Kinetis KE1xF Sub-Family Reference Manual

Supports: MKE1xF512VLL16, MKE1xF512VLH16, MKE1xF256VLL16, MKE1xF256VLH16

Document Number: KE1xFP100M168SF0RM
Rev. 4, 06/2019

Contents

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

Chapter 1
About This Manual

Chapter 2
Introduction

Chapter 3
Core Overview

Chapter 4
Interrupts

Chapter 5
System Integration Module (SIM)

Chapter 6
Miscellaneous Control Module (MCM)

Chapter 7
Crossbar Switch Lite (AXBS-Lite)

Chapter 8
Memory Protection Unit (MPU)

Chapter 10
Trigger MUX Control (TRGMUX)

Chapter 11
Direct Memory Access Multiplexer (DMAMUX)

Chapter 12
Enhanced Direct Memory Access (eDMA)

Chapter 13
Memory and memory map

Chapter 14
Local Memory Controller (LMEM)

Chapter 15
Miscellaneous System Control Module (MSCM)

Chapter 16
Flash Acceleration Unit (FAU)

Chapter 17
Flash Memory Module (FTFE)

Chapter 18
Clock Distribution

Chapter 19
System Clock Generator (SCG)

Chapter 20
RTC Oscillator (OSC32)

Chapter 21
Peripheral Clock Controller (PCC)

Chapter 22
Reset and Boot

Chapter 23
Kinetis ROM Bootloader

Chapter 24
Reset Control Module (RCM)

Chapter 25
Power Management

Chapter 26
System Mode Controller (SMC)

Chapter 27
Power Management Controller (PMC)

Chapter 28
Security

Chapter 29
External Watchdog Monitor (EWM)

Chapter 30
Watchdog timer (WDOG)

Chapter 31
Cyclic Redundancy Check (CRC)

Chapter 32
Debug

Chapter 33
JTAG Controller (JTAGC)

Chapter 34
Signal Multiplexing and Pin Assignment

Chapter 35
Port Control and Interrupts (PORT)

Chapter 36
General-Purpose Input/Output (GPIO)

Chapter 37
Analog-to-Digital Converter (ADC)

Chapter 38
Comparator (CMP)

Chapter 39
12-bit Digital-to-Analog Converter (DAC)

Chapter 40
Programmable Delay Block (PDB)

Chapter 41
FlexTimer Module (FTM)

Chapter 42
Low-power Periodic Interrupt Timer (LPIT)

Chapter 43
Pulse Width Timer (PWT)

Chapter 44
Low Power Timer (LPTMR)

Chapter 45
Real Time Clock (RTC)

Chapter 46
Low Power Serial Peripheral Interface (LPSPI)

Chapter 47
Low Power Inter-Integrated Circuit (LPI2C)

Chapter 48
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

Chapter 49
Flexible I/O (FlexIO)

Chapter 50
CAN (FlexCAN)

Chapter 1
About This Manual

1.1 Audience

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

1.2 Organization

This manual has two main sets of chapters.

1. Chapters in the first set contain information that applies to all components on the chip.
2. Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
   - Examples of these groupings are clocking, timers, and communication interfaces.
   - Each grouping includes chapters that provide a technical description of individual modules.

1.3 Module descriptions

Each module chapter has two main parts:

- **Chip-specific:** The first section, *Chip-specific [module name] information*, includes the number of module instances on the chip and possible implementation differences between the module instances, such as differences in FIFO depths or the number of

channels supported. It may also include functional connections between the module instances and other modules. Read this section *first* because its content is crucial to understanding the information in other sections of the chapter.

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

NOTE

If there is a conflict between the chip-specific module information (first section) and the general module information (subsequent sections), the chip-specific information supersedes the general information.

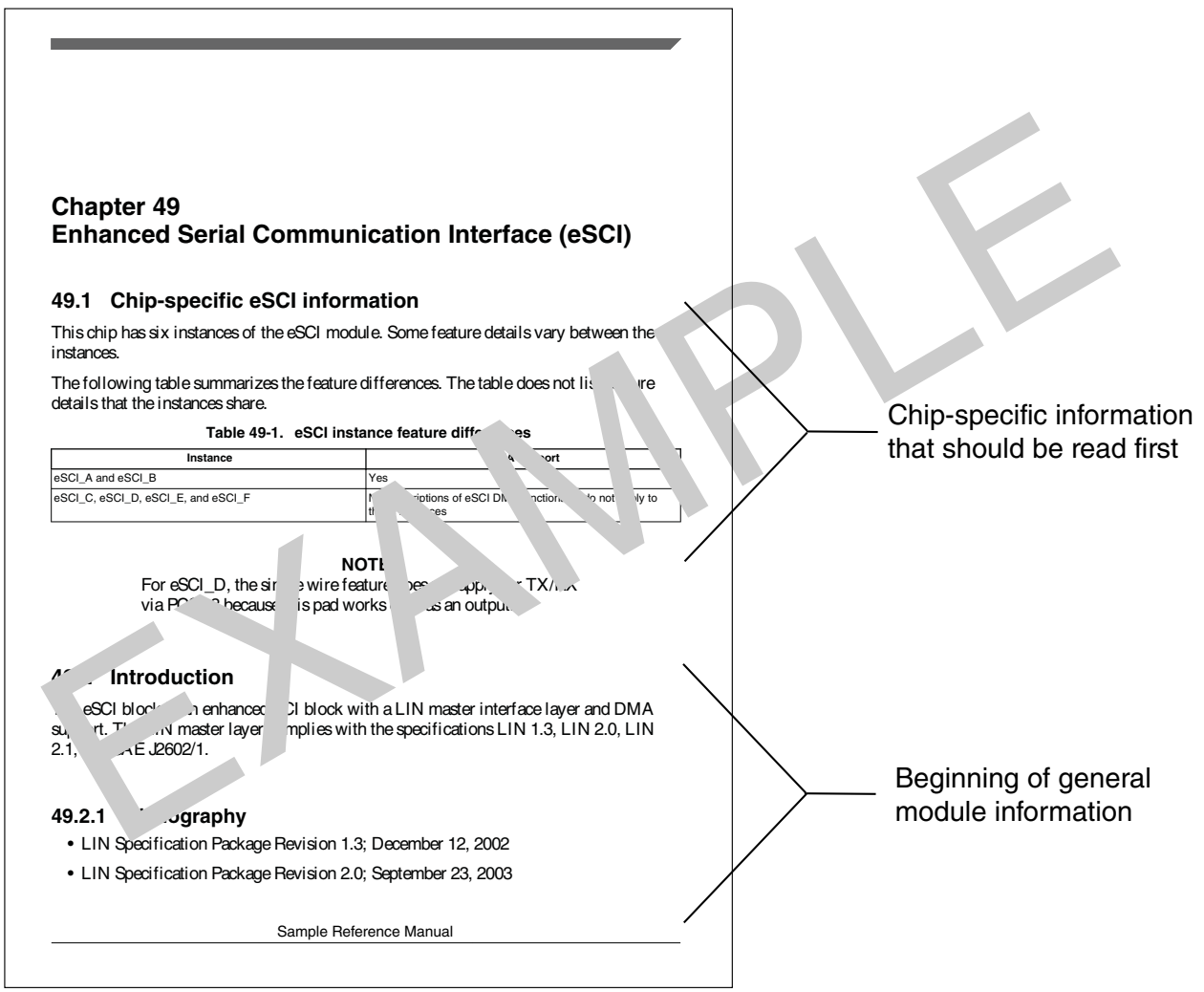
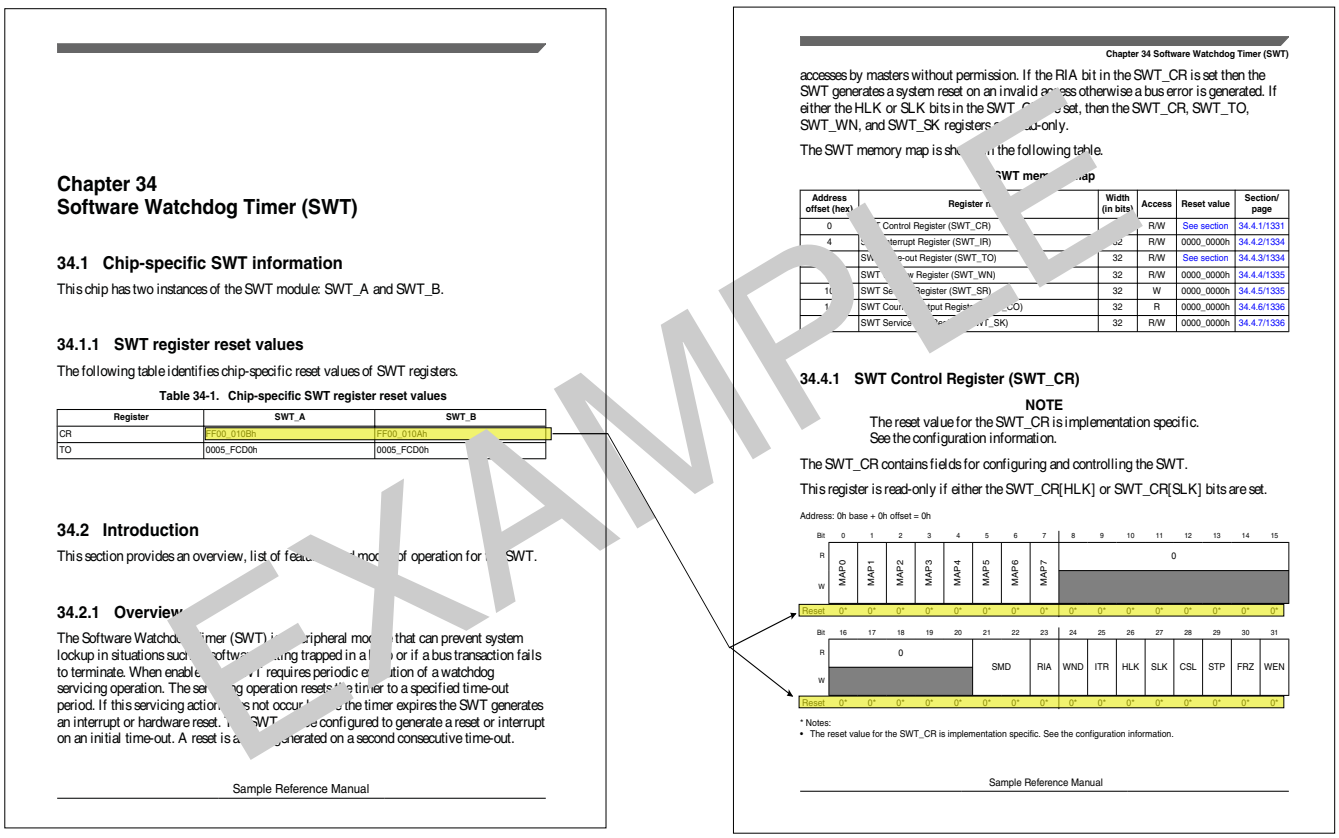


Figure 1-1. Example: chapter chip-specific information and general module information

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

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



Chapter 34
Software Watchdog Timer (SWT)

34.1 Chip-specific SWT information

This chip has two instances of the SWT module: SWT_A and SWT_B.

34.1.1 SWT register reset values

The following table identifies chip-specific reset values of SWT registers.

Table 34-1. Chip-specific SWT register reset values

Register	SWT_A	SWT_B
CR	FF00_010Bh	FF00_010Ah
TO	0005_FCD0h	0005_FCD0h

34.2 Introduction

This section provides an overview, list of feat... mod... of operation for ... SWT.

34.2.1 Overview

The Software Watchd... imer (SWT) i... ripheral mo... that can prevent system lockup in situations suc... softw... ting trapped in a l... or if a bus transaction fails to terminate. When enable... T requires periodic e... tion of a watchdog servicing operation. The se... g operation resets ... timer to a specified time-out period. If this servicing actio... s not occur ... the timer expires the SWT generates an interrupt or hardware reset. ... SWT ... e configured to generate a reset or interrupt on an initial time-out. A reset is a... enerated on a second consecutive time-out.

Sample Reference Manual

Chapter 34 Software Watchdog Timer (SWT)

accesses by masters without permission. If the RIA bit in the SWT_CR is set then the SWT generates a system reset on an invalid a... ss otherwise a bus error is generated. If either the HLK or SLK bits in the SWT_C... set, then the SWT_CR, SWT_TO, SWT_WN, and SWT_SK registers ... d-only.

The SWT memory map is sh... n the following table.

SWT mem... ap

Address offset (hex)	Register n...	Width (in bits)	Access	Reset value	Section/ page
0	... Control Register (SWT_CR)		R/W	See section	34.4.1/1331
4	... terrupt Register (SWT_IR)	...2	R/W	0000_0000h	34.4.2/1334
	SW... e-out Register (SWT_TO)	32	R/W	See section	34.4.3/1334
	SWT ... w Register (SWT_WN)	32	R/W	0000_0000h	34.4.4/1335
10	SWT Se... egister (SWT_SR)	32	W	0000_0000h	34.4.5/1335
1...	SWT Coun... tput Regis... CO)	32	R	0000_0000h	34.4.6/1336
	SWT Service ... T_SK)	32	R/W	0000_0000h	34.4.7/1336

34.4.1 SWT Control Register (SWT_CR)

NOTE
The reset value for the SWT_CR is implementation specific.
See the configuration information.

The SWT_CR contains fields for configuring and controlling the SWT.

This register is read-only if either the SWT_CR[HLK] or SWT_CR[SLK] bits are set.

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

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MAP0	MAP1	MAP2	MAP3	MAP4	MAP5	MAP6	MAP7	0							
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0					SMD	RIA	WND	ITR	HLK	SLK	CSL	STP	FRZ	WEN	
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

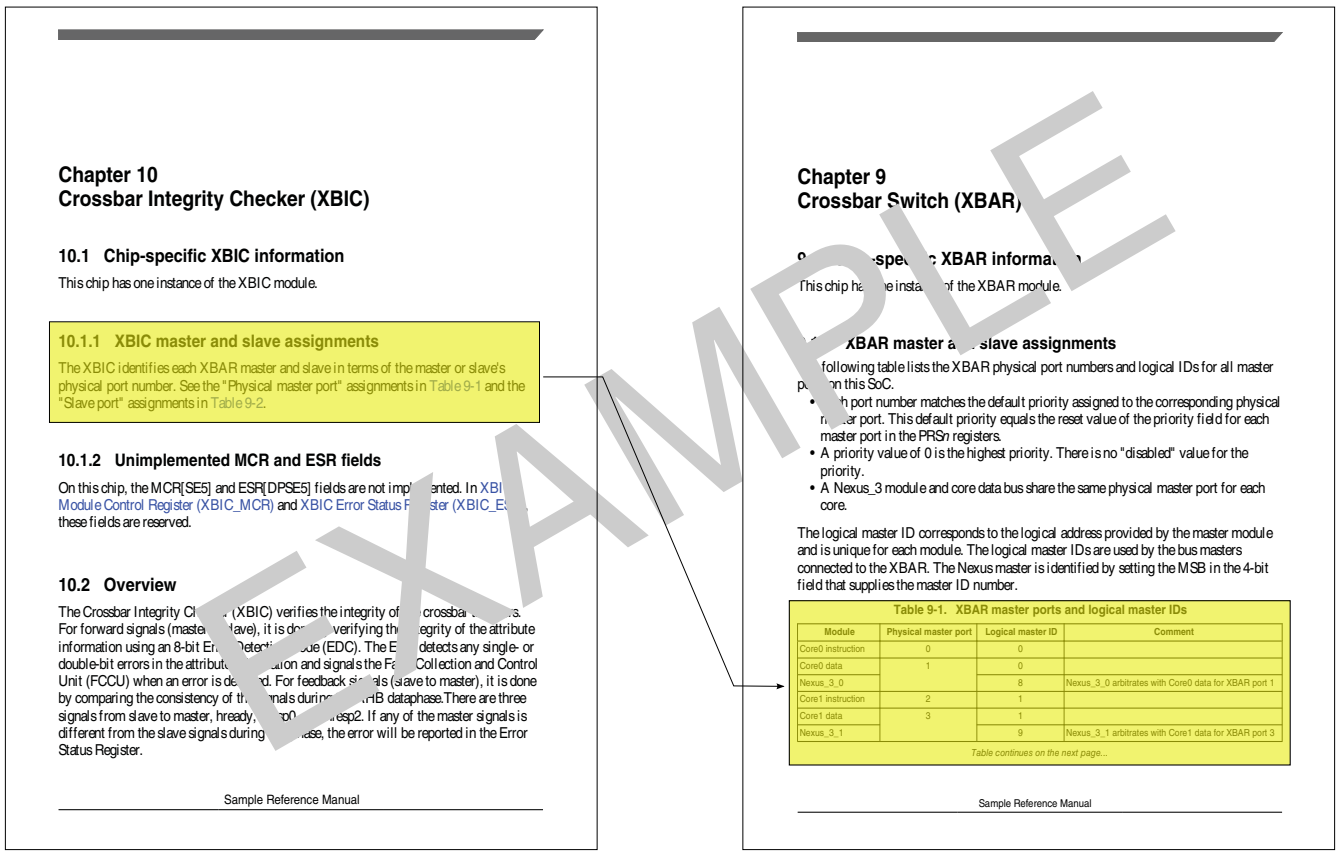
* Notes:
- The reset value for the SWT_CR is implementation specific. See the configuration information.

