiver are both disabled.
23 TDMAE	Transmitter DMA Enable TDMAE configures the transmit data register empty flag, LPUART_STAT[TDRE], to generate a DMA request. 0 DMA request disabled. 1 DMA request enabled.

Table continues on the next page...

LPUARTx_BAUD field descriptions (continued)

Field	Description
22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 RDMAE	Receiver Full DMA Enable RDMAE configures the receiver data register full flag, LPUART_STAT[RDRF], to generate a DMA request. 0 DMA request disabled. 1 DMA request enabled.
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 MATCFG	Match Configuration Configures the match addressing mode used. 00 Address Match Wakeup 01 Idle Match Wakeup 10 Match On and Match Off 11 Enables RWU on Data Match and Match On/Off for transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled. 0 Receiver samples input data using the rising edge of the baud rate clock. 1 Receiver samples input data using the rising and falling edge of the baud rate clock.
16 RESYNCDIS	Resynchronization Disable When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled. 0 Resynchronization during received data word is supported 1 Resynchronization during received data word is disabled
15 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, LBKDIF, to generate interrupt requests. 0 Hardware interrupts from LPUART_STAT[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[LBKDIF] flag is 1.
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the RXEDGIF to set. 0 Hardware interrupts from LPUART_STAT[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[RXEDGIF] flag is 1.
13 SBNS	Stop Bit Number Select SBNS determines whether data characters are one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled.

Table continues on the next page...

LPUARTx_BAUD field descriptions (continued)

Field	Description
	0 One stop bit. 1 Two stop bits.
SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) × SBR)". The 13-bit baud rate setting [SBR12:SBR0] must only be updated when the transmitter and receiver are both disabled (LPUART_CTRL[RE] and LPUART_CTRL[TE] are both 0).

46.3.2 LPUART Status Register (LPUARTx_STAT)

Address: 4002_A000h base + 4h offset = 4002_A004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	w1c	w1c										w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0													
W	w1c	w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPUARTx_STAT field descriptions

Field	Description
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it.

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

Field	Description
	0 No LIN break character has been detected. 1 LIN break character has been detected.
30 RXEDGIF	LPUART_RX Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the LPUART_RX pin occurs. RXEDGIF is cleared by writing a 1 to it. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
29 MSBF	MSB First Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1 MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE].
28 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. **NOTE:** Setting RXINV inverts the LPUART_RX input for all cases: data bits, start and stop bits, break, and idle. 0 Receive data not inverted. 1 Receive data inverted.
27 RWUID	Receive Wake Up Idle Detect For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled. 0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not get set when an address does not match. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does get set when an address does not match.
26 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled. 0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1). 1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 16 (if M10 = 1, SNBS = 1).
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer.

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

Field	Description
	0 Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1). 1 Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 14 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 15 (if M10 = 1, SNBS = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. 0 LPUART receiver idle waiting for a start bit. 1 LPUART receiver active (LPUART_RX input not idle).
23 TDRE	Transmit Data Register Empty Flag TDRE will set when the transmit data register (LPUART_DATA) is empty. To clear TDRE, write to the LPUART data register (LPUART_DATA). TDRE is not affected by a character that is in the process of being transmitted, it is updated at the start of each transmitted character. 0 Transmit data buffer full. 1 Transmit data buffer empty.
22 TC	Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to LPUART_DATA to transmit new data, queuing a preamble by clearing and then setting LPUART_CTRL[TE], queuing a break character by writing 1 to LPUART_CTRL[SBK]. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
21 RDRF	Receive Data Register Full Flag RDRF is set when the receive buffer (LPUART_DATA) is full. To clear RDRF, read the LPUART_DATA register. A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received. 0 Receive data buffer empty. 1 Receive data buffer full.
20 IDLE	Idle Line Flag IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot become set again until after a new character has been stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period.

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

Field	Description
	0 No idle line detected. 1 Idle line was detected.
19 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received. While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag. 0 No overrun. 1 Receive overrun (new LPUART data lost).
18 NF	Noise Flag The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from LPUART_DATA was received with noise detected within the character. To clear NF, write logic one to the NF. 0 No noise detected. 1 Noise detected in the received character in LPUART_DATA.
17 FE	Framing Error Flag FE is set whenever the next character to be read from LPUART_DATA was received with logic 0 detected where a stop bit was expected. To clear FE, write logic one to the FE. 0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error.
16 PF	Parity Error Flag PF is set whenever the next character to be read from LPUART_DATA was received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic one to the PF. 0 No parity error. 1 Parity error.
15 MA1F	Match 1 Flag MA1F is set whenever the next character to be read from LPUART_DATA matches MA1. To clear MA1F, write a logic one to the MA1F. 0 Received data is not equal to MA1 1 Received data is equal to MA1
14 MA2F	Match 2 Flag MA2F is set whenever the next character to be read from LPUART_DATA matches MA2. To clear MA2F, write a logic one to the MA2F. 0 Received data is not equal to MA2 1 Received data is equal to MA2

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

Field	Description
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

46.3.3 LPUART Control Register (LPUARTx_CTRL)

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

Address: 4002_A000h base + 8h offset = 4002_A008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0			IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPUARTx_CTRL field descriptions

Field	Description
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading LPUART_DATA. T9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing LPUART_DATA. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.
30 R9T8	Receive Bit 9 / Transmit Bit 8 R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading LPUART_DATA T8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing LPUART_DATA. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

Field	Description
29 TXDIR	LPUART_TX Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the LPUART_TX pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the LPUART_TX pin. 0 LPUART_TX pin is an input in single-wire mode. 1 LPUART_TX pin is an output in single-wire mode.
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the LPUART_TX output for all cases: data bits, start and stop bits, break, and idle. 0 Transmit data not inverted. 1 Transmit data inverted.
27 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled; use polling. 1 Hardware interrupt requested when OR is set.
26 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled; use polling. 1 Hardware interrupt requested when NF is set.
25 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled; use polling. 1 Hardware interrupt requested when FE is set.
24 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled; use polling). 1 Hardware interrupt requested when PF is set.
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests. 0 Hardware interrupts from TDRE disabled; use polling. 1 Hardware interrupt requested when TDRE flag is 1.
22 TCIE	Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 Hardware interrupts from TC disabled; use polling. 1 Hardware interrupt requested when TC flag is 1.

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

Field	Description
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests. 0 Hardware interrupts from RDRF disabled; use polling. 1 Hardware interrupt requested when RDRF flag is 1.
20 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests. 0 Hardware interrupts from IDLE disabled; use polling. 1 Hardware interrupt requested when IDLE flag is 1.
19 TE	Transmitter Enable Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the LPUART_TX pin is tristated. 0 Transmitter disabled. 1 Transmitter enabled.
18 RE	Receiver Enable Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any). 0 Receiver disabled. 1 Receiver enabled.
17 RWU	Receiver Wakeup Control This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear. **NOTE:** RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted. 0 Normal receiver operation. 1 LPUART receiver in standby waiting for wakeup condition.
16 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 to 13, or 13 to 16 if LPUART_STATBRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.
15 MA1IE	Match 1 Interrupt Enable

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

Field	Description
	0 MA1F interrupt disabled 1 MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0 MA2F interrupt disabled 1 MA2F interrupt enabled
13–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000 1 idle character 001 2 idle characters 010 4 idle characters 011 8 idle characters 100 16 idle characters 101 32 idle characters 110 64 idle characters 111 128 idle characters
7 LOOPS	Loop Mode Select When LOOPS is set, the LPUART_RX pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation - LPUART_RX and LPUART_TX use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).
6 DOZEEN	Doze Enable 0 LPUART is enabled in Doze mode. 1 LPUART is disabled in Doze mode.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the LPUART_RX pin. 1 Single-wire LPUART mode where the LPUART_TX pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0 Receiver and transmitter use 8-bit data characters. 1 Receiver and transmitter use 9-bit data characters.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1: • Address mark in the most significant bit position of a received data character, or • An idle condition on the receive pin input signal.

Table continues on the next page...

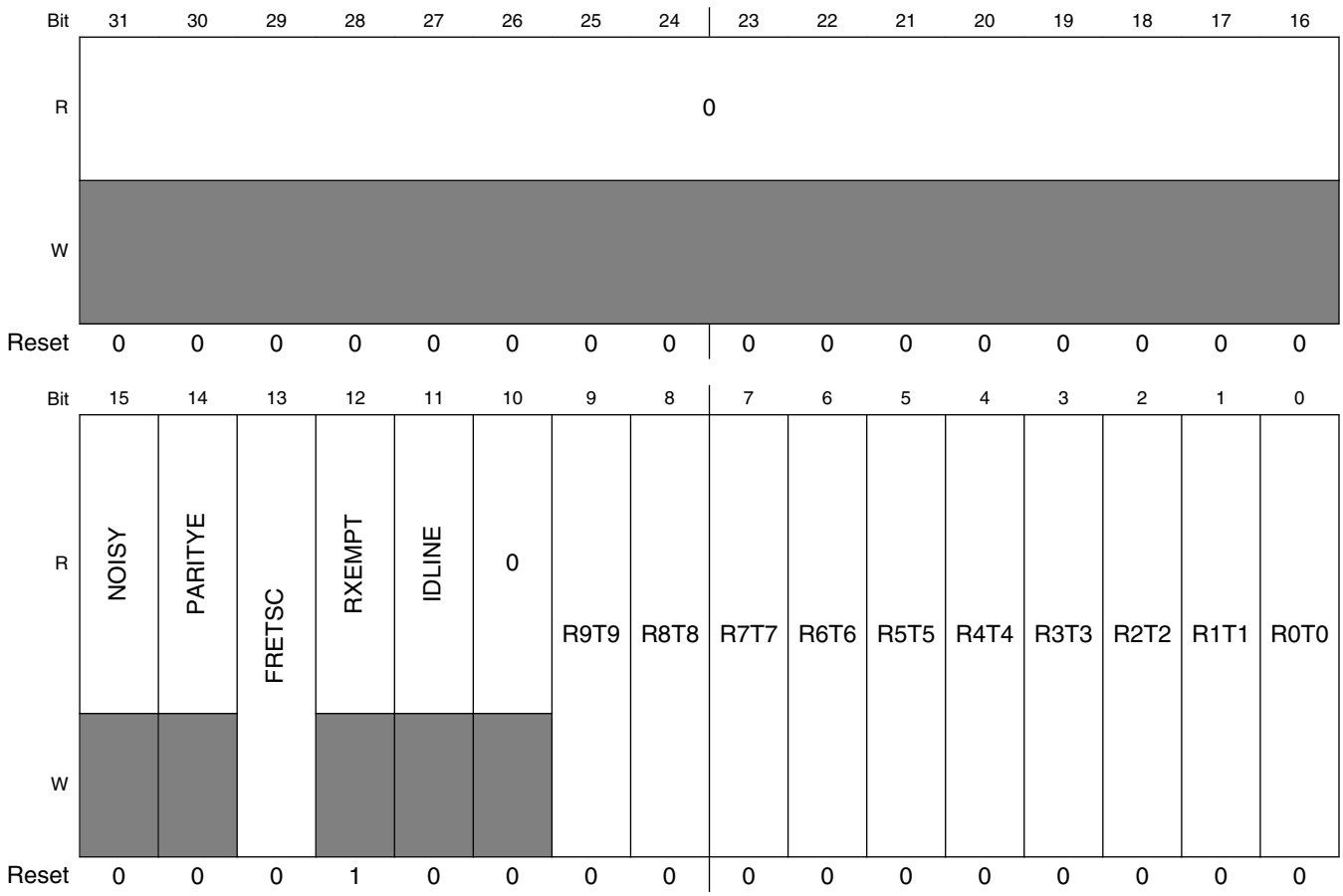
LPUARTx_CTRL field descriptions (continued)

Field	Description
	0 Configures RWU for idle-line wakeup. 1 Configures RWU with address-mark wakeup.
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0 No hardware parity generation or checking. 1 Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0 Even parity. 1 Odd parity.

46.3.4 LPUART Data Register (LPUARTx_DATA)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags.

Address: 4002_A000h base + Ch offset = 4002_A00Ch



LPUARTx_DATA field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 NOISY	The current received dataword contained in DATA[R9:R0] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.

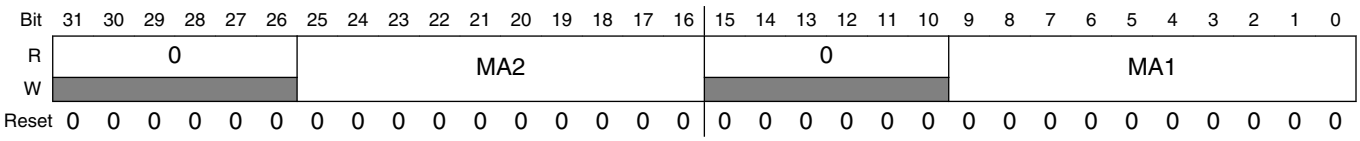
Table continues on the next page...

LPUARTx_DATA field descriptions (continued)

Field	Description
14 PARITYE	The current received dataword contained in DATA[R9:R0] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
13 FRETSC	Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, he contents of DATA[T8:T0] should be zero. 0 The dataword was received without a frame error on read, transmit a normal character on write. 1 The dataword was received with a frame error, transmit an idle or break character on transmit.
12 RXEMPT	Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer contains valid data. 1 Receive buffer is empty, data returned on read is not valid.
11 IDLINE	Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0 Receiver was not idle before receiving this character. 1 Receiver was idle before receiving this character.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 R9T9	Read receive data buffer 9 or write transmit data buffer 9.
8 R8T8	Read receive data buffer 8 or write transmit data buffer 8.
7 R7T7	Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	Read receive data buffer 0 or write transmit data buffer 0.

46.3.5 LPUART Match Address Register (LPUARTx_MATCH)

Address: 4002_A000h base + 10h offset = 4002_A010h



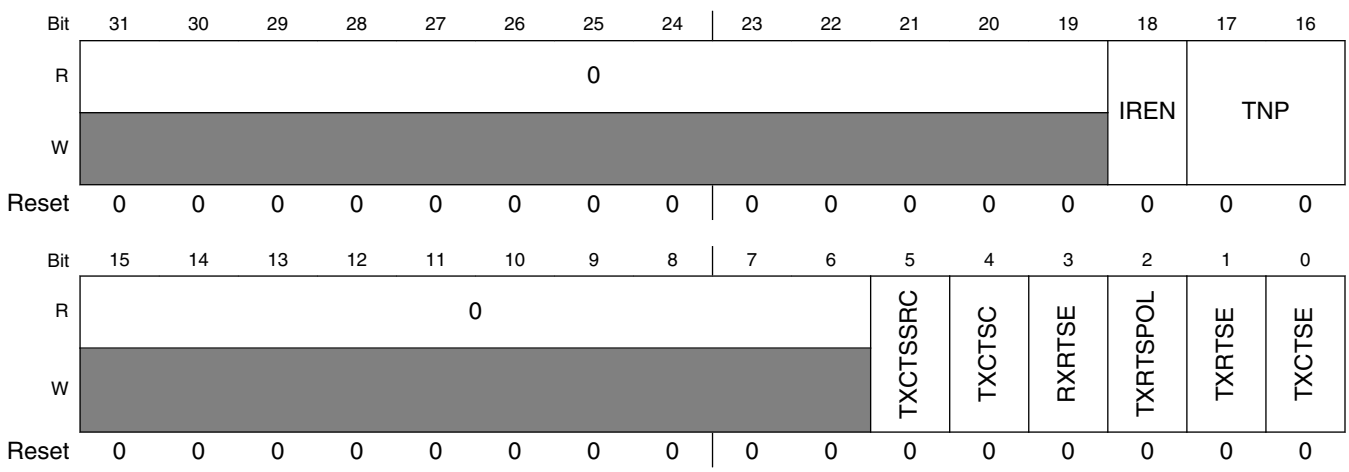
LPUARTx_MATCH field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–16 MA2	Match Address 2 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MA1	Match Address 1 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.

46.3.6 LPUART Modem IrDA Register (LPUARTx_MODIR)

The MODEM register controls options for setting the modem configuration.

Address: 4002_A000h base + 14h offset = 4002_A014h



LPUARTx_MODIR field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. 0 IR disabled. 1 IR enabled.
17–16 TNP	Transmitter narrow pulse Enables whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse. 00 1/OSR. 01 2/OSR. 10 3/OSR. 11 4/OSR.
15–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0 CTS input is the LPUART_CTS pin. 1 CTS input is the inverted Receiver Match result.
4 TXCTSC	Transmit CTS Configuration Configures if the CTS state is checked at the start of each character or only when the transmitter is idle. 0 CTS input is sampled at the start of each character. 1 CTS input is sampled when the transmitter is idle.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. **NOTE:** Do not set both RXRTSE and TXRTSE. 0 The receiver has no effect on RTS. 1 RTS is deasserted if the receiver data register is full or a start bit has been detected that would cause the receiver data register to become full. RTS is asserted if the receiver data register is not full and has not detected a start bit that would cause the receiver data register to become full.
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set. 0 Transmitter RTS is active low. 1 Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission.

Table continues on the next page...

LPUARTx_MODIR field descriptions (continued)

Field	Description
	0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0 CTS has no effect on the transmitter. 1 Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

46.4 Functional description

The LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the LPUART.

46.4.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the baud clock divisor for the asynchronous LPUART baud clock. The SBR bits are in the LPUART baud rate registers, BDH and BDL. The baud rate clock drives the receiver, while the transmitter is driven by the baud rate clock divided by the over sampling ratio. Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.

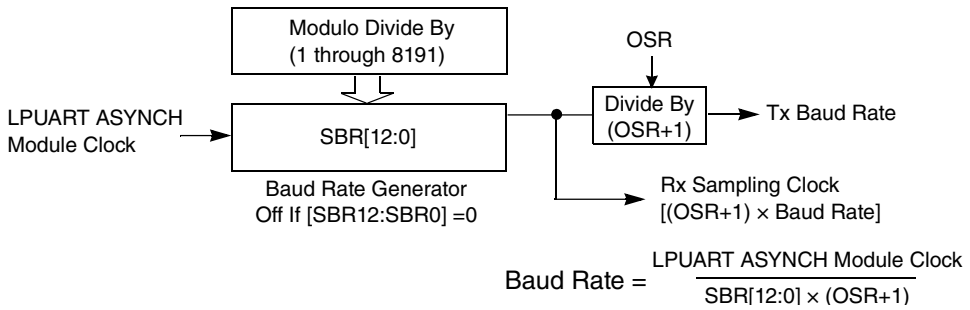


Figure 46-4. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

46.4.2 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (LPUART_TX) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the LPUART data register.

The central element of the LPUART transmitter is the transmit shift register that is 10-bit to 13 bits long depending on the setting in the CTRL[M], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at LPUART_DATA.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the LPUART_TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with LPUART_TX high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

46.4.2.1 Send break and queued idle

The LPUART_CTRL[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting LPUART_STAT[BRK13]. Normally, a program would wait for LPUART_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the LPUART_CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If LPUART_CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another Freescale Semiconductor LPUART, the break characters are received as 0s in all data bits and a framing error (LPUART_STAT[FE] = 1) occurs.

A break character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows the DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for LPUART_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the LPUART_CTRL[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while LPUART_CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the LPUART_TX pin.

An idle character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits also set. This supports transmitting the idle character as part of the normal data stream and also allows the DMA to transmit a break character.

The length of the break character is affected by the LPUART_STAT[BRK13], LPUART_CTRL[M], LPUART_BAUD[M10] and LPUART_BAUD[SNBS] bits as shown below.

Table 46-1. Break character length

BRK13	M	M10	SBNS	Break character length
0	0	0	0	10 bit times
0	0	0	1	11 bit times
0	1	0	0	11 bit times
0	1	0	1	12 bit times
0	X	1	0	12 bit times
0	X	1	1	13 bit times
1	0	0	0	13 bit times
1	0	0	1	13 bit times
1	1	0	0	14 bit times
1	1	0	1	14 bit times
1	X	1	0	15 bit times
1	X	1	1	15 bit times

46.4.2.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the clear-to-send operation is enabled, the character is transmitted when CTS is asserted. If CTS is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and LPUART_TX remains in the mark state until CTS is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS.

The transmitter's CTS signal can also be enabled even if the same LPUART receiver's RTS signal is disabled.

46.4.2.3 Transceiver driver enable

The transmitter can use LPUART_RTS as an enable signal for the driver of an external transceiver. See Transceiver driver enable using LPUART_RTS for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, LPUART_RTS asserts one bit time before the start bit is transmitted. LPUART_RTS remains asserted for the whole time that the transmitter data buffer has any characters. LPUART_RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts LPUART_RTS, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's LPUART_RTS signal asserts only when the transmitter is enabled. However, the transmitter's LPUART_RTS signal is unaffected by its LPUART_CTS signal. LPUART_RTS will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

46.4.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting LPUART_STAT[RXINV]. The receiver is enabled by setting the LPUART_CTRL[RE] bit. Character frames consist of a start bit of logic 0, eight to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 9-bit or 10-bit data mode, refer to 8-bit, 9-bit and 10-bit data modes. For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (LPUART_STAT[RDRF]) status flag is set. If LPUART_STAT[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the LPUART receiver is double-buffered, the program has one full character time after LPUART_STAT[RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. Refer to Interrupts and status flags for details about flag clearing.

46.4.3.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the LPUART_RX serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and

(OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (LPUART_STAT[NF]) is set when the received character is transferred to the receive data buffer.

When the LPUART receiver is configured to sample on both edges of the baud rate clock, the number of segments in each received bit is effectively doubled (from 1 to OSR×2). The start and data bits are then sampled at OSR, OSR+1 and OSR+2. Sampling on both edges of the clock must be enabled for oversampling rates of 4× to 7× and is optional for higher oversampling rates.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times (unless resynchronization has been disabled). This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

46.4.3.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit (LPUART_CTRL[RWU]). When RWU bit and LPUART_S2[RWUID] bit are set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force LPUART_CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver will ignore all characters that do not meet the address match requirements.

Table 46-2. Receiver Wakeup Options

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, IDLE flag not set
1	0	00	01	Receiver wakeup on idle line, IDLE flag set
1	0	00	10	Receiver wakeup on address mark
1	1	11	X0	Receiver wakeup on data match
0	1	00	X0	Address mark address match, IDLE flag not set for discarded characters
0	1	00	X1	Address mark address match, IDLE flag set for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Address match on and address match off, IDLE flag not set for discarded characters
0	1	10	X1	Address match on and address match off, IDLE flag set for discarded characters

46.4.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The LPUART_CTRL[M] and LPUART_BAUD[M10] control bit selects 8-bit to 10-bit data mode and the LPUART_BAUD[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 to 13 bit times because of the start and stop bits.

When LPUART_CTRL[RWU] is one and LPUART_STAT[RWUID] is zero, the idle condition that wakes up the receiver does not set the LPUART_STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the

LPUART_STAT[RDRF] flag and generates an interrupt if enabled. When LPUART_STAT[RWUID] is one, any idle condition sets the LPUART_STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether LPUART_CTRL[RWU] is zero or one.

The idle-line type (LPUART_CTRL[ILT]) control bit selects one of two ways to detect an idle line. When LPUART_CTRL[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When LPUART_CTRL[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

46.4.3.2.2 Address-mark wakeup

When LPUART_CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character.

Address-mark wakeup allows messages to contain idle characters, but requires the MSB be reserved for use in address frames. The logic 1 in the MSB of an address frame clears the LPUART_CTRL[RWU] bit before the stop bits are received and sets the LPUART_STAT[RDRF] flag. In this case, the character with the MSB set is received even though the receiver was sleeping during most of this character time.

46.4.3.2.3 Data match wakeup

When LPUART_CTRL[RWU] is set and LPUART_BAUD[MATCFG] equals 11, the receiver is configured for data match wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a character that matches MATCH[MA1] field when BAUD[MAEN1] is set, or that matches MATCH[MA2] when BAUD[MAEN2] is set.

46.4.3.2.4 Address Match operation

Address match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 00. In this function, a character received by the LPUART_RX pin with a logic 1 in the bit position immediately preceding the stop bit is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters received with a logic 0 in the bit position immediately preceding the stop bit are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs then no transfer

is made to the receive data buffer, and all following characters with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

46.4.3.2.5 Idle Match operation

Idle match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 01. In this function, the first character received by the LPUART_RX pin after an idle line condition is considered an address and is compared with the associated MA1 or MA2 register. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive data buffer until the next idle line condition is detected. If no address match occurs then no transfer is made to the receive data buffer, and all following frames until the next idle condition are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Idle match operation functions in the same way for both MA1 and MA2 registers.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] is asserted, the first character after an idle line is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, the first character after an idle line is compared with both match registers and data is transferred only on a match with either register.

46.4.3.2.6 Match On Match Off operation

Match on, match off operation is enabled when both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are set and LPUART_BAUD[MATCFG] is equal to 10. In this function, a character received by the LPUART_RX pin that matches MATCH[MA1] is received and transferred to the receive buffer, and LPUART_STAT[RDRF] is set. All subsequent characters are considered to be data and are also transferred to the receive data buffer, until a character is received that matches MATCH[MA2] register. The character that matches MATCH[MA2] and all following characters are discarded, this continues until another character that matches MATCH[MA1] is received. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

NOTE

Match on, match off operation requires both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] to be asserted.

46.4.3.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert LPUART_RTS.

- LPUART_RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using LPUART_RTS for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts LPUART_RTS if the number of characters in the receiver data register is full or a start bit is detected that will cause the receiver data register to be full.
- The receiver asserts LPUART_RTS when the number of characters in the receiver data register is not full and has not detected a start bit that will cause the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if LPUART_RTS is deasserted, the receiver continues to receive characters until the receiver data buffer is overrun.
- If the receiver request-to-send functionality is disabled, the receiver LPUART_RTS remains deasserted.

46.4.3.4 Infrared decoder

The infrared decoder converts the received character from the IrDA format to the NRZ format used by the receiver. It also has a OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received.

46.4.3.4.1 Start bit detection

When STAT[RXINV] is cleared, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently from each other.

46.4.3.4.2 Noise filtering

Any further rising edges detected during the first half of the infrared decoder counter are ignored by the decoder. Any pulses less than one oversampling baud clock can be undetected by it regardless of whether it is seen in the first or second half of the count.

46.4.3.4.3 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as a 0, which is sent to the receiver. The decoder counter is also reset.

46.4.3.4.4 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, then the decoder sends a 1 to the receiver.

If the next bit is a 0, which arrives late, then a low-bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to a 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, then the delay of a 0 is not recorded as noise.

46.4.4 Additional LPUART functions

The following sections describe additional LPUART functions.

46.4.4.1 8-bit, 9-bit and 10-bit data modes

The LPUART transmitter and receiver can be configured to operate in 9-bit data mode by setting the LPUART_CTRL[M] or 10-bit data mode by setting LPUART_CTRL[M10]. In 9-bit mode, there is a ninth data bit in 10-bit mode there is a tenth data bit. For the transmit data buffer, these bits are stored in LPUART_CTRL[T8] and LPUART_CTRL[T9]. For the receiver, these bits are held in LPUART_CTRL[R8] and LPUART_CTRL[R9]. They are also accessible via 16-bit or 32-bit accesses to the LPUART_DATA register.

For coherent 8-bit writes to the transmit data buffer, write to LPUART_CTRL[T8] and LPUART_CTRL[T9] before writing to LPUART_DATA[7:0]. For 16-bit and 32-bit writes to the LPUART_DATA register all 10 transmit bits are written to the transmit data buffer at the same time.

If the bit values to be transmitted as the ninth and tenth bit of a new character are the same as for the previous character, it is not necessary to write to LPUART_CTRL[T8] and LPUART_CTRL[T9] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in LPUART_CTRL[T8] and LPUART_CTRL[T9] is copied at the same time data is transferred from LPUART_DATA[7:0] to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

46.4.4.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark postion. The CTRL[ILT] register can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

The number of idle characters that must be received before an idle line condition is detected can also be configured using the CTRL[IDLECFG] field. This field configures the number of idle characters that must be received before the STAT[IDLE] flag is set, the STAT[RAF] flag is cleared and the DATA[IDLINE] flag is set with the next received character.

Idle-line wakeup and idle match operation are also affected by the CTRL[IDLECFG] field. When address match or match on/off operation is enabled, setting the STAT[RWUID] bit will cause any discarded characters to be treated as if they were idle characters.

46.4.4.3 Loop mode

When LPUART_CTRL[LOOPS] is set, the LPUART_CTRL[RSRC] bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[RSRC] = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the LPUART_RX pin is not used by the LPUART.

46.4.4.4 Single-wire operation

When LPUART_CTRL[LOOPS] is set, the RSRC bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the LPUART_TX pin (the LPUART_RX pin is not used).

In single-wire mode, the LPUART_CTRL[TXDIR] bit controls the direction of serial data on the LPUART_TX pin. When LPUART_CTRL[TXDIR] is cleared, the LPUART_TX pin is an input to the receiver and the transmitter is temporarily disconnected from the LPUART_TX pin so an external device can send serial data to the receiver. When LPUART_CTRL[TXDIR] is set, the LPUART_TX pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

46.4.5 Infrared interface

The LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the LPUART. The IrDA physical layer specification defines a half-duplex infrared communication link for exchanging data. The full standard includes data rates up to 16 Mbits/s. This design covers data rates only between 2.4 kbits/s and 115.2 kbits/s.

The LPUART has an infrared transmit encoder and receive decoder. The LPUART transmits serial bits of data that are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses are detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder, external from the LPUART. The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the UART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

46.4.5.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the LPUART_TX signal. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent at the start of the bit with a duration of 1/OSR, 2/OSR, 3/OSR, or 4/OSR of a bit time. A narrow low pulse is transmitted for a zero bit when LPUART_CTRL[TXINV] is cleared, while a narrow high pulse is transmitted for a zero bit when LPUART_CTRL[TXINV] is set.

46.4.5.2 Infrared receive decoder

The infrared receive block converts data from the LPUART_RX signal to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow low pulse is expected for a zero bit when LPUART_STAT[RXINV] is cleared, while a narrow high pulse is expected for a zero bit when LPUART_STAT[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

46.4.6 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty LPUART_STAT[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to LPUART_DATA. If the transmit interrupt enable LPUART_CTRL[TIE]) bit is set, a hardware interrupt is requested when LPUART_STAT[TDRE] is set. Transmit complete

(LPUART_STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with LPUART_TX at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (LPUART_CTRL[TCIE]) bit is set, a hardware interrupt is requested when LPUART_STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the LPUART_STAT[TDRE] and LPUART_STAT[TC] status flags if the corresponding LPUART_CTRL[TIE] or LPUART_CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. The LPUART_STAT[RDRF] flag is cleared by reading LPUART_DATA.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the LPUART_RX line remains idle for an extended period of time. IDLE is cleared by writing 1 to the LPUART_STAT[IDLE] flag. After LPUART_STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set LPUART_STAT[RDRF].

If the associated error was detected in the received character that caused LPUART_STAT[RDRF] to be set, the error flags - noise flag (LPUART_STAT[NF]), framing error (LPUART_STAT[FE]), and parity error flag (LPUART_STAT[PF]) - are set at the same time as LPUART_STAT[RDRF]. These flags are not set in overrun cases.

If LPUART_STAT[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (LPUART_STAT[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2] then the LPUART_STAT[MA1F] and/or LPUART_STAT[MA2F] flags are set at the same time that LPUART_STAT[RDRF] is set.

At any time, an active edge on the LPUART_RX serial data input pin causes the LPUART_STAT[RXEDGIF] flag to set. The LPUART_STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (LPUART_CTRL[RE] = 1).