Sample Reference Manual

Figure 1-2. Example: chip-specific information that supersedes content in the same chapter

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

The chip-specific information below refers to another chapter's chip-specific information. In this case, read both sets of chip-specific information before reading further in the chapter.



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC master and slave assignments

The XBIC identifies each XBAR master and slave in terms of the master or slave's physical port number. See the "Physical master port" assignments in Table 9-1 and the "Slave port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not implemented. In XBI Module Control Register (XBIC_MCR) and XBIC Error Status Register (XBIC_E these fields are reserved.

10.2 Overview

The Crossbar Integrity Ch (XBIC) verifies the integrity of crossba For forward signals (maste lave), it is do verifying th egrity of the attribute information using an 8-bit Er Detecti ue (EDC). The E detects any single- or double-bit errors in the attribut tion and signals the Fa Collection and Control Unit (FCCU) when an error is de d. For feedback si ls (slave to master), it is done by comparing the consistency of th nals duri hB dataphase.There are three signals from slave to master, hready, so0 esp2. If any of the master signals is different from the slave signals during ase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9 -spe c XBAR informa n

This chip ha e insta of the XBAR module.

XBAR master a slave assignments

following table lists the XBAR physical port numbers and logical IDs for all master p on this SoC.

- h port number matches the default priority assigned to the corresponding physical r er port. This default priority equals the reset value of the priority field for each master port in the PRSn registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical master port for each core.

The logical master ID corresponds to the logical address provided by the master module and is unique for each module. The logical master IDs are used by the bus masters connected to the XBAR. The Nexus master is identified by setting the MSB in the 4-bit field that supplies the master ID number.

Table 9-1. XBAR master ports and logical master IDs

Module	Physical master port	Logical master ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

Figure 1-3. Example: chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in:

- Memory maps including:
  - Addresses
  - The name and acronym/abbreviation of each register
  - The width of each register (in bits)
  - Each register's reset value
  - The page number on which each register is described
- Register figures
- Field-description tables
- Associated text

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

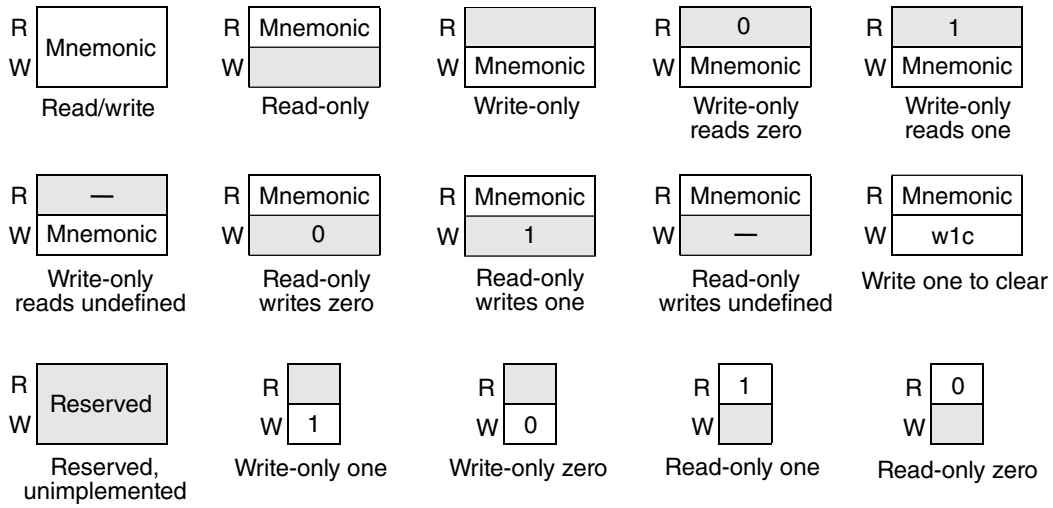


Figure 1-4. Register figure conventions

1.5 Conventions

1.5.1 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*.

1.5.2 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

Table continues on the next page...

Example	Description
	is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.5.3 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, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. • Do not modify the default value of a reserved programming setting, such as the reset value of a reserved register field. • Consider undefined locations in memory to be reserved.
w1c	Write 1 to clear: Refers to a register bitfield that must be written as 1 to be "cleared."

Chapter 2
Introduction

2.1 Overview

Information found here provides an overview of this MCU, which is a part of Kinetis E-series of ARM® Cortex®-M4 MCUs and product family. It also presents high-level descriptions of the modules available on the device covered by this document.

2.2 Block Diagram

The following figure shows a top-level block diagram of the MCU superset device.

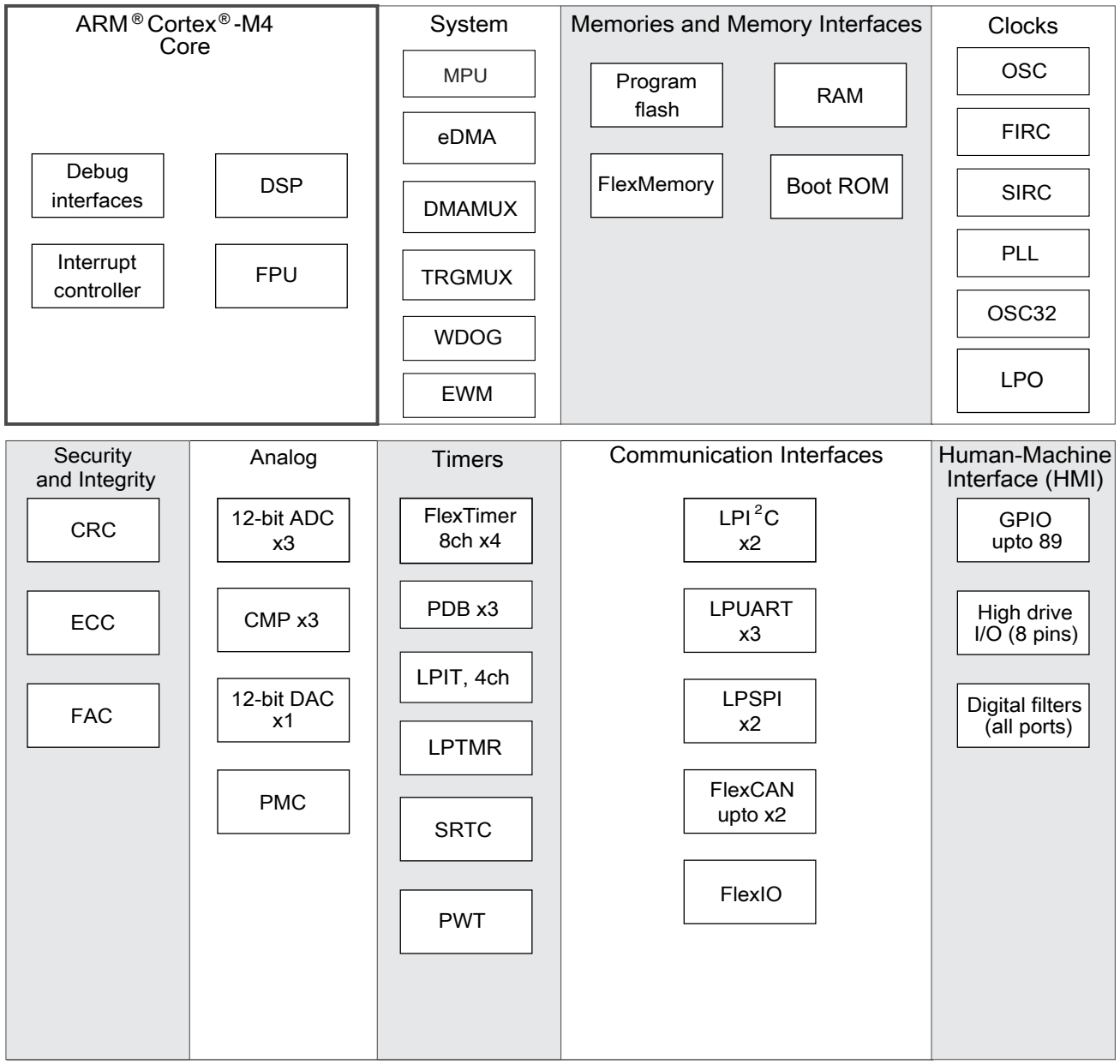


Figure 2-1. MCU block diagram

2.3 Module Functional Categories

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

Table 2-1. Module functional categories

Module category	Description
ARM® Cortex®-M4 core	• 32-bit MCU core from ARM's Cortex-M class adding DSP instructions and single-precision floating point unit based on ARMv7 architecture
System modules	• System integration module (SIM) • System mode controller (SMC) • Miscellaneous control module (MCM) • Crossbar switch (AXBS-Lite) • Memory protection unit • Peripheral bridge (AIPS-Lite) • Direct memory access (DMA) controller with multiplexer (DMAMUX) to increase available DMA requests. DMA can now handle transfers in VLPS mode • Watchdog (WDOG) • External watchdog monitor (EWM)

Memories and memory interfaces	• Internal memories include:
	  • Program flash memory
	  • FlexMemory
	  • FlexNVM
	  • FlexRAM
	  • SRAM
	  • Boot ROM
	  • Cache memory

Clocks	• System clock generator (SCG)
	  • Phase-locked loop (PLL)
	  • Fast internal reference clock (FIRC)
	  • Slow internal reference clock (SIRC)
	  • System oscillator (OSC)
	• Low Power Oscillator (LPO)
	• Peripheral Clock Control (PCC)

Security and integrity modules	• Cyclic Redundancy Check (CRC) module for error detection • Error-correcting code (ECC) on Flash and SRAM memories • Flash Access Control (FAC) • 128-bit unique identification (ID) number • Memory Protection Unit (MPU) module • ADC self-test and calibration feature
Analog modules	• High speed analog-to-digital converter (ADC) • Comparator (CMP) • Digital-to-analog converter (DAC) • Bandgap voltage reference (1V reference voltage) • Power management controllers (PMC) • Multiple power modes available based on high speed run, run, wait, stop, and power-down modes
Timer modules	• Programmable delay block (PDB) • FlexTimers (FTM) • Low-power periodic interrupt timer (LPIT) • Low power timer (LPTMR) • Independent real time clock (RTC)
Communication interfaces	• FlexCAN • Low-power Serial peripheral interface (LPSPI) • Low-power Inter-integrated circuit (LPI^{2}C) • Low-power UART (LPUART) • FlexIO
Human-machine interfaces (HMI)	• General purpose input/output controller (GPIO)

Table 2-1. Module functional categories

Module category	Description
	• High drive I/O pins, see Pin properties. • Digital filters, see "Ports summary" table in Port control and interrupt module features.

Chapter 3
Core Overview

3.1 ARM Cortex-M4

The ARM Cortex-M4 is the member of Cortex-M Series of processors designed for the micro-controller market. It is a 32-bit processor core with a 3-stage pipeline Harvard architecture. The Cortex-M4 processor implements a large variety of highly efficient digital signal processing (DSP) features applicable to digital signal control markets including single-cycle multiply accumulate (MAC) instructions, optimized SIMD arithmetic, saturating arithmetic instructions, dedicated hardware division and an IEEE754-compliant single-precision Floating Point Unit (FPU).

The Cortex-M4 processor implements a version of the Thumb® instruction set based on Thumb-2 technology, ensuring high code density and reduced program memory requirements. The Cortex-M4 instruction set provides the exceptional performance expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers.

Cortex-M4 Processor Features

- Thumb/Thumb-2 Instruction Set
- Nested Vectored Interrupt Controller (NVIC)
- Digital signal processing instruction and hardware (DSP)
- IEEE 755 compliant single precision Floating Point Unit (FPU)
- Bit-banding for bit manipulation
- SWD/JTAG debug port (SWD/JTAG)
- Instrumentation Trace Macrocell (ITM)
- Data Watchpoint and Trace Unit (DWT)
- Trace Port Interface Unit (TPIU)
- Flash Patch and Breakpoint Unit (FPB)
- 24-bit system tick timer (SysTick)

The detailed architecture and programming model of Cortex-M4 processor are discussed in the following documents from ARM.

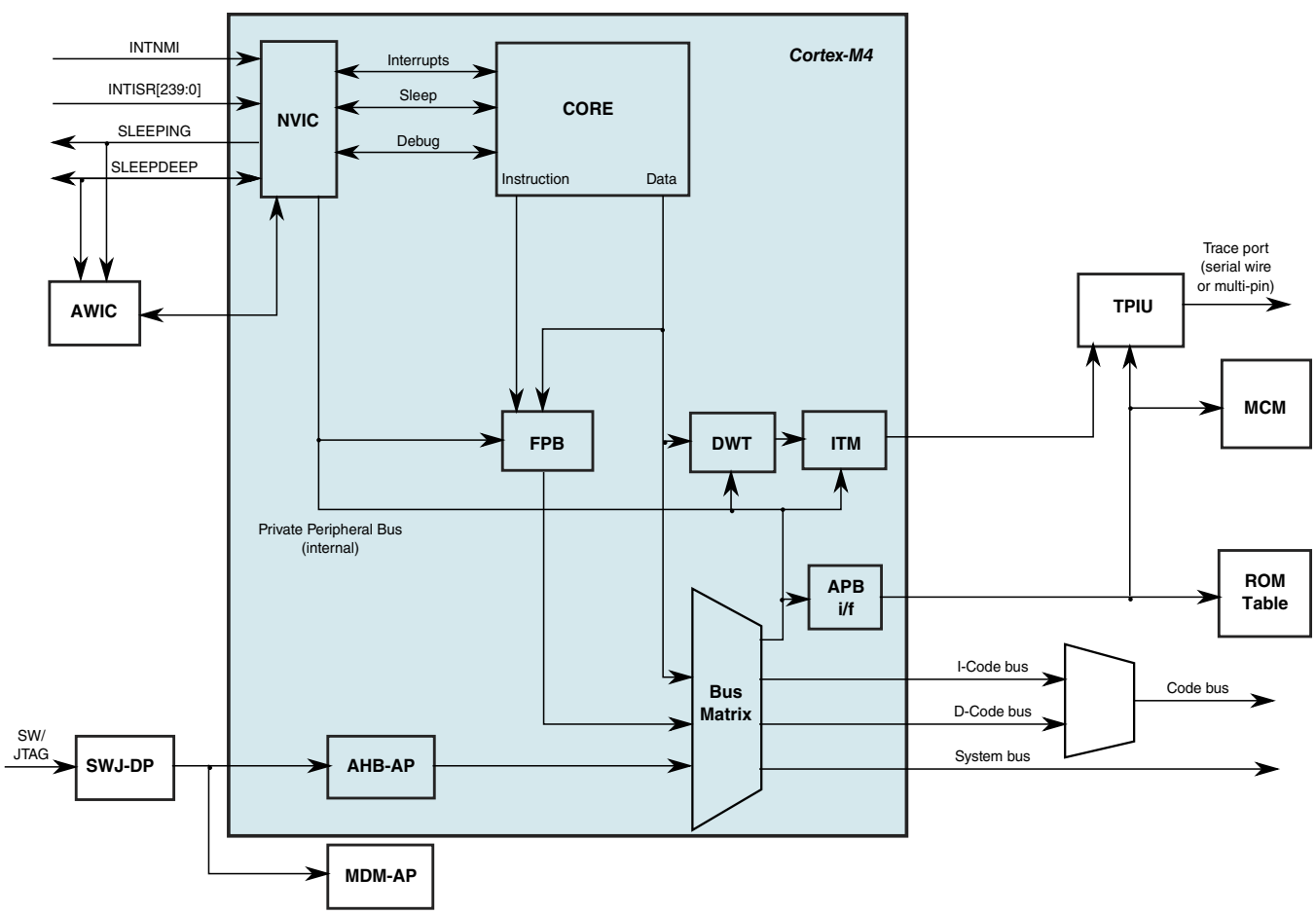
- Cortex-M4 Devices Generic User Guide
- ARM Cortex-M4 Processor Technical Reference Manual
- ARMv7-M Architecture Reference Manual

3.2 Core Buses and Interfaces

The Cortex-M4 processor provides multiple buses and interfaces using AMBA® technology to provide memory and peripheral accesses, a NVIC interface for interrupt handling, a Debug Access Port (DAP) for SWD/JTAG debug and an TPIU interface for trace.

The following interfaces are implemented on the Cortex-M4 processor of this device.

- I-Code, D-Code, and System bus
- PPB bus
- NVIC interface
- Trace port interface
- Debug port interface



3.3 Core Component Configuration

The processor supports optional tightly-coupled system components. The following table lists the specific configuration of the Cortex-M4 core on this device.

Component name	Present on this device	Note
FPU	YES	
MPU	Not present	A NXP proprietary MPU is used on this device
Bit-banding	YES	
FPB	YES	
DWT	YES	
ITM	YES	
ETM	Not present	
HTM	Not present	
TPIU	YES	
WIC	YES	
JTAG	YES	
SWD	YES	

3.4 SysTick Clock Configuration

The System Tick Timer's clock source is always the core clock (CORE_CLK) on this device. This results in the following:

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

Chapter 4
Interrupts

4.1 Introduction

The ARM Cortex-M4 processor includes an interrupt controller called the Nested Vectored Interrupt Controller (NVIC). It is closely coupled to the processor core to provide outstanding interrupt handling abilities and low latency interrupt processing. The NVIC supports nested interrupt, dynamic priority changes, interrupt masking and interrupt tail-chaining. In addition, the NVIC also supports re-locatable vector table and an external Nonmaskable Interrupt (NMI).

The NVIC registers are located within the processor's internal System Control Space (SCS) with base address of 0xE000E000. Most of the NVIC registers are accessible only in privileged mode. The detailed NVIC functionalities and registers descriptions are discussed in the following documents from ARM web.

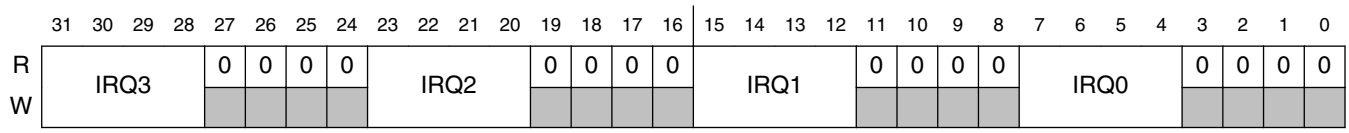
- Cortex-M4 Devices Generic User Guide
- ARM Cortex-M4 Processor Technical Reference Manual

4.2 NVIC configuration

The NVIC supports configurable interrupt number and level of priority. The following sections speficy the exact priority level and interrupt vectors implemented on this device.

4.2.1 Interrupt priority levels

The NVIC on this device supports 16 interrupt priority levels. Therefore, the NVIC_IPR registers contains 4 bits for each interrupt request (IRQ). For example, NVIC_IPR0 is shown below:



4.2.2 Non-maskable interrupt

This device supports non-maskable interrupt (NMI) to the NVIC. It is controlled by the external NMI signal from the pin. The pin which the NMI signal is multiplexed on, must be configured for the NMI function to generate the non-maskable interrupt request.

4.3 Interrupt channel assignments

The interrupt source assignments are defined in the following table.

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

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

Table 4-2. Interrupt vector assignments

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
ARM Core System Handler Vectors						
0x0000_0000	0	–	–	–	ARM core	Initial Stack Pointer
0x0000_0004	1	–	–	–	ARM core	Initial Program Counter
0x0000_0008	2	–	–	–	ARM core	Non-maskable Interrupt (NMI)
0x0000_000C	3	–	–	–	ARM core	Hard Fault
0x0000_0010	4	–	–	–	ARM core	MemManage Fault
0x0000_0014	5	–	–	–	ARM core	Bus Fault
0x0000_0018	6	–	–	–	ARM core	Usage Fault
0x0000_001C	7	–	–	–	—	—

Table continues on the next page...

Table 4-2. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0020	8	–	–	–	—	—
0x0000_0024	9	–	–	–	—	—
0x0000_0028	10	–	–	–	—	—
0x0000_002C	11	–	–	–	ARM core	Supervisor call (SVCall)
0x0000_0030	12	–	–	–	ARM core	Debug Monitor
0x0000_0034	13	–	–	–	—	—
0x0000_0038	14	–	–	–	ARM core	Pendable request for system service (PendableSrvReq)
0x0000_003C	15	–	–	–	ARM core	System tick timer (SysTick)
Non-Core Vectors						
0x0000_0040	16	0	0	0	DMA	DMA channel 0 transfer complete
0x0000_0044	17	1	0	0	DMA	DMA channel 1 transfer complete
0x0000_0048	18	2	0	0	DMA	DMA channel 2 transfer complete
0x0000_004C	19	3	0	0	DMA	DMA channel 3 transfer complete
0x0000_0050	20	4	0	1	DMA	DMA channel 4 transfer complete
0x0000_0054	21	5	0	1	DMA	DMA channel 5 transfer complete
0x0000_0058	22	6	0	1	DMA	DMA channel 6 transfer complete
0x0000_005C	23	7	0	1	DMA	DMA channel 7 transfer complete
0x0000_0060	24	8	0	2	DMA	DMA channel 8 transfer complete
0x0000_0064	25	9	0	2	DMA	DMA channel 9 transfer complete
0x0000_0068	26	10	0	2	DMA	DMA channel 10 transfer complete
0x0000_006C	27	11	0	2	DMA	DMA channel 11 transfer complete
0x0000_0070	28	12	0	3	DMA	DMA channel 12 transfer complete
0x0000_0074	29	13	0	3	DMA	DMA channel 13 transfer complete
0x0000_0078	30	14	0	3	DMA	DMA channel 14 transfer complete
0x0000_007C	31	15	0	3	DMA	DMA channel 15 transfer complete
0x0000_0080	32	16	0	4	DMA	DMA error interrupt channels 0-15
0x0000_0084	33	17	0	4	MCM	FPU sources
0x0000_0088	34	18	0	4	Flash memory	Command complete
0x0000_008C	35	19	0	4	Flash memory	Read collision
0x0000_0090	36	20	0	5	PMC	Low-voltage detect, low-voltage warning
0x0000_0094	37	21	0	5	Flash memory	Double bit fault detect interrupt
0x0000_0098	38	22	0	5	WDOG or EWM	Both watchdog modules share this interrupt.
0x0000_009C	39	23	0	5	—	—
0x0000_00A0	40	24	0	6	LPI^{2}C0	—
0x0000_00A4	41	25	0	6	LPI^{2}C1	—

Table continues on the next page...

Table 4-2. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_00A8	42	26	0	6	LPSPI0	Single interrupt vector for all sources
0x0000_00AC	43	27	0	6	LPSPI1	Single interrupt vector for all sources
0x0000_00B0	44	28	0	7	—	—
0x0000_00B4	45	29	0	7	PWT	—
0x0000_00B8	46	30	0	7	—	—
0x0000_00BC	47	31	0	7	LPUART0	LPUART0 transmit interrupt
0x0000_00C0	48	32	1	8	LPUART0	LPUART0 receive interrupt
0x0000_00C4	49	33	1	8	LPUART1	LPUART1 transmit interrupt
0x0000_00C8	50	34	1	8	LPUART1	LPUART1 receive interrupt
0x0000_00CC	51	35	1	8	LPUART2	LPUART2 transmit interrupt
0x0000_00D0	52	36	1	9	LPUART2	LPUART2 receive interrupt
0x0000_00D4	53	37	1	9	—	—
0x0000_00D8	54	38	1	9	—	—
0x0000_00DC	55	39	1	9	ADC0	—
0x0000_00E0	56	40	1	10	CMP0	—
0x0000_00E4	57	41	1	10	CMP1	—
0x0000_00E8	58	42	1	10	FTM0	Single interrupt vector for all sources
0x0000_00EC	59	43	1	10	FTM1	Single interrupt vector for all sources
0x0000_00F0	60	44	1	11	FTM2	Single interrupt vector for all sources
0x0000_00F4	61	45	1	11	—	—
0x0000_00F8	62	46	1	11	RTC	Single interrupt vector for Time Alarm, Time Invalid and Time Overflow interrupts
0x0000_00FC	63	47	1	11	RTC	Seconds interrupt
0x0000_0100	64	48	1	12	LPIT	Channel 0
0x0000_0104	65	49	1	12	LPIT	Channel 1
0x0000_0108	66	50	1	12	LPIT	Channel 2
0x0000_010C	67	51	1	12	LPIT	Channel 3
0x0000_0110	68	52	1	13	PDB0	—
0x0000_0114	69	53	1	13	—	—
0x0000_0118	70	54	1	13	—	—
0x0000_011C	71	55	1	13	—	—
0x0000_0120	72	56	1	14	DAC0	—
0x0000_0124	73	57	1	14	SCG or RCM	—
0x0000_0128	74	58	1	14	Low Power Timer	—
0x0000_012C	75	59	1	14	Port control module	Pin detect (Port A)
0x0000_0130	76	60	1	15	Port control module	Pin detect (Port B)

Table continues on the next page...

Table 4-2. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0134	77	61	1	15	Port control module	Pin detect (Port C)
0x0000_0138	78	62	1	15	Port control module	Pin detect (Port D)
0x0000_013C	79	63	1	15	Port control module	Pin detect (Port E)
0x0000_0140	80	64	2	16	Software	Software interrupt [4]
0x0000_0144	81	65	2	16	—	—
0x0000_0148	82	66	2	16	—	—
0x0000_014C	83	67	2	16	—	—
0x0000_0150	84	68	2	17	PDB1	—
0x0000_0154	85	69	2	17	FlexIO	—
0x0000_0158	86	70	2	17	CMP2	—
0x0000_015C	87	71	2	17	FTM3	Single interrupt vector for all sources
0x0000_0160	88	72	2	18	—	—
0x0000_0164	89	73	2	18	ADC1	—
0x0000_0168	90	74	2	18	ADC2	—
0x0000_016C	91	75	2	18	—	—
0x0000_0170	92	76	2	19	—	—
0x0000_0174	93	77	2	19	PDB2	—
0x0000_0178	94	78	2	19	CAN0	OR’ed [Bus Off OR Transmit Warning OR Receive Warning]
0x0000_017C	95	79	2	19	CAN0	Error
0x0000_0180	96	80	2	20	CAN0	Wake Up
0x0000_0184	97	81	2	20	CAN0	OR'ed Message buffer
0x0000_0188	98	82	2	20	CAN0	Reserved
0x0000_018C	99	83	2	20	CAN0	Reserved (MB extension 32-47)
0x0000_0190	100	84	2	21	CAN0	Reserved (MB extension 48-63)
0x0000_0194	101	85	2	21	CAN1	OR’ed [Bus Off OR Transmit Warning OR Receive Warning]
0x0000_0198	102	86	2	21	CAN1	Error
0x0000_019C	103	87	2	21	CAN1	Wake Up
0x0000_01A0	104	88	2	22	CAN1	OR'ed Message buffer (0-15)
0x0000_01A4	105	89	2	22	CAN1	Reserved (MB extension 16-31)
0x0000_01A8	106	90	2	22	CAN1	Reserved (MB extension 32-47)
0x0000_01AC	107	91	2	22	CAN1	Reserved (MB extension 48-63)

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

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

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

Table 4-3. LPTMR interrupt vector assignment (example only)

Address	Vector	IRQ[1]	NVIC non-IPR register number[2]	NVIC IPR register number[3]	Source module	Source description
0x0000_0128	74	58	1	14	Low Power Timer	—

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

- The NVIC registers you would use to configure the interrupt are:
  - NVIC_ISER1
  - NVIC_ICER1
  - NVIC_ISPR1
  - NVIC_ICPR1
  - NVIC_IABR1
  - NVIC_IPR14
- To determine the particular IRQ's bitfield location within these particular registers:
  - NVIC_ISER1, NVIC_ICER1, NVIC_ISPR1, NVIC_ICPR1, NVIC_IABR1 bit location = IRQ mod 32 = 26
  - NVIC_IPR14 bitfield starting location = 8 × (IRQ mod 4) + 4 = 20

    Since the NVIC_IPR bitfields are 4-bit wide (16 priority levels), the NVIC_IPR14 bitfield range is 20-23

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

- NVIC_ISER1[26]
- NVIC_ICER1[26]
- NVIC_ISPR1[26]
- NVIC_ICPR1[26]
- NVIC_IABR1[26]
- NVIC_IPR14[23:20]

Chapter 5
System Integration Module (SIM)

5.1 Introduction

The System Integration Module (SIM) provides system control and chip configuration registers.

5.1.1 Features

Features of the SIM include:

- System clocking configuration
- Flash and system RAM size configuration
- FlexTimer clock and channel selection and configuration
- ADC trigger selection
- Flash configuration
- System device unique identification (UID)

5.2 Memory map and register definition

NOTE

The SIM registers can only be written in the supervisor mode. In the user mode, write accesses are blocked and will result in a bus error.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_8004	Chip Control register (SIM_CHIPCTL)	32	R/W	0000_0000h	5.2.1/68

Table continues on the next page...

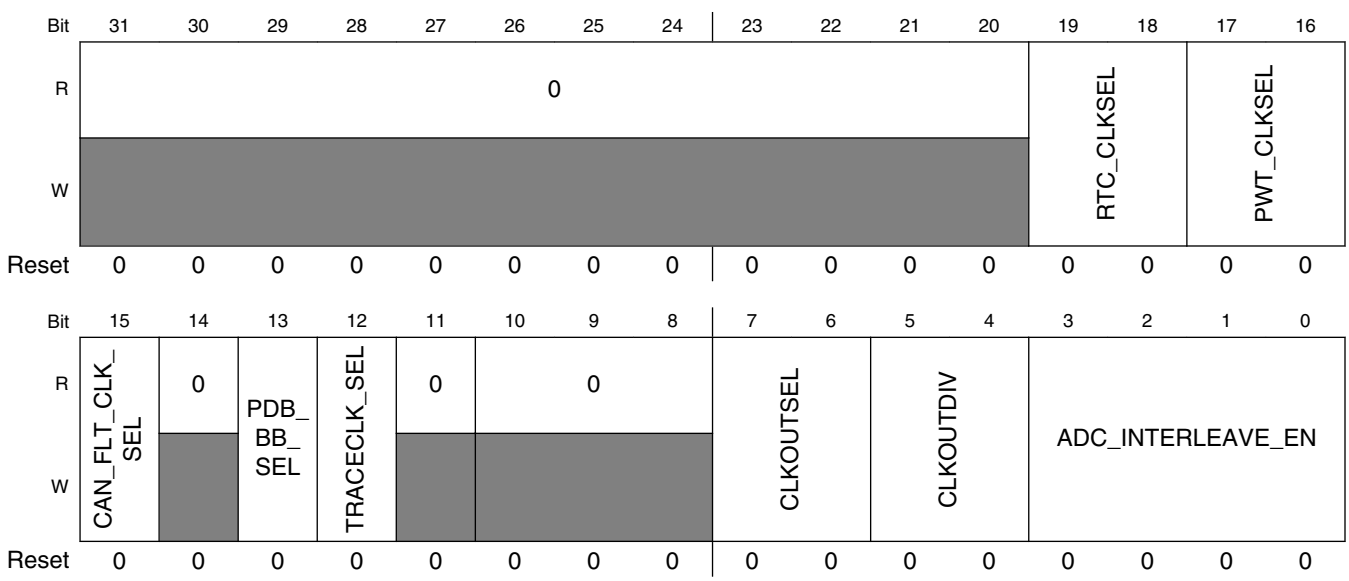
SIM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_800C	FTM Option Register 0 (SIM_FTMOPT0)	32	R/W	0000_0000h	5.2.2/70
4004_8018	ADC Options Register (SIM_ADCOPT)	32	R/W	0000_0000h	5.2.3/72
4004_801C	FTM Option Register 1 (SIM_FTMOPT1)	32	R/W	0000_0000h	5.2.4/74
4004_8024	System Device Identification Register (SIM_SDID)	32	R	See section	5.2.5/76
4004_8040	Platform Clock Gating Control Register (SIM_PLATCGC)	32	R/W	0000_0007h	5.2.6/77
4004_804C	Flash Configuration Register 1 (SIM_FCFG1)	32	R	See section	5.2.7/78
4004_8050	Flash Configuration Register 2 (SIM_FCFG2)	32	R	See section	5.2.8/81
4004_8054	Unique Identification Register High (SIM_UIDH)	32	R	See section	5.2.9/82
4004_8058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R	See section	5.2.10/82
4004_805C	Unique Identification Register Mid Low (SIM_UIDML)	32	R	See section	5.2.11/83
4004_8060	Unique Identification Register Low (SIM_UIDL)	32	R	See section	5.2.12/83
4004_8068	System Clock Divider Register 4 (SIM_CLKDIV4)	32	R/W	1000_0000h	5.2.13/84
4004_806C	Miscellaneous Control register (SIM_MISCTRL)	32	R/W	0000_0000h	5.2.14/85

5.2.1 Chip Control register (SIM_CHIPCTL)

SIM_CHIPCTL contains the controls for selecting ADC COCO trigger, trace clock, clock out source, PDB back-to-back mode and ADC interleave channel.