Chapter 47
Memory Mapped Cryptographic Acceleration Unit (MMCAU)

47.1 Chip-specific MMCAU information

47.1.1 Overview

The MMCAU provides security encrypt/decrypt acceleration, to allow users to secure external data with other devices via the various series communication ports.

The MMCAU clock source is the CPU/platform clock.

47.2 Introduction

The memory-mapped Cryptographic Acceleration Unit (CAU) is a coprocessor that is connected to the processor's Private Peripheral Bus (PPB). It supports the hardware implementation of a set of specialized operations to improve the throughput of software-based security encryption/decryption operations and message digest functions.

The CAU supports acceleration of the DES, 3DES, AES, MD5, SHA-1, and SHA-256 algorithms. Freecsale provides an optimized C-function library that provides the appropriate software building blocks to implement higher-level security functions.

47.3 CAU Block Diagram

A simplified block diagram is given below that illustrates the CAU and a table to show its parts.

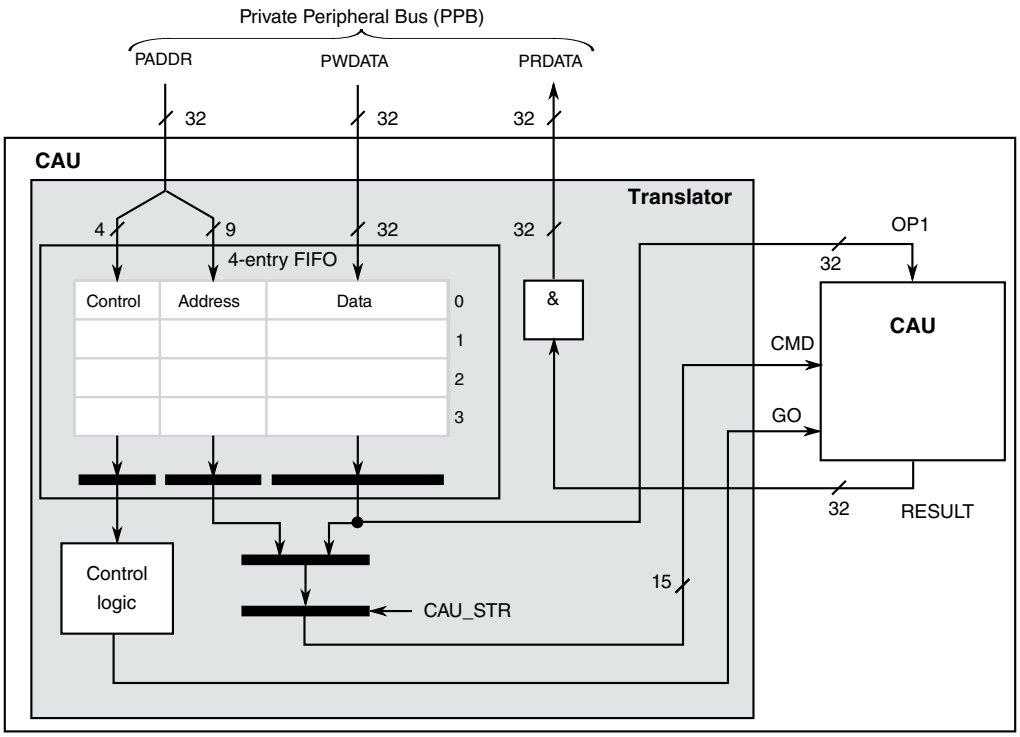


Figure 47-1. CAU block diagram

Table 47-1. CAU parts table

Item	Description
Translator submodule	Provides the bridge between the PPB interface and the CAU module. Passes memory-mapped commands and data on the PPB to/from the CAU
4-entry FIFO	Contains commands and input operands and the associated control captured from the PPB and sent to the CAU
CAU	3-terminal block with a command and optional input operand and a result bus. More details in following figure.

The following figure shows the CAU block in more detail.

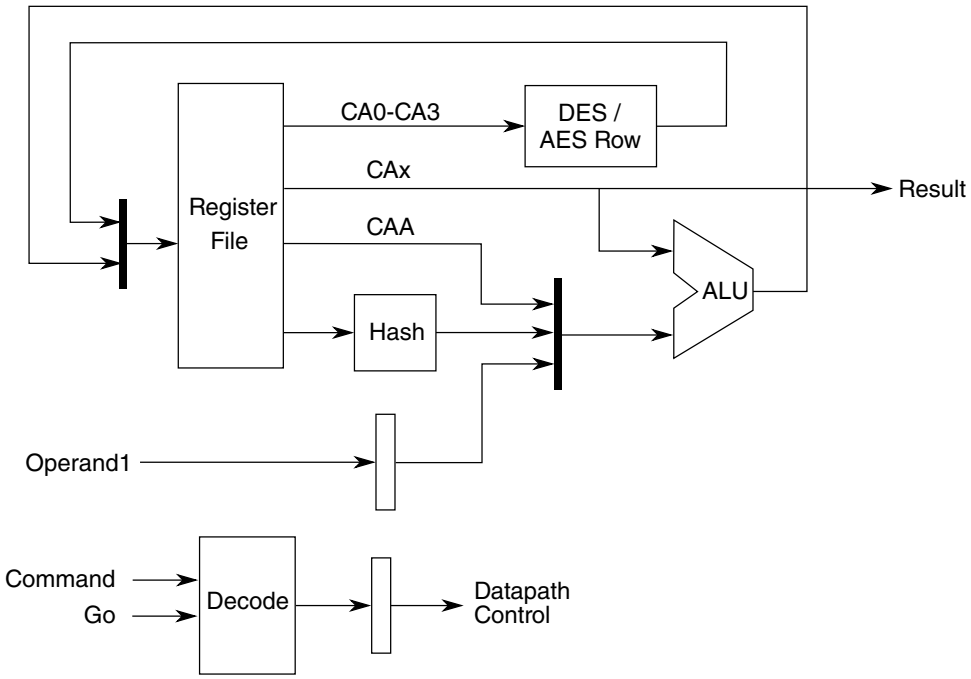


Figure 47-2. Top-level CAU block diagram

47.4 Overview

As the name suggests, the CAU provides a mechanism for memory-mapped register reads and writes to be transformed into specific commands and operands sent to the CAU coprocessor.

The CAU translator module performs the following functions:

- All the required functions affecting the transmission of commands to the CAU module.
- If needed, stalling the PPB transactions based on the state of the 4-entry command/ data FIFO.
- Some basic integrity checks on PPB operations.

The set of implemented algorithms provides excellent support for network security standards, such as SSL and IPsec. Additionally, using the CAU efficiently permits the implementation of any higher level functions or modes of operation, such as HMAC, CBC, and so on based on the supported algorithms.

The cryptographic algorithms are implemented partially in software with only functions critical to increasing performance implemented in hardware. The CAU allows for efficient, fine-grained partitioning of functions between hardware and software:

- Implement the innermost security kernel functions using the coprocessor instructions.
- Implement higher level functions in software by using the standard processor instructions.

This partitioning of functions is key to minimizing size of the CAU while maintaining a high level of throughput. Using software for some functions also simplifies the CAU design. The CAU implements a set of coprocessor commands that operate on a register file of 32-bit registers.

47.5 Features

The CAU includes the following distinctive features:

- Supports DES, 3DES, AES, MD5, SHA-1, and SHA-256 algorithms
- Simple, flexible programming model
- Ability to send up to three commands in one data write operation

47.6 Memory map/Register definition

The CAU contains multiple registers used by each of the supported algorithms. The following table shows registers that are applicable to each supported algorithm and indicates the corresponding letter designations for each algorithm.

For more information on these letter designations, see the supported algorithm specifications.

Code	Register	DES	AES	MD5	SHA-1	SHA-256
0	CAU Status Register (CASR)	—	—	—	—	—
1	CAU Accumulator (CAA)	—	—	a	T	T
2	General-Purpose Register 0 (CA0)	C	W0	—	A	A
3	General-Purpose Register 1 (CA1)	D	W1	b	B	B

Table continues on the next page...

Code	Register	DES	AES	MD5	SHA-1	SHA-256
4	General-Purpose Register 2 (CA2)	L	W2	c	C	C
5	General-Purpose Register 3 (CA3)	R	W3	d	D	D
6	General-Purpose Register 4 (CA4)	—	—	—	E	E
7	General-Purpose Register 5 (CA5)	—	—	—	W	F
8	General-Purpose Register 6 (CA6)	—	—	—	—	G
9	General-Purpose Register 7 (CA7)	—	—	—	—	H
10	General-Purpose Register 8 (CA8)	—	—	—	—	W/T_1

The CAU supports only 32-bit operations and register accesses. All registers support read, write, and ALU operations. However, only bits 1–0 of the CASR are writeable. Bits 31–2 of the CASR must be written as 0 for compatibility with future versions of the CAU.

The codes listed in this section are used in the memory-mapped commands. For more details on this, see CAU programming model.

NOTE

In the following table, the "address" or "offset" refers to the command code value for the CAU registers.

CAU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_5000	Status Register (CAU_CASR)	32	R/W	2000_0000h	47.6.1/1054
F000_5001	Accumulator (CAU_CAA)	32	R/W	0000_0000h	47.6.2/1055

Table continues on the next page...

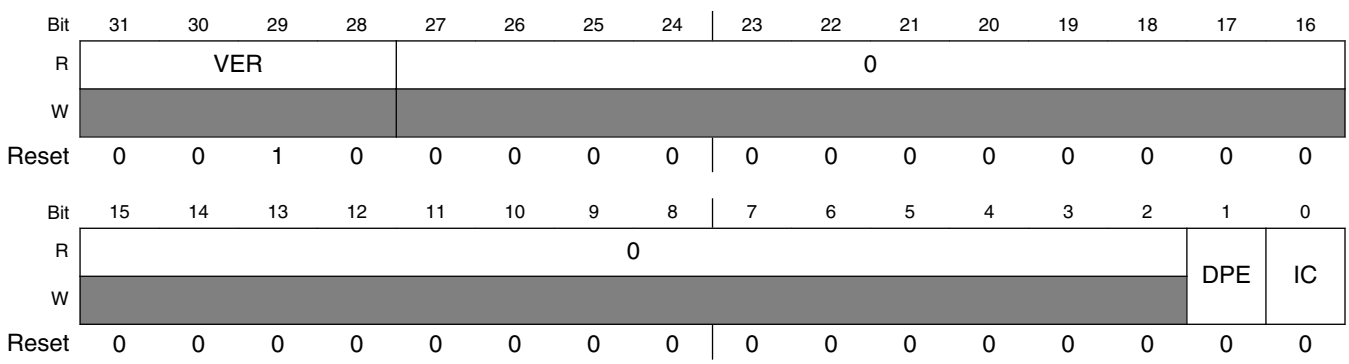
CAU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_5002	General Purpose Register (CAU_CA0)	32	R/W	0000_0000h	47.6.3/1055
F000_5003	General Purpose Register (CAU_CA1)	32	R/W	0000_0000h	47.6.3/1055
F000_5004	General Purpose Register (CAU_CA2)	32	R/W	0000_0000h	47.6.3/1055
F000_5005	General Purpose Register (CAU_CA3)	32	R/W	0000_0000h	47.6.3/1055
F000_5006	General Purpose Register (CAU_CA4)	32	R/W	0000_0000h	47.6.3/1055
F000_5007	General Purpose Register (CAU_CA5)	32	R/W	0000_0000h	47.6.3/1055
F000_5008	General Purpose Register (CAU_CA6)	32	R/W	0000_0000h	47.6.3/1055
F000_5009	General Purpose Register (CAU_CA7)	32	R/W	0000_0000h	47.6.3/1055
F000_500A	General Purpose Register (CAU_CA8)	32	R/W	0000_0000h	47.6.3/1055

47.6.1 Status Register (CAU_CASR)

CASR contains the status and configuration for the CAU.

Address: F000_5000h base + 0h offset = F000_5000h



CAU_CASR field descriptions

Field	Description
31–28 VER	CAU Version Indicates CAU version. 0x1 Initial CAU version. 0x2 Second version, added support for SHA-256 algorithm (This is the value on this device).
27–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DPE	DES Parity Error Indicates whether the DES parity error is detected. 0 No error detected. 1 DES key parity error detected.

Table continues on the next page...

CAU_CASR field descriptions (continued)

Field	Description
0 IC	Illegal Command Indicates an illegal instruction has been executed. 0 No illegal commands issued. 1 Illegal command issued.

47.6.2 Accumulator (CAU_CAA)

Commands use the CAU accumulator for storage of results and as an operand for the cryptographic algorithms.

Address: F000_5000h base + 1h offset = F000_5001h

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

CAU_CAA field descriptions

Field	Description
ACC	Accumulator Stores results of various CAU commands.

47.6.3 General Purpose Register (CAU_CA*n*)

The General Purpose Register is used in the CAU commands for storage of results and as operands for various cryptographic algorithms.

Address: F000_5000h base + 2h offset + (1d × i), where i=0d to 8d

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

CAU_CA*n* field descriptions

Field	Description
CAn	General Purpose Registers Used by the CAU commands. Some cryptographic operations work with specific registers.

47.7 Functional description

This section discusses the programming model and operation of the CAU.

47.7.1 CAU programming model

The 4-entry FIFO is indirectly mapped into a 4-KB address space associated with the CAU located on this device. This address space is effectively split into two equal regions:

- one used to directly write commands for CAU load operations
- the other used to send commands and input operands for CAU loads

Data writes on the PPB are loaded into this FIFO and automatically converted into CAU load operands by the CAU translator. Data reads on the PPB are converted into CAU store register operations where the result is returned to the processor as the read data value.

The CAU requires a 15-bit command, and optionally, a 32-bit input operand, for each CAU load, PPB write. The 15-bit command includes the 9-bit opcode and other bits statically formed by the CAU translator logic controlling the CAU.

The following figure shows the 4-KB address space and the mapping of the CAU commands in this space.

NOTE

- Although the indirect store/load portion of the address space in the figure below shows only the indirect load/store commands, direct load commands can also be used in this space. However, it is more efficient to use the direct load portion of the address space.
- Accesses to the reserved space in the direct load space are terminated with an error, while accesses to the reserved space in the indirect load/store space are detected as an illegal CAU command. See CAU integrity checks for details.

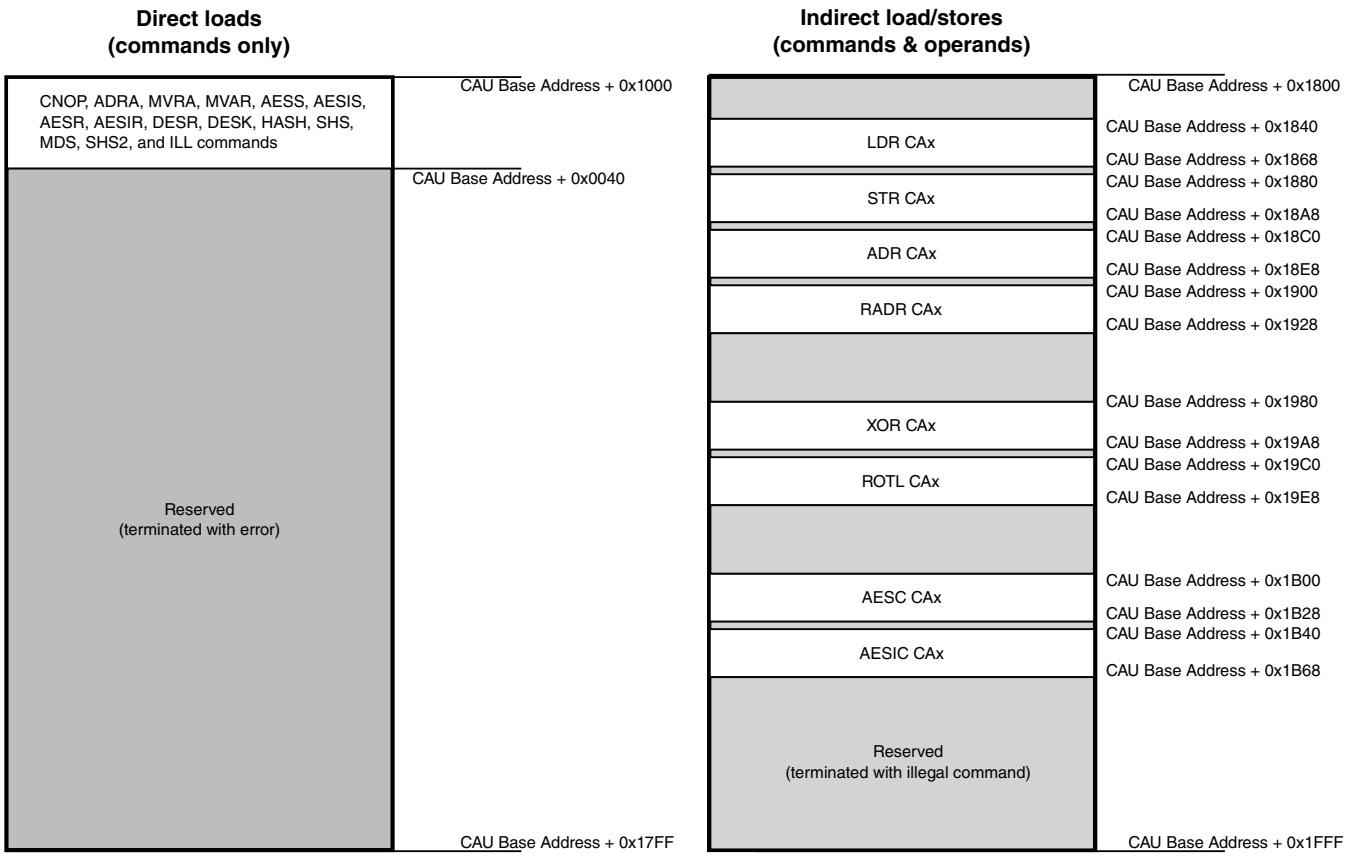


Figure 47-3. CAU memory map

47.7.1.1 Direct loads

The CAU supports writing multiple commands in each 32-bit direct write operation. Each 9-bit opcode also includes a valid bit. Therefore, one, two, or three commands can be transmitted in a single 32-bit PPB write. The following figure illustrates the accepted formats for the 32-bit CAU write data value:

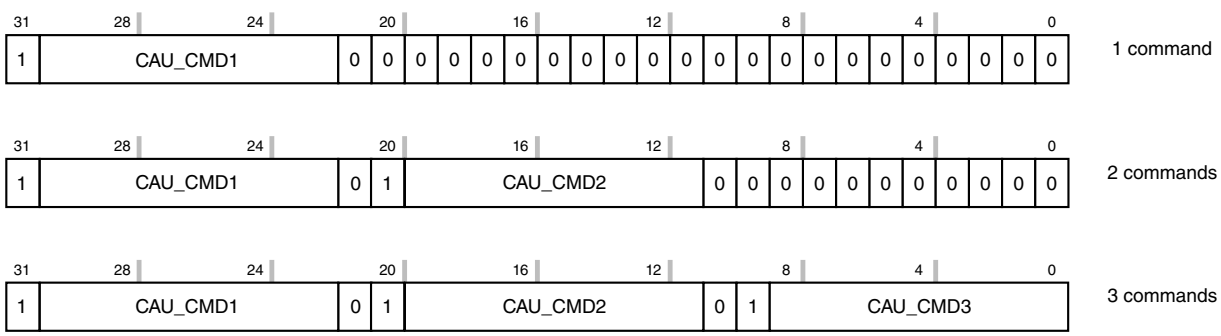


Figure 47-4. Direct loads

47.7.1.2 Indirect loads

For CAU load operations requiring a 32-bit input operand, the address contains the 9-bit opcode to be passed to the CAU while the data is the 32-bit operand. Specifically, the CAU address and data for these indirect writes is shown in the figure below.

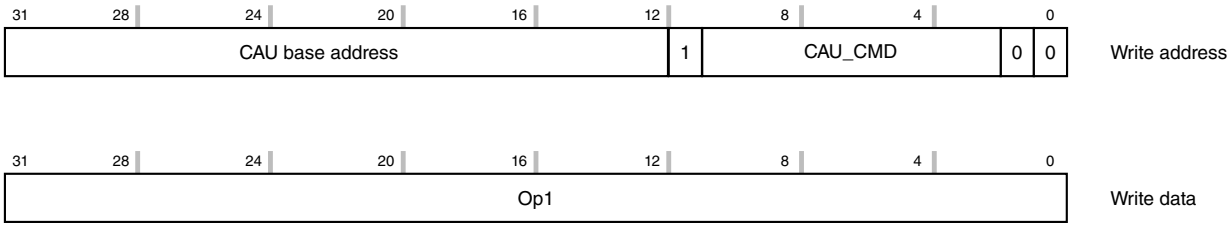


Figure 47-5. Indirect loads

47.7.1.3 Indirect stores

For CAU store operations, a PPB read is performed with the appropriate CAU store register opcode embedded in the address. This appears as another indirect command. The detail of Indirect stores is shown in the figure below.

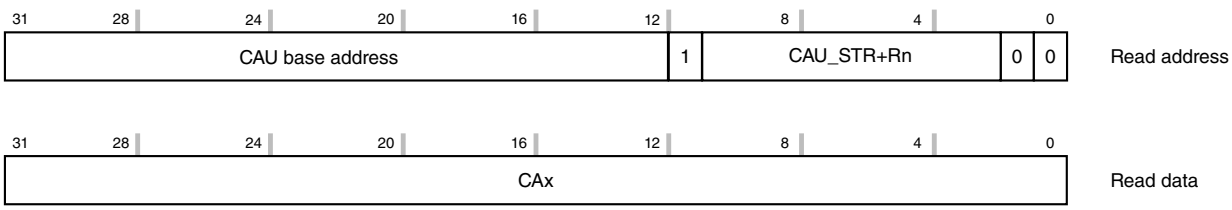


Figure 47-6. Indirect store

47.7.2 CAU integrity checks

If an illegal operation or access is attempted, the PPB bus cycle is terminated with an error response and the operation is aborted and not sent to the CAU.

The CAU performs a series of address and data integrity checks as described in the following sections. The results of these checks are logically summed together and, if appropriate, a PPB error termination is generated.

47.7.2.1 Address integrity checks

The CAU address checking includes the following. See Figure 47-3 for the CAU memory map details.

- Any CAU reference using a non-0-modulo-4 byte address (addr[1:0] ≠ 00) generates an error termination.
- For CAU writes:
  - Only the first 64 bytes of the 2-KB direct write address space can be referenced. Attempting to access regions beyond the first 64 bytes terminates with an error.
  - The second 2-KB space defines the indirect address-as-command region and any reference in this space is allowed by the CAU.

NOTE

The CAU contains error logic to detect any illegal command sent to it. Accordingly, there are address values in this upper 2-KB region of the address space that are passed to the CAU, and then detected as illegal commands. If the CAU detects an illegal command, it sets the CASR[IC] flag and performs no operation.

- For CAU reads:
  - Any attempted read from the first 2-KB region of the address space (an attempted direct read) is illegal and produces an error termination.
  - Within the second 2-KB region of the address space, i.e., addr[11] = 1, only a 64-byte space is treated as a legal CAU store operation. The allowable addresses are defined as:

    addr[11:0] = 1000_10xx_xx_00

    where the 4-bit xxxx value specifies the CAU register number. The CAU supports a subset of the allowable register numbers, 0x0 - 0xA. Attempting a store of a reserved register produces an undefined result.

47.7.2.2 Data integrity checks

Direct writes can send 1, 2, or 3 commands to the CAU in a single 32-bit transfer. As shown in Figure 47-4, the commands include a valid bit located at bits 31, 20, and 9 of the write data where:

- Bit 31 is the valid bit for the first command
- Bit 20 is the valid bit for the second command
- Bit 9 is the valid bit for the third command

The direct write data check validates the combination of these three valid bits. The following table presents the three legal states associated with these bits:

Value of bits 31, 20, and 9	Number of commands included
100	1
110	2
111	3

All other combinations of bits 31, 20, and 9 are illegal and generate an error termination.

47.7.3 CAU commands

The CAU supports the commands shown in the following table. All other encodings are reserved. The CASR[IC] bit is set if an undefined command is issued. A specific illegal command (ILL) is defined to allow software self-checking. Reserved commands must not be issued so as to ensure compatibility with future implementations.

The CMD field specifies the 9-bit CAU opcode for the operation command.

See Assembler equate values for a set of assembly constants used in the command descriptions here. If supported by the assembler, macros can also be created for each instruction. The value CA*x* should be interpreted as any CAU register (CASR, CAA, and CA*n*).

Table 47-2. CAU commands

Type	Command name	Description	CMD									Operation
			8	7	6	5	4	3	2	1	0	
Direct load	CNOP	No Operation	0x000									—
Indirect load	LDR	Load Reg	0x01					CAx				Op1 → CAx
Indirect store	STR	Store Reg	0x02					CAx				CAx → Result
Indirect load	ADR	Add	0x03					CAx				CAx + Op1 → CAx
Indirect load	RADR	Reverse and Add	0x04					CAx				CAx + ByteRev(Op1) → CAx
Direct load	ADRA	Add Reg to Acc	0x05					CAx				CAx + CAA → CAA
Indirect load	XOR	Exclusive Or	0x06					CAx				CAx ^ Op1 → CAx
Indirect load	ROTL	Rotate Left	0x07					CAx				(CAx <<< (Op1 % 32)) | (CAx >>> (32 - (Op1 % 32))) → CAx
Direct load	MVRA	Move Reg to Acc	0x08					CAx				CAx → CAA
Direct load	MVAR	Move Acc to Reg	0x09					CAx				CAA → CAx
Direct load	AESS	AES Sub Bytes	0x0A					CAx				SubBytes(CAx) → CAx
Direct load	AESIS	AES Inv Sub Bytes	0x0B					CAx				InvSubBytes(CAx) → CAx
Indirect load	AESC	AES Column Op	0x0C					CAx				MixColumns(CAx)^Op1→ CAx
Indirect load	AESIC	AES Inv Column Op	0x0D					CAx				InvMixColumns(CAx^Op1) → CAx

Table continues on the next page...

Table 47-2. CAU commands (continued)

Type	Command name	Description	CMD									Operation
			8	7	6	5	4	3	2	1	0	
Direct load	AESR	AES Shift Rows	0x0E0									ShiftRows(CA0-CA3) → CA0-CA3
Direct load	AESIR	AES Inv Shift Rows	0x0F0									InvShiftRows(CA0-CA3)→ CA0-CA3
Direct load	DESR	DES Round	0x10					IP	FP	KS[1:0]		DES Round(CA0-CA3) →CA0-CA3
Direct load	DESK	DES Key Setup	0x11					0	0	CP	DC	DES Key Op(CA0-CA1)→ CA0-CA1 Key Parity Error & CP → CASR[1]
Direct load	HASH	Hash Function	0x12					0	HF[2:0]			Hash Func(CA1-CA3)+CAA→ CAA
Direct load	SHS	Secure Hash Shift	0x130									CAA <<< 5→ CAA, CAA→CA0, CA0→CA1, CA1 <<< 30 → CA2, CA2→CA3, CA3→CA4
Direct load	MDS	Message Digest Shift	0x140									CA3→CAA, CAA→CA1, CA1→CA2, CA2→CA3,
Direct load	SHS2	Secure Hash Shift 2	0x150									CAA→CA0, CA0→CA1, CA1 → CA2, CA2→CA3, CA3 + CA8 →CA4, CA4 → CA5, CA5 → CA6, CA6 → CA7
Direct load	ILL	Illegal Command	0x1F0									0x1→CASR[IC]

47.7.3.1 Coprocessor No Operation (CNOP)

The CNOP command is the coprocessor no-op. It is issued by the CAU and consumes a location in the CAU FIFO, but has no effect on any CAU register.

47.7.3.2 Load Register (LDR)

The LDR command loads CAx with the source data specified by the write data.

47.7.3.3 Store Register (STR)

The STR command returns the value of CAx specified in the read address to the destination specified as read data.

47.7.3.4 Add to Register (ADR)

The ADR command adds the source operand specified by the write data to CAx and stores the result in CAx.

47.7.3.5 Reverse and Add to Register (RADR)

The RADR command performs a byte reverse on the source operand specified by the write data, adds that value to CAx, and stores the result in CAx. The table below shows an example.

Table 47-3. RADR command example

Operand	CAx before	CAx after
0x0102_0304	0xA0B0_C0D0	0xA4B3_C2D1

47.7.3.6 Add Register to Accumulator (ADRA)

The ADRA command adds CAx to CAA and stores the result in CAA.

47.7.3.7 Exclusive Or (XOR)

The XOR command performs an exclusive-or of the source operand specified by the write data with CAx and stores the result in CAx.

47.7.3.8 Rotate Left (ROTL)

ROTL rotates the CAx bits to the left with the result stored back to CAx. The number of bits to rotate is the value specified by the write data modulo 32.

47.7.3.9 Move Register to Accumulator (MVRA)

The MVRA command moves the value from the source register CAx to the destination register CAA.

47.7.3.10 Move Accumulator to Register (MVAR)

The MVAR command moves the value from source register CAA to the destination register CAx.

47.7.3.11 AES Substitution (AESS)

The AESS command performs the AES byte substitution operation on CAx and stores the result back to CAx.

47.7.3.12 AES Inverse Substitution (AESIS)

The AESIS command performs the AES inverse byte substitution operation on CAx and stores the result back to CAx.

47.7.3.13 AES Column Operation (AESC)

The AESC command performs the AES column operation on the contents of CAx. It then performs an exclusive-or of that result with the source operand specified by the write data and stores the result in CAx.

47.7.3.14 AES Inverse Column Operation (AESIC)

The AESIC command performs an exclusive-or operation of the source operand specified by the write data on the contents of CAx followed by the AES inverse mix column operation on that result and stores the result back in CAx.

47.7.3.15 AES Shift Rows (AESR)

The AESR command performs the AES shift rows operation on registers CA0, CA1, CA2, and CA3. The table below shows an example.

Table 47-4. AESR command example

Register	Before	After
CA0	0x0102_0304	0x0106_0B00
CA1	0x0506_0708	0x050A_0F04
CA2	0x090A_0B0C	0x090E_0308
CA3	0x0D0E_0F00	0x0D02_070C

Where:

- row 1 = CA0[31:24], CA1[31:24], CA2[31:24], CA3[31:24]
- row 2 = CA0[23:16], CA1[23:16], CA2[23:16], CA3[23:16]
- row 3 = CA0[15:8], CA1[15:8], CA2[15:8], CA3[15:8]
- row 4 = CA0[7:0], CA1[7:0], CA2[7:0], CA3[7:0]

47.7.3.16 AES Inverse Shift Rows (AESIR)

The AESIR command performs the AES inverse shift rows operation on registers CA0, CA1, CA2, and CA3. The table below shows an example.

Table 47-5. AESIR command example

Register	Before	After
CA0	0x0106_0B00	0x0102_0304
CA1	0x050A_0F04	0x0506_0708
CA2	0x090E_0308	0x090A_0B0C
CA3	0x0D02_070C	0x0D0E_0F00

Where:

- row 1 = CA0[31:24], CA1[31:24], CA2[31:24], CA3[31:24]
- row 2 = CA0[23:16], CA1[23:16], CA2[23:16], CA3[23:16]
- row 3 = CA0[15:8], CA1[15:8], CA2[15:8], CA3[15:8]
- row 4 = CA0[7:0], CA1[7:0], CA2[7:0], CA3[7:0]

47.7.3.17 DES Round (DESR)

The DESR command performs a round of the DES algorithm and a key schedule update with the following source and destination designations: CA0=C, CA1=D, CA2=L, CA3=R. If the IP bit is set, DES initial permutation performs on CA2 and CA3 before the round operation. If the FP bit is set, DES final permutation, that is, inverse initial permutation, performs on CA2 and CA3 after the round operation. The round operation

uses the source values from registers CA0 and CA1 for the key addition operation. The KSx field specifies the shift for the key schedule operation to update the values in CA0 and CA1. The following table defines the specific shift function performed based on the KSx field.