Address: 4004_8000h base + 4h offset = 4004_8004h



SIM_CHIPCTL 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 RTC_CLKSEL	RTC clock select 00 OSC32_CLK 01 RTC_CLKIN 10 SOSC_CLK 11 reserved
17–16 PWT_CLKSEL	PWT clock select 00 TCLK0 01 TCLK1 10 TCLK2 11 reserved
15 CAN_FLT_CLK_SEL	CAN filter clock select 0 LPO clock 1 SIRC clock
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 PDB_BB_SEL	PDB back-to-back select Selects ADC COCO source as pdb back-to-back mode, see Back-to-back acknowledgement connections for details. 0 PDB0 channel 0 back-to-back operation with ADC0 COCO[7:0]; PDB1 channel 0 back-to-back operation with ADC1 COCO[7:0] ; PDB2 channel 0 back-to-back operation with ADC2 COCO[7:0]. 1 Channel 0 of PDB0, PDB1 and PDB2 back-to-back operation with COCO[7:0] of ADC0, ADC1 and ADC2.
12 TRACECLK_SEL	Debug trace clock select Selects core clock or platform clock as the trace clock source. **NOTE:** The platform clock has the same frequency as the core clock, but they have different phases. 0 core clock 1 platform clock
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 CLKOUTSEL	CLKOUT Select Selects the clock to output on the CLKOUT pin. 00 Reserved 01 SCGCLKOUT(SIRC/FIRC/SOSC/LPFLL), see the SCG_CLKOUTCNFG register 10 RTC oscillator (OSC32) clock (32 kHz) 11 LPO clock (128 kHz)

Table continues on the next page...

SIM_CHIPCTL field descriptions (continued)

Field	Description
5–4 CLKOUTDIV	CLKOUT divider 00 no divider 01 div 2 10 div 4 11 div 8
ADC_ INTERLEAVE_ EN	ADC interleave channel enable Select ADC interleave pins. Bit 3 to 0 are for PTB14, PTB13, PTB1 and PTB0 respectively. 0000 No interleave channel Bit 3: PTB14 to ADC1_SE9 and ADC2_SE9 Bit 2: PTB13 to ADC1_SE8 and ADC2_SE8 Bit 1: PTB1 to ADC0_SE5 and ADC1_SE15 Bit 0: PTB0 to ADC0_SE4 and ADC1_SE14

5.2.2 FTM Option Register 0 (SIM_FTMOPT0)

Address: 4004_8000h base + Ch offset = 4004_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FTM3CLKSEL		FTM2CLKSEL		FTM1CLKSEL		FTM0CLKSEL		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	FTM3FLTxSEL			0	FTM2FLTxSEL			0	FTM1FLTxSEL			0	FTM0FLTxSEL		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_FTMOPT0 field descriptions

Field	Description
31–30 FTM3CLKSEL	FTM3 External Clock Pin Select Selects the external pin used to drive the clock to the FTM3 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM3 external clock driven by TCLK0 pin. 01 FTM3 external clock driven by TCLK1 pin. 10 FTM3 external clock driven by TCLK2 pin. 11 No clock input
29–28 FTM2CLKSEL	FTM2 External Clock Pin Select Selects the external pin used to drive the clock to the FTM2 module.

Table continues on the next page...

SIM_FTMOPT0 field descriptions (continued)

Field	Description
	NOTE: The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM2 external clock driven by TCLK0 pin. 01 FTM2 external clock driven by TCLK1 pin. 10 FTM2 external clock driven by TCLK2 pin. 11 No clock input
27–26 FTM1CLKSEL	FTM1 External Clock Pin Select Selects the external pin used to drive the clock to the FTM1 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM1 external clock driven by TCLK0 pin. 01 FTM1 external clock driven by TCLK1 pin. 10 FTM1 external clock driven by TCLK2 pin. 11 No clock input
25–24 FTM0CLKSEL	FTM0 External Clock Pin Select Selects the external pin used to drive the clock to the FTM0 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM0 external clock driven by TCLK0 pin. 01 FTM0 external clock driven by TCLK1 pin. 10 FTM0 external clock driven by TCLK2 pin. 11 No clock input
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 FTM3FLTxSEL	FTM3 Fault X Select Selects the source of FTM3 fault. Every bit means one fault input respectively. **NOTE:** The pin source for fault must be configured for the FTM module fault function through the appropriate pin control register in the port control module when it comes from external fault pin. TRGMUX_FTM3 SELx is corresponding to FTM3 Fault x input . Bit value = 0: FTM3_FLTx pin Bit value = 1: TRGMUX_FTM3 out
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FTM2FLTxSEL	FTM2 Fault X Select Selects the source of FTM2 fault. Every bit means one fault input respectively. **NOTE:** The pin source for fault must be configured for the FTM module fault function through the appropriate pin control register in the port control module when it comes from external fault pin. TRGMUX_FTM2 SELx is corresponding to FTM2 Fault x input . Bit value = 0: FTM2_FLTx pin Bit value = 1: TRGMUX_FTM2 out

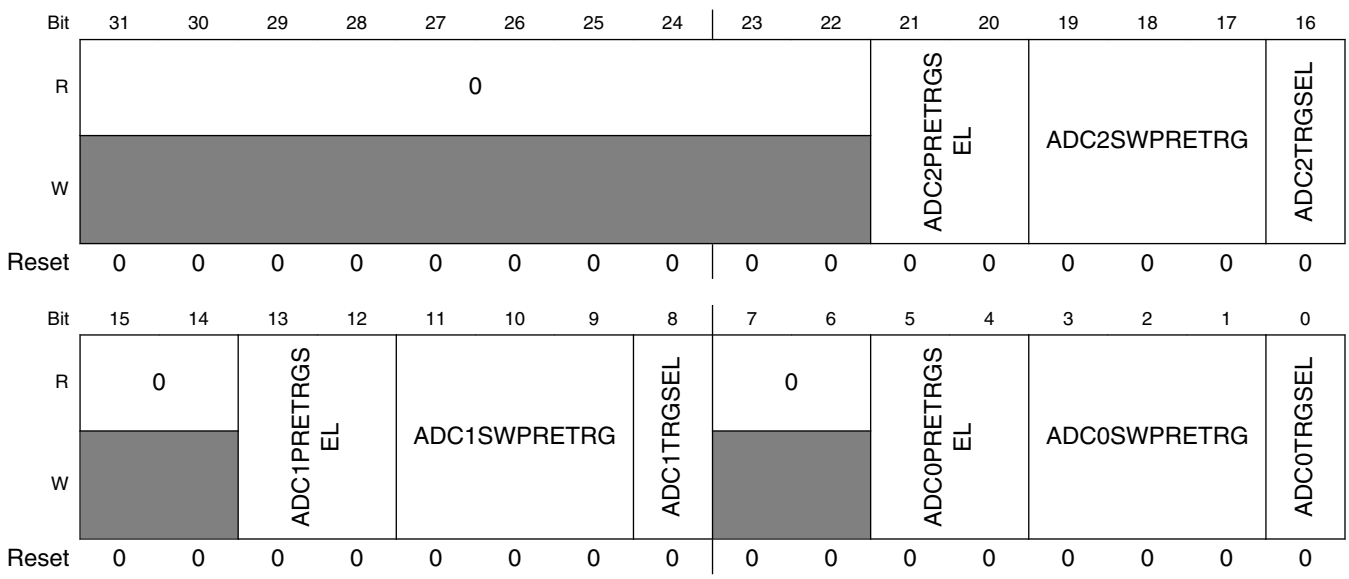
Table continues on the next page...

SIM_FTMOPT0 field descriptions (continued)

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FTM1FLTxSEL	FTM1 Fault X Select Selects the source of FTM1 fault. Every bit means one fault input respectively. **NOTE:** The pin source for fault must be configured for the FTM module fault function through the appropriate pin control register in the port control module when it comes from external fault pin. TRGMUX_FTM1 SELx is corresponding to FTM1 Fault x input. Bit value = 0: FTM1_FLTx pin Bit value = 1: TRGMUX_FTM1 out
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FTM0FLTxSEL	FTM0 Fault X Select Selects the source of FTM0 fault. Every bit means one fault input respectively. **NOTE:** The pin source for fault must be configured for the FTM module fault function through the appropriate pin control register in the port control module when it comes from external fault pin. TRGMUX_FTM0 SELx is corresponding to FTM0 Fault x input. Bit value = 0: FTM0_FLTx pin Bit value = 1: TRGMUX_FTM0 out

5.2.3 ADC Options Register (SIM_ADCOPT)

Address: 4004_8000h base + 18h offset = 4004_8018h



SIM_ADCOPT field descriptions

Field	Description
31–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–20 ADC2PRETRGSEL	ADC2 pre-trigger source select Selects pre-trigger source for ADC2 00 PDB pre-trigger (default) 01 TRGMUX pre-trigger 10 Software pre-trigger 11 Reserved
19–17 ADC2SWPRETRG	ADC2 software pre-trigger sources 000 software pre-trigger disabled 001 - 011 Reserved (do not use) 100 software pre-trigger 0 101 software pre-trigger 1 110 software pre-trigger 2 111 software pre-trigger 3
16 ADC2TRGSEL	ADC2 trigger source select Selects trigger source for ADC2. **NOTE:** Each PDB channel will have up to 8 pre-triggers for this ADC2 channel control. 0 PDB output 1 TRGMUX output
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–12 ADC1PRETRGSEL	ADC1 pre-trigger source select Selects pre-trigger source for ADC1. 00 PDB pre-trigger (default) 01 TRGMUX pre-trigger 10 Software pre-trigger 11 Reserved
11–9 ADC1SWPRETRG	ADC1 software pre-trigger sources 000 software pre-trigger disabled 001 - 011 Reserved (do not use) 100 software pre-trigger 0 101 software pre-trigger 1 110 software pre-trigger 2 111 software pre-trigger 3
8 ADC1TRGSEL	ADC1 trigger source select Selects trigger source for ADC1. 0 PDB output 1 TRGMUX output

Table continues on the next page...

SIM_ADCOPT field descriptions (continued)

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 ADC0PRETRGSEL	ADC0 pre-trigger source select Selects pre-trigger source for ADC0. 00 PDB pre-trigger (default) 01 TRGMUX pre-trigger 10 Software pre-trigger 11 Reserved
3–1 ADC0SWPRETRG	ADC0 software pre-trigger sources 000 software pre-trigger disabled 001 - 011 Reserved (do not use) 100 software pre-trigger 0 101 software pre-trigger 1 110 software pre-trigger 2 111 software pre-trigger 3
0 ADC0TRGSEL	ADC0 trigger source select Selects trigger source for ADC0. 0 PDB output 1 TRGMUX output

5.2.4 FTM Option Register 1 (SIM_FTMOPT1)

Address: 4004_8000h base + 1Ch offset = 4004_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FTM3_OUTSEL								FTM0_OUTSEL							
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							FTM2CH1SEL	FTM2CH0SEL		FTM1CH0SEL		FTM3SYNCBIT	FTM2SYNCBIT	FTM1SYNCBIT	FTM0SYNCBIT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_FTMOPT1 field descriptions

Field	Description
31–24 FTM3_OUTSEL	FTM3 channel modulation select with FTM2_CH1 Bit 7 to 0 are for channel 7 to 0 respectively. 0 No modulation with FTM2_CH1 1 Modulation with FTM2_CH1
23–16 FTM0_OUTSEL	FTM0 channel modulation select with FTM1_CH1 Bit 7 to 0 are for channel 7 to 0 respectively. 0 No modulation with FTM1_CH1 1 Modulation with FTM1_CH1
15–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 FTM2CH1SEL	FTM2 CH1 Select Selects FTM2 CH1 input 0 FTM2_CH1 input 1 exclusive OR of FTM2_CH0,FTM2_CH1, and FTM1_CH1
7–6 FTM2CH0SEL	FTM2 CH0 Select Selects FTM2 CH0 input 00 FTM2_CH0 input 01 CMP0 output 10 CMP1 output 11 CMP2 output
5–4 FTM1CH0SEL	FTM1 CH0 Select Selects FTM1 CH0 input 00 FTM1_CH0 input 01 CMP0 output 10 CMP1 output 11 CMP2 output
3 FTM3SYNCBIT	FTM3 Sync Bit Software control for FTM3 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM3. Software must clear this bit to allow other trigger sources to assert.
2 FTM2SYNCBIT	FTM2 Sync Bit Software control for FTM2 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM2. Software must clear this bit to allow other trigger sources to assert.

Table continues on the next page...

SIM_FTMOPT1 field descriptions (continued)

Field	Description
1 FTM1SYNCBIT	FTM1 Sync Bit Software control for FTM1 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM1. Software must clear this bit to allow other trigger sources to assert.
0 FTM0SYNCBIT	FTM0 Sync Bit Software control for FTM0 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM0. Software must clear this bit to allow other trigger sources to assert.

5.2.5 System Device Identification Register (SIM_SDID)

NOTE

Reset value loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 24h offset = 4004_8024h

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

* Notes:
- x = Undefined at reset.

SIM_SDID field descriptions

Field	Description
31–28 FAMILYID	Kinetis E-series Family ID Specifies the Kinetis E-series family of the device. 0001 KE1x Family (Enhanced features)
27–24 SUBFAMID	Kinetis E-series Sub-Family ID Specifies the Kinetis E-series sub-family of the device.
23–20 SERIESID	Kinetis Series ID Specifies the Kinetis series of the device. 0010 Kinetis E+ series

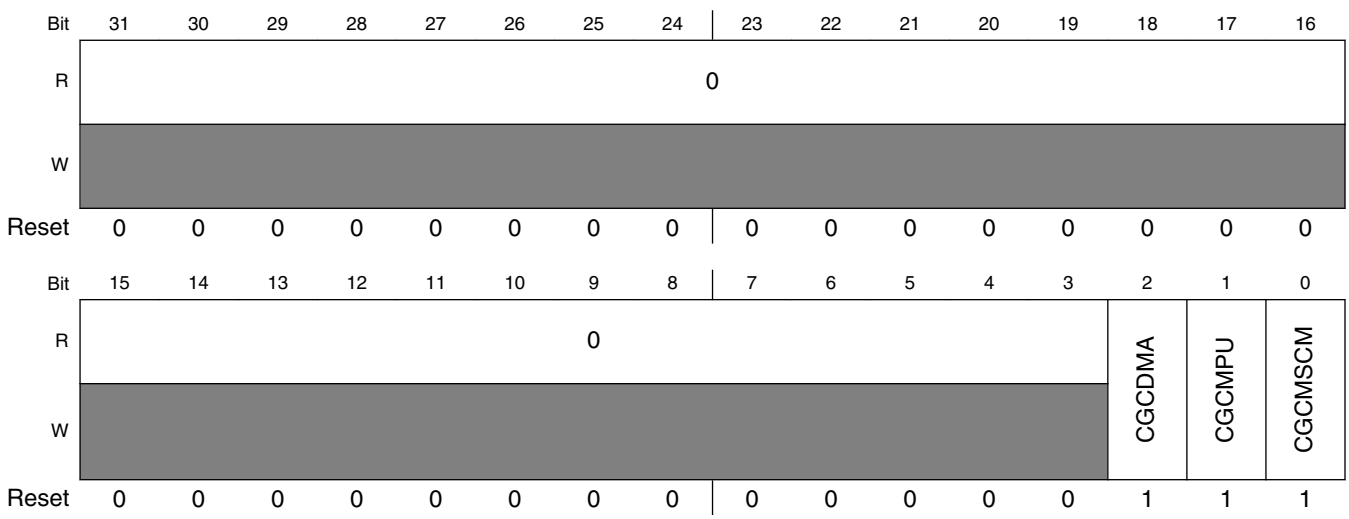
Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
19–16 RAMSIZE	RAM size This field specifies the amount of system RAM available on the device. 0101 32 KB 0111 64 KB Others Reserved
15–12 REVID	Device revision number Specifies the silicon implementation number for the device.
11–7 PROJECTID	Project ID Specifies the silicon feature set identication number for the device. 00001 for this device
PINID	Pin identification Specifies the pin count of the device. 0000111 64-pin 0001010 100-pin

5.2.6 Platform Clock Gating Control Register (SIM_PLATCGC)

Address: 4004_8000h base + 40h offset = 4004_8040h



SIM_PLATCGC field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SIM_PLATCGC field descriptions (continued)

Field	Description
2 CGCDMA	DMA Clock Gating Control Controls the clock gating to the DMA module. 0 Clock disabled 1 Clock enabled
1 CGCMPU	MPU Clock Gating Control Controls the clock gating to the MPU module. 0 Clock disabled 1 Clock enabled
0 CGCMSCM	MSCM Clock Gating Control Controls the clock gating to the MSCM module. 0 Clock disabled 1 Clock enabled

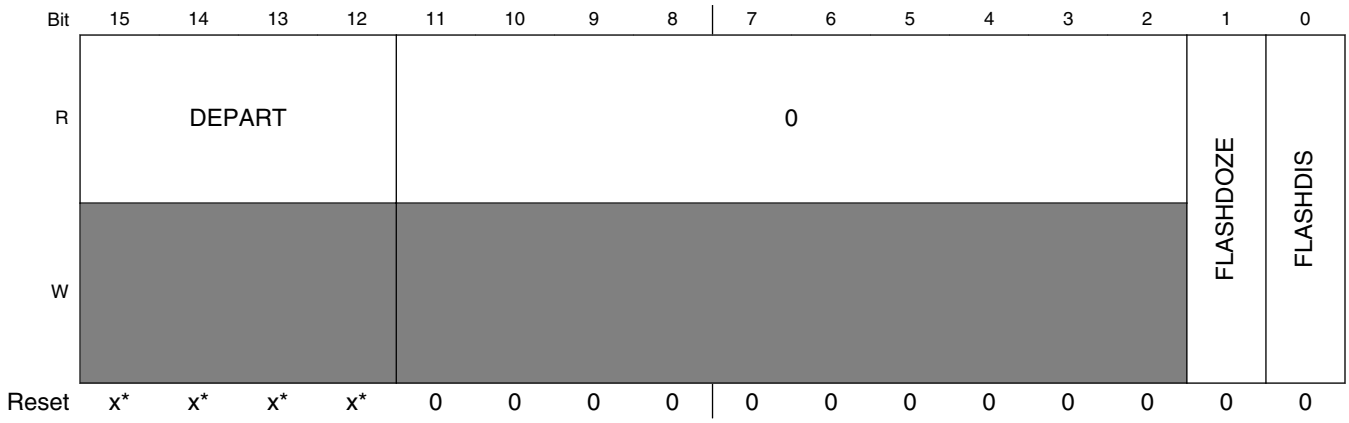
5.2.7 Flash Configuration Register 1 (SIM_FCFG1)

NOTE

Reset value of NVMSIZE, PFSIZE, EEERAM_SIZE, DEPART loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 4Ch offset = 4004_804Ch

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



* Notes:
- x = Undefined at reset.

SIM_FCFG1 field descriptions

Field	Description
31–28 NVMSIZE	FlexNVM size This field specifies the amount of FlexNVM memory available on the device. Undefined values are reserved. 0000 0 KB of FlexNVM 0011 32 KB of FlexNVM 0101 64 KB of FlexNVM 1111 64 KB of FlexNVM
27–24 PFSIZE	Program flash size This field specifies the amount of program flash memory available on the device . Undefined values are reserved. 0000 8 KB of program flash memory, 0.25 KB protection region 0001 16 KB of program flash memory, 0.5 KB protection region 0011 32 KB of program flash memory, 1 KB protection region 0101 64 KB of program flash memory, 2 KB protection region 0111 128 KB of program flash memory, 4 KB protection region 1001 256 KB of program flash memory, 8 KB protection region 1011 / 1111 512 KB of program flash memory, 16 KB protection region
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 EEERAMSIZE	EEE SRAM SIZE EEE SRAM data size . 0000 Reserved 0001 Reserved 0010 4 KB 0011 2 KB 0100 1 KB 0101 512 Bytes

Table continues on the next page...

SIM_FCFG1 field descriptions (continued)

Field	Description
	0110 256 Bytes 0111 128 Bytes 1000 64 Bytes 1001 32 Bytes 1010-1111 Reserved
15–12 DEPART	FlexNVM partition Data flash / EEPROM backup split . See DEPART bit description in FTFE chapter.
11–2 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

5.2.8 Flash Configuration Register 2 (SIM_FCFG2)

NOTE

Reset values of MAXADDR0 and MAXADDR1 are loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 50h offset = 4004_8050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAXADDR0							1	MAXADDR1						
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.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 MAXADDR0	Max address block 0 This field concatenated with 13 trailing zeros indicates the first invalid address of program flash (block 0). For example, if MAXADDR0 = 0x10, the first invalid address of program flash (block 0) is 0x0002_0000. This would be the MAXADDR0 value for a device with 128 KB program flash in flash block 0.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
22–16 MAXADDR1	Max address block 1 This field concatenated with 13 trailing zeros indicates the first invalid address of data flash (block 1).
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

5.2.9 Unique Identification Register High (SIM_UIDH)

Address: 4004_8000h base + 54h offset = 4004_8054h

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDH field descriptions

Field	Description
UID127_96	Unique Identification Unique identification for the device.

5.2.10 Unique Identification Register Mid-High (SIM_UIDMH)

Address: 4004_8000h base + 58h offset = 4004_8058h

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDMH field descriptions

Field	Description
UID95_64	Unique Identification Unique identification for the device.

5.2.11 Unique Identification Register Mid Low (SIM_UIDML)

Address: 4004_8000h base + 5Ch offset = 4004_805Ch

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDML field descriptions

Field	Description
UID63_32	Unique Identification Unique identification for the device.

5.2.12 Unique Identification Register Low (SIM_UIDL)

Address: 4004_8000h base + 60h offset = 4004_8060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID31_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*

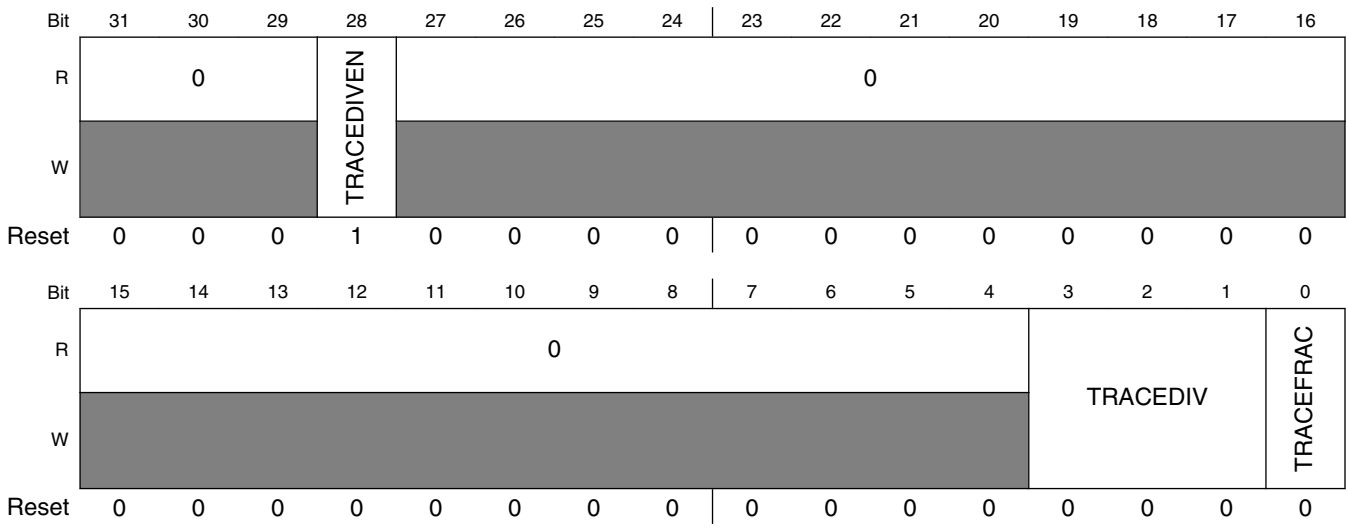
* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDL field descriptions

Field	Description
UID31_0	Unique Identification Unique identification for the device.

5.2.13 System Clock Divider Register 4 (SIM_CLKDIV4)

Address: 4004_8000h base + 68h offset = 4004_8068h

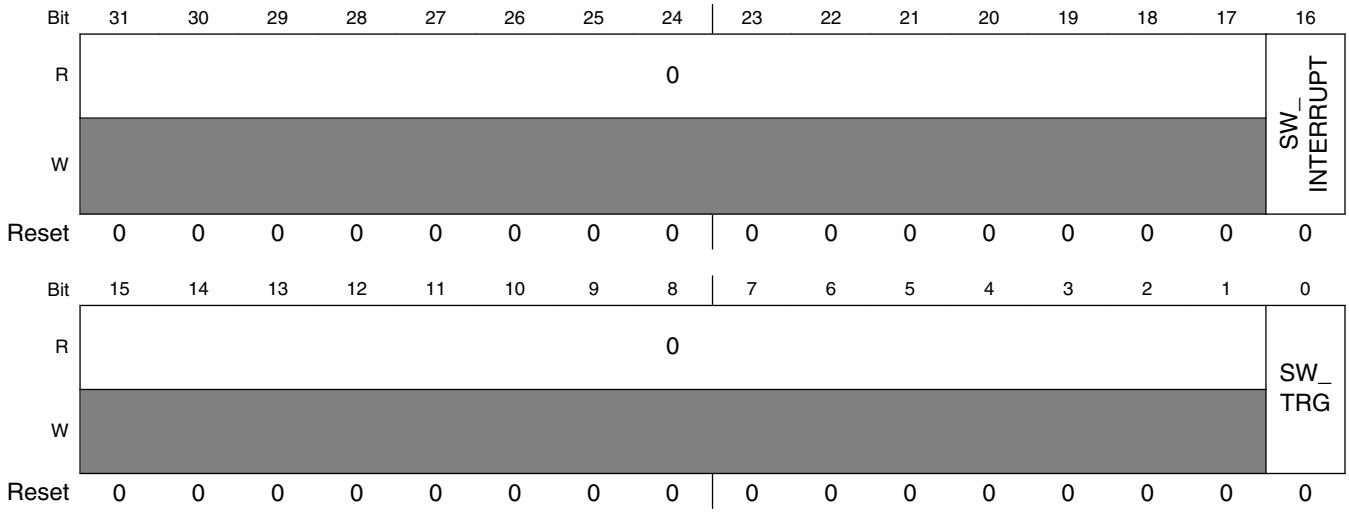


SIM_CLKDIV4 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 TRACEDIVEN	Debug Trace Divider Control This bit controls the Debug Trace Divider. 0 Debug trace divider disabled 1 Debug trace divider enabled
27–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–1 TRACEDIV	Trace clock divider divisor **NOTE** To configure TRACEDIV, the user must disable TRACEDIVEN at first, and then enable it after setting TRACEDIV. This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the trace clock is set by the SIM_CHIPCTL[TRACECLK_SEL]. Divider output clock = Divider input clock × [(TRACEFRAC+1)/(TRACEDIV+1)].
0 TRACEFRAC	Trace clock divider fraction **NOTE** To configure TRACEDIV and TRACEFRAC, the user must clear TRACEDIVEN at first to disable the trace clock divide function. This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the trace clock is set by the SIM_CHIPCTL[TRACECLK_SEL]. Divider output clock = Divider input clock × [(TRACEFRAC+1)/(TRACEDIV+1)].

5.2.14 Miscellaneous Control register (SIM_MISCTRL)

Address: 4004_8000h base + 6Ch offset = 4004_806Ch



SIM_MISCTRL field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 SW_INTERRUPT	Software Interrupt 0 Disables software interrupt. 1 Software can send an interrupt to CPU.
15–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 SW_TRG	Software Trigger bit to TRGMUX

Chapter 6
Miscellaneous Control Module (MCM)

6.1 Introduction

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

6.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision
- Error status and interrupts for the cache write buffer
- Error status and interrupts for the core's floating-point unit (FPU)

6.2 Memory map/register descriptions

The memory map and register descriptions below describe the registers using byte addresses.

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_0008	Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)	16	R	0007h	6.2.1/88
E008_000A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	0007h	6.2.2/89
E008_000C	Core Platform Control Register (MCM_CPCR)	32	R/W	0000_0000h	6.2.3/89
E008_0010	Interrupt Status and Control Register (MCM_ISCR)	32	R	0000_0000h	6.2.4/91
E008_0020	Store Buffer Fault address register (MCM_FADR)	32	R	Undefined	6.2.5/94

Table continues on the next page...

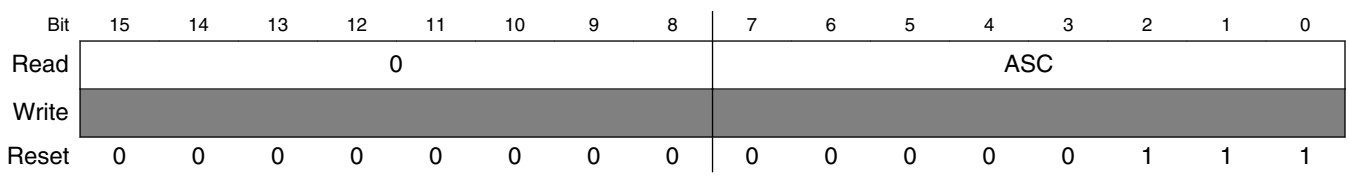
MCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_0024	Store Buffer Fault Attributes register (MCM_FATR)	32	R	Undefined	6.2.6/94
E008_0028	Store Buffer Fault Data Register (MCM_FDR)	32	R	Undefined	6.2.7/96
E008_0030	Process ID register (MCM_PID)	32	R/W	0000_0000h	6.2.8/97
E008_0040	Compute Operation Control Register (MCM_CPO)	32	R/W	0000_0000h	6.2.9/98
E008_0400	Local Memory Descriptor Register (MCM_LMDR0)	32	R/W	See section	6.2.10/99
E008_0404	Local Memory Descriptor Register (MCM_LMDR1)	32	R/W	See section	6.2.10/99
E008_0408	Local Memory Descriptor Register (MCM_LMDR2)	32	R/W	See section	6.2.10/99
E008_0480	LMEM Parity & ECC Control Register (MCM_LMPECR)	32	R/W	0000_0000h	6.2.11/103
E008_0488	LMEM Parity & ECC Interrupt Register (MCM_LMPEIR)	32	R/W	0000_0000h	6.2.12/104
E008_0490	LMEM Fault Address Register (MCM_LMFAR)	32	R	0000_0000h	6.2.13/105
E008_0494	LMEM Fault Attribute Register (MCM_LMFATR)	32	R/W	0000_0000h	6.2.14/106
E008_04A0	LMEM Fault Data High Register (MCM_LMFDHR)	32	R	0000_0000h	6.2.15/107
E008_04A4	LMEM Fault Data Low Register (MCM_LMFDLR)	32	R	0000_0000h	6.2.16/107

6.2.1 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: E008_0000h base + 8h offset = E008_0008h



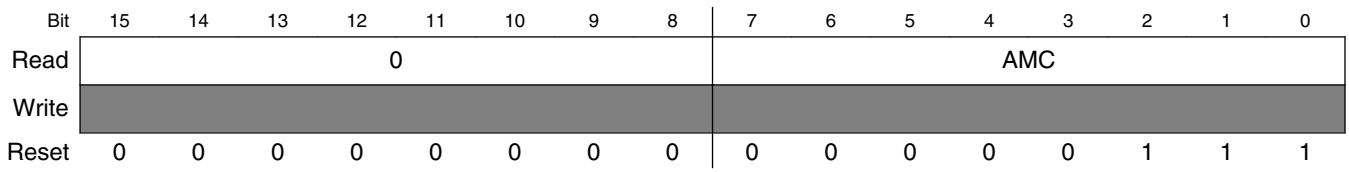
MCM_PLASC field descriptions

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

6.2.2 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: E008_0000h base + Ah offset = E008_000Ah



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

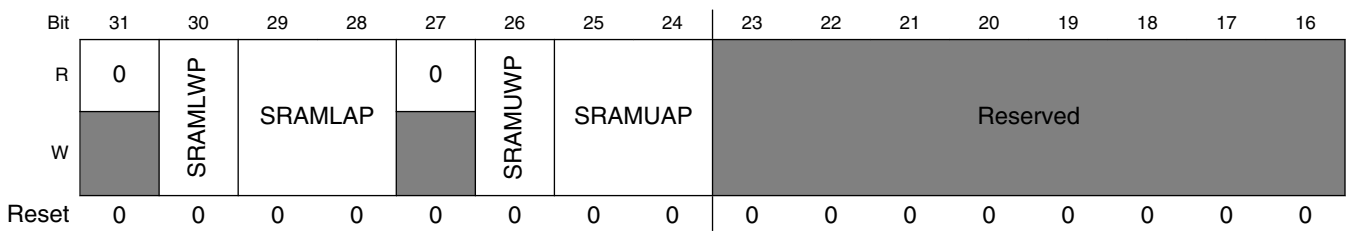
6.2.3 Core Platform Control Register (MCM_CPCR)

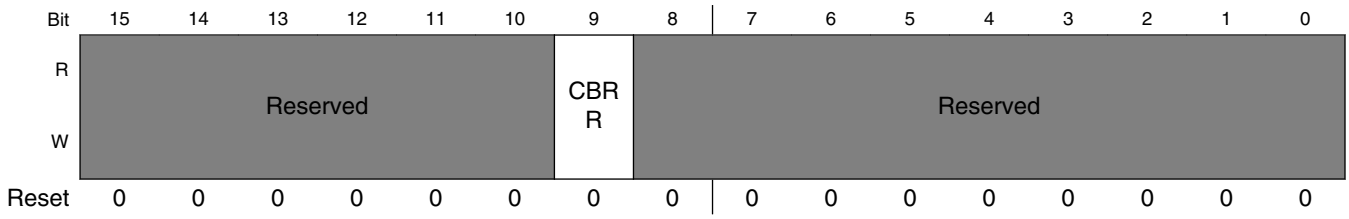
Control Register defines the arbitration and protection schemes for the two system RAM arrays.