Table 47-6. Key shift function codes

KSx code	KSx define	Shift function
0	KSL1	Left 1
1	KSL2	Left 2
2	KSR1	Right 1
3	KSR2	Right 2

47.7.3.18 DES Key setup (DESK)

The DESK command performs the initial key transformation, permuted choice 1, defined by the DES algorithm on CA0 and CA1 with CA0 containing bits 1–32 of the key and CA1 containing bits 33–64 of the key[1] . If the DC bit is set, no shift operation performs and the values C_0 and D_0 store back to CA0 and CA1, respectively. The DC bit must be set for decrypt operations. If the DC bit is not set, a left shift by one also occurs and the values C_1 and D_1 store back to CA0 and CA1, respectively. The DC bit should be cleared for encrypt operations. If the CP bit is set and a key parity error is detected, CASR[DPE] bit is set; otherwise, it is cleared.

47.7.3.19 Hash Function (HASH)

The HASH command performs a hashing operation on a set of registers and adds that result to the value in CAA and stores the result in CAA. The specific hash function performed is based on the HFx field as defined in the table below.

This table uses the following terms:

- ROTR^n(CAx): rotate CAx register right *n* times
- SHR^n(CAx): shift CAx register right *n* times

Table 47-7. Hash Function codes

HFx code	HFx define	Hash Function	Hash logic
0	HFF	MD5 F()	(CA1 & CA2) | ($\overline{\text{CA1}}$ & CA3)
1	HFG	MD5 G()	(CA1 & CA3) | (CA2 & $\overline{\text{CA3}}$)

Table continues on the next page...

1. The DES algorithm numbers the most significant bit of a block as bit 1 and the least significant as bit 64.

Table 47-7. Hash Function codes (continued)

HFx code	HFx define	Hash Function	Hash logic
2	HFH	MD5 H(), SHA Parity()	CA1 ^ CA2 ^ CA3
3	HFI	MD5 I()	CA2 ^ (CA1 | $\overline{CA3}$)
4	HFC	SHA Ch()	(CA1 & CA2) ^ ($\overline{CA1}$ & CA3)
5	HFM	SHA Maj()	(CA1 & CA2) ^ (CA1 & CA3) ^ (CA2 & CA3)
6	HF2C	SHA-256 Ch()	(CA4 & CA5) ^ ($\overline{CA1}$ & CA6)
7	HF2M	SHA-256 Maj()	(CA0 & CA1) ^ (CA0 & CA2) ^ (CA1 & CA2)
8	HF2S	SHA-256 Sigma 0	$ROTR^{2}$(CA0) ^ $ROTR^{13}$(CA0) ^ $ROTR^{22}$(CA0)
9	HF2T	SHA-256 Sigma 1	$ROTR^{6}$(CA4) ^ $ROTR^{11}$(CA4) ^ $ROTR^{25}$(CA4)
A	HF2U	SHA-256 Sigma 0	$ROTR^{7}$(CA8) ^ $ROTR^{18}$(CA8) ^ SHR^{3}(CA8)
B	HF2V	SHA-256 Sigma 1	$ROTR^{17}$(CA8) ^ $ROTR^{19}$(CA8) ^ SHR^{10}(CA8)

47.7.3.20 Secure Hash Shift (SHS)

The SHS command does a set of parallel register-to-register move and shift operations for implementing SHA-1. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA4	CA4	CA3
CA3	CA3	CA2
CA2	CA2	CA1<<<30
CA1	CA1	CA0
CA0	CA0	CAA
CAA	CAA	CAA<<<5

47.7.3.21 Message Digest Shift (MDS)

The MDS command does a set of parallel register-to-register move operations for implementing MD5. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA3	CA3	CA2
CA2	CA2	CA1
CA1	CA1	CAA
CAA	CAA	CA3

47.7.3.22 Secure Hash Shift 2 (SHS2)

The SHS2 command does an addition and a set of register to register moves in parallel for implementing SHA-256. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA7	CA7	CA6
CA6	CA6	CA5
CA5	CA5	CA4
CA4	CA4	CA3+CA8
CA3	CA3	CA2
CA2	CA2	CA1
CA1	CA1	CA0
CA0	CA0	CAA

47.7.3.23 Illegal command (ILL)

The ILL command is a specific illegal command that sets CASR[IC]. All other illegal commands are reserved for use in future implementations.

47.8 Application/initialization information

This section discusses how to initialize and use the CAU.

47.8.1 Code example

A code fragment is shown below as an example of how the CAU is used. This example shows the round function of the AES algorithm. Core registers are defined as follows:

- R1 points to the key schedule
- R3 contains 3 direct CAU commands
- R8 contains 2 direct CAU commands
- R9 contains an indirect CAU command
- FP points to the CAU indirect command address space
- IP points to the CAU direct command space

```
movw    fp, #:lower16:MMCAU_PPB_INDIRECT            @ fp -> MMCAU_PPB_INDIRECT
movt    fp, #:upper16:MMCAU_PPB_INDIRECT
movw    ip, #:lower16:MMCAU_PPB_DIRECT              @ ip -> MMCAU_PPB_DIRECT
movt    ip, #:upper16:MMCAU_PPB_DIRECT
```

```
# r3 = mmcau_3_cmds(AESS+CA0,AESS+CA1,AESS+CA2)
    movw    r3, #:lower16:(0x80100200+(AESS+CA0)<<22+(AESS+CA1)<<11+AESS+CA2)
    movt    r3, #:upper16:(0x80100200+(AESS+CA0)<<22+(AESS+CA1)<<11+AESS+CA2)

# r8 = mmcau_2_cmds(AESS+CA3,AESR)
    movw    r8, #:lower16:(0x80100000+(AESS+CA3)<<22+(AESR)<<11)
    movt    r8, #:upper16:(0x80100000+(AESS+CA3)<<22+(AESR)<<11)

    add     r9, fp, $((AESC+CA0)<<2)                        @ r9 = mmcau_cmd(AESC+CA0)

    str     r3, [ip]                                      @ sub bytes w0, w1, w2
    str     r8, [ip]                                      @ sub bytes w3, shift rows
    ldmia   r1!, {r4-r7}                                  @ get next 4 keys; r1++
    stmia   r9, {r4-r7}                                   @ mix columns, add keys
```

47.8.2 Assembler equate values

The following equates ease programming of the CAU.

```
; CAU Registers (CAx)
     .set CASR,0x0
     .set CAA,0x1
     .set CA0,0x2
     .set CA1,0x3
     .set CA2,0x4
     .set CA3,0x5
     .set CA4,0x6
     .set CA5,0x7
     .set CA6,0x8
     .set CA7,0x9
     .set CA8,0xA
; CAU Commands
     .set CNOP,0x000
     .set LDR,0x010
     .set STR,0x020
     .set ADR,0x030
     .set RADR,0x040
     .set ADRA,0x050
     .set XOR,0x060
     .set ROTL,0x070
     .set MVRA,0x080
     .set MVAR,0x090
     .set AESS,0x0A0
     .set AESIS,0x0B0
     .set AESC,0x0C0
     .set AESIC,0x0D0
     .set AESR,0x0E0
     .set AESIR,0x0F0
     .set DESR,0x100
     .set DESK,0x110
     .set HASH,0x120
     .set SHS,0x130
     .set MDS,0x140
     .set SHS2,0x150
     .set ILL,0x1F0
; DESR  Fields
     .set IP,0x08         ; initial permutation
     .set FP,0x04         ; final permutation
     .set KSL1,0x00       ; key schedule left 1 bit
     .set KSL2,0x01       ; key schedule left 2 bits
     .set KSR1,0x02       ; key schedule right 1 bit
     .set KSR2,0x03       ; key schedule right 2 bits
; DESK Field
     .set DC,0x01         ; decrypt key schedule
```

```
     .set CP,0x02          ; check parity
; HASH Functions Codes
     .set HFF,0x0          ; MD5 F() CA1&CA2 | ~CA1&CA3
     .set HFG,0x1          ; MD5 G() CA1&CA3 | CA2&~CA3
     .set HFH,0x2          ; MD5 H(), SHA Parity() CA1^CA2^CA3
     .set HFI,0x3          ; MD5 I()  CA2^(CA1|~CA3)
     .set HFC,0x4          ; SHA Ch() CA1&CA2 ^ ~CA1&CA3
     .set HFM,0x5          ; SHA Maj() CA1&CA2 ^ CA1&CA3 ^ CA2&CA3
     .set HF2C,0x6         ; SHA-256 Ch() CA4&CA5 ^ ~CA4&CA6
     .set HF2M,0x7         ; SHA-256 Maj() CA0&CA1 ^ CA0&CA2 ^ CA1&CA2
     .set HF2S,0x8         ; SHA-256 Sigma 0 ROTR2(CA0)^ROTR13(CA0)^ROTR22(CA0)
     .set HF2T,0x9         ; SHA-256 Sigma 1 ROTR6(CA4)^ROTR11(CA4)^ROTR25(CA4)
     .set HF2U,0xA         ; SHA-256 sigma 0 ROTR7(CA8)^ROTR18(CA8)^SHR3(CA8)
     .set HF2V,0xB         ; SHA-256 sigma 1 ROTR17(CA8)^ROTR19(CA8)^SHR10(CA8)
```

Chapter 48
Cyclic Redundancy Check (CRC)

48.1 Introduction

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 32 bits of data at a time.

48.1.1 Features

Features of the CRC module include:

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 32-bit CPU register programming interface

48.1.2 Block diagram

The following is a block diagram of the CRC.

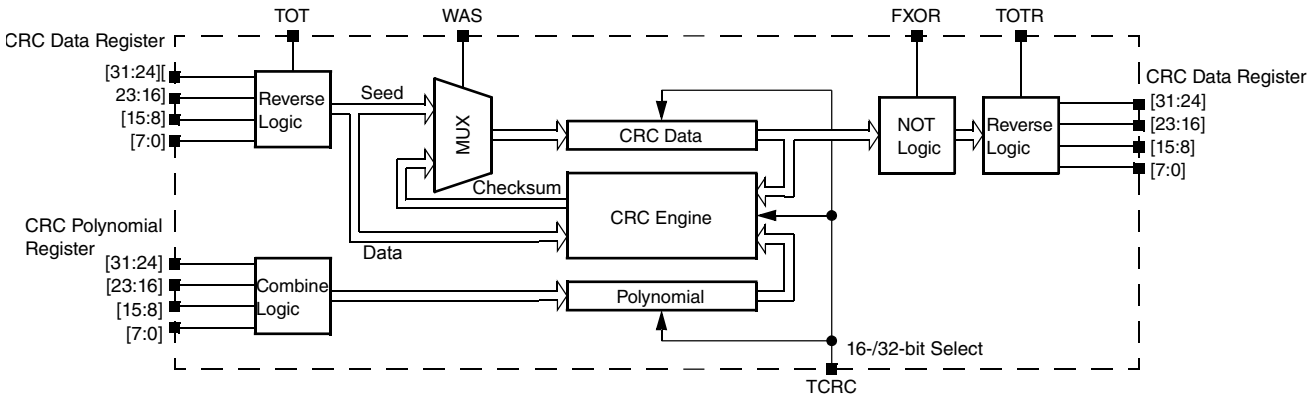


Figure 48-1. Programmable cyclic redundancy check (CRC) block diagram

48.1.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

48.1.3.1 Run mode

This is the basic mode of operation.

48.1.3.2 Low-power modes (Wait or Stop)

Any CRC calculation in progress stops when the MCU enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the MCU.

48.2 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_4000	CRC Data register (CRC_DATA)	32	R/W	FFFF_FFFFh	48.2.1/1073
4003_4004	CRC Polynomial register (CRC_GPOLY)	32	R/W	0000_1021h	48.2.2/1074
4003_4008	CRC Control register (CRC_CTRL)	32	R/W	0000_0000h	48.2.3/1074

48.2.1 CRC Data register (CRC_DATA)

The CRC Data register contains the value of the seed, data, and checksum. When CTRL[WAS] is set, any write to the data register is regarded as the seed value. When CTRL[WAS] is cleared, any write to the data register is regarded as data for general CRC computation.

In 16-bit CRC mode, the HU and HL fields are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with MSB of data value written first.

After all data values are written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is available in the LU and LL fields. In 32-bit CRC mode, all fields contain the result. Reads of this register at any time return the intermediate CRC value, provided the CRC module is configured.

Address: 4003_4000h base + 0h offset = 4003_4000h

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

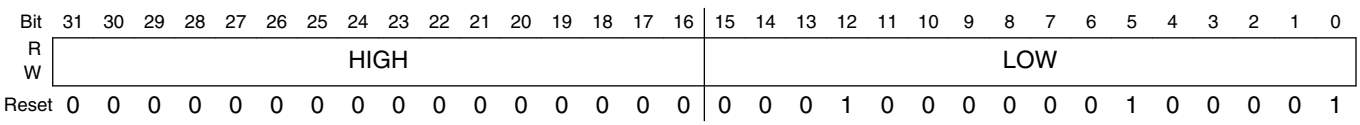
CRC_DATA field descriptions

Field	Description
31–24 HU	CRC High Upper Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
23–16 HL	CRC High Lower Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
15–8 LU	CRC Low Upper Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.
LL	CRC Low Lower Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.

48.2.2 CRC Polynomial register (CRC_GPOLY)

This register contains the value of the polynomial for the CRC calculation. The HIGH field contains the upper 16 bits of the CRC polynomial, which are used only in 32-bit CRC mode. Writes to the HIGH field are ignored in 16-bit CRC mode. The LOW field contains the lower 16 bits of the CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 4003_4000h base + 4h offset = 4003_4004h



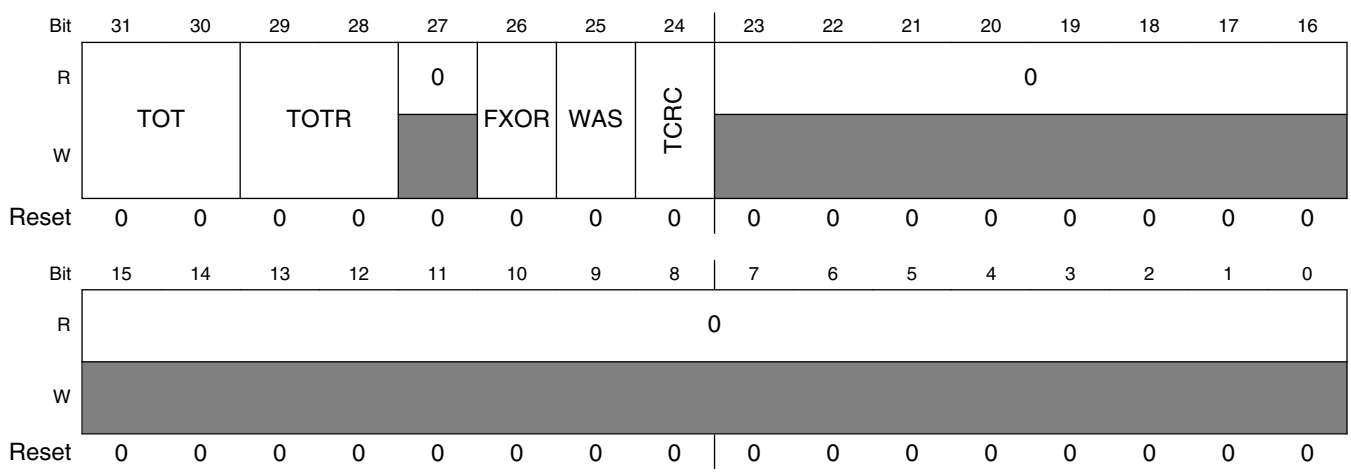
CRC_GPOLY field descriptions

Field	Description
31–16 HIGH	High Polynominal Half-word Writable and readable in 32-bit CRC mode (CTRL[TCRC] is 1). This field is not writable in 16-bit CRC mode (CTRL[TCRC] is 0).
LOW	Low Polynominal Half-word Writable and readable in both 32-bit and 16-bit CRC modes.

48.2.3 CRC Control register (CRC_CTRL)

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC data register.

Address: 4003_4000h base + 8h offset = 4003_4008h



CRC_CTRL field descriptions

Field	Description
31–30 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
29–28 TOTR	Type Of Transpose For Read Identifies the transpose configuration of the value read from the CRC Data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 FXOR	Complement Read Of CRC Data Register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on the fly complementing of read data. 0 No XOR on reading. 1 Invert or complement the read value of the CRC Data register.
25 WAS	Write CRC Data Register As Seed When asserted, a value written to the CRC data register is considered a seed value. When deasserted, a value written to the CRC data register is taken as data for CRC computation. 0 Writes to the CRC data register are data values. 1 Writes to the CRC data register are seed values.
24 TCRC	Width of CRC protocol. 0 16-bit CRC protocol. 1 32-bit CRC protocol.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

48.3 Functional description

48.3.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_GPOLY,necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DATA register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

48.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

48.3.2.1 16-bit CRC

To compute a 16-bit CRC:

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_GPOLY[LOW] field. The CRC_GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DATA[LU:LL]. CRC_DATA[HU:HL] are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[LU:LL].

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

48.3.2.2 32-bit CRC

To compute a 32-bit CRC:

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_GPOLY[HIGH:LOW].
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DATA[HU:HL:LU:LL].
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[HU:HL:LU:LL]. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

48.3.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on the fly while being read or written.

Some protocols use little endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

48.3.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC data register:

1. CTRL[TOT] or CTRL[TOTR] is 00.

No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 01

   Bits in a byte are transposed, while bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}

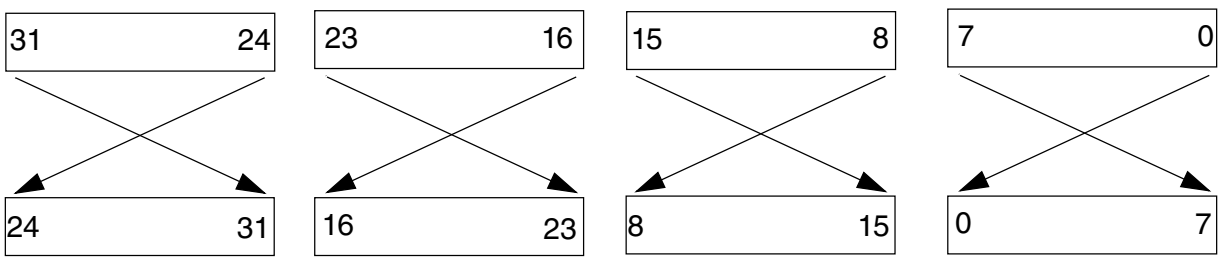


Figure 48-2. Transpose type 01

3. CTRL[TOT] or CTRL[TOTR] is 10.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes = {reg[0:7], reg[8:15],reg[16:23], reg[24:31]}

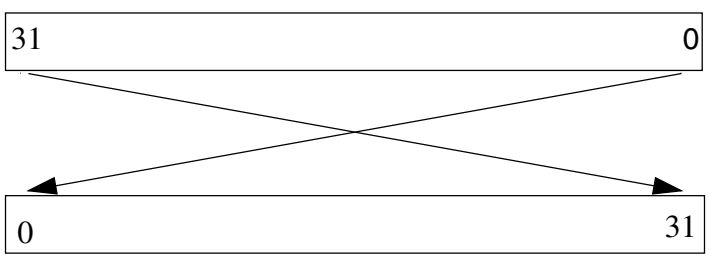


Figure 48-3. Transpose type 10

4. CTRL[TOT] or CTRL[TOTR] is 11.

   Bytes are transposed, but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}

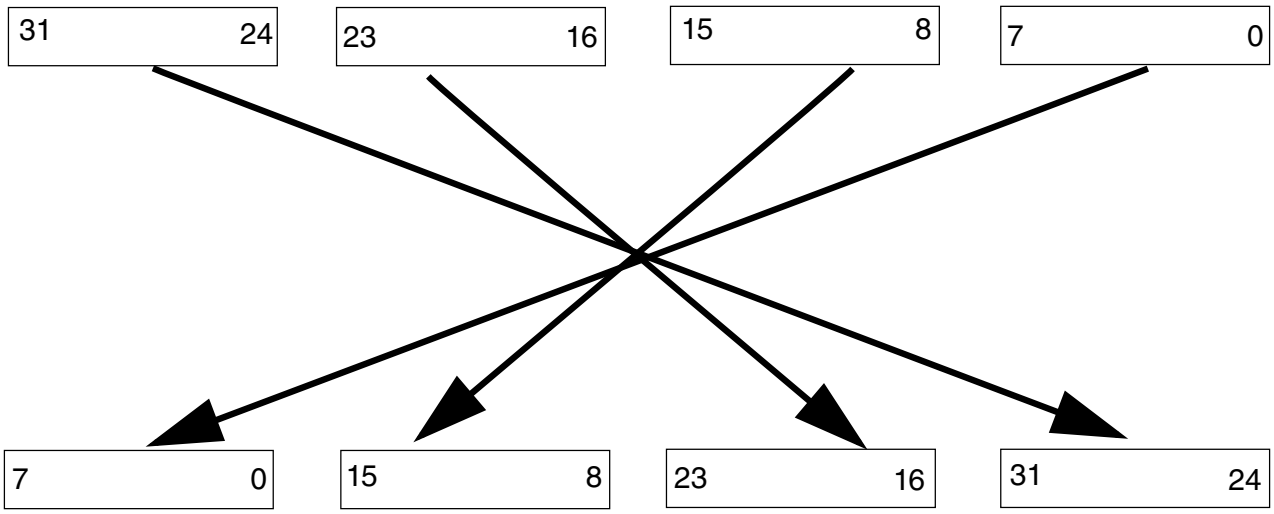


Figure 48-4. Transpose type 11

NOTE

For 8-bit and 16-bit write accesses to the CRC data register, the data is transposed with zeros on the unused byte or bytes (taking 32 bits as a whole), but the CRC is calculated on the valid byte(s) only. When reading the CRC data register for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in the CRC[HU:HL] fields. The user software must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

48.3.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC data register every time the CRC data register is read. When CTRL[FXOR] is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 49
Random Number Generator Accelerator (RNGA)

49.1 Introduction

This chapter describes the random-number-generator accelerator RNGA, including a programming model, functional description, and application information. Throughout this chapter, the terms "RNG" and "RNGA" are meant to be synonymous.

49.1.1 Overview

RNGA is a digital integrated circuit capable of generating 32-bit random numbers. The random bits are generated using shift registers with clocks derived from two free-running, independent ring oscillators. The configuration of the shift registers ensures statistically good data, that is, data that looks random. The oscillators, with their unknown frequencies and independent phases, provide the means of generating the required entropy needed to create random data. The random words generated by RNGA are loaded into an output register (OR). RNGA is designed to generate an error interrupt (if not masked), if OR is read and does not contain valid random data. OR contains valid random data if the LVL field in the status register (SR) is 1.

It is important to note there is no known cryptographic proof showing this is a secure method of generating random data. In fact, there may be an attack against this random number generator if its output is used directly in a cryptographic application. The attack is based on the linearity of the internal shift registers. Therefore, it is highly recommended that this random data produced by this module be used as an entropy source to provide an input seed to a NIST-approved pseudo-random-number generator based on DES or SHA-1 and defined in *NIST FIPS PUB 186-2 Appendix 3* and *NIST FIPS PUB SP 800-90*.

The requirement is to maximize the entropy of this input seed. In order to do this, when data is extracted from RNGA as quickly as the hardware allows, there are about one or two bits of added entropy per 32-bit word. Any single bit of that word contains that

entropy. Therefore, when used as an entropy source, a random number should be generated for each bit of entropy required, and the least significant bit (any bit would be equivalent) of each word retained. The remainder of each random number should then be discarded. Used this way, even with full knowledge of the internal state of RNGA and all prior random numbers, an attacker is not able to predict the values of the extracted bits.

Other sources of entropy can be used along with RNGA to generate the seed to the pseudorandom algorithm. The more random sources combined to create the seed, the better. The following is a list of sources that can be easily combined with the output of this module:

- Current time using highest precision possible
- Real-time system inputs that can be characterized as "random"
- Other entropy supplied directly by the user

49.2 Modes of operation

RNGA supports the following modes of operation.

Table 49-1. Modes of operation supported by RNGA

Mode	Description
Normal	The ring-oscillator clocks are active; RNGA generates entropy (randomness) from the clocks and stores it in shift registers.
Sleep	The ring-oscillator clocks are inactive; RNGA does not generate entropy.

49.2.1 Entering Normal mode

To enter Normal mode, write 0 to CR[SLP].

49.2.2 Entering Sleep mode

To enter Sleep mode, write 1 to CR[SLP].

49.3 Memory map and register definition

This section describes the RNGA registers.

RNG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_9000	RNGA Control Register (RNG_CR)	32	R/W	0000_0010h	49.3.1/1083
4002_9004	RNGA Status Register (RNG_SR)	32	R	0001_0000h	49.3.2/1085
4002_9008	RNGA Entropy Register (RNG_ER)	32	W (always reads 0)	0000_0000h	49.3.3/1087
4002_900C	RNGA Output Register (RNG_OR)	32	R	0000_0000h	49.3.4/1087

49.3.1 RNGA Control Register (RNG_CR)

Controls the operation of RNGA.

Address: 4002_9000h base + 0h offset = 4002_9000h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SLP	0	INTM	HA	GO
W													CLRI			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

RNG_CR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 SLP	Sleep Specifies whether RNGA is in Sleep or Normal mode. **NOTE:** You can also enter Sleep mode by asserting the DOZE signal.

Table continues on the next page...

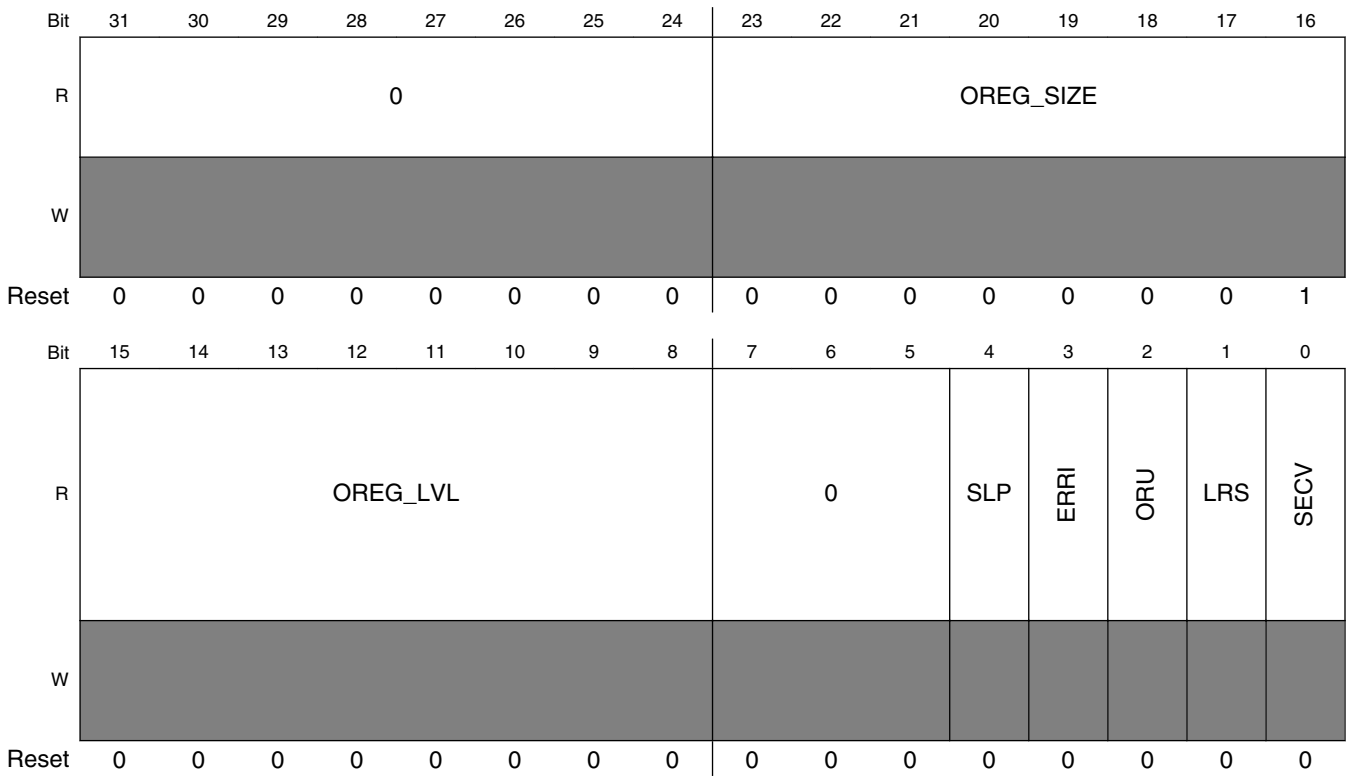
RNG_CR field descriptions (continued)

Field	Description
	0 Normal mode 1 Sleep (low-power) mode
3 CLRI	Clear Interrupt Clears the interrupt by resetting the error-interrupt indicator (SR[ERRI]). **NOTE:** Reading SR[ERRI] immediately after writing to CR[CLRI] is not recommended. 0 Do not clear the interrupt. 1 Clear the interrupt. When you write 1 to this field, RNGA then resets the error-interrupt indicator (SR[ERRI]). This bit always reads as 0.
2 INTM	Interrupt Mask Masks the triggering of an error interrupt to the interrupt controller when an OR underflow condition occurs. An OR underflow condition occurs when you read OR[RANDOUT] and SR[OREG_LVL]=0. See the Output Register (OR) description. 0 Not masked 1 Masked
1 HA	High Assurance Enables notification of security violations (via SR[SECV]). A security violation occurs when you read OR[RANDOUT] and SR[OREG_LVL]=0. **NOTE:** This field is sticky. After enabling notification of security violations, you must reset RNGA to disable them again. 0 Disabled 1 Enabled
0 GO	Go Specifies whether random-data generation and loading (into OR[RANDOUT]) is enabled. **NOTE:** This field is sticky. You must reset RNGA to stop RNGA from loading OR[RANDOUT] with data. 0 Disabled 1 Enabled

49.3.2 RNGA Status Register (RNG_SR)

Indicates the status of RNGA. This register is read-only.

Address: 4002_9000h base + 4h offset = 4002_9004h



RNG_SR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 OREG_SIZE	Output Register Size Indicates the size of the Output (OR) register in terms of the number of 32-bit random-data words it can hold. 1 One word (this value is fixed)
15–8 OREG_LVL	Output Register Level Indicates the number of random-data words that are in OR[RANDOUT], which indicates whether OR[RANDOUT] is valid. **NOTE:** If you read OR[RANDOUT] when SR[OREG_LVL] is not 0, then the contents of a random number contained in OR[RANDOUT] are returned, and RNGA writes 0 to both OR[RANDOUT] and SR[OREG_LVL]. 0 No words (empty) 1 One word (valid)

Table continues on the next page...