NOTE

Bits 23-0 are undefined after reset.

Address: E008_0000h base + Ch offset = E008_000Ch





MCM_CPCR field descriptions

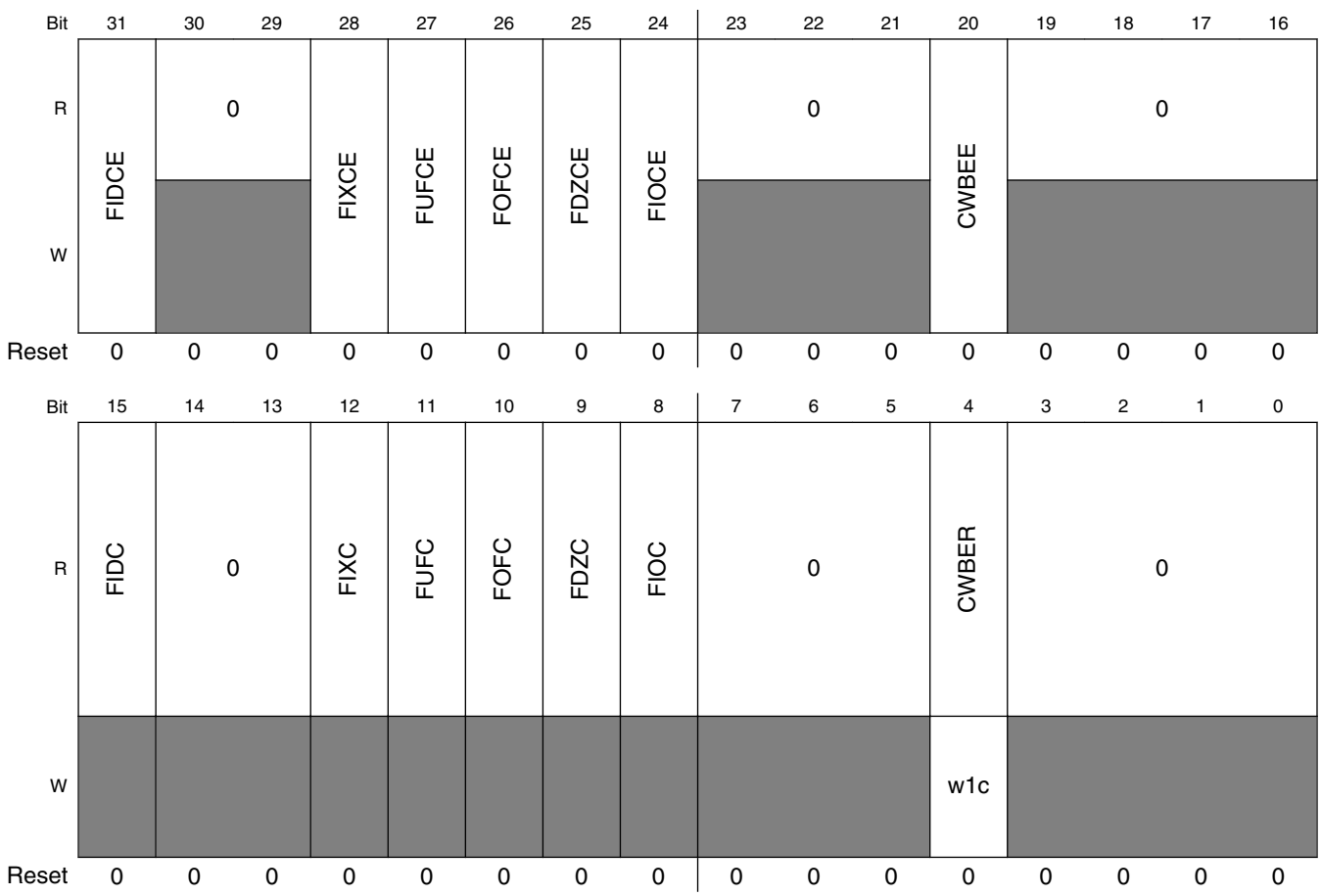
Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SRAMLWP	SRAM_L Write Protect When this bit is set, writes to SRAM_L array generates a bus error.
29–28 SRAMLAP	SRAM_L arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_L array. 00 Round robin 01 Special round robin (favors SRAM backdoor accesses over the processor) 10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SRAMUWP	SRAM_U write protect When this bit is set, writes to SRAM_U array generates a bus error.
25–24 SRAMUAP	SRAM_U arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_U array. 00 Round robin 01 Special round robin (favors SRAM backdoor accesses over the processor) 10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
23–10 Reserved	This field is reserved.
9 CBRR	Crossbar round-robin arbitration enable Configures the crossbar slave ports to fixed-priority or round-robin arbitration. 0 Fixed-priority arbitration 1 Round-robin arbitration
Reserved	This field is reserved.

6.2.4 Interrupt Status and Control Register (MCM_ISCR)

The MCM_ISCR register defines the configuration and reports status for a number of core-related interrupt exception conditions. It includes the enable and status bits associated with the core's floating-point exceptions, and bus errors associated with the core's cache write buffer. The individual event indicators are first qualified with their exception enables and then logically summed to form an interrupt request sent to the core's NVIC.

Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these bits are ignored. Once set, the flags remain asserted until software clears the corresponding FPSCR bit.

Address: E008_0000h base + 10h offset = E008_0010h



MCM_ISCR field descriptions

Field	Description
31 FIDCE	FPU input denormal interrupt enable

Table continues on the next page...

MCM_ISCR field descriptions (continued)

Field	Description
	0 Disable interrupt 1 Enable interrupt
30–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 FIXCE	FPU inexact interrupt enable 0 Disable interrupt 1 Enable interrupt
27 FUFCE	FPU underflow interrupt enable 0 Disable interrupt 1 Enable interrupt
26 FOFCE	FPU overflow interrupt enable 0 Disable interrupt 1 Enable interrupt
25 FDZCE	FPU divide-by-zero interrupt enable 0 Disable interrupt 1 Enable interrupt
24 FIOCE	FPU invalid operation interrupt enable 0 Disable interrupt 1 Enable interrupt
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 CWBEE	Cache write buffer error enable Enables the generation of an interrupt in response to a bus error termination reported on a system bus transfer initiated from the cache's write buffer. 0 Disable error interrupt 1 Enable error interrupt
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 FIDC	FPU input denormal interrupt status This read-only bit is a copy of the core's FPSCR[IDC] bit and signals input denormalized number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IDC] bit. 0 No interrupt 1 Interrupt occurred
14–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 FIXC	FPU inexact interrupt status This read-only bit is a copy of the core's FPSCR[IXC] bit and signals an inexact number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IXC] bit.

Table continues on the next page...

MCM_ISCR field descriptions (continued)

Field	Description
	0 No interrupt 1 Interrupt occurred
11 FUFC	FPU underflow interrupt status This read-only bit is a copy of the core's FPSCR[UFC] bit and signals an underflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[UFC] bit. 0 No interrupt 1 Interrupt occurred
10 FOFC	FPU overflow interrupt status This read-only bit is a copy of the core's FPSCR[OFC] bit and signals an overflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[OFC] bit. 0 No interrupt 1 Interrupt occurred
9 FDZC	FPU divide-by-zero interrupt status This read-only bit is a copy of the core's FPSCR[DZC] bit and signals a divide by zero has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[DZC] bit. 0 No interrupt 1 Interrupt occurred
8 FIOC	FPU invalid operation interrupt status This read-only bit is a copy of the core's FPSCR[IOC] bit and signals an illegal operation has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IOC] bit. 0 No interrupt 1 Interrupt occurred
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 CWBER	Cache write buffer error status Signals a data transfer from the core's cache write buffer was terminated with a bus error. This bit only sets when the corresponding enable bit (CWBEE) is set. The corresponding core fault address, attributes and write data are typically retrieved from the FADR, FATR, and FDR registers during the interrupt service routine before clearing the CWBER flag. 0 No error 1 Error occurred
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

6.2.5 Store Buffer Fault address register (MCM_FADR)

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

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes have no effect.

Address: E008_0000h base + 20h offset = E008_0020h

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

* Notes:
- x = Undefined at reset.

MCM_FADR field descriptions

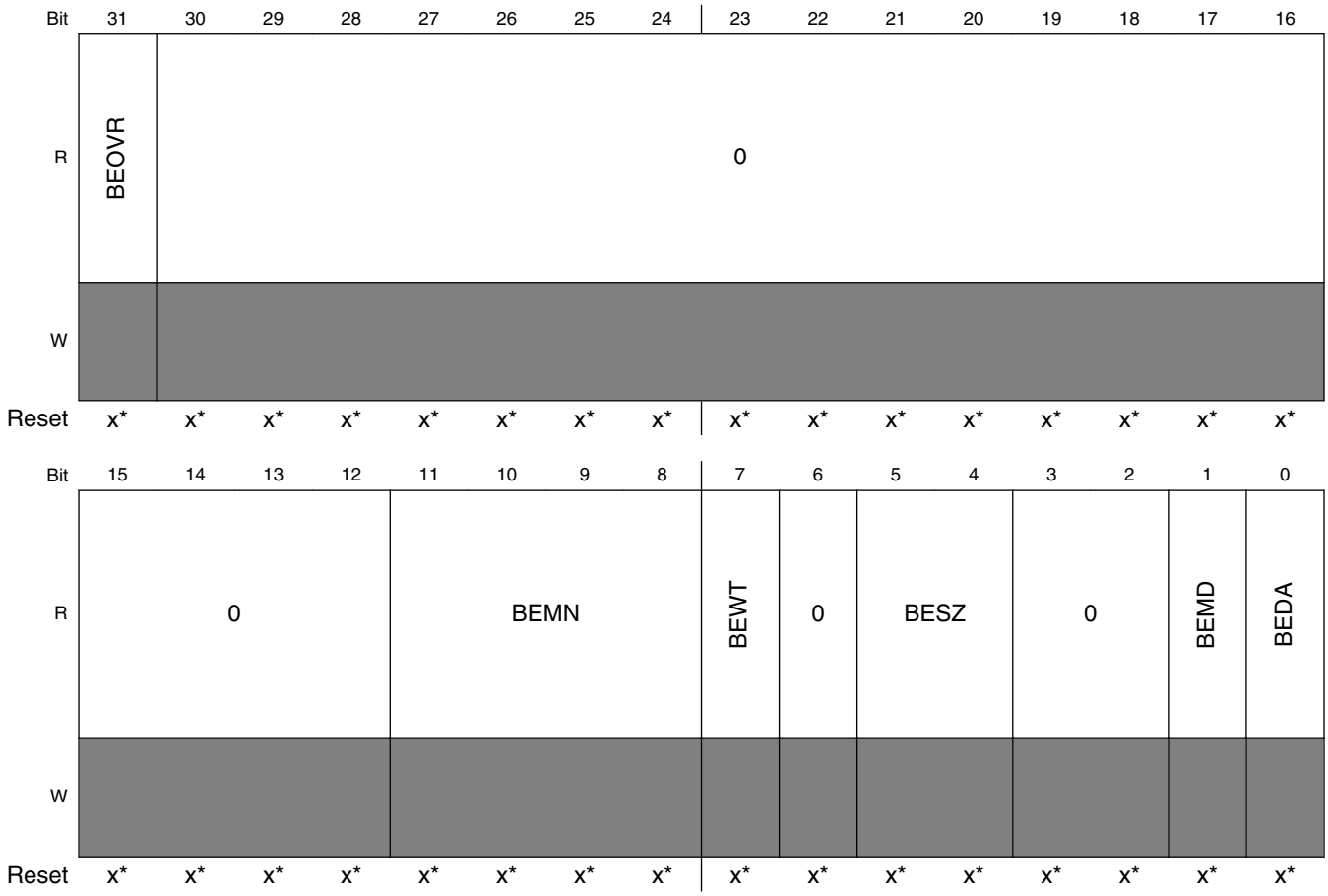
Field	Description
ADDRESS	Fault address

6.2.6 Store Buffer Fault Attributes register (MCM_FATR)

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

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes have no effect.

Address: E008_0000h base + 24h offset = E008_0024h



* Notes:
- x = Undefined at reset.

MCM_FATR field descriptions

Field	Description
31 BEOVR	Bus error overrun Indicates if another cache write buffer bus error is detected before system software has retrieved all the error information from the original event. The window of time is defined from the detection of the original cache write buffer error termination until the MCM_ISCR[CWBER] is written with a 1 to clear it and rearm the capture logic. This bit is set by the hardware and cleared whenever software writes a 1 to the CWBER bit. 0 No bus error overrun 1 Bus error overrun occurred. The FADR and FDR registers and the other FATR bits are not updated to reflect this new bus error.
30–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 BEMN	Bus error master number Crossbar switch bus master number of the captured cache write buffer bus error. For this device, this value is always 0x1.

Table continues on the next page...

MCM_FATR field descriptions (continued)

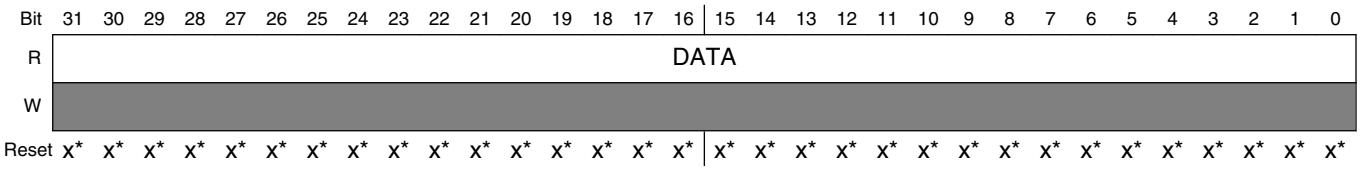
Field	Description
7 BEWT	Bus error write Indicates the type of system bus access when the error was detected. Since this logic is monitoring data transfers from the cache write buffer, this bit is always a logical one, signaling a write operation. 0 Read access 1 Write access
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 BESZ	Bus error size Indicates the size of the cache write buffer access when the error was detected. 00 8-bit access 01 16-bit access 10 32-bit access 11 Reserved
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 BEMD	Bus error privilege level Indicates the privilege level of the cache write buffer access when the error was detected. 0 User mode 1 Supervisor/privileged mode
0 BEDA	Bus Error Data Access type Indicates the type of cache write buffer access when the error was detected. This attribute is always a logical one signaling a data reference. 0 Instruction 1 Data

6.2.7 Store Buffer Fault Data Register (MCM_FDR)

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

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. For byte and halfword writes, only the accessed byte lanes contain valid data; the contents of the other bytes are undefined. Attempted writes have no effect.