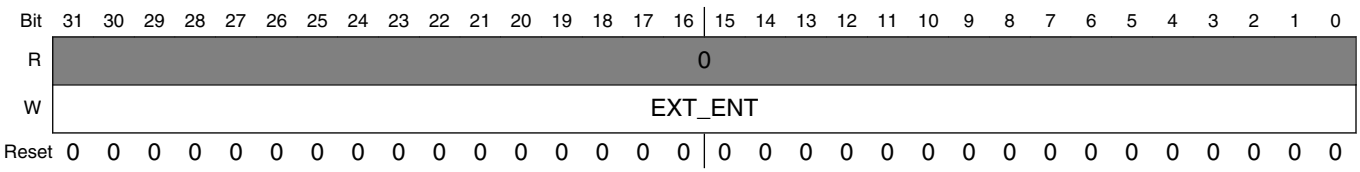
RNG_SR field descriptions (continued)

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 SLP	Sleep Specifies whether RNGA is in Sleep or Normal mode. **NOTE:** You can also enter Sleep mode by asserting the DOZE signal. 0 Normal mode 1 Sleep (low-power) mode
3 ERRI	Error Interrupt Indicates whether an OR underflow condition has occurred since you last cleared the error interrupt (CR[CLRI]) or RNGA was reset, regardless of whether the error interrupt is masked (CR[INTM]). An OR underflow condition occurs when you read OR[RANDOUT] and SR[OREG_LVL]=0. **NOTE:** After you reset the error-interrupt indicator (via CR[CLRI]), RNGA writes 0 to this field. 0 No underflow 1 Underflow
2 ORU	Output Register Underflow Indicates whether an OR underflow condition has occurred since you last read this register (SR) or RNGA was reset, regardless of whether the error interrupt is masked (CR[INTM]). An OR underflow condition occurs when you read OR[RANDOUT] and SR[OREG_LVL]=0. **NOTE:** After you read this register, RNGA writes 0 to this field. 0 No underflow 1 Underflow
1 LRS	Last Read Status Indicates whether the most recent read of OR[RANDOUT] caused an OR underflow condition, regardless of whether the error interrupt is masked (CR[INTM]). An OR underflow condition occurs when you read OR[RANDOUT] and SR[OREG_LVL]=0. **NOTE:** After you read this register, RNGA writes 0 to this field. 0 No underflow 1 Underflow
0 SECV	Security Violation Used only when high assurance is enabled (CR[HA]). Indicates that a security violation has occurred. **NOTE:** This field is sticky. To clear SR[SECV], you must reset RNGA. 0 No security violation 1 Security violation

49.3.3 RNGA Entropy Register (RNG_ER)

Specifies an entropy value that RNGA uses in addition to its ring oscillators to seed its pseudorandom algorithm. This is a write-only register; reads return all zeros.

Address: 4002_9000h base + 8h offset = 4002_9008h



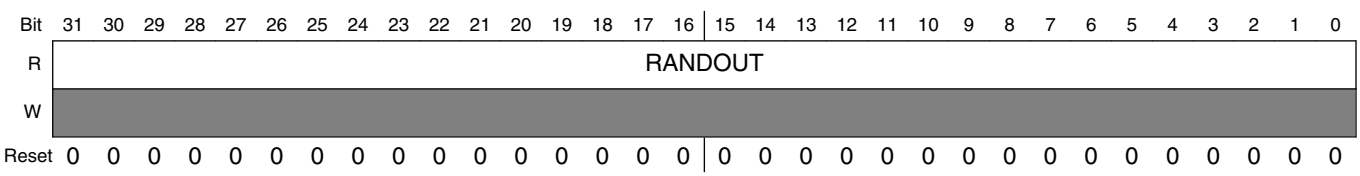
RNG_ER field descriptions

Field	Description
EXT_ENT	External Entropy Specifies an entropy value that RNGA uses in addition to its ring oscillators to seed its pseudorandom algorithm. **NOTE:** Specifying a value for this field is optional but recommended. You can write to this field at any time during operation.

49.3.4 RNGA Output Register (RNG_OR)

Stores a random-data word generated by RNGA.

Address: 4002_9000h base + Ch offset = 4002_900Ch



RNG_OR field descriptions

Field	Description
RANDOUT	Random Output Stores a random-data word generated by RNGA. This is a read-only field. **NOTE:** Before reading RANDOUT, be sure it is valid (SR[OREG_LVL]=1).

RNG_OR field descriptions (continued)

Field	Description	
	0	Invalid data (if you read this field when it is 0 and SR[OREG_LVL] is 0, RNGA then writes 1 to SR[ERRI], SR[ORU], and SR[LRS]; when the error interrupt is not masked (CR[INTM]=0), RNGA also asserts an error interrupt request to the interrupt controller).
	All other values	Valid data (if you read this field when SR[OREG_LVL] is not 0, RNGA returns RANDOUT, and then writes 0 to this field and to SR[OREG_LVL]).

49.4 Functional description

This is a block diagram of RNGA.

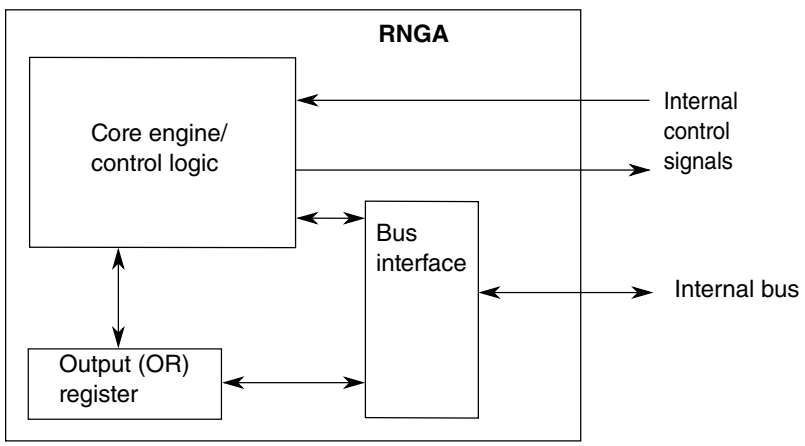


Figure 49-1. RNGA block diagram

49.4.1 Output (OR) register

The Output (OR) register provides temporary storage for random data generated by the core engine / control logic. The Status (SR) register allows the user to monitor the presence of valid random data in OR through SR[OREG_LVL].

If the OR is read while containing valid random data (as signaled by SR[OREG_LVL] = 1), the valid data is returned, then OR and SR[OREG_LVL] are both cleared. If the user reads from OR when it is empty, RNGA returns all zeros and, if the interrupt is enabled, RNGA drives a request to the interrupt controller. Polling SR[OREG_LVL] is very important to make sure random values are present before reading from OR.

49.4.2 Core engine / control logic

This block contains RNGA's control logic as well as its core engine used to generate random data.

49.4.2.1 Control logic

The control logic contains the address decoder, all addressable registers, and control state machines for RNGA. This block is responsible for communication with both the peripheral interface and the Output (OR) register interface. The block also controls the core engine to generate random data. The general functionality of the block is as follows:

After reset, RNGA operates in Normal mode as follows:

1. The core engine generates entropy and stores it in the shift registers.
2. After you enable random-data generation by loading CR[GO], every 256 clock cycles the core engine generates a new random-data word. If SR[OREG_LVL] = 0, then the control block loads the new random data into OR and set SR[OREG_LVL] = 1; else the new data is discarded.

49.4.2.2 Core engine

The core engine block contains the logic used to generate random data. The logic within the core engine contains the internal shift registers as well as the logic used to generate the two oscillator-based clocks. The control logic determines how the shift registers are configured as well as when the oscillator clocks are turned on.

49.5 Initialization/application information

The intended general operation of RNGA is as follows:

1. Reset/initialize.
2. Write 1 to CR[INTM], CR[HA], and CR[GO].
3. Poll SR[OREG_LVL] until it is not 0.
4. When SR[OREG_LVL] is not 0, read the available random data from OR[RANDOUT].
5. Repeat steps 3 and 4 as needed.

For application information, see Overview.

Chapter 50
Segment LCD Controller (SLCD)

50.1 Chip-specific SLCD information

50.1.1 Instantiation information

The following table lists the Segment LCD (SLCD) support information across the KM3x_256 series of devices.

Table 50-1. SLCD support[1]

Max. number of segments	Configurations
Up to 8×47	Up to 8 backplanes and up to 60 front planes 4×60, 6×58 and 8×56 configurations are supported

1. The LCD controller can support multiplexing from ×1 up to ×8.

NOTE

- 5 V LCD glass is not supported
- Default configuration of VSUPPLY is 0. Here VLL3 is derived from VDD_3V internally. If VLL3 is derived from any other source than VDD on board , there will be a current between VLL3 and VDD at bootup. It is recommended to have VLL3 derived from VDD on board
- Vsupply options to generate VLL3 from VLL2 (VSUPPLY=0) can generate maximum 3.6 V on VLL3 at MCU VDD (1.8 V — 3.6 V)
- KM family devices have a 1/3 bias controller that works with a 1/3 bias LCD glass. To avoid ghosting, the LCD OFF threshold should be greater than VLL1 level. If the LCD glass has an OFF threshold less than VLL1 level, use

the internal VREG mode and generate VLL1 internally using RVTRIM option. This can reduce VLL1 level to allow for a lower OFF threshold LCD glass.

50.2 Introduction

The SLCD module is a CMOS charge pump voltage inverter that is designed for low-voltage and low-power operation. SLCD is designed to generate the appropriate waveforms to drive multiplexed numeric, alphanumeric, or custom segment LCD panels. SLCD also has several timing and control settings that can be software configured depending on the application's requirements. Timing and control consists of registers and control logic for:

- LCD frame frequency
- Duty cycle selection
- Front plane/back plane selection and enabling
- Blink modes and frequency
- Operation in low-power modes

50.2.1 Features

SLCD includes these features:

- LCD waveforms functional in all low-power modes
- 64 LCD pins with selectable front plane/back plane configuration to:
  - Generate up to 63 front plane signals
  - Generate up to 8 back plane signals
- Programmable LCD frame frequency
- Programmable blink modes and frequency
  - All segments blank during blink period
  - Alternate display for each LCD segment in x4 or less mode
  - Blink operation in low-power modes

- Programmable LCD duty cycle from static to 1/8
- Programmable LCD power supply switch, making it an ideal solution for battery-powered and board-level applications
  - Charge pump that requires only four external capacitors
  - Internal LCD power using V_{DD}
  - Internal V_{IREG} regulated power supply option
  - External V_{LL3} power supply option
- Internal regulated voltage source with a 4-bit trim register to apply contrast control
- Integrated charge pump for generating LCD bias voltages
  - On-chip generation of bias voltages
- Waveform storage registers (WF)
- Low power consumption in standby modes
- Back plane reassignment to assist in vertical scrolling on dot-matrix displays
- Software configurable LCD frame frequency interrupt
- Support for segment fault detection

50.2.2 Modes of operation

SLCD supports the following operating modes:

Table 50-2. SLCD operation modes

Mode	LCD behavior
Run	SLCD can operate an LCD panel. SLCD continues displaying the current LCD panel contents based on the WF registers.
Wait	Depending on the configuration, SLCD can operate an LCD panel in Wait mode. If LCDDOZE = 1, SLCD clock generation is turned off and the SLCD enters a power-conservation state and is disabled. If LCDDOZE = 0, the LCD controller can operate an LCD panel in Wait mode, and SLCD continues displaying the current LCD panel contents based on the WF registers. In Wait mode, the LCD frame interrupt can cause the MCU to exit Wait.
Stop	Depending on the state of the LCDSTP bit, SLCD can operate an LCD panel in Stop mode. If LCDSTP = 1, SLCD clock generation is turned off and the LCD controller enters a power conservation state and is disabled. If LCDSTP = 0, the LCD controller can operate an LCD panel in Stop mode, and SLCD continues to display the current LCD panel contents based on the LCD operation prior to the Stop event.

Table continues on the next page...

Table 50-2. SLCD operation modes (continued)

Mode	LCD behavior
	If the LCD is enabled in Stop mode, the selected LCD clock source must be enabled to operate in Stop mode. The LCD frame interrupt does not cause the MCU to exit Stop mode.
VLPR	SLCD behavior is the same as in Run mode.
VLPW	SLCD behavior is the same as in Wait mode.
VLPS	SLCD behavior is the same as in Stop mode.
LLS	SLCD behavior is the same as in Stop mode.
VLLS3	SLCD behavior is the same as in Stop mode.
VLLS2	SLCD behavior is the same as in Stop mode.
VLLS1	SLCD behavior is the same as in Stop mode.

NOTE

LCD End of Frame wakeup is not supported in LLS and VLLSx modes.

NOTE

The clock source and power modes are chip-specific. For the clock source and power-mode assignments, see the chapter that describes how modules are configured.

50.2.3 Block diagram

- shows the SLCD block diagram.

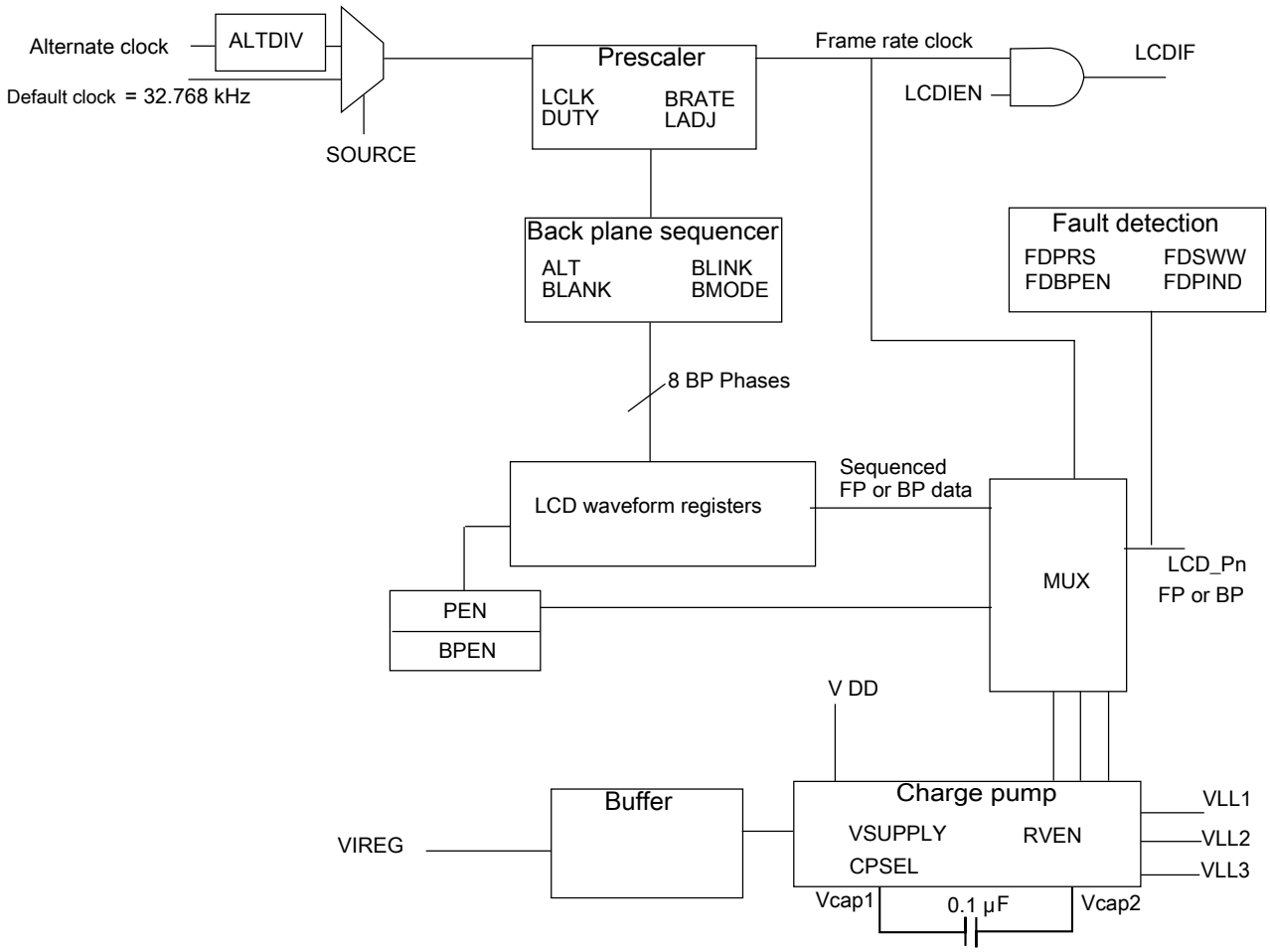


Figure 50-1. SLCD block diagram

50.3 LCD signal descriptions

SLCD has several external pins dedicated to power supply and LCD front plane/back plane signaling. SLCD can be configured to support up to eight back plane signals. This table lists and describes the LCD external signals or pins.

Table 50-3. LCD signal descriptions

Signal	Description	I/O
LCD_P[63:0] . 64 LCD front plane/back plane	Configurable front plane/back plane driver that connects directly to the display LCD_P[63:0] can operate as GPIO pins	O
V_{LL1}, V_{LL2}, V_{LL3}. LCD bias voltages	LCD bias voltages (requires external capacitors when charge pump is used)	I/O
V_{cap1}, V_{cap2}. LCD charge pump capacitance	Charge pump capacitor pins	O

50.3.1 LCD_P[63:0]

When LCD functionality is enabled by the PEN[63:0] bits in the PEN registers, the corresponding LCD_P[63:0] pin generates a front plane or back plane waveform depending on the configuration of the back plane enable field (BPEN[63:0]).

50.3.2 V_{LL1}, V_{LL2}, V_{LL3}

V_{LL1}, V_{LL2}, and V_{LL3} are bias voltages for the LCD controller driver waveforms that can be internally generated using the internal charge pump, when enabled. When using the LCD charge pump, CPSEL = 1, a 0.1 µF capacitor (C_{BYLCD}) must be placed from V_{LLx} to ground. The charge pump can also be configured to accept V_{LL3} as an input and generate V_{LL1} and V_{LL2}. V_{LL3} must never be connected to a voltage other than V_{DD}. Refer to VSUPPLY explanation.

NOTE

LCD pins assigned to VCAP1 and VCAP2 may only be enabled if the pins are in charge pump mode (CPSEL=1).

- When CPSEL=0 and LADJ = 00 or 01, VLL1/2 pins cannot be used as LCD_P pins.
- When CPSEL=0 and LADJ = 10 or 11, VLL1/2 pins can be configured as LCD_P pins or GPIO pins and cannot be configured as bias voltage.
- When CPSEL=1, VLL1/2 cannot be used as LCD_P pins.

50.3.3 V_{cap1}, V_{cap2}

The charge pump capacitor (C_{LCD}) is used to transfer charge from the input supply to the regulated output. Use a ceramic capacitor. A 0.1 µF capacitor must be placed between these two pins. .

50.4 Memory map and register definition

The total address for each register is the sum of the base address for SLCD and the address offset for each register.

This section consists of register descriptions. Each description includes a standard register diagram. Details of register bit and field function follow the register diagrams, in bit order.

LCD memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_3000	LCD General Control Register (LCD_GCR)	32	R/W	See section	50.4.1/1098
4004_3004	LCD Auxiliary Register (LCD_AR)	32	R/W	0000_0000h	50.4.2/1102
4004_3008	LCD Fault Detect Control Register (LCD_FDCR)	32	R/W	0000_0000h	50.4.3/1104
4004_300C	LCD Fault Detect Status Register (LCD_FDSR)	32	R/W	0000_0000h	50.4.4/1106
4004_3010	LCD Pin Enable register (LCD_PENL)	32	R/W	0000_0000h	50.4.5/1107
4004_3014	LCD Pin Enable register (LCD_PENH)	32	R/W	0000_0000h	50.4.5/1107
4004_3018	LCD Back Plane Enable register (LCD_BPENL)	32	R/W	0000_0000h	50.4.6/1108
4004_301C	LCD Back Plane Enable register (LCD_BPENH)	32	R/W	0000_0000h	50.4.6/1108
4004_3020	LCD Waveform register (LCD_WF3TO0)	32	R/W	0000_0000h	50.4.7/1108
4004_3024	LCD Waveform register (LCD_WF7TO4)	32	R/W	0000_0000h	50.4.8/1109
4004_3028	LCD Waveform register (LCD_WF11TO8)	32	R/W	0000_0000h	50.4.9/1110
4004_302C	LCD Waveform register (LCD_WF15TO12)	32	R/W	0000_0000h	50.4.10/ 1111
4004_3030	LCD Waveform register (LCD_WF19TO16)	32	R/W	0000_0000h	50.4.11/ 1111
4004_3034	LCD Waveform register (LCD_WF23TO20)	32	R/W	0000_0000h	50.4.12/ 1112
4004_3038	LCD Waveform register (LCD_WF27TO24)	32	R/W	0000_0000h	50.4.13/ 1112
4004_303C	LCD Waveform register (LCD_WF31TO28)	32	R/W	0000_0000h	50.4.14/ 1113
4004_3040	LCD Waveform register (LCD_WF35TO32)	32	R/W	0000_0000h	50.4.15/ 1113
4004_3044	LCD Waveform register (LCD_WF39TO36)	32	R/W	0000_0000h	50.4.16/ 1114
4004_3048	LCD Waveform register (LCD_WF43TO40)	32	R/W	0000_0000h	50.4.17/ 1115
4004_304C	LCD Waveform register (LCD_WF47TO44)	32	R/W	0000_0000h	50.4.18/ 1115
4004_3050	LCD Waveform register (LCD_WF51TO48)	32	R/W	0000_0000h	50.4.19/ 1116
4004_3054	LCD Waveform register (LCD_WF55TO52)	32	R/W	0000_0000h	50.4.20/ 1116
4004_3058	LCD Waveform register (LCD_WF59TO56)	32	R/W	0000_0000h	50.4.21/ 1117
4004_305C	LCD Waveform register (LCD_WF63TO60)	32	R/W	0000_0000h	50.4.22/ 1117

50.4.1 LCD General Control Register (LCD_GCR)

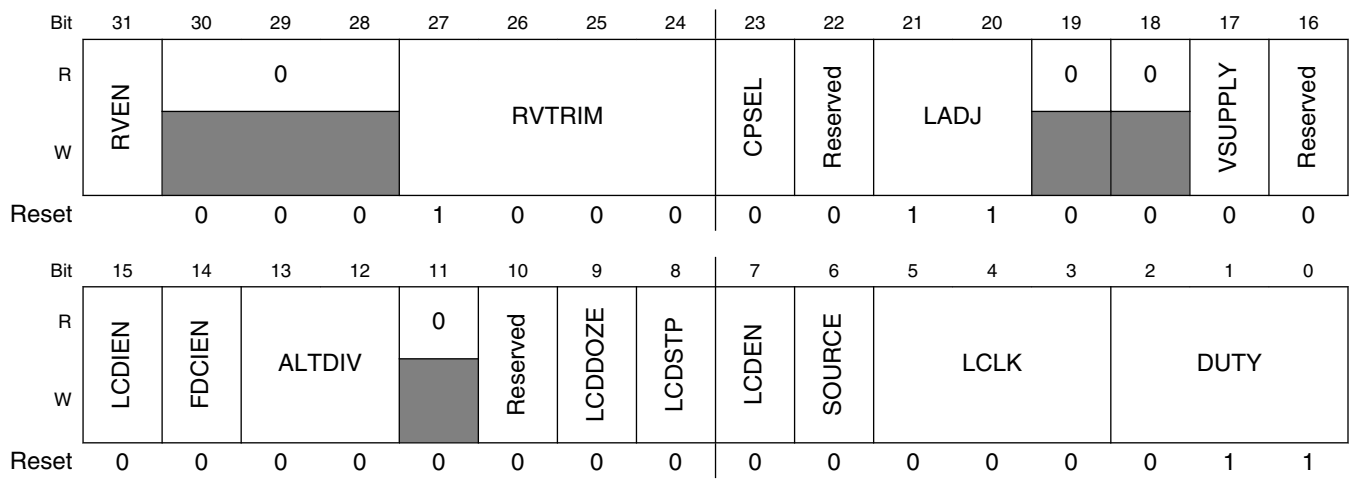
Write: LCDEN anytime. Do not change SOURCE, LCLK, or DUTY while LCDEN = 1. For proper operation, do not modify VSUPPLY while the LCDEN bit is asserted. VSUPPLY must also be configured according to the external hardware power supply configuration.

NOTE

The reset value of this register depends on the reset type:

- POR -- 0x0830_0003

Address: 4004_3000h base + 0h offset = 4004_3000h



LCD_GCR field descriptions

Field	Description
31 RVEN	Regulated Voltage Enable Enables internal voltage regulator. It must have the charge pump enabled. See Table 50-12 for more information about this bitfield configuration. 0 Regulated voltage disabled. 1 Regulated voltage enabled.
30–28 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
27–24 RVTRIM	Regulated Voltage Trim This 4-bit trim register is used to adjust the regulated input. Each bit in the register has equal weight. The regulated input is changed by 1.5% for each count.
23 CPSEL	Charge Pump or Resistor Bias Select

Table continues on the next page...

LCD_GCR field descriptions (continued)

Field	Description
	Selects the LCD controller charge pump or a resistor network to supply the LCD voltages V_{LL1}, V_{LL2}, and V_{LL3}. 0 LCD charge pump is disabled. Resistor network selected. (The internal 1/3-bias is forced.) 1 LCD charge pump is selected. Resistor network disabled. (The internal 1/3-bias is forced.)
22 Reserved	Reserved This field is reserved. **NOTE:** This bit should always be zero and nothing should be written to it
21–20 LADJ	Load Adjust Configures SLCD to handle different LCD glass capacitance.

For CPSEL = 0	Adjust the resistor bias network for different LCD glass capacitance.
	00 — Low Load (LCD glass capacitance 2000 pF or lower).
	01 — Low Load (LCD glass capacitance 2000 pF or lower).
	10 — High Load (LCD glass capacitance 8000 pF or lower) .
	11 — High Load (LCD glass capacitance 8000 pF or lower).
For CPSEL = 1	• Adjust the clock source for the charge pump. • Higher loads require higher charge pump clock rates.
	00 — Fastest clock source for charge pump (LCD glass capacitance 8000 pF or).
	01 — Intermediate clock source for charge pump (LCD glass capacitance 4000 pF or).
	10 — Intermediate clock source for charge pump (LCD glass capacitance 2000 pF or).
	11 — Slowest clock source for charge pump (LCD glass capacitance 1000 pF or).

19 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
18 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
17 VSUPPLY	Voltage Supply Control Configures whether the LCD controller power supply is external or internal. Avoid modifying this field while the LCD controller is enabled, for example, if LCDEN = 1.

Table continues on the next page...

LCD_GCR field descriptions (continued)

Field	Description
	0 Drive V_{LL3} internally from V_{DD} 1 Drive V_{LL3} externally from V_{DD} or drive V_{LL1} internally from v_{IREG}
16 Reserved	This field is reserved.
15 LCDIEN	LCD Frame Frequency Interrupt Enable Enables an LCD interrupt event that coincides with the LCD controller frame frequency. 0 No interrupt request is generated by this event. 1 When LCDIF bit is set, this event causes an interrupt request.
14 FDCIEN	LCD Fault Detection Complete Interrupt Enable Enables an LCD interrupt event when fault detection is completed. 0 No interrupt request is generated by this event. 1 When a fault is detected and FDCF bit is set, this event causes an interrupt request.
13–12 ALTDIV	LCD Alternate Clock Divider Functions as a clock divider to divide the alternate clock before it is selected as LCD clock source. 00 Divide factor = 1 (No divide) 01 Divide factor = 64 10 Divide factor = 256 11 Divide factor = 512
11 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	Reserved This field is reserved. **NOTE:** This bit should always be zero and nothing should be written to it
9 LCDDOZE	LCD Doze enable LCD driver, charge pump, resistor bias network, and voltage regulator stop while in Doze mode. 0 Allows the LCD driver, charge pump, resistor bias network, and voltage regulator to continue running during Doze mode. 1 Disables the LCD driver, charge pump, resistor bias network, and voltage regulator when MCU enters Doze mode.
8 LCDSTP	LCD Stop LCD driver, charge pump, resistor bias network, and voltage regulator stop while in Stop mode. 0 Allows the LCD driver, charge pump, resistor bias network, and voltage regulator to continue running during Stop mode. 1 Disables the LCD driver, charge pump, resistor bias network, and voltage regulator when MCU enters Stop mode.
7 LCDEN	LCD Driver Enable

Table continues on the next page...

LCD_GCR field descriptions (continued)

Field	Description
	Starts LCD controller waveform generator. 0 All front plane and back plane pins are disabled. The LCD controller system is also disabled, and all LCD waveform generation clocks are stopped. V_{LL3} is connected to V_{DD} internally. 1 LCD controller driver system is enabled, and front plane and back plane waveforms are generated. All LCD pins, LCD_Pn, enabled using the LCD Pin Enable register, output an LCD driver waveform. The back plane pins output an LCD driver back plane waveform based on the settings of DUTY[2:0]. Charge pump or resistor bias is enabled.
6 SOURCE	LCD Clock Source Select The LCD controller has two possible clock sources. This bit is used to select which clock source is the basis for LCD clock. **NOTE:** The clock sources are chip-specific. For the clock source assignments, see the chapter that describes how modules are configured. Alternate clock can be only MCGIRCLK, and therefore when SOURCE=1, SIM_SOPT1[OSC32KSEL] bit plays no role. Default clock can be either MCGIRCLK, ERCLK32K, OSC32KCLK, according to the selection in SIM_SOPT1[OSC32KSEL]. 0 Selects the default clock as the LCD clock source. 1 Selects the alternate clock as the LCD clock source.
5–3 LCLK	LCD Clock Prescaler Used as a clock divider to generate the SLCD frame frequency. LCD controller duty cycle configuration is used to determine the LCD controller frame frequency. LCD controller frame frequency calculations are provided in SLCD base clock and frame frequency . `LCD controller frame frequency = LCD clock/((DUTY + 1) × 8 × (4 + LCLK[2:0]) × Y)` *where:* • 30 < LCD clock < 39.063 kHz • Y = 2, 2, 3, 3, 4, 5, 8, 16 chosen by module duty cycle configuration
DUTY	LCD duty select Selects the duty cycle of the LCD controller driver. 000 Use 1 BP (1/1 duty cycle). 001 Use 2 BP (1/2 duty cycle). 010 Use 3 BP (1/3 duty cycle). 011 Use 4 BP (1/4 duty cycle). (Default) 100 Use 5 BP (1/5 duty cycle). 101 Use 6 BP (1/6 duty cycle). 110 Use 7 BP (1/7 duty cycle). 111 Use 8 BP (1/8 duty cycle).