Address: E008_0000h base + 28h offset = E008_0028h



* Notes:
- x = Undefined at reset.

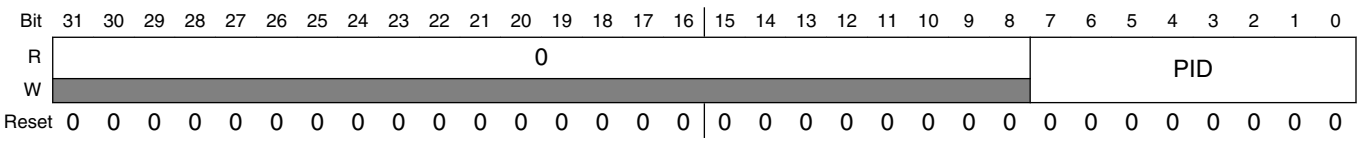
MCM_FDR field descriptions

Field	Description
DATA	Fault data

6.2.8 Process ID register (MCM_PID)

This register drives the M0_PID and M1_PID values 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: E008_0000h base + 30h offset = E008_0030h



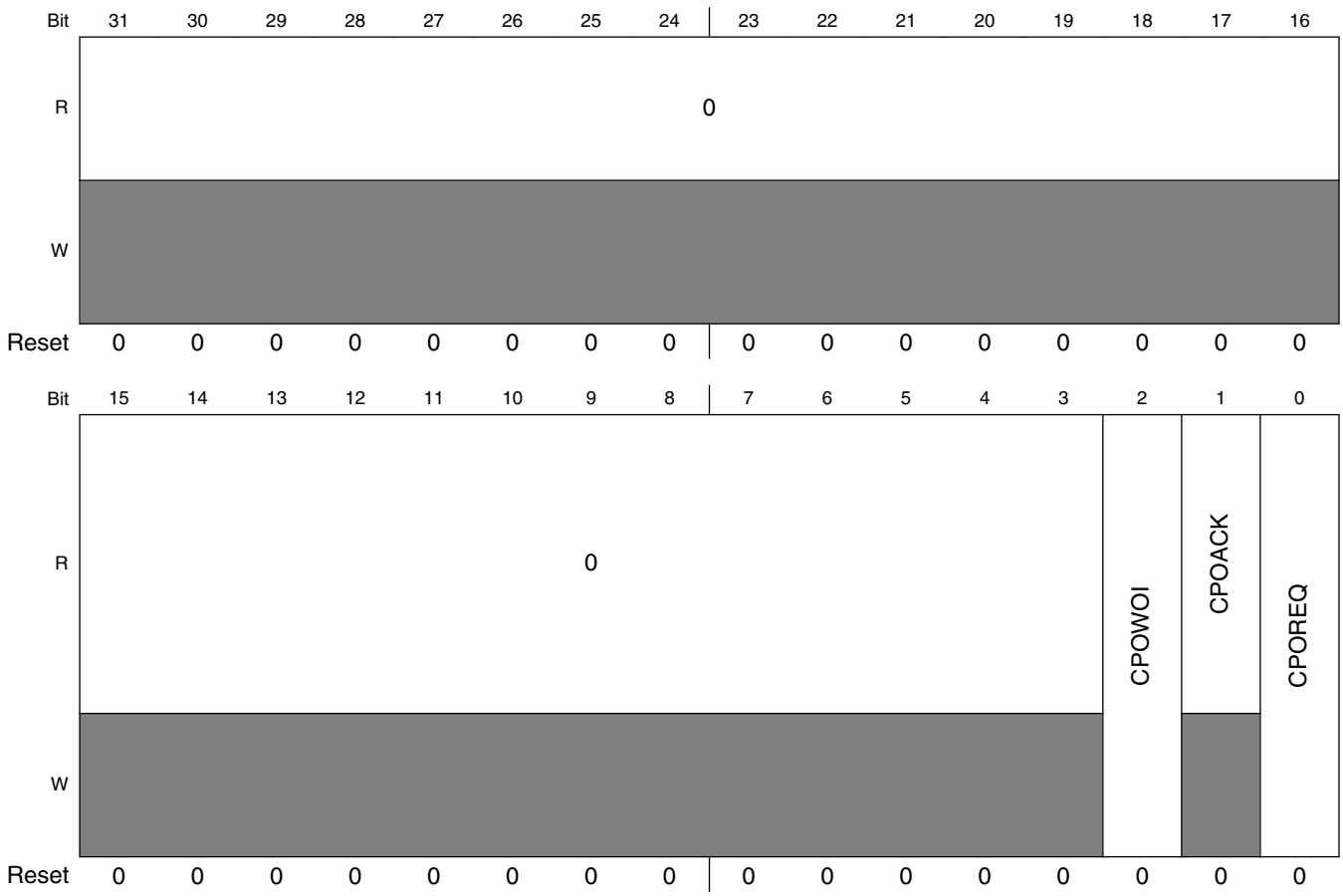
MCM_PID field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PID	M0_PID And M1_PID For MPU Drives the M0_PID and M1_PID values in the MPU.

6.2.9 Compute Operation Control Register (MCM_CPO)

This register controls the Compute Operation.

Address: E008_0000h base + 40h offset = E008_0040h



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 wakeup 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.

6.2.10 Local Memory Descriptor Register (MCM_LMDR*n*)

NOTE

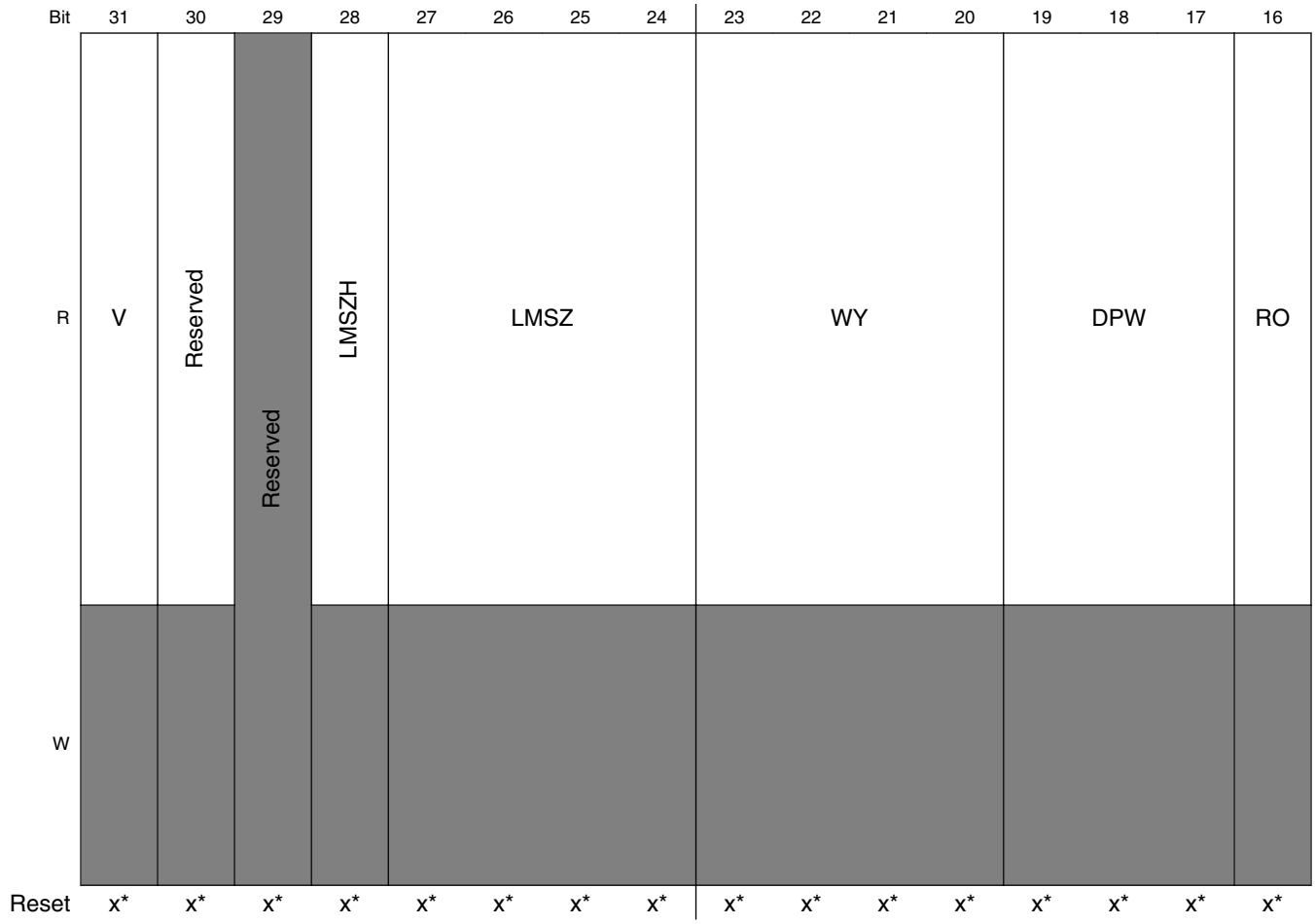
The LMDRn registers mapping to the LMEMs is as follows:

- LMDR0: SRAM_L
- LMDR1: SRAM_U
- LMDR2: PC CACHE

This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information on the attached memories as well as configurable controls (where appropriate).

Privileged 32-bit reads from a processor core or the debugger return the appropriate processor information. Reads from any other bus master return all zeroes. Privileged writes from a processor core or the debugger to writeable registers update the appropriate fields. Privileged writes from other bus masters are ignored. Attempted user mode accesses or any access with a size other than 32 bits are terminated with an error.

Address: E008_0000h base + 400h offset + (4d × i), where i=0d to 2d



* Notes:

- The reset values are different for the individual LMDR registers. LMDR0: 0x8604_0003; LMDR1: 0x8604_2003; LMDR2: 0x8424_40A0. x = Undefined at reset.

MCM_LMDR*n* field descriptions

Field	Description
31 V	Local memory Valid bit. This read-only field defines the validity (presence) of the local memory. 0 LMEMn is not present. 1 LMEMn is present.
30 Reserved	This field is reserved.
29 Reserved	This field is reserved.
28 LMSZH	LMEM Size “Hole”. For local memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0 LMEMn is a power-of-2 capacity. 1 LMEMn is not a power-of-2, with a capacity is 0.75 × LMSZ.
27–24 LMSZ	LMEM Size. This read-only field provides an encoded value of the local memory size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+SZ)}$ where SZ is non-zero; a SZ = 0 indicates the memory is not present. 0000 no LMEMn (0 KB) 0001 1 KB LMEMn 0010 2 KB LMEMn 0011 4 KB LMEMn 0100 8 KB LMEMn 0101 16 KB LMEMn 0110 32 KB LMEMn 0111 64 KB LMEMn 1000 128 KB LMEMn 1001 256 KB LMEMn 1010 512 KB LMEMn 1011 1024 KB LMEMn 1100 2048 KB LMEMn 1101 4096 KB LMEMn 1110 8192 KB LMEMn 1111 16384 KB LMEMn
23–20 WY	Level 1 Cache Ways 0000 No Cache 0010 2-Way Set Associative 0100 4-Way Set Associative
19–17 DPW	LMEM Data Path Width. This read-only field defines the width of the local memory. 000-001 Reserved 010 LMEMn 32-bits wide 011 LMEMn 64-bits wide 100-111 Reserved

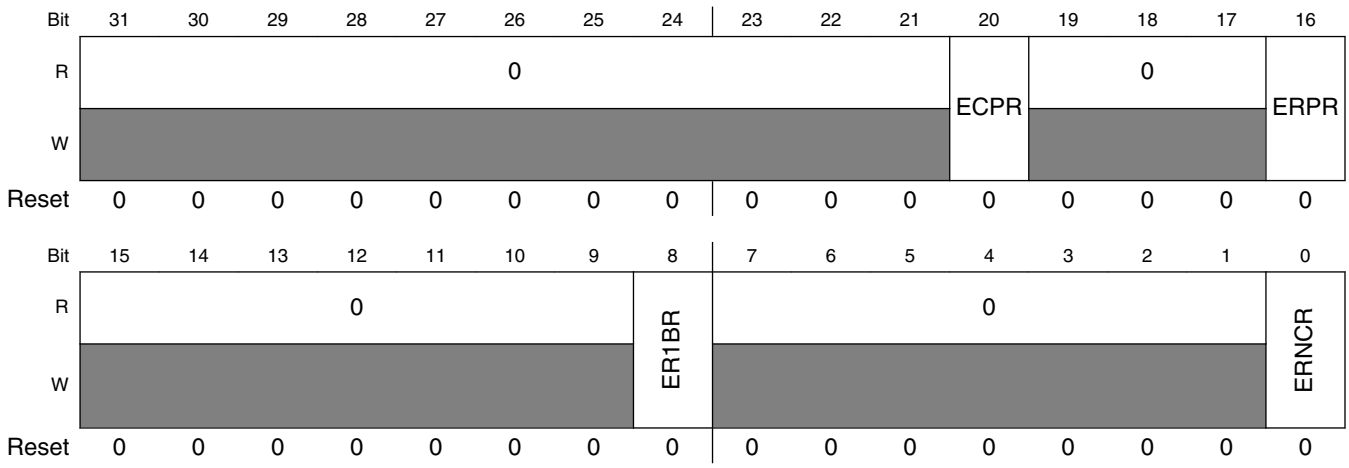
Table continues on the next page...

MCM_LMDR*n* field descriptions (continued)

Field	Description
16 RO	Read-Only. This register bit provides a mechanism to “lock” the configuration state defined by LMDRn[7:0]. Once asserted, attempted writes to the LMDRn[7:0] register are ignored until the next reset clears the flag. 0 Writes to the LMDRn[7:0] are allowed. 1 Writes to the LMDRn[7:0] are ignored.
15–13 MT	Memory Type This field defines the type of the local memory. 000 SRAM_L 001 SRAM_U 010 PC Cache 011 PS Cache
12 Reserved	This field is reserved.
11–8 Reserved	This field is reserved.
7–4 CF1	Control Field 1 - for Cache Parity control functions • CF1[3] - PCPFE = PC Parity Fault Enable • CF1[2] - PSPFE = PS Parity Fault Enable • CF1[1] - PCPME = PC Parity Miss Enable • CF1[0] - PSPME = PS Parity Miss Enable
CF0	Control Field 0 - for TCM ECC control functions • CF0[3] - PFE = Parity Fault Enable • CF0[2] - RESERVED • CF0[1] - EERC = ECC Enable Read Check • CF0[0] - EEWG = ECC Enable Write Generation

6.2.11 LMEM Parity & ECC Control Register (MCM_LMPECR)

Address: E008_0000h base + 480h offset = E008_0480h



MCM_LMPECR field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 ECPR	Enable Cache Parity Reporting 1 reporting enabled 0 reporting disabled
19–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ERPR	Enable RAM Parity Reporting 1 reporting enabled 0 reporting disabled
15–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 ER1BR	Enable RAM ECC 1 Bit Reporting 1 reporting enabled 0 reporting disabled
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 ERNCR	Enable RAM ECC Noncorrectable Reporting 1 reporting enabled 0 reporting disabled

6.2.12 LMEM Parity & ECC Interrupt Register (MCM_LMPEIR)

NOTE

Write 1 to the error bit in MCM_LMPEIR[23:0] can clear the interrupt flag.

Address: E008_0000h base + 488h offset = E008_0488h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	V	0		PEELOC					PE							
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	E1B								ENC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCM_LMPEIR field descriptions

Field	Description
31 V	Valid bit
30–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28–24 PEELOC	Parity or ECC Error Location • 5'h00 - a non-correctable ECC event from SRAM_L • 5'h01 - a non-correctable ECC event from SRAM_U • 5'h08 - a 1-bit correctable ECC event from SRAM_L • 5'h09 - a 1-bit correctable ECC event from SRAM_U • 5'h14 - a PC Tag Parity Error • 5'h15 - a PC Data Parity Error
23–16 PE	Parity Error • [21] - PC Data Parity Error • [20] - PC Tag Parity Error • [19] - RESERVED • [18] - RESERVED
15–8 E1B	E1Bn = ECC 1-bit Error n • PEIR[15:10] - Reserved • PEIR[9] - 1-bit Error detected on SRAM_U • PEIR[8] - 1-bit Error detected on SRAM_L
ENC	ENCn = ECC Noncorrectable Error n

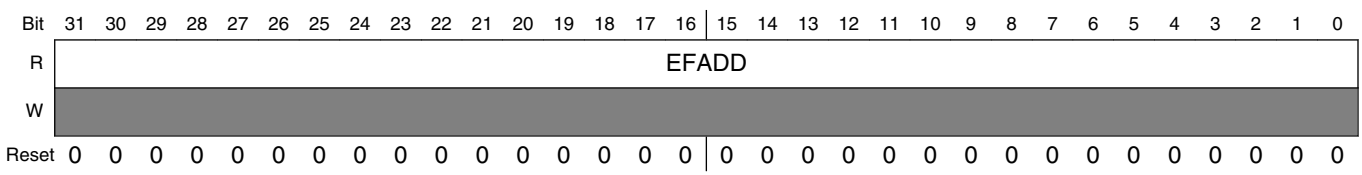
Table continues on the next page...