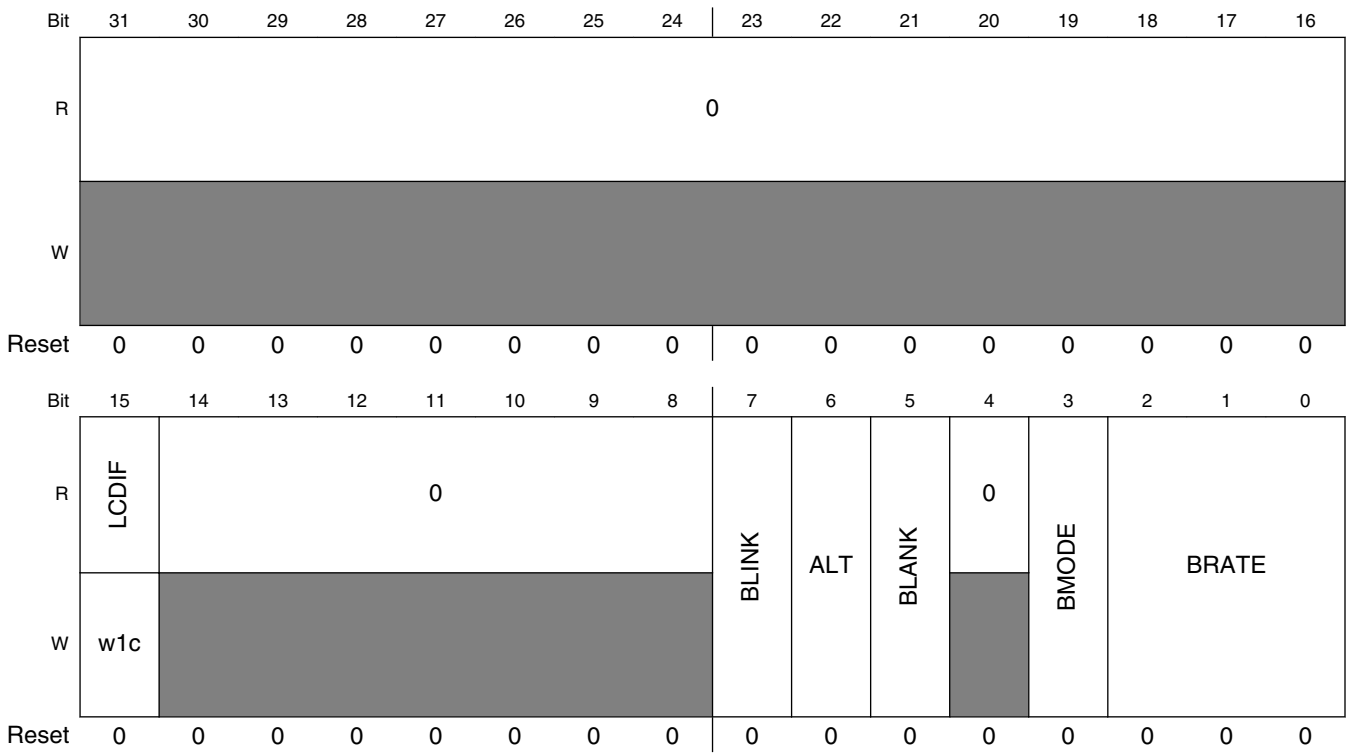
50.4.2 LCD Auxiliary Register (LCD_AR)

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 4h offset = 4004_3004h



LCD_AR field descriptions

Field	Description
31–16 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
15 LCDIF	LCD Frame Frequency Interrupt flag Indicates a frame frequency interrupt condition has occurred. To clear the interrupt, write 1 to LCDIF. 0 Frame frequency interrupt condition has not occurred. 1 Start of SLCD frame has occurred.
14–8 Reserved	Reserved This field is reserved.

Table continues on the next page...

LCD_AR field descriptions (continued)

Field	Description
	This read-only field is reserved and always has the value 0.
7 BLINK	Blink command Starts or stops SLCD blinking. 0 Disables blinking. 1 Starts blinking at blinking frequency specified by LCD blink rate calculation.
6 ALT	Alternate display mode For four back planes or less, the LCD back plane sequencer changes to output an alternate display. ALT bit is ignored if DUTY[2:0] is 100 or greater. 0 Normal display mode. 1 Alternate display mode.
5 BLANK	Blank display mode Asserting this bit clears all segments in the LCD. 0 Normal or alternate display mode. 1 Blank display mode.
4 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
3 BMODE	Blink mode Selects the blink mode displayed during the blink period. 0 Display blank during the blink period. 1 Display alternate display during blink period (Ignored if duty is 5 or greater).
BRATE	Blink-rate configuration Selects frequency at which the LCD blinks when the BLINK bit is asserted. The following equation provides an expression for the LCD controller blink rate and shows how BRATE field is used in the LCD blink-rate calculation. LCD controller blink rate = LCD clock /$2^{(12 + BRATE)}$

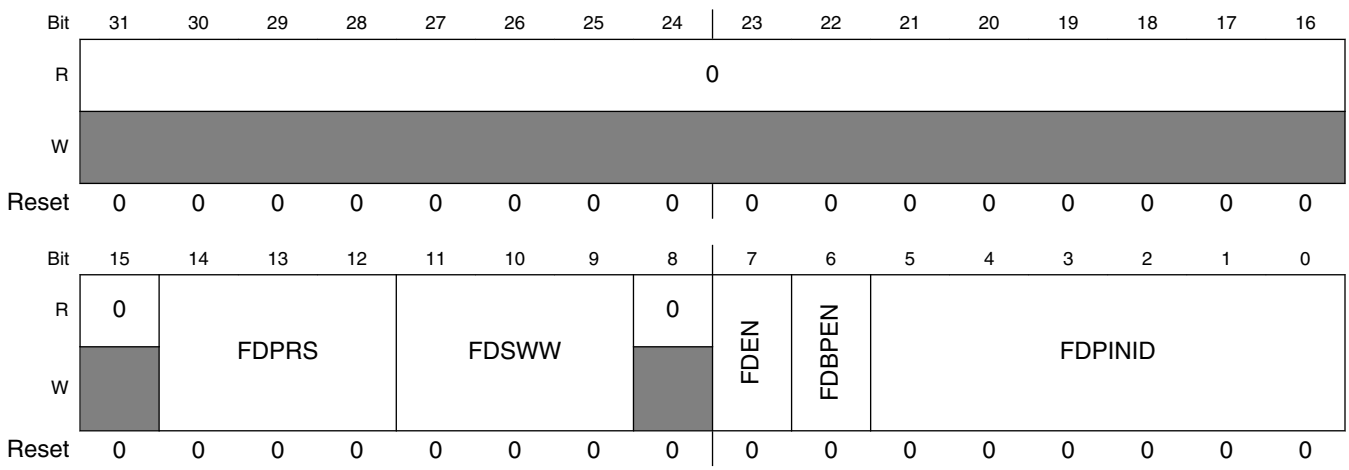
50.4.3 LCD Fault Detect Control Register (LCD_FDCR)

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 8h offset = 4004_3008h



LCD_FDCR field descriptions

Field	Description
31–16 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
14–12 FDPRS	Fault Detect Clock Prescaler Fault detect sample clock frequency is: 0 1/1 bus clock. 1 1/2 bus clock. 2 1/4 bus clock. 3 1/8 bus clock. 4 1/16 bus clock. 5 1/32 bus clock. 6 1/64 bus clock. 7 1/128 bus clock.

Table continues on the next page...

LCD_FDCR field descriptions (continued)

Field	Description
11–9 FDSWW	Fault Detect Sample Window Width Specifies the sample window width of fault detection, in number of cycles in the range from 4–512 (Sample window = $4*2^N$). 0 Sample window width is 4 sample clock cycles. 1 Sample window width is 8 sample clock cycles. 2 Sample window width is 16 sample clock cycles. 3 Sample window width is 32 sample clock cycles. 4 Sample window width is 64 sample clock cycles. 5 Sample window width is 128 sample clock cycles. 6 Sample window width is 256 sample clock cycles. 7 Sample window width is 512 sample clock cycles.
8 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
7 FDEN	Fault Detect Enable If LCDEN is 1, asserting FDEN inserts a test frame after normal LCD refresh frame is completed. After the test frame is done, Fault detection complete flag (FDCF) is set. When the test frame is done, a normal LCD refresh frame starts. FDEN is one-shot register, it clears after FDCF is set. To initiate another fault detection, FDEN must be set again. 0 Disable fault detection. 1 Enable fault detection.
6 FDBPEN	Fault Detect Back Plane Enable Enables "back plane" timing for the fault detect circuit. FDBPEN = 0 generates front plane timing. This bit specifies the type of pin selected under fault detect test. 0 Type of the selected pin under fault detect test is front plane. 1 Type of the selected pin under fault detect test is back plane.
FDPINID	Fault Detect Pin ID Specifies the LCD pin to be checked by pullup fault detection. 0 Fault detection for LCD_P0 pin. 1 Fault detection for LCD_P1 pin. ... 63 Fault detection for LCD_P63 pin.

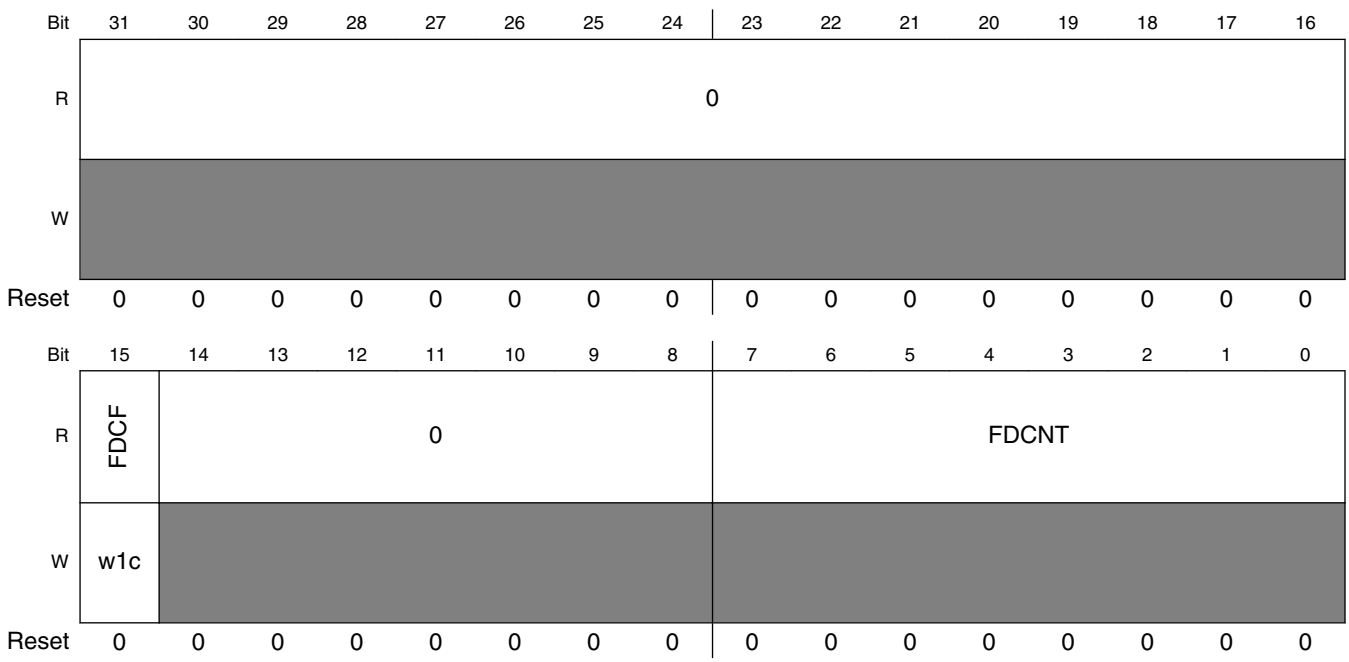
50.4.4 LCD Fault Detect Status Register (LCD_FDSR)

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + Ch offset = 4004_300Ch



LCD_FDSR field descriptions

Field	Description
31–16 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
15 FDCF	Fault Detection Complete Flag FDCF indicates that the fault detection is completed. Writing 1 to this bit clears it to zero. This bit also acts as an interrupt flag when FDCIEN is set. Software can use either interrupt or polling to check whether one pin fault detection is completed. 0 Fault detection is not completed. 1 Fault detection is completed.
14–8 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

LCD_FDSR field descriptions (continued)

Field	Description
FDCNT	Fault Detect Counter Contains how many “one/high” are sampled inside the fault detect sample window. 0 No "one" samples. 1 1 "one" samples. 2 2 "one" samples. ... 254 254 "one" samples. 255 255 or more "one" samples. The FDCNT can overflow. Therefore, FDSWW and FDPRS must be reconfigured for proper sampling.

50.4.5 LCD Pin Enable register (LCD_PEN*n*)

When LCDEN = 1, each PEN bit enables the corresponding LCD pin (LCD_Pn) for LCD waveform generation.

Initialize these registers before enabling the LCD controller.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 10h offset + (4d × i), where i=0d to 1d

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

LCD_PEN*n* field descriptions

Field	Description
PEN	LCD Pin Enable The PEN[63:0] bits enable the LCD_P[63:0] pins for LCD operation. PENL contains PEN[31:0], and PENH contains PEN[63:32]. Each LCD_P[63:0] pin can be configured as a back plane or a front plane based on the corresponding BPEN[*n*] bit in the Back Plane Enable register (BPEN). If LCDEN = 0, these bits have no effect on the state of the I/O pins. Set PEN[63:0] bits before LCDEN is set. 0 LCD operation disabled on LCD_Pn. 1 LCD operation enabled on LCD_Pn.

50.4.6 LCD Back Plane Enable register (LCD_BPEN*n*)

When PEN[n] = 1, the BPEN[63:0] bits configure the corresponding LCD pin to operate as an LCD back plane or an LCD front plane. Most applications set a maximum of eight of these bits. Initialize these registers before enabling the LCD controller.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 18h offset + (4d × i), where i=0d to 1d

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

LCD_BPEN*n* field descriptions

Field	Description
BPEN	Back Plane Enable The BPEN[63:0] bits configure the LCD_P[63:0] pins to operate as an LCD back plane or LCD front plane. BPENL contains BPEN[31:0], and BPENH contains BPEN[63:32]. If LCDEN = 0, these bits have no effect on the state of the I/O pins. It is recommended to set BPEN[63:0] bits before LCDEN is set. 0 Front plane operation enabled on LCD_Pn. 1 Back plane operation enabled on LCD_Pn.

50.4.7 LCD Waveform register (LCD_WF3TO0)

Each of the WFyTOx registers contains four waveform control (WFn) fields, where x is the n index value of the WFn field in the least significant byte (bits 7-0) and y is the n index value of the WFn field in the most significant byte (bits 31-24). The bits in each WFn field control the front plane segments or back plane phases connected to the LCD_Pn signal.

In an LCD controller, each element consists of a front plane segment and a back plane phase. These segments and phases are labeled A through H (x8 multiplexing, 1/8 duty cycle). Each LCD_Pn signal can be connected to one or more segments (in front plane operation) or one or more phases (in back plane operation). An LCD element is turned on when the associated back plane phase is activated and the front plane segment is on.

If LCD_Pn is configured for front plane operation, the bits in WFn turn on or off each of the front plane segments connected to LCD_Pn: bit 0 controls segment A, bit 1 controls segment B, and so on.

If LCD_Pn is configured for back plane operation, the bits in WFn activate or deactivate each of the back plane phases connected to LCD_Pn: bit 0 controls phase A, bit 1 controls phase B, and so on.

Software can write to this register with 8-bit, 16-bit, or 32-bit writes. After reset, the WFyTOx register is cleared to 0.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 20h offset = 4004_3020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF3								WF2								WF1								WF0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF3TO0 field descriptions

Field	Description
31–24 WF3	**Segment-on front plane operation —** Each bit turns on or off the segments associated with LCD_P3 in the following pattern: HGFEDCBA (most significant bit controls segment H and least significant bit controls segment A). **Segment-on back plane operation —** Each bit activates or deactivates the phases associated with LCD_P3 in the following pattern: HGFEDCBA (most significant bit controls phase H and least significant bit controls phase A). For each bit: 0 Segment off or phase deactivated 1 Segment on or phase activated
23–16 WF2	Controls segments or phases connected to LCD_P2 as described above for WF3.
15–8 WF1	Controls segments or phases connected to LCD_P1 as described above for WF3.
WF0	Controls segments or phases connected to LCD_P0 as described above for WF3.

50.4.8 LCD Waveform register (LCD_WF7TO4)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 24h offset = 4004_3024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF7								WF6								WF5								WF4							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF7TO4 field descriptions

Field	Description
31–24 WF7	Controls segments or phases connected to LCD_P7 as described above for WF3TO0[WF3].
23–16 WF6	Controls segments or phases connected to LCD_P6 as described above for WF3TO0[WF3].
15–8 WF5	Controls segments or phases connected to LCD_P5 as described above for WF3TO0[WF3].
WF4	Controls segments or phases connected to LCD_P4 as described above for WF3TO0[WF3].

50.4.9 LCD Waveform register (LCD_WF11TO8)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 28h offset = 4004_3028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF11								WF10								WF9								WF8							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF11TO8 field descriptions

Field	Description
31–24 WF11	Controls segments or phases connected to LCD_P11 as described above for WF3TO0[WF3].
23–16 WF10	Controls segments or phases connected to LCD_P10 as described above for WF3TO0[WF3].
15–8 WF9	Controls segments or phases connected to LCD_P9 as described above for WF3TO0[WF3].
WF8	Controls segments or phases connected to LCD_P8 as described above for WF3TO0[WF3].

50.4.10 LCD Waveform register (LCD_WF15TO12)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 2Ch offset = 4004_302Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF15								WF14								WF13								WF12							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF15TO12 field descriptions

Field	Description
31–24 WF15	Controls segments or phases connected to LCD_P15 as described above for WF3TO0[WF3].
23–16 WF14	Controls segments or phases connected to LCD_P14 as described above for WF3TO0[WF3].
15–8 WF13	Controls segments or phases connected to LCD_P13 as described above for WF3TO0[WF3].
WF12	Controls segments or phases connected to LCD_P12 as described above for WF3TO0[WF3].

50.4.11 LCD Waveform register (LCD_WF19TO16)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 30h offset = 4004_3030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF19								WF18								WF17								WF16							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF19TO16 field descriptions

Field	Description
31–24 WF19	Controls segments or phases connected to LCD_P19 as described above for WF3TO0[WF3].

Table continues on the next page...

LCD_WF19TO16 field descriptions (continued)

Field	Description
23–16 WF18	Controls segments or phases connected to LCD_P18 as described above for WF3TO0[WF3].
15–8 WF17	Controls segments or phases connected to LCD_P17 as described above for WF3TO0[WF3].
WF16	Controls segments or phases connected to LCD_P16 as described above for WF3TO0[WF3].

50.4.12 LCD Waveform register (LCD_WF23TO20)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 34h offset = 4004_3034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF23								WF22								WF21								WF20							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF23TO20 field descriptions

Field	Description
31–24 WF23	Controls segments or phases connected to LCD_P23 as described above for WF3TO0[WF3].
23–16 WF22	Controls segments or phases connected to LCD_P22 as described above for WF3TO0[WF3].
15–8 WF21	Controls segments or phases connected to LCD_P21 as described above for WF3TO0[WF3].
WF20	Controls segments or phases connected to LCD_P20 as described above for WF3TO0[WF3].

50.4.13 LCD Waveform register (LCD_WF27TO24)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 38h offset = 4004_3038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF27								WF26								WF25								WF24							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF27TO24 field descriptions

Field	Description
31–24 WF27	Controls segments or phases connected to LCD_P27 as described above for WF3TO0[WF3].
23–16 WF26	Controls segments or phases connected to LCD_P26 as described above for WF3TO0[WF3].
15–8 WF25	Controls segments or phases connected to LCD_P25 as described above for WF3TO0[WF3].
WF24	Controls segments or phases connected to LCD_P24 as described above for WF3TO0[WF3].

50.4.14 LCD Waveform register (LCD_WF31TO28)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 3Ch offset = 4004_303Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF31								WF30								WF29								WF28							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF31TO28 field descriptions

Field	Description
31–24 WF31	Controls segments or phases connected to LCD_P31 as described above for WF3TO0[WF3].
23–16 WF30	Controls segments or phases connected to LCD_P30 as described above for WF3TO0[WF3].
15–8 WF29	Controls segments or phases connected to LCD_P29 as described above for WF3TO0[WF3].
WF28	Controls segments or phases connected to LCD_P28 as described above for WF3TO0[WF3].

50.4.15 LCD Waveform register (LCD_WF35TO32)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 40h offset = 4004_3040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF35								WF34								WF33								WF32							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF35TO32 field descriptions

Field	Description
31–24 WF35	Controls segments or phases connected to LCD_P35 as described above for WF3TO0[WF3].
23–16 WF34	Controls segments or phases connected to LCD_P34 as described above for WF3TO0[WF3].
15–8 WF33	Controls segments or phases connected to LCD_P33 as described above for WF3TO0[WF3].
WF32	Controls segments or phases connected to LCD_P32 as described above for WF3TO0[WF3].

50.4.16 LCD Waveform register (LCD_WF39TO36)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 44h offset = 4004_3044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF39								WF38								WF37								WF36							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF39TO36 field descriptions

Field	Description
31–24 WF39	Controls segments or phases connected to LCD_P39 as described above for WF3TO0[WF3].
23–16 WF38	Controls segments or phases connected to LCD_P38 as described above for WF3TO0[WF3].
15–8 WF37	Controls segments or phases connected to LCD_P37 as described above for WF3TO0[WF3].
WF36	Controls segments or phases connected to LCD_P36 as described above for WF3TO0[WF3].

50.4.17 LCD Waveform register (LCD_WF43TO40)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 48h offset = 4004_3048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF43								WF42								WF41								WF40							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF43TO40 field descriptions

Field	Description
31–24 WF43	Controls segments or phases connected to LCD_P43 as described above for WF3TO0[WF3].
23–16 WF42	Controls segments or phases connected to LCD_P42 as described above for WF3TO0[WF3].
15–8 WF41	Controls segments or phases connected to LCD_P41 as described above for WF3TO0[WF3].
WF40	Controls segments or phases connected to LCD_P40 as described above for WF3TO0[WF3].

50.4.18 LCD Waveform register (LCD_WF47TO44)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 4Ch offset = 4004_304Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF47								WF46								WF45								WF44							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF47TO44 field descriptions

Field	Description
31–24 WF47	Controls segments or phases connected to LCD_P47 as described above for WF3TO0[WF3].

Table continues on the next page...

LCD_WF47TO44 field descriptions (continued)

Field	Description
23–16 WF46	Controls segments or phases connected to LCD_P46 as described above for WF3TO0[WF3].
15–8 WF45	Controls segments or phases connected to LCD_P45 as described above for WF3TO0[WF3].
WF44	Controls segments or phases connected to LCD_P44 as described above for WF3TO0[WF3].

50.4.19 LCD Waveform register (LCD_WF51TO48)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 50h offset = 4004_3050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF51								WF50								WF49								WF48							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF51TO48 field descriptions

Field	Description
31–24 WF51	Controls segments or phases connected to LCD_P51 as described above for WF3TO0[WF3].
23–16 WF50	Controls segments or phases connected to LCD_P50 as described above for WF3TO0[WF3].
15–8 WF49	Controls segments or phases connected to LCD_P49 as described above for WF3TO0[WF3].
WF48	Controls segments or phases connected to LCD_P48 as described above for WF3TO0[WF3].

50.4.20 LCD Waveform register (LCD_WF55TO52)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 54h offset = 4004_3054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF55								WF54								WF53								WF52							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF55TO52 field descriptions

Field	Description
31–24 WF55	Controls segments or phases connected to LCD_P55 as described above for WF3TO0[WF3].
23–16 WF54	Controls segments or phases connected to LCD_P54 as described above for WF3TO0[WF3].
15–8 WF53	Controls segments or phases connected to LCD_P53 as described above for WF3TO0[WF3].
WF52	Controls segments or phases connected to LCD_P52 as described above for WF3TO0[WF3].

50.4.21 LCD Waveform register (LCD_WF59TO56)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 58h offset = 4004_3058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF59								WF58								WF57								WF56							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF59TO56 field descriptions

Field	Description
31–24 WF59	Controls segments or phases connected to LCD_P59 as described above for WF3TO0[WF3].
23–16 WF58	Controls segments or phases connected to LCD_P58 as described above for WF3TO0[WF3].
15–8 WF57	Controls segments or phases connected to LCD_P57 as described above for WF3TO0[WF3].
WF56	Controls segments or phases connected to LCD_P56 as described above for WF3TO0[WF3].

50.4.22 LCD Waveform register (LCD_WF63TO60)

See the LCD Waveform register (WFC3TO0) for register and field descriptions.

NOTE

The reset value of this register depends on the reset type:

- POR — 0x0000_0000

Address: 4004_3000h base + 5Ch offset = 4004_305Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WF63								WF62								WF61								WF60							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCD_WF63TO60 field descriptions

Field	Description
31–24 WF63	Controls segments or phases connected to LCD_P63 as described above for WF3TO0[WF3].
23–16 WF62	Controls segments or phases connected to LCD_P62 as described above for WF3TO0[WF3].
15–8 WF61	Controls segments or phases connected to LCD_P61 as described above for WF3TO0[WF3].
WF60	Controls segments or phases connected to LCD_P60 as described above for WF3TO0[WF3].

50.5 Functional description

This section provides a complete functional description of SLCD.

Before enabling SLCD by asserting GCR[LCDEN], configure SLCD based on the end application requirements. Out of reset, SLCD is configured with default settings, but these settings are not optimal for every application. SLCD provides several versatile configuration settings and options to support varied implementation requirements, including:

- Frame frequency
- Duty cycle (number of back planes)
- Back plane assignment (which LCD_P[63:0] pins operate as back planes)
- Frame frequency interrupt enable
- Blinking frequency and options
- Power-supply configurations
- Fault detection configuration

SLCD also provides an LCD pin enable control. Setting the LCD pin enable bit in the PENn register for a particular LCD_Pn pin enables SLCD front plane or back plane waveform functionality of that pin after GCR[LCDEN] is set. When GCR[LCDEN] is clear and the LCD pin enable bit is set, the LCD_Pn pin will be driven disabled. When the back plane enable bit in the BPENn is set, the associated pin operates as a back plane. The WFyTOx registers can then activate (display) the corresponding LCD segments on an LCD panel.

The WFyTOx registers control the on/off state for the segments controlled by the LCD pins defined as front planes and the active phase for the back planes. Blank display modes do not use the data from the WFyTOx registers. When using the WFyTOx register for front plane operation, writing a 0 turns the segment off.

For pins enabled as back plane, the bits in WFn fields assign the phases of the back plane (A-H) for the corresponding back plane pin. For a detailed description of SLCD operation for a basic seven-segment LCD display, see LCD seven segment example description.

50.5.1 LCD controller driver description

The LCD controller driver has eight modes of operation:

- 1/1 duty (one back plane) (Phase A), static mode
- 1/2 duty (two back planes) (Phase A, B), 1/3 bias (four voltage levels)
- 1/3 duty (three back planes) (Phase A, B, C), 1/3 bias (four voltage levels)
- 1/4 duty (four back planes) (Phase A, B, C, D), 1/3 bias (four voltage levels)
- 1/5 duty (five back planes) (Phase A, B, C, D, E), 1/3 bias (four voltage levels)
- 1/6 duty (six back planes) (Phase A, B, C, D, E, F), 1/3 bias (four voltage levels)
- 1/7 duty (seven back planes) (Phase A, B, C, D, E, F, G), 1/3 bias (four voltage levels)
- 1/8 duty (eight back planes) (Phase A, B, C, D, E, F, G, H), 1/3 bias (four voltage levels)

All modes are 1/3 bias. These modes of operation are described in more detail in the next sections.

50.5.1.1 LCD duty cycle

The denominator of the duty cycle indicates the number of LCD panel segments capable of being driven by each individual front plane output driver. Depending on the duty cycle, the LCD waveform drive can be categorized as static or multiplexed.

In static-driving method, the LCD is driven with two square waveforms. The static-driving method is the most basic method to drive an LCD panel, but because each front plane driver can drive only one LCD segment, static driving limits the LCD segments that can be driven with a given number of front plane pins. In static mode, only one back plane is required.

In multiplexed mode, the LCD waveforms are multi-level and depend on the bias mode. Multiplex mode, depending on the number of back planes, can drive multiple LCD segments with a single front plane driver. This reduces the number of driver circuits and connections to LCD segments. For multiplex mode operation, at least two back plane drivers are needed. The LCD controller is optimized for multiplex mode.

The numerator of the duty cycle indicates the amount of time the LCD panel segment is energized during each LCD controller frame cycle. The denominator of the duty cycle indicates the number of back planes that are being used to drive an LCD panel and the number of phase sequences, 1 to 8 .

The duty cycle is used by the back plane phase generator to set the phase outputs. The phase outputs A-H are driven according to the sequence shown below. The sequence is repeated at the LCD frame frequency. The duty cycle is configured using GCR[DUTY[2:0]], as shown in Table 50-4.

Table 50-4. LCD controller duty cycle modes

Duty	GCR			Number of back planes	Phase sequence
	DUTY2	DUTY1	DUTY0		
1/1	0	0	0	1	A
1/2	0	0	1	2	A B
1/3	0	1	0	3	A B C
1/4	0	1	1	4	A B C D
1/5	1	0	0	5	A B C D E
1/6	1	0	1	6	A B C D E F
1/7	1	1	0	7	A B C D E F G
1/8	1	1	1	8	A B C D E F G H

50.5.1.2 LCD bias

Because a single front plane driver is configured to drive more and more individual LCD segments, three voltage levels are required to generate the appropriate waveforms to drive the segment. SLCD is designed to operate using the 1/3 bias mode.

50.5.1.3 SLCD base clock and frame frequency

SLCD is optimized to operate using a 32.768 kHz clock input. Two clock sources are available to the SLCD, which are selectable by configuring the GCR[SOURCE]. The two clock sources include:

- Default clock (GCR[SOURCE] = 0)
- Alternate clock (GCR[SOURCE] = 1)

NOTE

The clock source is chip-specific. For the clock source assignments, see the chapter that describes how modules are configured.

Figure 50-2 shows the LCD clock tree. The clock tree shows the two possible clock sources, and the LCD frame frequency and blink frequency clock source. The LCD blink frequency is discussed in Blink frequency.

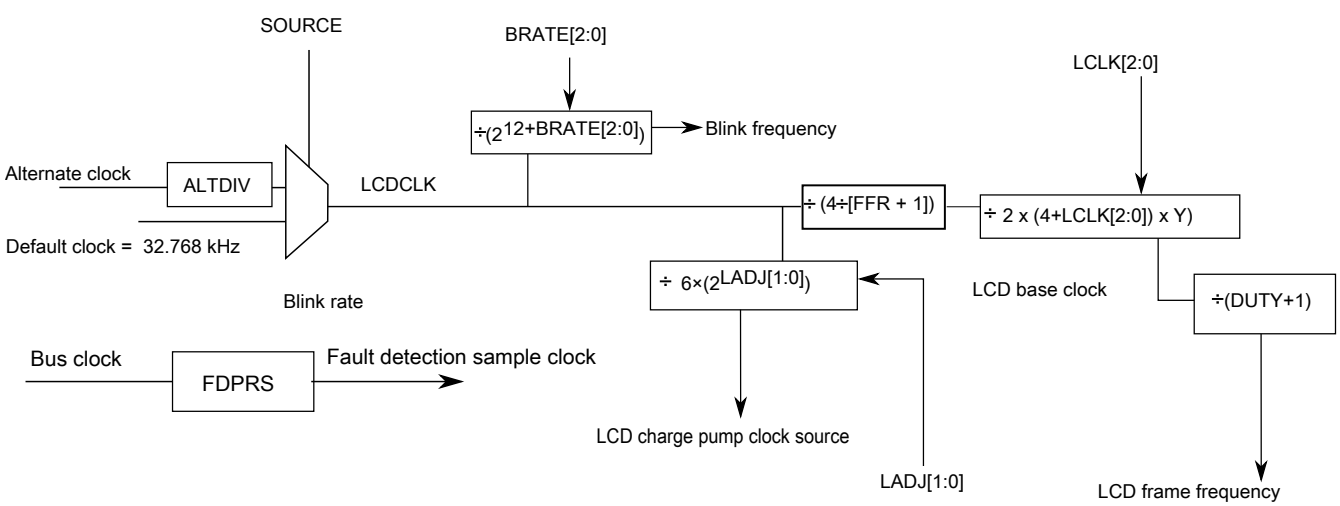


Figure 50-2. LCD clock tree

An external 32.768 kHz clock input is required to achieve lowest power consumption.

The value of GCR[LCLK] is important because it is used to generate the SLCD frame frequency.

The following two tables show SLCD frame frequency calculations that consider several possible SLCD configurations of GCR[LCLK[2:0]], and GCR[DUTY[2:0]].

SLCD frame frequency is defined as the number of times the LCD segments are energized per second. The SLCD frame frequency must be selected to achieve the right balance between flickering (SLCD frame frequency too close to the frequency of an interfering light source or too low), contrast (SLCD frame frequency too low for the display chosen), and power (SLCD frame frequency too high).. Selecting lower values for the LCD base and frame frequency results in lower current consumption for the SLCD.

The SLCD base clock frequency is the SLCD frame frequency multiplied by the number of back plane phases that are being generated. The number of back plane phases is selected using the GCR[DUTY[2:0]] bits. The SLCD base clock is used by the back plane sequencer to generate the LCD waveform data for the enabled phases (A-H).

Table 50-5. SLCD frame frequency calculations for clock input ~ 32.768 kHz

Duty cycle	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8
Y	16	8	5	4	3	3	2	2
GCR[LCLK[2:0]]								
0	64	64	68.3	64	68.3	56.9	73.1	64
1	51.2	51.2	54.6	51.2	54.6	45.5	58.5	51.2
2	42.7	42.7	45.5	42.7	45.5	37.9	48.8	42.7
3	36.6	36.6	39	36.6	39	32.5	41.8	36.6
4	32	32	34.1	32	34.1	28.4	36.6	32
5	28.4	28.4	30.3	28.4	30.3	25.3	32.5	28.4
6	25.6	25.6	27.3	25.6	27.3	22.8	29.3	25.6
7	23.3	23.3	24.8	23.3	24.8	20.7	26.6	23.3

Table 50-6. SLCD frame frequency calculations for clock input ~ 39.063 kHz

Duty cycle	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8
Y	16	8	5	4	3	3	2	2
GCR[LCLK[2:0]]								
0	76.3	76.3	81.4	76.3	81.4	67.8	87.2	76.3
1	61	61	65.1	61	65.1	54.3	69.8	61
2	50.9	50.9	54.3	50.9	54.3	45.2	58.1	50.9
3	43.6	43.6	46.5	43.6	46.5	38.8	49.8	43.6
4	38.1	38.1	40.7	38.1	40.7	33.9	43.6	38.1
5	33.9	33.9	36.2	33.9	36.2	30.1	38.8	33.9

Table continues on the next page...

Table 50-6. SLCD frame frequency calculations for clock input ~ 39.063 kHz (continued)

Duty cycle	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8
Y	16	8	5	4	3	3	2	2
GCR[LCLK[2:0]]								
6	30.5	30.5	32.6	30.5	32.6	27.1	34.9	30.5
7	27.7	27.7	29.6	27.7	29.6	24.7	31.7	27.7

50.5.1.4 LCD waveform examples

This section shows the timing examples of the LCD output waveforms for the several modes of operation. As shown in Table 50-7, all examples use 1/3 bias mode.

Table 50-7. Configurations for example LCD waveforms

Examples	Bias mode	GCR[DUTY[2:0]]	Duty cycle
Example 1	1/3	001	1/2
Example 2		011	1/4
Example 3		111	1/8

50.5.1.4.1 1/2 duty multiplexed with 1/3 bias mode (low-power waveform)

Duty=1/2: DUTY[2:0] = 001

LCD_P[1:0] enabled as back planes:

BPEN0 =1 and BPEN1 =1 in the BPEN0

LCD_P0 assigned to Phase A: WF0 = 0x01

LCD_P1 assigned to Phase B: WF1 = 0x02, WF1TO0 = 0x00000201

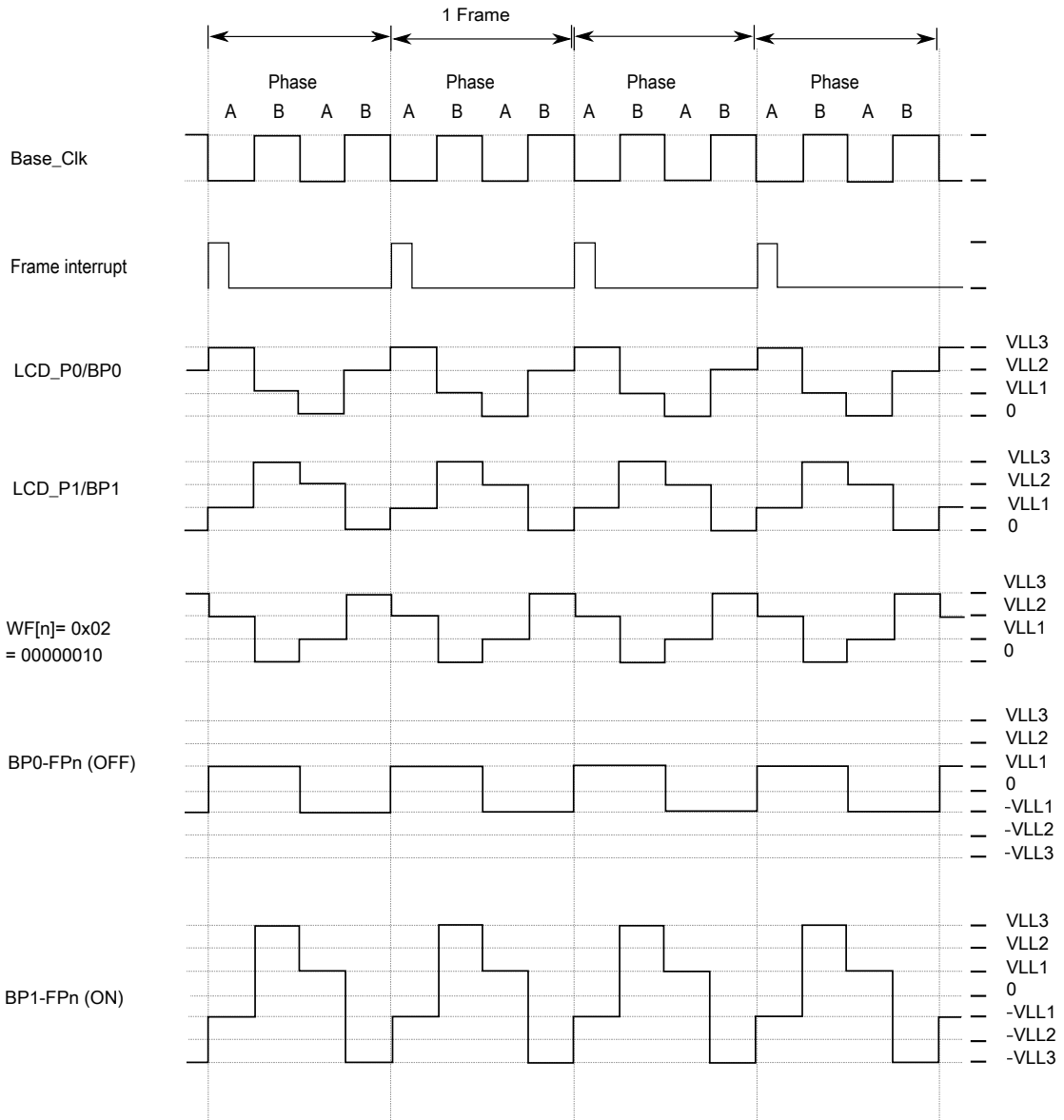


Figure 50-3. 1/2 duty and 1/3 bias (low-power waveform)

50.5.1.4.2 1/4 duty multiplexed with 1/3 bias mode (low-power waveform)

Duty = 1/4: DUTY[2:0] = 011

LCD_P[3:0] enabled as back planes: BPEN0 = 0x0F

LCD_P0 assigned to Phase A: WF0 = 0x01

LCD_P1 assigned to Phase B: WF1 = 0x02

LCD_P2 assigned to Phase C: WF2 = 0x04

LCD_P3 assigned to Phase D: WF3 = 0x08, , WF3TO0 = 0x08040201

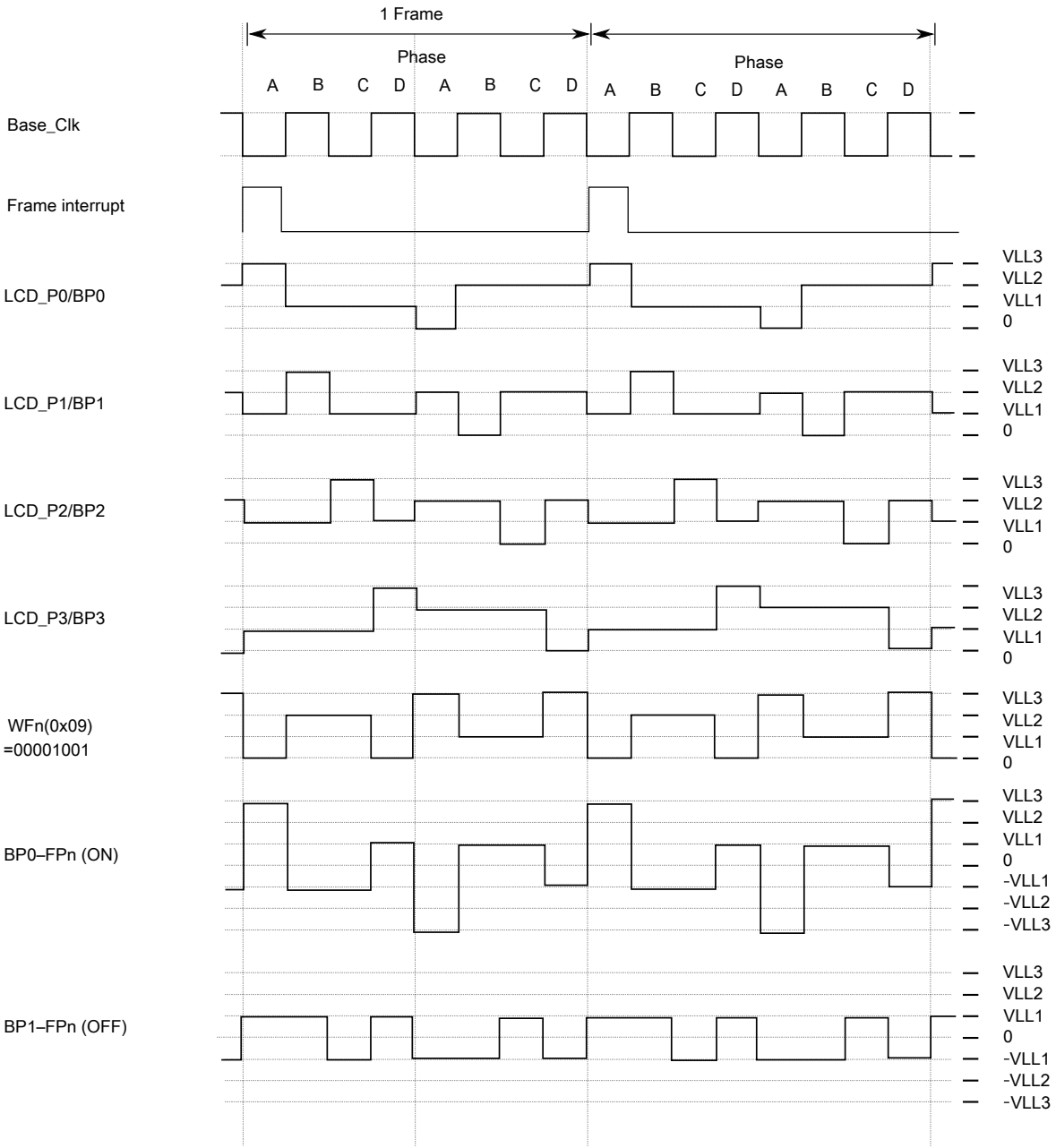


Figure 50-4. 1/4 duty and 1/3 bias (low-power waveform)

50.5.1.4.3 1/8 Duty multiplexed with 1/3 bias mode (low-power waveform)

Duty = 1/8:DUTY[2:0] = 111

LCD_P[7:0] enabled as backplanes: BPEN0 = 0xFF

LCD_P0 assigned to Phase A: WF0 = 0x01

LCD_P1 assigned to Phase B: WF1 = 0x02

LCD_P2 assigned to Phase C: WF2 = 0x04

LCD_P3 assigned to Phase D: WF3 = 0x08

LCD_P4 assigned to Phase E: WF4 = 0x10

LCD_P5 assigned to Phase F: WF5 = 0x20

LCD_P6 assigned to Phase G: WF6 = 0x40

LCD_P7 assigned to Phase H: WF7 = 0x80, WF7TO4 = 0x80402010, WF3TO0 =0x08040201

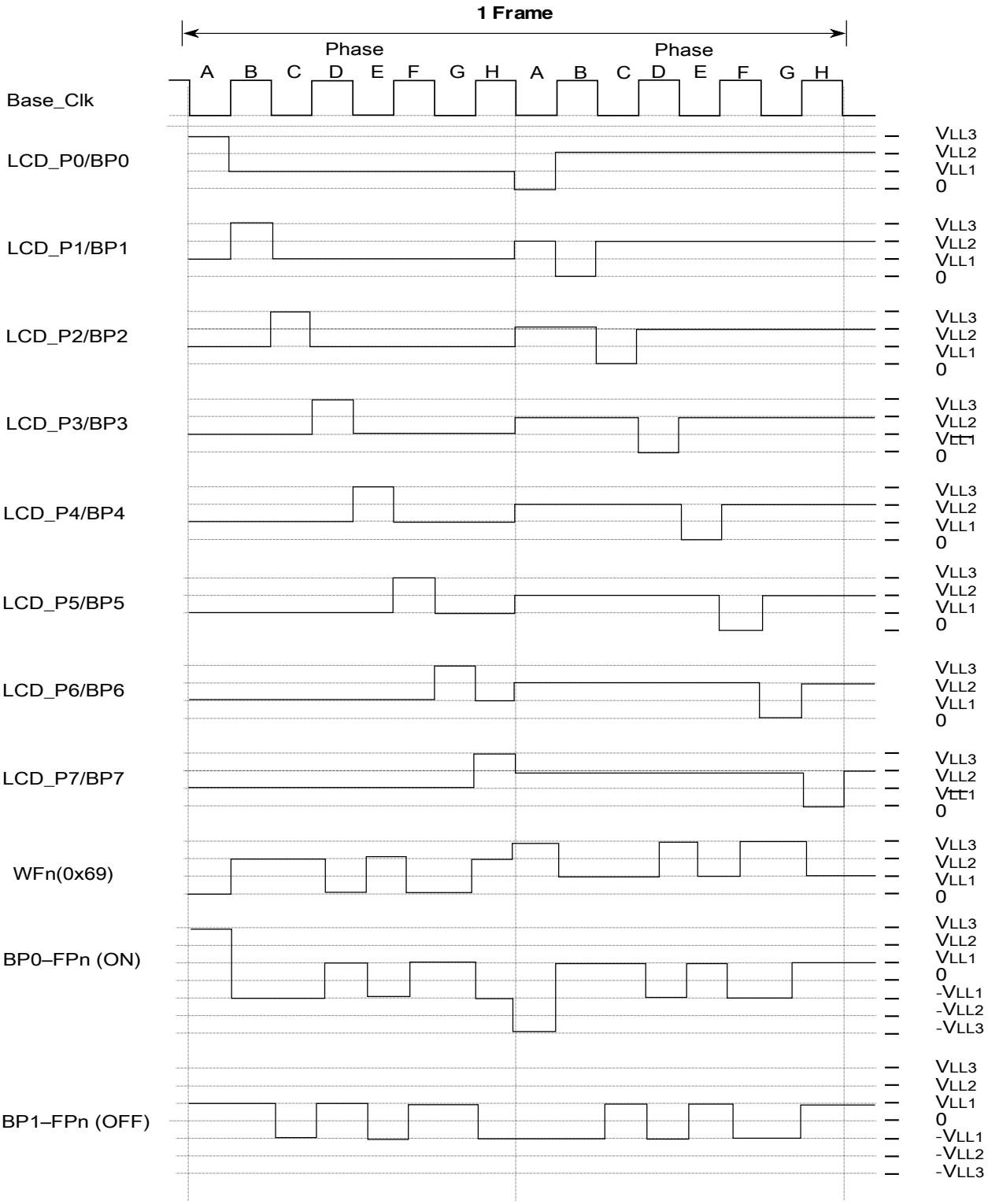


Figure 50-5. 1/8 Duty and 1/3 bias (low-power waveform)

50.5.2 WFyTOx registers

For a segment on the LCD panel to be displayed, data must be written to the WFyTOx registers. For LCD pins enabled as front planes, each bit in the WFyTOx registers corresponds to a segment on an LCD panel. The different phases (A-H) represent the different back planes of the LCD panel. The selected LCD duty cycle controls the number of implemented phases. Refer to Table 50-4 for normal LCD operation the phases follow the sequence shown.

For LCD pins enabled as a back plane, the WFyTOx register assigns the phase in which the back plane pin is active. This is how back plane assignment is done.

An example of normal operation follows.

To enable LCD_P0 to operate as back plane 0:

1. Enable the LCD_P0 by setting PEN0 bit in the PENL register, PENL = 0x00000001.
2. Configure LCD_P0 as a back plane pin by setting the BPEN0 bit in the BPEN0 register.
3. The WF0 bit in the WF3TO0 register is set to associate LCD_P0 with back plane phase A, WF3tO0 =0x00000001.

This configures LCD_P0 to operate as a back plane that is active in phase A.

For LCD pins enabled as a front plane:

- Writing 1 to a given WF location results in the corresponding display segment being driven with the differential root mean square (RMS) voltage necessary to turn the segment on during the phase selected.
- Writing 0 to a given location results in the corresponding display segment being driven with the differential RMS voltage necessary to turn the segment off during the phase selected.

50.5.3 LCD display modes

SLCD can be configured to implement several different display modes. In the LCD auxiliary register (AR), the bits ALT and BLANK configure the different display modes:

Normal display mode (default)

The LCD segments are controlled by the data placed in the WFyTOx registers, as described in WFyTOx registers.

Blank display mode

The WF data is bypassed and the front plane and back plane pins are configured to clear all segments.

Alternate display mode

The backplane sequence is modified for duty cycles of 1/4, 1/3, 1/2, and 1/1.

For four back planes or less, the back plane sequence is modified as shown below. The altered sequence allows two complete displays to be placed in the WFyTOx registers. The first display is placed in phases A-D and the second in phases E-H in the case of four back planes. If the LCD duty cycle is five back planes or greater, the ALT bit is ignored and creates a blank display. Refer to Table 50-9 for additional information.

NOTE

For alternate display modes, both the segment data and the back plane configuration must be configured for both the upper and lower nibble. For example, if LCD_P0 is configured as back plane 0 (phase A) then for proper operation in Alternate Display mode, the WF3TO0 must be set to 0x00000011 to set both BPEWF0 and BPAWF0.

Using the alternate display function, an inverse display can be accomplished for x4 mode and less by placing inverse data in the alternate phases of the WFyTOx registers.

Table 50-8. Alternate display back plane sequence

Duty	Back plane sequence	Alternate back plane sequence
1/1	A	E
1/2	A B	E F
1/3	A B C	E F G
1/4	A B C D	E F G H

50.5.3.1 LCD blink modes

Blink mode is used as a means of alternating among different LCD display modes at a defined frequency. The LCD controller can be configured to implement two blink modes. The AR[BMODE] bit in the LCD auxiliary register configures the different blink modes. Blink modes are activated by setting AR[BLINK]. If AR[BLINK] = 0, SLCD operates normally as described in LCD display modes. If AR[BLINK] = 1, AR[BMODE] configures the blinking operation. During a blink, the display data driven by SLCD changes to the mode selected by AR[BMODE]. The AR[BMODE] bit selects two different blink modes, blank and alternate modes. They operate in the same way, as defined in LCD display modes. Table 50-9 shows the interaction between display modes and blink modes. If the LCD duty cycle is five back planes or greater, AR[BMODE] = 1 is ignored and reverts to create a blank display during the blink period.

Table 50-9. Display mode interaction

BLANK	ALT	BMODE	LCD duty cycle	BLINK = 1	
				Normal period	Blink period
0	0	0	1-4	Normal display	Blank display
0	0	1	1-4	Normal display	Alternate display[1]
0	1	0	1-4	Alternate display[1]	Blank display
0	1	1	1-4	Alternate display[1]	Alternate display[1]
1	x	0	1-4	Blank display	Blank display
1	x	1	1-4	Blank display	Alternate display[1]
0	x	x	8	Normal display	Blank display
1	x	x	8	Blank display	Blank display

1. For alternate display modes, both the segment data and the back plane configuration must be configured for both the upper and lower nibble. For example, if LCD_P0 is configured as back plane 0 (phase A) then for proper operation in Alternate Display mode, the WF3TO0 should be set to 0x00000011 to set both BPEWF0 and BPAWF0.

50.5.3.2 Blink frequency

The LCD clock is the basis for the calculation of the LCD controller blink frequency. The LCD controller blink frequency is equal to the LCD clock (GCR[LCLK]) divided by the factor selected by the AR[BRATE[2:0]] bits. The following table shows LCD controller blink frequency calculations for all values of AR[BRATE[2:0]] at a few common GCR[LCLK] selections.

Table 50-10. Blink frequency calculations (Blink rate = LCD clock (Hz) ÷ Blink divider)

AR[BRATE[2:0]]	0	1	2	3	4	5	6	7
LCD clock (GCR[LCLK])	Blink frequency (Hz)							
30 kHz	7.32	3.66	1.831	0.916	0.46	0.23	0.11	0.06
32.768 kHz	8	4	2	1	0.5	0.25	0.13	0.06
39.063 kHz	9.54	4.77	2.38	1.19	0.6	0.30	0.15	0.075

50.5.4 LCD charge pump and power supply operation

This section describes the LCD charge pump and LCD power supply configuration options.

The LCD bias voltages (V_{LL1}, V_{LL2} and V_{LL3}) can be generated by the LCD charge pump or a resistor divider network that is connected using the GCR[CPSEL] bit. The input source to the LCD charge pump is controlled by GCR[VSUPPLY].

GCR[VSUPPLY] indicates the state of internal signals used to configure power switches as shown in the table in the following figure. The block diagram in the following figure illustrates several potential operational modes for SLCD including configuration of the SLCD power supply source using V_{DD}, internal regulated voltage V_{IREG}, or an external supply on V_{LL3} . V_{LL3} must never be externally driven to any level other than V_{DD}.

Upon reset GCR[VSUPPLY] is configured to connect V_{LL3} to V_{DD}. This configuration must be changed to match the application requirements before SLCD is enabled.

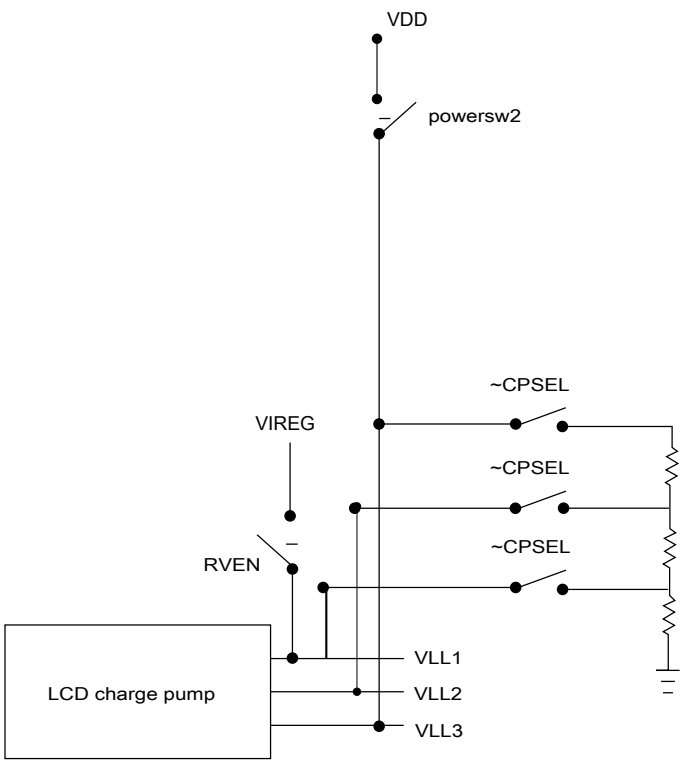


Figure 50-6. LCD charge pump block diagram

Table 50-11. LCD power supply configuration options

GCR[VSUPPLY]	Configuration	—	powersw2
0	Drive V_{LL3} internally from V_{DD}	—	1
1	Drive V_{LL3} externally from V_{DD} or Drive V_{LL1} internally from V_{IREG}	—	0

Note

The charge pump is optimized for 1/3 bias mode operation only.

The charge pump requires external capacitance for its operation. Place 0.1 µF ceramic capacitors across V_{cap1} and V_{cap2}, and from each V_{LLx} pin to ground.

50.5.4.1 LCD power supply configuration

The LCD bias voltages can be internally derived from:

- V_{DD}
- A voltage source connected to VLL3. This voltage source must not be externally driven to any level other than VDD.
- Internally derived from a regulated voltage source that can be configured to supply 1.0 V (V_{IREG})

Table 50-12 provides a more detailed description of the power state of SLCD that depends on the configuration of the GCR[VSUPPLY], GCR[CPSEL] and GCR[RVEN] bits. The table shows all possible configurations of the LCD power supply. All other combinations of the configuration bits above are not permissible LCD power supply modes and must be avoided.

Table 50-12. LCD power supply options

LCD operational state	LCD power supply configuration	VSUPPLY	CPSEL	RVEN
V_{LL3} connected to V_{DD} internally	For 3 V glass operation, V_{DD} must be in valid range for contrast desired for display. Charge pump is used to generate V_{LL1} and V_{LL2}.	0	1	0
V_{LL3} is driven externally	For 3 V glass operation, V_{LL3} must be in valid range for contrast desired on the display. Charge pump is used to generate V_{LL1} and V_{LL2}.	1	1	0
V_{LL3} is driven externally Resistor bias network enabled.	For 3 V glass operation V_{LL3} must be in valid range for contrast desired on the display. Charge pump is disabled. Resistor bias network is used to create V_{LL1} and V_{LL2}.	1	0	0

Table continues on the next page...

Table 50-12. LCD power supply options (continued)

LCD operational state	LCD power supply configuration	VSUPPLY	CPSEL	RVEN
V_{IREG} is connected to V_{LL1}	For 3 V glass operation,, V_{IREG} = 1 V. V_{IREG} may be adjusted to achieve desired contrast V_{IREG} is connected to V_{LL1} internally. Charge pump is used to generate V_{LL2} and V_{LL3}. .	1	1	1

50.5.4.1.1 LCD external power supply, VSUPPLY = 1

When VSUPPLY = 1, powersw2 is deasserted. V_{DD} is not available to power SLCD internally, so an alternate (internal RBIAS or external) power source is required for V_{LL1}, V_{LL2}, and V_{LL3} when the charge pump is disabled.

If the charge pump is enabled, external power must be applied to V_{LL3}, V_{LL2}, or V_{LL1}; or internal power applied to V_{LL1} through the V_{IREG}. If V_{LL3} is applied externally, it MUST be equal to V_{DD}. With this configuration, the charge pump generates the other LCD bias voltages not provided internally or externally.

Internal V_{IREG}

If the charge pump is enabled, the internal regulated voltage, V_{IREG}, can be used as an input to generate the LCD bias voltages. In this state, the external voltage source must not be connected to V_{LL1}, V_{LL2}, or V_{LL3}. V_{IREG} is controlled by the LCD general control register (GCR). LCD Charge Pump shows that V_{IREG} is connected to V_{LL1} when the RVEN bit is set.

V_{LL1} is connected to the internal charge pump. Using the charge pump, the value of V_{LL1} is tripled and output as V_{LL3}. V_{IREG} should be adjusted so that V_{LL3} is in an acceptable range for the desired contrast on the glass.

GCR contains trim bits that can be used to make changes to the regulated voltage so that contrast can be adjusted as desired. The trim can be used to increase or decrease the regulated voltage by 1.5% for each count. A total of ±12% of change can be done to the regulated voltage.

50.5.4.1.2 LCD internal power supply, VSUPPLY = 0

V_{DD} is used as the SLCD power supply when VSUPPLY = 0. When powering SLCD using V_{DD}, the charge pump must be enabled (GCR[CPSEL] = 1).

When VSUPPLY = 0, V_{DD}=V_{LL3} and V_{DD} should be kept in a range that supports the desired contrast on the glass.

50.5.5 Resets

During a reset except V_{LLS3}/V_{LLS1} recovery reset, SLCD is configured in Default mode. Default mode includes the following settings:

- GCR[LCDEN] is cleared, thereby forcing all front plane and back plane driver outputs to the high impedance state.
- 1/4 duty
- 1/3 bias
- LCLK[2:0], VSUPPLY , CPSEL, RVEN, and BRATE[2:0] revert to their reset values; V_{LL3} is internally connected to V_{DD}

50.5.6 Interrupts

When an SLCD frame-frequency interrupt event occurs, the AR[LCDIF] bit is asserted. It remains asserted until software clears the SLCD frame frequency interrupt. The interrupt can be cleared by software writing a 1 to the LCDIF bit. The SLCD frame frequency interrupt is not available in all low-power modes.

If both the AR[LCDIF] bit and the GCR[LCDIEN] bit are set, an LCD interrupt signal asserts.

When a fault detection complete event occurs, FDSR[FDCF] bit is set. The FDSR[FDCF] bit remains asserted until software writes 1 to clear. If both FDSR[FDCF] and GCR[FDCIEN] bits are set, an LCD interrupt signal is asserted.

50.5.7 LCD display fault detect circuit (LFD)

Most failures, where an LCD segment is either unexpectedly on or off, result in erroneous information that can mislead a user and cause a dangerous situation. The LCD display fault detect circuit's (LFD) function finds faults in the LCD display, display connector, and board connections between the MCU and the display.

The LCD panel can be regarded as a matrix of segments, as shown in Figure 50-7, Figure 50-8, and Figure 50-9. Any open/short connection changes the capacitive characteristics of the segment matrix. The pullup fault detection checks the capacitive characteristic of an LCD_Pn pin by applying a weak pullup voltage to the display capacitor matrix. The rise time response is sampled and summed within a user-defined time frame and stored for postprocessing in FDSR[FDCNT]. The summing of the digitized values uses the principle that the response of a capacitor for each charge/discharge is constant for the same loading conditions.

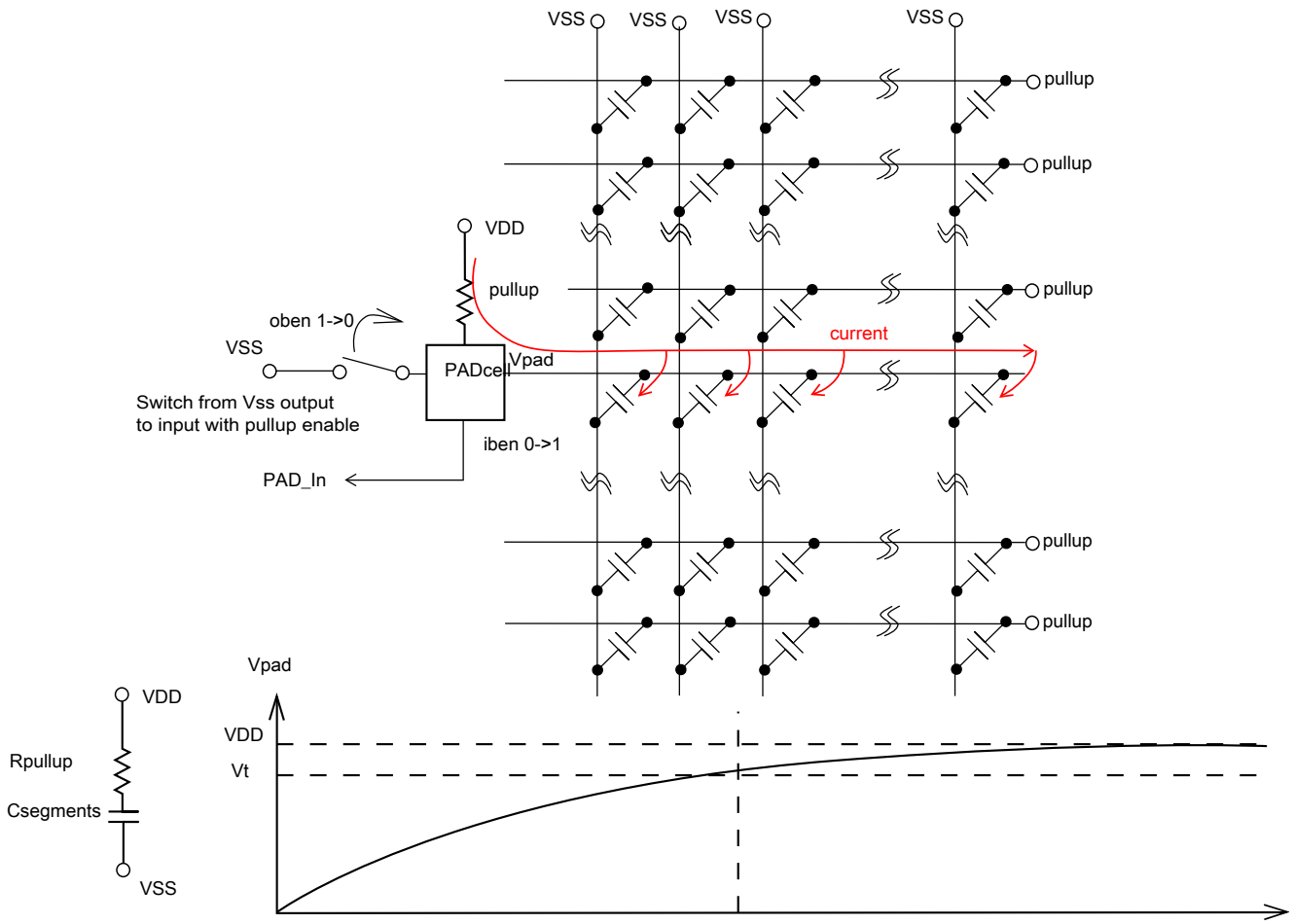
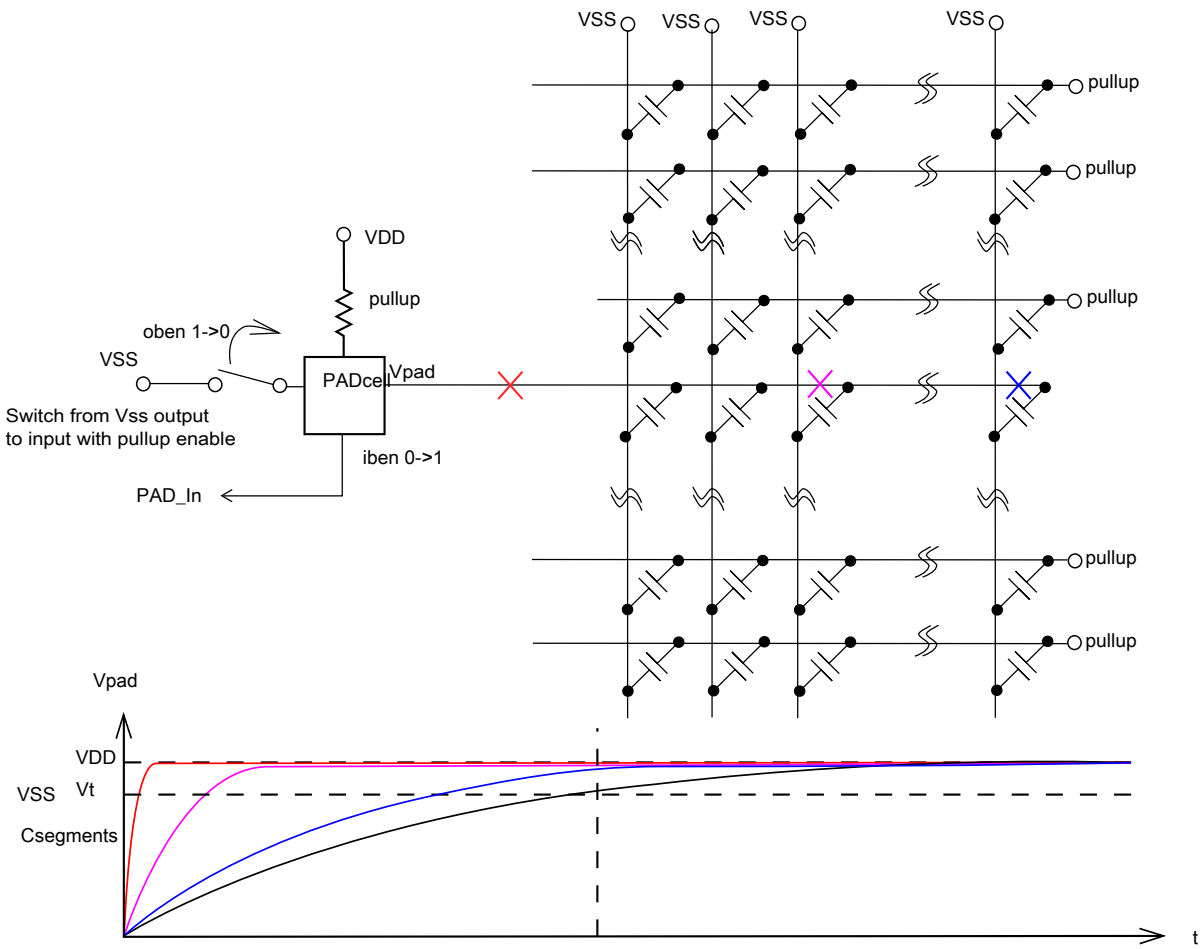
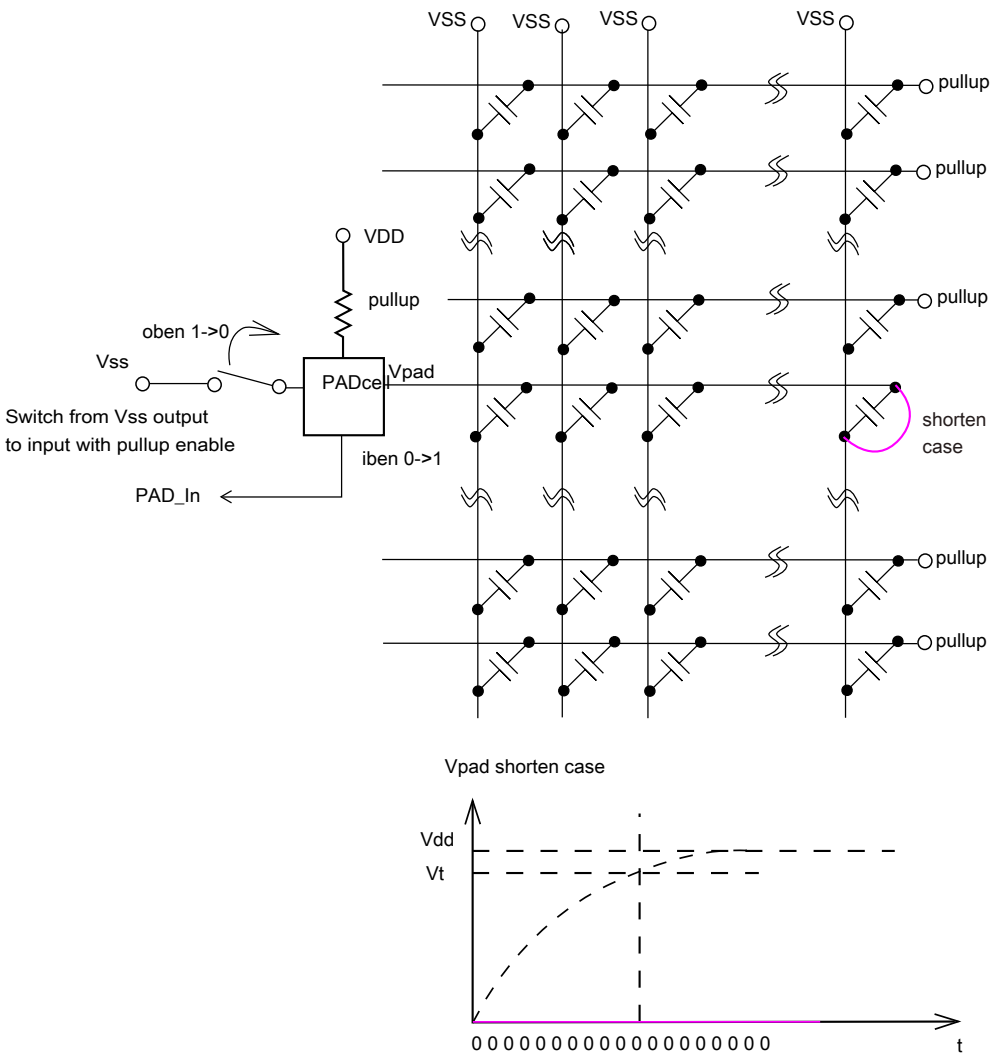


Figure 50-7. Pullup fault detect in connection/segment normal case



If there's any open case in connection or panel, the Csegments will become smaller, which makes the pullup time shorter.

Figure 50-8. Pullup fault detect in connection/segment open case



If there's any short case between BP and FP, Vpad will be shortened to VSS and never be pulled up.

Figure 50-9. Pullup fault detect in connection/segment short case

This circuit works by measuring the capacitance of the LCD front plane or back plane connection and comparing it to a reference number. This reference number is usually measured at the user's production facility (by this circuit) and stored in flash memory for later comparison. The comparison must account for fluctuations due to temperature, voltage, and other environmental effects. Intelligent software may be able to account for these effects more accurately by using time-dependent algorithms; but the basic comparison to a factory-measured number should be sufficient for most applications.

This circuit can measure the following:

- Front plane or back plane rise time response changes, including pad, pin, board, and all pixels connected to the measured front plane or back plane.

- Short-circuits in the front plane or back plane connection (all silicon or board locations), see Figure 50-9.
- Excessive leakage in the front plane or back plane connection (silicon or board).
- Supply connectivity through the driver for the V_{SS} level.
- Opens in the front plane or back plane connection (bondwire, package, silicon, or board), see Figure 50-9.

The purpose of this circuit is not to:

- Measure independent pixel capacitances.
- Measure voltage levels or contrast.
- Measure the open connections between the front plane/back plane and the V_{LL1}, V_{LL2}, and V_{LL3} drivers.

If there is any open case in connection or panel, $C_{segments}$ becomes smaller, which changes the rise time response.

Pullup fault detection can be performed while LCDEN is asserted.

1. Set the target pin number by writing FDCR[FDPINID[5:0]].
2. Set FDCR[FDBPEN] if the target pin is working as back plane, or set FDCR[FDBPEN] as 0 if the target pin is working as front plane.
3. Select the sample clock frequency by setting FDCR[FDPRS[2:0]].
4. Set the sample window by writing to FDCR[FDSWW[2:0]].
5. Enable detection by writing 1 to FDCR[FDEN].
6. Wait until the FDSR[FDCF] bit is asserted (by polling or by interrupt).
7. Read FDSR[FDCNT].
8. Write 1 to clear FSCR[FDCF].

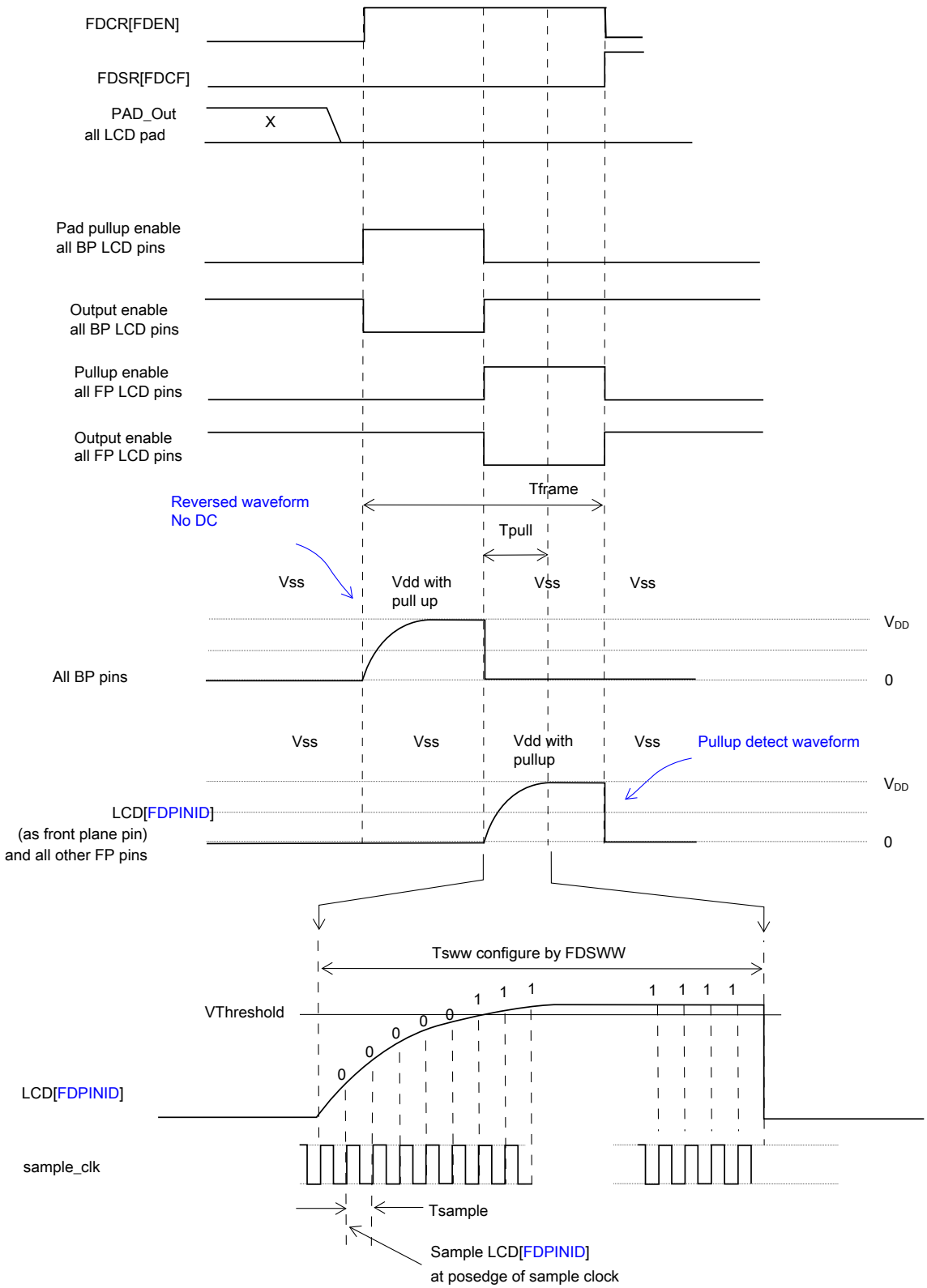


Figure 50-10. Pullup fault detection timing diagram, while FDBPEN is 0

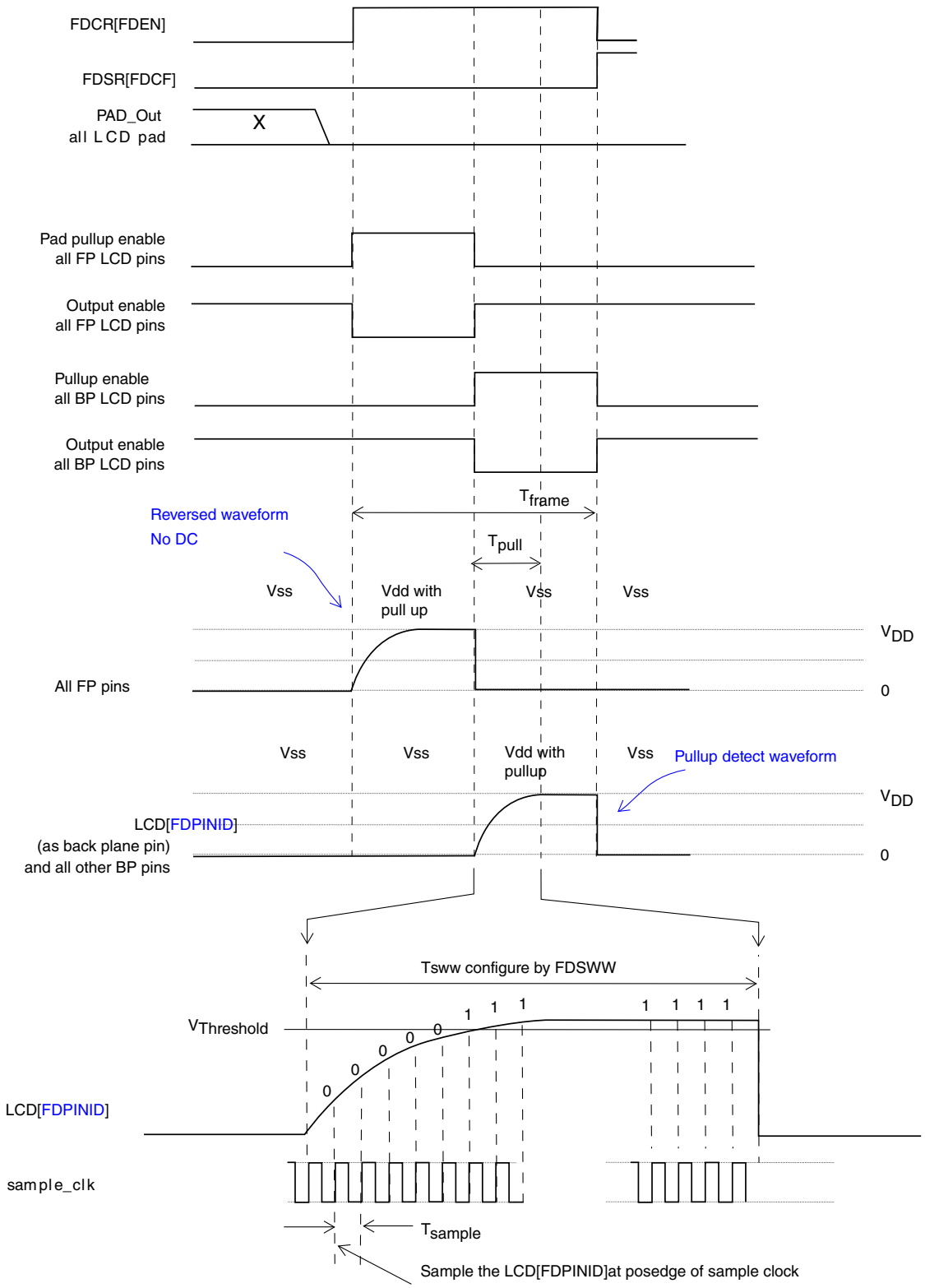


Figure 50-11. Pullup fault detection timing diagram, while FDBPEN is 1

Note

The voltage waveforms applied to the LCD electrodes must have no DC component. Therefore, a reversed pullup voltage waveform must be applied before the pullup detect waveform.

Fault detection occupies one frame and the pullup time is half-frame duration. You must ensure that the fault detection window set by the sample window width is less than half-frame duration.

50.6 Initialization section

This section provides a recommended initialization sequence for the LCD controller and also includes initialization examples for several LCD application scenarios.

50.6.1 Initialization sequence

The list below provides a recommended initialization sequence for the LCD controller.

You must write to all PEN, BPEN, and WFyTOx registers to initialize their values after a reset.

1. GCR
    a. Configure LCD clock source (SOURCE bit).
    b. Enable regulated voltage (RVEN).
    c. Trim the regulated voltage (RVTRIM).
    d. Enable charge pump (CPSEL bit).
    e. Configure charge pump clock (LADJ[1:0]).
    f. Configure LCD power supply (VSUPPLY).
    g. Configure LCD frame frequency interrupt (LCDIEN bit).
    h. Configure LCD behavior in low-power mode (LCDDOZE and LCDSTP bits).
    i. Configure LCD duty cycle (DUTY[2:0]).
    j. Select and configure LCD frame frequency (LCLK[2:0]).
2. AR

a. Configure display mode (ALT and BLANK bits).

b. Configure blink mode (BMODE).

c. Configure blink frequency (BRATE[2:0]).

3. PENn

   a. Enable LCD controller pins (PEN[63:0] bits).

4. BPENn

   a. Enable LCD pins to operate as an LCD back plane (BPEN[63:0]).

5. WFyTOx

   a. Initialize WFyTOx registers with back plane configuration and an initial display screen.

6. GCR

   a. Enable LCD controller (LCDEN bit).

7. LCD pins assigned to VCAP1 and VCAP2 may only be enabled if the pins are in charge pump mode (CPSEL=1).
   - When CPSEL=0 and LADJ = 00 or 01, VLL1/2 pins cannot be used as LCD_P pins.
   - When CPSEL=0 and LADJ = 10 or 11, VLL1/2 pins can be configured as LCD_P pins or GPIO pins and cannot be configured as bias voltage.
   - When CPSEL=1, VLL1/2 cannot be used as LCD_P pins.

50.6.2 Initialization examples

This section provides initialization information for LCD configuration. Each example details the register and field values required to achieve the appropriate LCD configuration for a given LCD application scenario. Table 50-13 lists each example and the setup requirements.

Table 50-13. LCD application scenario

Example	Operating voltage, V_{DD}	LCD clock source	LCD glass operating voltage	Required LCD segments	LCD frame rate	Blinking mode/ rate	Behavior in Wait/Stop modes	LCD power input
1	2.7 V	External 32.768 kHz	3 V	128	30 Hz	None	Wait: on Stop: on	Power via external resistor bias network

These examples illustrate the flexibility of SLCD to be configured to meet a wide range of application requirements, including:

- Clock inputs/sources
- LCD power supply
- LCD glass operating voltage
- LCD segment count
- Varied blink modes/frequencies
- LCD frame rate

50.6.2.1 Initialization example 1

Table 50-14. LCD setup requirements for example 1

Example	Operating voltage, V_{DD}	LCD clock source	LCD glass operating voltage	Required LCD segments	LCD frame rate	Blinking mode/rate	Behavior in Stop and Wait modes	LCD power input
1	2.7 V	External 32.768 kHz	3 V	128	30 Hz	None	Wait: on Stop: on	Power via External resistor bias network

Table 50-15 lists the setup values required to initialize the LCD as specified by example 1:

Table 50-15. Initialization register values for example 1

Register	Field	Binary value	Comment
GCR	CPSEL	0	Disable charge pump.
	LADJ[1:0]	XX	Adjust the resistor bias network for different LCD glass capacitance. or Configure LCD charge pump clock source.
	VSUPPLY	1	When VSUPPLY = 1 and CPSEL = 0, the LCD must be externally powered via V_{LL3} (see Table 50-12).
	LCDIEN	0	LCD frame interrupts disabled.
	LCDDOZE	1	LCD is "on" in Wait mode.
	LCDSTP	1	LCD is "on" in Stop mode.
	LCDEN	0	Initialization is done before initializing the LCD controller.

Table continues on the next page...

Table 50-15. Initialization register values for example 1 (continued)

Register	Field	Binary value	Comment
AR	SOURCE	0	Selects the external clock reference as the LCD clock input.
	LCLK[2:0]	101	For 1/8 duty cycle, select closest value to the desired 30 Hz LCD frame frequency (see Table 50-5).
	DUTY[2:0]	111	For 128 segments (8x16), select 1/8 duty cycle.
	RVEN	0	V_{IREG} is not used for this configuration.
	RVTRIM[3:0]	XXXX	Trim value may be adjusted to achieve desired contrast. For a given voltage, trim value is determined by characterization.
	BLINK	0	No blinking.
	ALT	X	Alternate bit is configured during LCD operation.
	BLANK	X	Blank bit is configured during LCD operation.
	BMODE	X	N/A; Blink Blank = 0; Blink Alternate = 1
	BRATE[2:0]	XXX	N/A
PEN[3:0]	PEN0	11111111	Only 29 LCD pins need to be enabled. **Note:** Any of the 63 LCD pins can be used, this allows flexibility in the hardware design.
	PEN1	11111111	
	PEN2	11111111	
	PEN3	00000000	
BPEN[3:0]	BPEN0	11111111	Eight back plane pins needed. **Note:** Any LCD pin can be enabled to operate as a back plane.
	BPEN1	00000000	
	BPEN2	00000000	
	BPEN3	00000000	
WFyTOx	WF0	00000001	Configure which phase the eight back plane pins are active in. This configuration sets LCD_P0 to be active in Phase A, LCD_P1 to be active in Phase B, and so on. This configuration sets LCD pins 0–7 to represent back plane 1–8. **Note:** Any back plane pin can be active in any phase.
	WF1	00000010	
	WF2	00000100	
	WF3	00001000	
	WF4	00010000	
	WF5	00100000	
	WF6	01000000	
	WF7	10000000	

50.7 Application information

Figure 50-12 is a programmer's model of SLCD. The programmer's model groups the SLCD register fields into functional groups. The model is a high-level illustration of SLCD showing the module's functional hierarchy including initialization and runtime control.

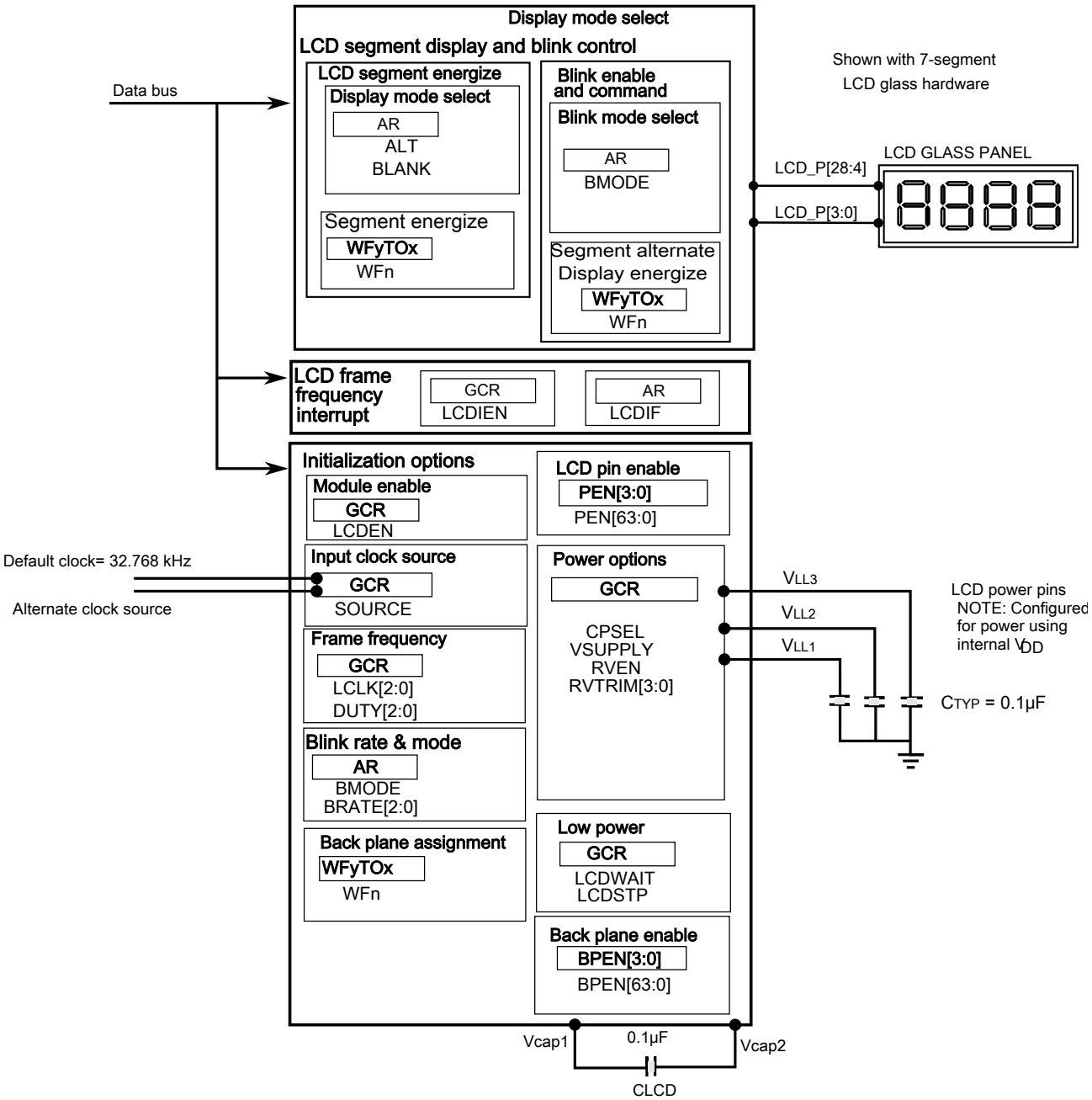
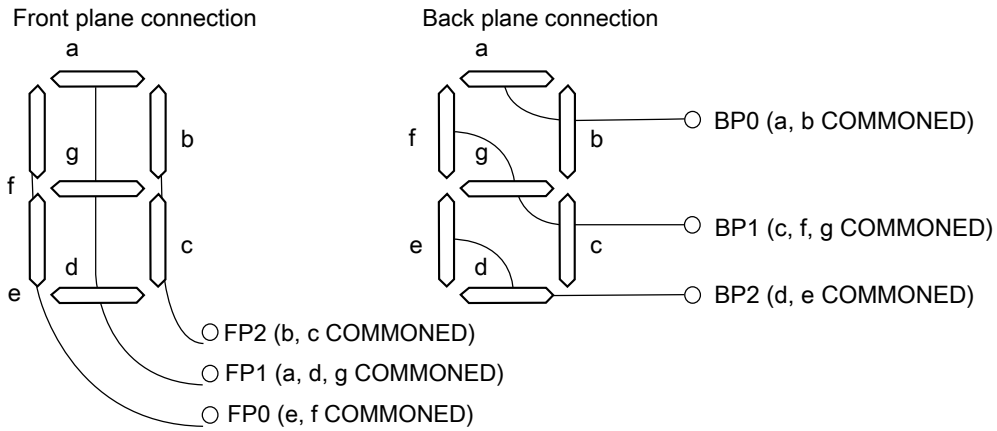


Figure 50-12. LCD programmer's model diagram

50.7.1 LCD seven segment example description

A description of the connection between SLCD and a seven segment LCD character is illustrated in Figure 50-13 to provide a basic example for a 1/3 duty cycle LCD implementation. The example uses three back plane pins (LCD_P3, LCD_P4, and

LCD_P5) and three front plane pins (LCD_P0, LCD_P1, and LCD_P2). The contents of the WFyTOx registers and output waveforms are also shown. Output waveforms are illustrated in the following two figures.



The above segment assignments are provided by the specification for the LCD glass for this example.
For this LCD controller any of the LCD pins can be configured to be front plane 0-2 or back plane 0-2.
For this example, set LCD_P0 as FP0, LCD_P1 as FP1, and LCD_P2 as FP2.
For this example, set LCD_P3 as BP0, LCD_P4 as BP1 and LCD_P5 as BP2.

Back plane assignment is done in the LCD waveform registers as shown below:

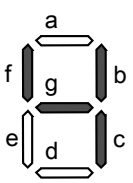
WF3	0	0	0	0	0	0	0	1
WF4	0	0	0	0	0	0	1	0
WF5	0	0	0	0	0	1	0	0

With the above conditions the segment assignment is shown below:

WF0	–	–	–	–	–	e	f	–
WF1	–	–	–	–	–	d	g	a
WF2	–	–	–	–	–	–	c	b

To display the character "4": WF0 = XXXXX01X, WF1 = XXXXX010, WF2 =XXXXXX11

WF0	X	X	X	X	X	0	1	X
WF1	X	X	X	X	X	0	1	0
WF2	X	X	X	X	X	X	1	1



X = don't care

Figure 50-13. Waveform output from WFyTOx registers

50.7.1.1 LCD controller waveforms

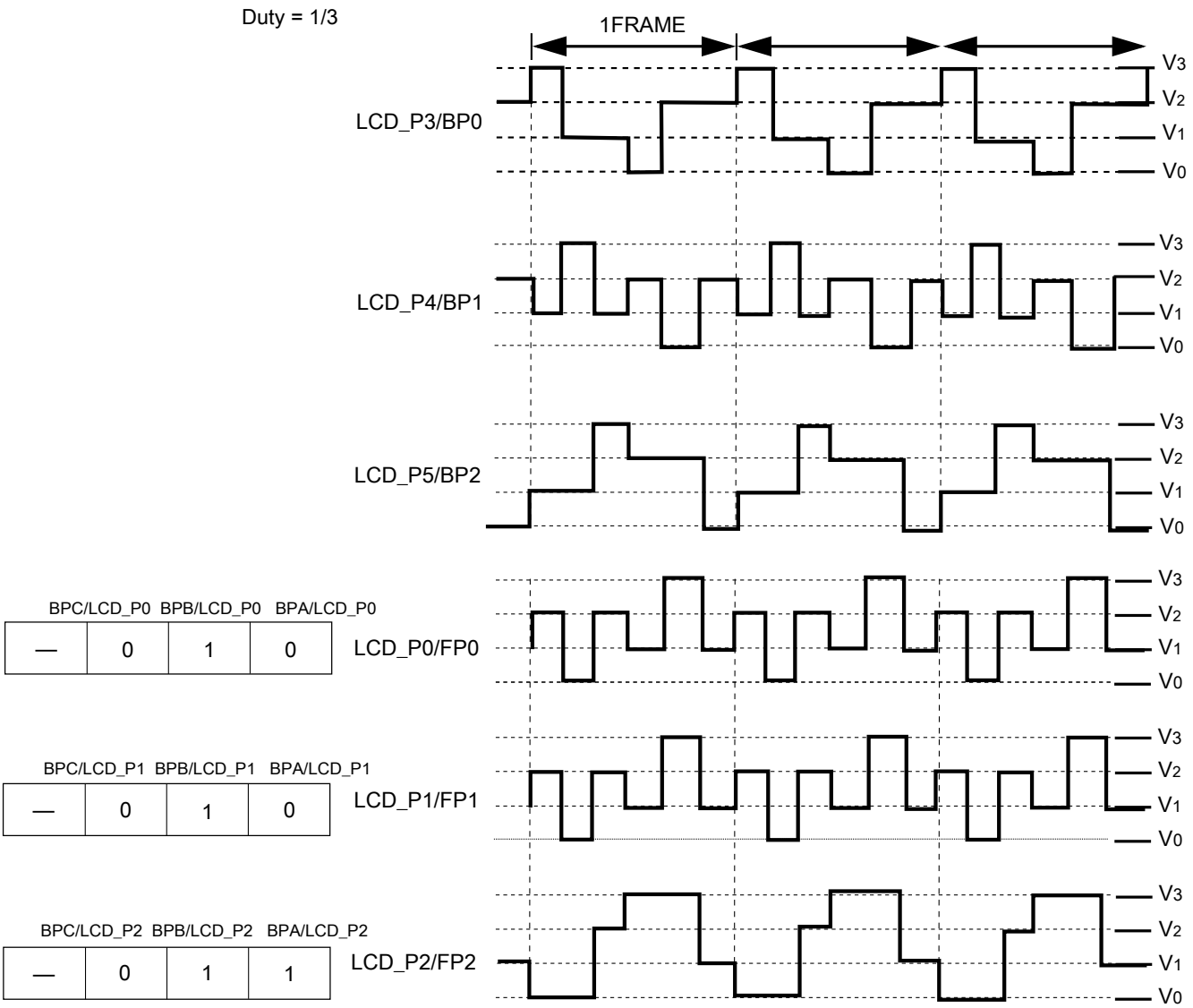


Figure 50-14. LCD waveforms

50.7.1.2 Segment on driving waveform

The voltage waveform across the "f" segment of the LCD (between LCD_P4/BP1 and LCD_P0/FP0) is illustrated in Figure 50-15. As shown in the waveform, the voltage level reaches the value V_3 — therefore, the segment will be on.

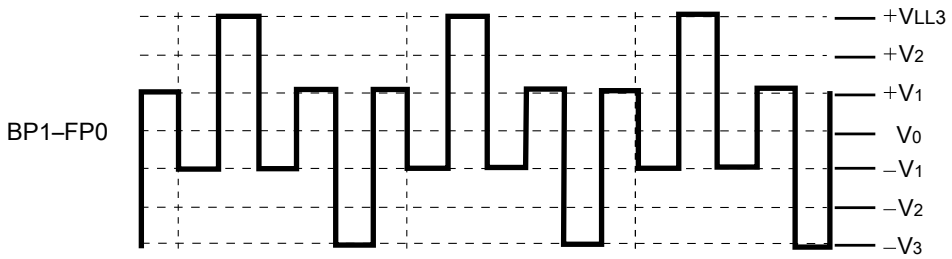


Figure 50-15. "f" segment voltage waveform

50.7.1.3 Segment off driving waveform

The voltage waveform across the "e" segment of the LCD (between LCD_P5/BP2 and LCD_P0/FP0) is illustrated in Figure 50-16. As shown in the waveform, the voltage does not reach the voltage V_3 threshold — therefore the segment will be off.

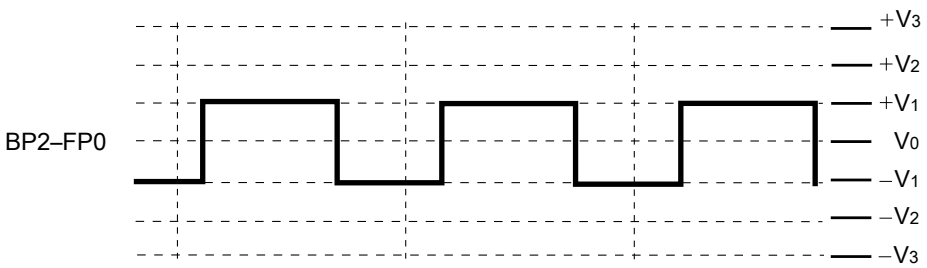


Figure 50-16. "e" segment voltage waveform

50.7.2 LCD contrast control

Contrast control for SLCD is achieved when the LCD power supply is adjusted to a value greater than or less than the LCD threshold voltage. The LCD threshold voltage is the nominal voltage required to energize the LCD segments. For 3 V LCD glass, the LCD threshold voltage is usually about 3 V. Increasing the value of the LCD voltage makes the energized segments on the LCD glass become more opaque. Decreasing the value of the LCD voltage makes the energized segments on the LCD glass become more transparent. The LCD power supply can be adjusted to facilitate contrast control by using the internal regulator V_{IREG} and adjusting its trim (RVTRIM), or for more extreme contrast control, by using external components like a variable resistor.

NOTE:
Contrast control configuration when LCD is powered using internal VDD

VDD is specified between 2.7 and 5.5V .

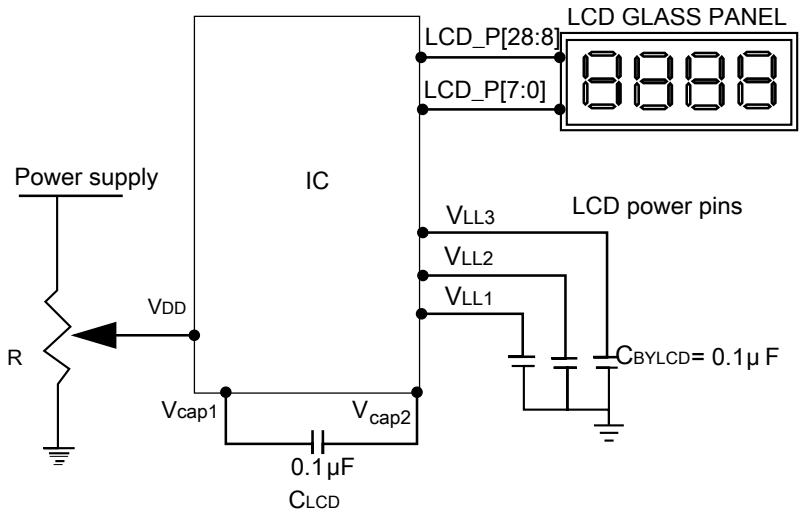


Figure 50-17. Power connections for contrast control

Chapter 51
General-Purpose Input/Output (GPIO)

51.1 Chip-specific GPIO information

51.1.1 Overview

The Enhanced General Purpose Input / Output controller (eGPIO) supports common pin input and output functions. This module combines keyboard interrupt functionality for rising/falling edges or level sensitivity.

The eGPIO module is a dual port eGPIO module. eGPIO shall have one port connected to the M0+ core directly (via IOPORT) to allow direct accesses the GPIO (single cycle access). The other port will be connected to AIPS to allow normal IPS accesses to the module.

Some features of the module:

- The module can support any number of GPIO pins via a number of control and data registers. All these registers are also accessible at the byte and halfword level, so the configuration and data values associated with smaller sets of GPIO pins can be independently controlled.
- The module has access control registers to control access to GPIOs. One access control register will include four 4-bit GPIO Access Control Registers (GACRn) modeled after the same functionality present in the AIPS's PACRn registers. This includes a lock bit and a 3-bit attribute. The attribute controls the access permission for privileged secure, user secure and user non-secure mode. The lock bit can be set to "lock" the configuration and prevent any subsequent writes until the next reset.
- There will be one access control register per 8-pins of GPIO. For 8-bit or 16-bit or 32-bit accesses, the appropriate number of GACRn registers are selected, their flags logically summed together (OR'ed) and the access evaluated.

- After the "effective" protect bits are formed, the access is checked. If the access is permitted, the operation completes. If the access is denied, the response is dependent on the source of the reference:
  - If the access is from the IPS port, the transaction is aborted and a bus error termination generated - the read data is zeroed and the write is ignored.
  - If the access is from the core's IOPORT, the required response is different - recall this bus does not support any type of wait state insertion nor any type of error termination. For IOPORT accesses that are not allowed, the reference is treated as a RAZ/WI, the ARM notation for Read As Zero, Write Ignored.
- The GACRn registers will be accessible in Supervisor mode only. Protection to GACRn registers provided via lock bit (in module itself) and via PACR register of AIPS.

51.1.2 Instantiation Information

The maximum total number of GPIO pins for general purpose is 99 (number varies from device to device). GPIO pins are configured as 8-pins per port. The peripheral bus interface for the eGPIO operates at the bus clock.

Only one eGPIO ports have the digital filter feature for reduced noise and hardware switch de-bouncing: Port E. The other ports do not have this feature .

51.2 Introduction

The general-purpose input and output (GPIO) module is accessible via the peripheral bus. The GPIO registers support 8-bit, 16-bit or 32-bit accesses.

The GPIO data direction and output data registers control the direction and output data of each pin when the pin is configured for the GPIO function. The GPIO input data register displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt module for that pin is enabled.

Efficient bit manipulation of the general-purpose outputs is supported through the addition of set, clear, and toggle write-only registers for each port output data register.

51.2.1 Features

Features of the GPIO module include:

- Port Data Input register visible in all digital pin-multiplexing modes

- Port Data Output register with corresponding set/clear/toggle registers
- Port Data Direction register

NOTE

The GPIO module is clocked by system clock.

51.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 51-1. Modes of operation

Modes of operation	Description
Run	The GPIO module operates normally.
Stop	The GPIO module is disabled.
Debug	The GPIO module operates normally.

51.2.3 GPIO signal descriptions

Table 51-2. GPIO signal descriptions

GPIO signal descriptions	Description	I/O
PORTA7–PORTA0	General-purpose input/output	I/O
PORTB7–PORTB0	General-purpose input/output	I/O
PORTC7–PORTC0	General-purpose input/output	I/O
PORTD7–PORTD0	General-purpose input/output	I/O
PORTE7–PORTE0	General-purpose input/output	I/O
PORTF7–PORTF0	General-purpose input/output	I/O
PORTG7–PORTG0	General-purpose input/output	I/O
PORTH7–PORTH0	General-purpose input/output	I/O
PORTI7–PORTI0	General-purpose input/output	I/O
PORTJ7–PORTJ0	General-purpose input/output	I/O
PORTK7–PORTK0	General-purpose input/output	I/O
PORTL7–PORTL0	General-purpose input/output	I/O
PORTM7–PORTM0	General-purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

NOTE

PORT I1 is true open drain. The internal pullup resistor may be enabled when this pin is configured as an input. An external pullup resistor is required when this pin is configured as an output.

51.2.3.1 Detailed signal description

Table 51-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA7–PORTA0	I/O	General-purpose input/output	
PORTB7–PORTB0		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
PORTC7–PORTC0			
PORTD7–PORTD0		Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.
PORTE7–PORTE0			
PORTF7–PORTF0			
PORTG7–PORTG0			
PORTH7–PORTH0			
PORTI7–PORTI0			
PORTJ7–PORTJ0			
PORTK7–PORTK0			
PORTL7–PORTL0			
PORTM7–PORTM0			

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

51.3 Memory map and register definition

Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

GPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F000	Port Data Output Register (GPIOA_PDOR)	8	R/W	00h	51.3.1/1159
400F_F004	Port Set Output Register (GPIOA_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F008	Port Clear Output Register (GPIOA_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F00C	Port Toggle Output Register (GPIOA_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F010	Port Data Input Register (GPIOA_PDIR)	8	R	00h	51.3.5/1161
400F_F014	Port Data Direction Register (GPIOA_PDDR)	8	R/W	00h	51.3.6/1161
400F_F01C	GPIO Attribute Checker Register (GPIOA_GACR)	8	R/W	00h	51.3.7/1162
400F_F001	Port Data Output Register (GPIOB_PDOR)	8	R/W	00h	51.3.1/1159
400F_F005	Port Set Output Register (GPIOB_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F009	Port Clear Output Register (GPIOB_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F00D	Port Toggle Output Register (GPIOB_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F011	Port Data Input Register (GPIOB_PDIR)	8	R	00h	51.3.5/1161
400F_F015	Port Data Direction Register (GPIOB_PDDR)	8	R/W	00h	51.3.6/1161
400F_F01D	GPIO Attribute Checker Register (GPIOB_GACR)	8	R/W	00h	51.3.7/1162
400F_F002	Port Data Output Register (GPIOC_PDOR)	8	R/W	00h	51.3.1/1159
400F_F006	Port Set Output Register (GPIOC_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F00A	Port Clear Output Register (GPIOC_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F00E	Port Toggle Output Register (GPIOC_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F012	Port Data Input Register (GPIOC_PDIR)	8	R	00h	51.3.5/1161
400F_F016	Port Data Direction Register (GPIOC_PDDR)	8	R/W	00h	51.3.6/1161
400F_F01E	GPIO Attribute Checker Register (GPIOC_GACR)	8	R/W	00h	51.3.7/1162
400F_F003	Port Data Output Register (GPIOD_PDOR)	8	R/W	00h	51.3.1/1159
400F_F007	Port Set Output Register (GPIOD_PSOR)	8	W (always reads 0)	00h	51.3.2/1159

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F00B	Port Clear Output Register (GPIOD_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F00F	Port Toggle Output Register (GPIOD_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F013	Port Data Input Register (GPIOD_PDIR)	8	R	00h	51.3.5/1161
400F_F017	Port Data Direction Register (GPIOD_PDDR)	8	R/W	00h	51.3.6/1161
400F_F01F	GPIO Attribute Checker Register (GPIOD_GACR)	8	R/W	00h	51.3.7/1162
400F_F040	Port Data Output Register (GPIOE_PDOR)	8	R/W	00h	51.3.1/1159
400F_F044	Port Set Output Register (GPIOE_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F048	Port Clear Output Register (GPIOE_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F04C	Port Toggle Output Register (GPIOE_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F050	Port Data Input Register (GPIOE_PDIR)	8	R	00h	51.3.5/1161
400F_F054	Port Data Direction Register (GPIOE_PDDR)	8	R/W	00h	51.3.6/1161
400F_F05C	GPIO Attribute Checker Register (GPIOE_GACR)	8	R/W	00h	51.3.7/1162
400F_F041	Port Data Output Register (GPIOF_PDOR)	8	R/W	00h	51.3.1/1159
400F_F045	Port Set Output Register (GPIOF_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F049	Port Clear Output Register (GPIOF_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F04D	Port Toggle Output Register (GPIOF_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F051	Port Data Input Register (GPIOF_PDIR)	8	R	00h	51.3.5/1161
400F_F055	Port Data Direction Register (GPIOF_PDDR)	8	R/W	00h	51.3.6/1161
400F_F05D	GPIO Attribute Checker Register (GPIOF_GACR)	8	R/W	00h	51.3.7/1162
400F_F042	Port Data Output Register (GPIOG_PDOR)	8	R/W	00h	51.3.1/1159
400F_F046	Port Set Output Register (GPIOG_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F04A	Port Clear Output Register (GPIOG_PCOR)	8	W (always reads 0)	00h	51.3.3/1160

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F04E	Port Toggle Output Register (GPIOG_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F052	Port Data Input Register (GPIOG_PDIR)	8	R	00h	51.3.5/1161
400F_F056	Port Data Direction Register (GPIOG_PDDR)	8	R/W	00h	51.3.6/1161
400F_F05E	GPIO Attribute Checker Register (GPIOG_GACR)	8	R/W	00h	51.3.7/1162
400F_F043	Port Data Output Register (GPIOH_PDOR)	8	R/W	00h	51.3.1/1159
400F_F047	Port Set Output Register (GPIOH_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F04B	Port Clear Output Register (GPIOH_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F04F	Port Toggle Output Register (GPIOH_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F053	Port Data Input Register (GPIOH_PDIR)	8	R	00h	51.3.5/1161
400F_F057	Port Data Direction Register (GPIOH_PDDR)	8	R/W	00h	51.3.6/1161
400F_F05F	GPIO Attribute Checker Register (GPIOH_GACR)	8	R/W	00h	51.3.7/1162
400F_F080	Port Data Output Register (GPIOI_PDOR)	8	R/W	00h	51.3.1/1159
400F_F084	Port Set Output Register (GPIOI_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F088	Port Clear Output Register (GPIOI_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F08C	Port Toggle Output Register (GPIOI_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F090	Port Data Input Register (GPIOI_PDIR)	8	R	00h	51.3.5/1161
400F_F094	Port Data Direction Register (GPIOI_PDDR)	8	R/W	00h	51.3.6/1161
400F_F09C	GPIO Attribute Checker Register (GPIOI_GACR)	8	R/W	00h	51.3.7/1162
400F_F081	Port Data Output Register (GPIOJ_PDOR)	8	R/W	00h	51.3.1/1159
400F_F085	Port Set Output Register (GPIOJ_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F089	Port Clear Output Register (GPIOJ_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F08D	Port Toggle Output Register (GPIOJ_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F091	Port Data Input Register (GPIOJ_PDIR)	8	R	00h	51.3.5/1161
400F_F095	Port Data Direction Register (GPIOJ_PDDR)	8	R/W	00h	51.3.6/1161

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F09D	GPIO Attribute Checker Register (GPIOJ_GACR)	8	R/W	00h	51.3.7/1162
400F_F082	Port Data Output Register (GPIOK_PDOR)	8	R/W	00h	51.3.1/1159
400F_F086	Port Set Output Register (GPIOK_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F08A	Port Clear Output Register (GPIOK_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F08E	Port Toggle Output Register (GPIOK_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F092	Port Data Input Register (GPIOK_PDIR)	8	R	00h	51.3.5/1161
400F_F096	Port Data Direction Register (GPIOK_PDDR)	8	R/W	00h	51.3.6/1161
400F_F09E	GPIO Attribute Checker Register (GPIOK_GACR)	8	R/W	00h	51.3.7/1162
400F_F083	Port Data Output Register (GPIOL_PDOR)	8	R/W	00h	51.3.1/1159
400F_F087	Port Set Output Register (GPIOL_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F08B	Port Clear Output Register (GPIOL_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F08F	Port Toggle Output Register (GPIOL_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F093	Port Data Input Register (GPIOL_PDIR)	8	R	00h	51.3.5/1161
400F_F097	Port Data Direction Register (GPIOL_PDDR)	8	R/W	00h	51.3.6/1161
400F_F09F	GPIO Attribute Checker Register (GPIOL_GACR)	8	R/W	00h	51.3.7/1162
400F_F0C0	Port Data Output Register (GPIOM_PDOR)	8	R/W	00h	51.3.1/1159
400F_F0C4	Port Set Output Register (GPIOM_PSOR)	8	W (always reads 0)	00h	51.3.2/1159
400F_F0C8	Port Clear Output Register (GPIOM_PCOR)	8	W (always reads 0)	00h	51.3.3/1160
400F_F0CC	Port Toggle Output Register (GPIOM_PTOR)	8	W (always reads 0)	00h	51.3.4/1160
400F_F0D0	Port Data Input Register (GPIOM_PDIR)	8	R	00h	51.3.5/1161
400F_F0D4	Port Data Direction Register (GPIOM_PDDR)	8	R/W	00h	51.3.6/1161
400F_F0DC	GPIO Attribute Checker Register (GPIOM_GACR)	8	R/W	00h	51.3.7/1162

51.3.1 Port Data Output Register (GPIOx_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All un-bonded pins not available in your package will default to DISABLE state for lowest power consumption.

Bit	7	6	5	4	3	2	1	0
Read Write	PDO							
Reset	0	0	0	0	0	0	0	0

GPIOx_PDOR field descriptions

Field	Description
PDO	Port Data Output Unimplemented pins for a particular device read as zero. 0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

51.3.2 Port Set Output Register (GPIOx_PSOR)

This register configures whether to set the fields of the PDOR.

Bit	7	6	5	4	3	2	1	0
Read	0							
Write	PTSO							
Reset	0	0	0	0	0	0	0	0

GPIOx_PSOR field descriptions

Field	Description
PTSO	Port Set Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1.

51.3.3 Port Clear Output Register (GPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Bit	7	6	5	4	3	2	1	0
Read	0							
Write	PTCO							
Reset	0	0	0	0	0	0	0	0

GPIOx_PCOR field descriptions

Field	Description
PTCO	Port Clear Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0.

51.3.4 Port Toggle Output Register (GPIOx_PTOR)

Bit	7	6	5	4	3	2	1	0
Read	0							
Write	PTTO							
Reset	0	0	0	0	0	0	0	0

GPIOx_PTOR field descriptions

Field	Description
PTTO	Port Toggle Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

51.3.5 Port Data Input Register (GPIOx_PDIR)

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All un-bonded pins not available in your package will default to DISABLE state for lowest power consumption.

Bit	7	6	5	4	3	2	1	0
Read	PDI							
Write								
Reset	0	0	0	0	0	0	0	0

GPIOx_PDIR field descriptions

Field	Description
PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1.

51.3.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Bit	7	6	5	4	3	2	1	0
Read Write	PDD							
Reset	0	0	0	0	0	0	0	0

GPIOx_PDDR field descriptions

Field	Description
PDD	Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

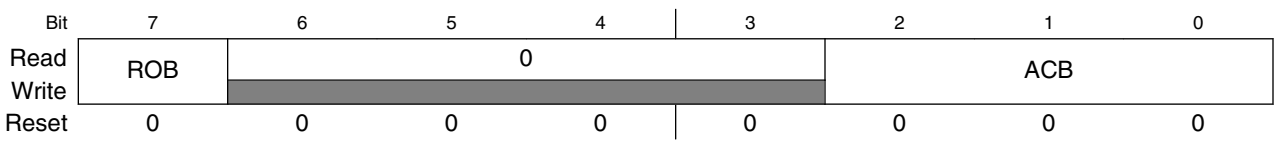
51.3.7 GPIO Attribute Checker Register (GPIOx_GACR)

The GPIO module supports a device-specific number of data ports

NOTE

GPIO supports single-cycle accesses from the processor core IOPORT bus as well as references from the PBRIDGE-mapped address space. Since the core's IOPORT does not support any type of bus stalls or error terminations, if an IOPORT access fails the attribute checks, it is treated as a RAZ/WI access, that is, reads-as-zero and writes ignored.

GPIO GACRn register is checked for PBRDIGE accesses only if the corresponding PACRn register allowed the reference. The appropriate PACRn controls access rights to the entire GPIO address space, while the GACRn registers define the access rights down to the byte level for the actual data ports.



GPIOx_GACR field descriptions

Field	Description
7 ROB	Read-Only Byte This register bit provides a mechanism to "lock" the configuration state defined by ACB. Once asserted, this bit remains set and attempted writes to the ACB are ignored until the next system reset clears the flag. 0 Writes to the ACB are allowed. 1 Writes to the ACB are ignored.
6–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ACB	Attribute Check Byte This 3 bit field defines the attributes required to access the corresponding GPIO port's programming model. 000 User nonsecure: Read + Write; User Secure: Read + Write; Privileged Secure: Read + Write 001 User nonsecure: Read; User Secure: Read + Write; Privileged Secure: Read + Write 010 User nonsecure: None; User Secure: Read + Write; Privileged Secure: Read + Write 011 User nonsecure: Read; User Secure: Read; Privileged Secure: Read + Write 100 User nonsecure: None; User Secure: Read; Privileged Secure: Read + Write 101 User nonsecure: None; User Secure: None; Privileged Secure: Read + Write 110 User nonsecure: None; User Secure: None; Privileged Secure: Read 111 User nonsecure: None; User Secure: None; Privileged Secure: None

GPIOx_GACR field descriptions (continued)

Field	Description

51.4 Functional description

51.4.1 General-purpose input

The logic state of each pin is available via the Port Data Input registers, provided the pin is configured for a digital function and the corresponding Port Control and Interrupt module is enabled.

The Port Data Input registers return the synchronized pin state after any enabled digital filter in the Port Control and Interrupt module. The input pin synchronizers are shared with the Port Control and Interrupt module, so that if the corresponding Port Control and Interrupt module is disabled, then synchronizers are also disabled. This reduces power consumption when a port is not required for general-purpose input functionality.

51.4.2 General-purpose output

The logic state of each pin can be controlled via the port data output registers and port data direction registers, provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then
A pin is configured for the GPIO function and the corresponding port data direction register bit is clear.	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding port data direction register bit is set.	The pin is configured as an output and and the logic state of the pin is equal to the corresponding port data output register.

To facilitate efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers exist to allow one or more outputs within one port to be set, cleared, or toggled from a single register write.

The corresponding Port Control and Interrupt module does not need to be enabled to update the state of the port data direction registers and port data output registers including the set/clear/toggle registers.

Appendix A
Missing Neutral Scenario Recommendations

Assuming that on power up, the meter could be tampered with and neutral is disconnected, the SoC must boot in lowest power consumption mode and enable only the required modules. The process after reset could be as follows:

1. There needs to be a mechanism on board to signal missing neutral. The indication can be connected to GPIO or TAMPERx pin for CPU to monitor.
2. CPU should not enable any peripheral (only iRTC is functional or GPIO port which has the GPIO for missing neutral indication) and first check the status of this TAMPERx pin or GPIO pin
3. CPU can configure and enable minimal peripherals required for missing neutral operation. If in VLPR mode, CPU should enable peripherals in a staggered fashion to control the current increase (since PMC is in loose regulation mode).
4. Based on the state of this pin, the next steps are decided. If the pin indicates no tamper, CPU can start enabling other peripherals and the AFEs and continue normal operation in RUN mode. CPU can also enable FLL and PLL to give clocks to core and peripherals.
5. If the pin indicates missing neutral tamper, CPU will enter into VLPR mode (or continue in VLPR if boot happens in VLPR mode) to conserve power in ICS and Flash.
6. While in VLPR, the time stamp is recorded in iRTC (if TAMPERx pin is used) and CPU will work with one AFE (for phase current measurement with PGA disabled to save power), VREF, one CMP (for frequency measurement). Other system modules that will be needed are AXBS, AIPS. Communication peripherals can be enabled based on application needs. This can vary depending on customer application requirements. AFE works in low power mode with modulator clock on less than 1 MHz (768 kHz nominal). AFE will be running at reduced accuracy since PGA is disabled. VREF is also configured to work in "low power buffer mode".
7. CPU will continue to operate in VLPR (or go into VLPW) until the missing neutral situation is resolved. When operating in VLPR, the software code must written in an optimized way so that current due to code execution is not high.

The table below summarizes the key points of the boot sequence (broken down into major steps) in both RUN and VLPR modes assuming a missing neutral condition exists.

Table A-1. Boot Modes Summary

	Power Up	Reset Sequence	Flash Calibration	Flash Initialization - NVOPT fetch	Flash Initialization - Remainder operation	Boot Up	Phase Measurement
RUN	Flash along with other analog components charges up	Mode for boot selected. Default clock source selected is 4 MHz IRC.	Flash calibrates itself using internal clock source (25 MHz)	Flash initializes and fetches NVOPT information. Based on NVOPT bit, flash clock source is changed to externally supplied clock (ipg_clk)	Flash initializes using externally supplied clock.	core_hold signal de-asserted and CPU starts to boot up. CPU checks for missing neutral condition and enables minimal set of peripherals.	AFE enabled in low power mode with PGA disabled. The clock to AFE is less than 1 MHz.
VLPR	Flash along with other analog components charges up	Mode for boot selected. Default clock source selected is 4 MHz IRC	Flash calibrates itself using externally supplied clock (ipg_clk). Flash clock should be ensured to be =< 1 MHz	Flash initializes using externally supplied clock. NVOPT bit is not checked.	Flash initializes using externally supplied clock.	core_hold signal de-asserted and CPU starts to boot up. CPU checks for missing neutral condition and enables minimal set of peripherals in a staggered fashion	AFE enabled in low power mode with PGA disabled. The clock to AFE is less than 1 MHz.

NOTE

To handle current spikes during Flash charging and Flash NVOPT fetching (when booting in RUN), a capacitor on VDD supply will be required. A capacitor on VDD supply can sustain more than 1.6 mA current if the high current consumption is of short duration (3-5 µs).

Appendix B
Revision history

The following table provides a revision history for this document.

Table B-1. Revision History

Rev. No.	Date	Substantial Changes
2	05/2015	Initial public release

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Document Number KM34P144M75SF0RM
Revision 2, May 2015

ARM® POWERED

freescale™