MCM_LMPEIR field descriptions (continued)

Field	Description
	• PEIR[7:2] - Reserved • PEIR[1] - Noncorrectable Error detected on SRAM_U • PEIR[0] - Noncorrectable Error detected on SRAM_L

6.2.13 LMEM Fault Address Register (MCM_LMFAR)

Address: E008_0000h base + 490h offset = E008_0490h

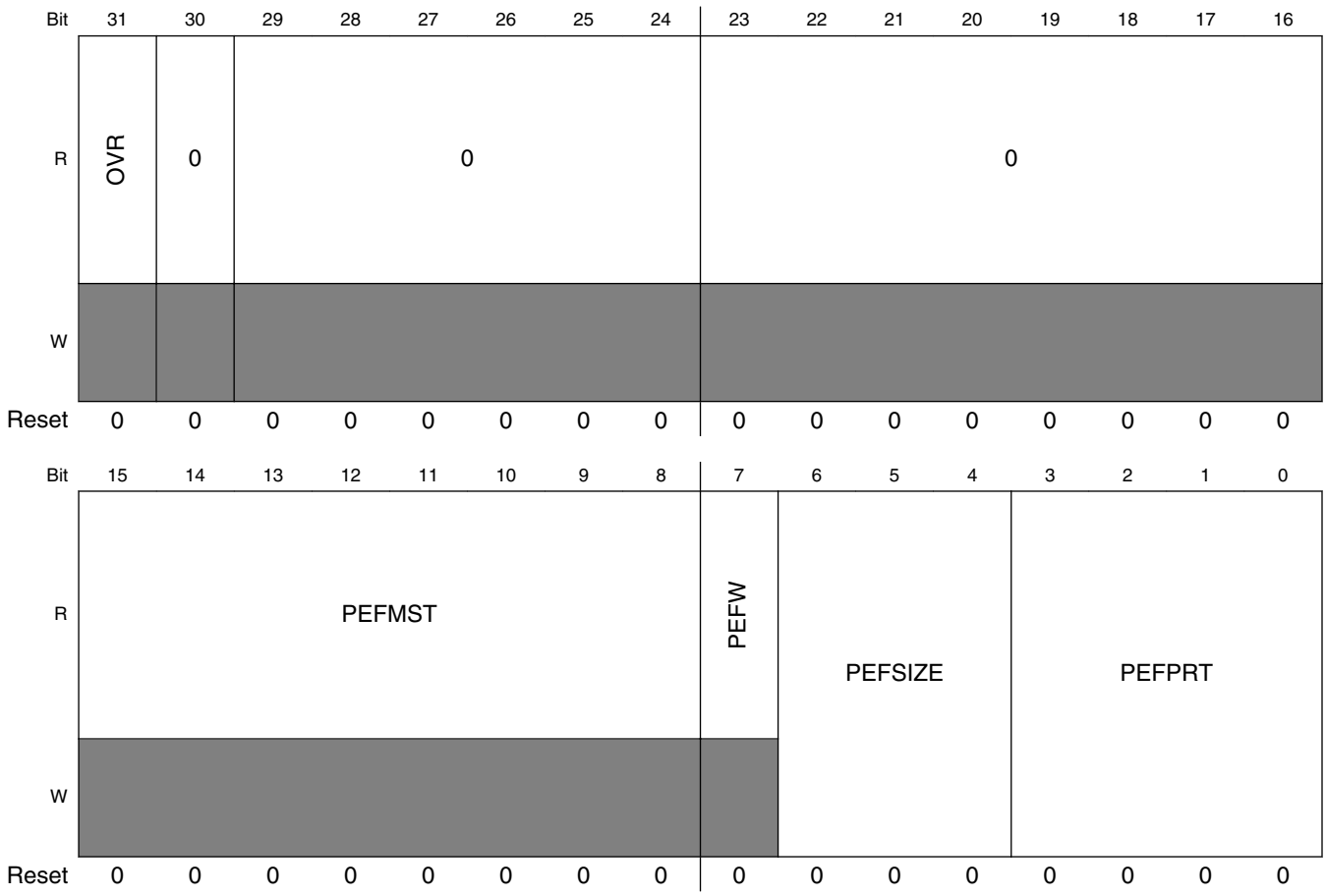


MCM_LMFAR field descriptions

Field	Description
EFADD	ECC Fault Address

6.2.14 LMEM Fault Attribute Register (MCM_LMFATR)

Address: E008_0000h base + 494h offset = E008_0494h



MCM_LMFATR field descriptions

Field	Description
31 OVR	Overrun
30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 PEFMST	Parity/ECC Fault Master Number
7 PEFW	Parity/ECC Fault Write

Table continues on the next page...

MCM_LMFATR field descriptions (continued)

Field	Description
6–4 PEFSIZE	Parity/ECC Fault Master Size • 3'b000 = 8-bit access • 3'b001 = 16-bit access • 3'b010 = 32-bit access • 3'b011 = 64-bit access • 3'b1xx = Reserved
PEFPRT	Parity/ECC Fault Protection • FATR[3] is Cacheable: 0=Non-cacheable, 1=Cacheable • FATR[2] is Bufferable: 0=Non-bufferable, 1=Bufferable • FATR[1] is Mode: 0=User mode, 1=Supervisor mode • FATR[0] is Type: 0=I-Fetch, 1=Data

6.2.15 LMEM Fault Data High Register (MCM_LMFDHR)

Address: E008_0000h base + 4A0h offset = E008_04A0h

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

MCM_LMFDHR field descriptions

Field	Description
PEFDH	Parity or ECC Fault Data High

6.2.16 LMEM Fault Data Low Register (MCM_LMFDLR)

Address: E008_0000h base + 4A4h offset = E008_04A4h

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

MCM_LMFDLR field descriptions

Field	Description
PEFDL	Parity or ECC Fault Data Low

MCM_LMFDLR field descriptions (continued)

Field	Description

6.3 Functional description

This section describes the functional description of MCM module.

6.3.1 Interrupts

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

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC)
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC)
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC)
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC)
- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC)
- FPU invalid operation interrupt is enabled (FIOCE) and an invalid occurs (FIOC)

- SRAM_L correctable(1-bit) ECC error
- SRAM_L uncorrectable ECC error
- SRAM_U correctable (1-bit) ECC error
- SRAM_U uncorrectable ECC error
- PC data parity error
- PC tag parity error
- Cache write buffer error

6.3.1.1 Determining source of the interrupt

To determine the exact source of the interrupt qualify the interrupt status flags with the corresponding interrupt enable bits.

1. From MCM_ISCR[31:16] && MCM_ISCR[15:0]
2. Search the result for asserted flags, which indicate the exact interrupt sources

NOTE

ECC and Parity interrupts are determined by LMEPECR (interrupt enable) and LMPEIR (interrupt source).

Chapter 7
Crossbar Switch Lite (AXBS-Lite)

7.1 Chip-specific information for this module

7.1.1 Instantiation Information

7.1.1.1 Crossbar Switch Master Assignments

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

Master module	Master port number
ARM core code bus	0
ARM core system bus	1
DMA	2

7.1.1.2 Crossbar Switch Slave Assignments

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

Slave module	Slave port number	Protected by MPU?
Flash memory controller	0	Yes
SRAM controllers	1	Yes
Peripheral bridge 0 / GPIO[1]	2	No. Protection built into Peripheral Bridge (AIPS).

1. See System memory map for access restrictions.

7.1.1.3 Information of crossbar arbitration scheme on this device

NOTE

The selection of the global slave port arbitration is controlled in the MCM module. For fixed priority, set MCM_CPCR[CBRR] to 0. For round robin, set MCM_CPCR[CBRR] to 1. This arbitration setting applies to all slave ports.

7.2 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.

7.2.1 Features

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves
- Up to single-clock 32-bit transfer
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).

7.3 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.

7.4 Functional Description

7.4.1 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.

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.

7.4.2 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round-robin

The selection of the global slave port arbitration algorithm is described in the crossbar switch chip-specific information.

7.4.2.1 Arbitration during undefined length bursts

All lengths of burst accesses lock out arbitration until the last beat of the burst.

7.4.2.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (for example, in a system with 5 masters, master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the proper master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port:

Table 7-1. 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
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

7.4.2.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and master 0, 4, and 5 make simultaneous requests, they are serviced in the order: 4 then 5 then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) as the fixed master priority does not affect the master selection.

7.5 Initialization/application information

No initialization is required for the crossbar switch. See the chip-specific crossbar switch information for the reset state of the arbitration scheme.

Chapter 8
Memory Protection Unit (MPU)

8.1 Chip-specific information for this module

8.1.1 Instantiation Information

8.1.1.1 MPU Slave Port Assignments

The memory-mapped resources protected by the MPU are:

Table 8-1. MPU Slave Port Assignments

Source	MPU Slave Port Assignment	Destination
Crossbar slave port 0	MPU slave port 0	Flash Controller and boot ROM
Crossbar slave port 1	MPU slave port 1	SRAM backdoor
Code Bus	MPU slave port 2	all code bus address (0x0000_0000 to 0x1FFF_FFFF)
System Bus	MPU slave port 3	all system bus address (0x2000_0000 to 0xFFFF_FFFF)

8.1.1.2 MPU Logical Bus Master Assignments

The logical bus master assignments for the MPU are:

Table 8-2. MPU Logical Bus Master Assignments

MPU Logical Bus Master Number	Bus Master
0	Core
1	Debugger
2	DMA

8.2 Introduction

The memory protection unit (MPU) provides hardware access control for all memory references generated in the device.

8.3 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.

8.3.1 Block diagram

A simplified block diagram of the MPU module is shown in the following figure.

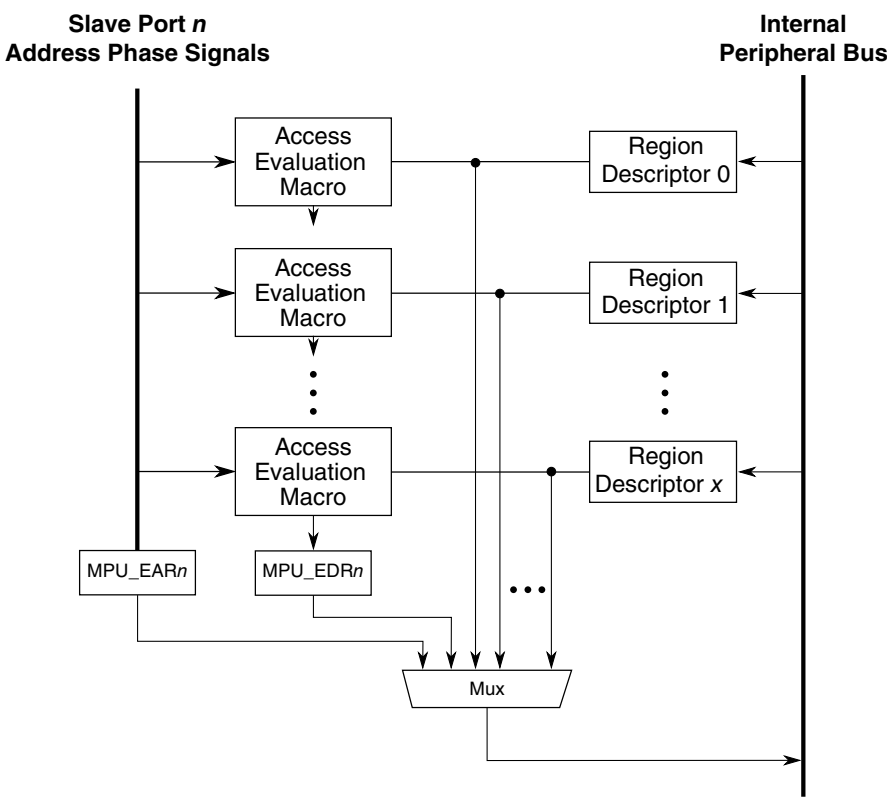


Figure 8-1. MPU block diagram

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.

8.3.2 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

8.4 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 be referenced using only 32-bit accesses. The programming model can be accessed only in supervisor mode.

Attempted references of the following types generate an error termination:

- Non-32-bit references
- Accesses in user mode
- References to undefined—that is, reserved—addresses
- References with a non-supported access type, such as a write access to a read-only register or a read access of a write-only register

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_D000	Control/Error Status Register (MPU_CESR)	32	R/W	0081_4001h	8.4.1/120
4000_D010	Error Address Register, slave port n (MPU_EAR0)	32	R	0000_0000h	8.4.2/121
4000_D014	Error Detail Register, slave port n (MPU_EDR0)	32	R	0000_0000h	8.4.3/122
4000_D018	Error Address Register, slave port n (MPU_EAR1)	32	R	0000_0000h	8.4.2/121
4000_D01C	Error Detail Register, slave port n (MPU_EDR1)	32	R	0000_0000h	8.4.3/122
4000_D020	Error Address Register, slave port n (MPU_EAR2)	32	R	0000_0000h	8.4.2/121
4000_D024	Error Detail Register, slave port n (MPU_EDR2)	32	R	0000_0000h	8.4.3/122
4000_D028	Error Address Register, slave port n (MPU_EAR3)	32	R	0000_0000h	8.4.2/121
4000_D02C	Error Detail Register, slave port n (MPU_EDR3)	32	R	0000_0000h	8.4.3/122
4000_D400	Region Descriptor n, Word 0 (MPU_RGD0_WORD0)	32	R/W	0000_0000h	8.4.4/123

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D404	Region Descriptor n, Word 1 (MPU_RGD0_WORD1)	32	R/W	See section	8.4.5/124
4000_D408	Region Descriptor n, Word 2 (MPU_RGD0_WORD2)	32	R/W	See section	8.4.6/124
4000_D40C	Region Descriptor n, Word 3 (MPU_RGD0_WORD3)	32	R/W	See section	8.4.7/127
4000_D410	Region Descriptor n, Word 0 (MPU_RGD1_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D414	Region Descriptor n, Word 1 (MPU_RGD1_WORD1)	32	R/W	See section	8.4.5/124
4000_D418	Region Descriptor n, Word 2 (MPU_RGD1_WORD2)	32	R/W	See section	8.4.6/124
4000_D41C	Region Descriptor n, Word 3 (MPU_RGD1_WORD3)	32	R/W	See section	8.4.7/127
4000_D420	Region Descriptor n, Word 0 (MPU_RGD2_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D424	Region Descriptor n, Word 1 (MPU_RGD2_WORD1)	32	R/W	See section	8.4.5/124
4000_D428	Region Descriptor n, Word 2 (MPU_RGD2_WORD2)	32	R/W	See section	8.4.6/124
4000_D42C	Region Descriptor n, Word 3 (MPU_RGD2_WORD3)	32	R/W	See section	8.4.7/127
4000_D430	Region Descriptor n, Word 0 (MPU_RGD3_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D434	Region Descriptor n, Word 1 (MPU_RGD3_WORD1)	32	R/W	See section	8.4.5/124
4000_D438	Region Descriptor n, Word 2 (MPU_RGD3_WORD2)	32	R/W	See section	8.4.6/124
4000_D43C	Region Descriptor n, Word 3 (MPU_RGD3_WORD3)	32	R/W	See section	8.4.7/127
4000_D440	Region Descriptor n, Word 0 (MPU_RGD4_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D444	Region Descriptor n, Word 1 (MPU_RGD4_WORD1)	32	R/W	See section	8.4.5/124
4000_D448	Region Descriptor n, Word 2 (MPU_RGD4_WORD2)	32	R/W	See section	8.4.6/124
4000_D44C	Region Descriptor n, Word 3 (MPU_RGD4_WORD3)	32	R/W	See section	8.4.7/127
4000_D450	Region Descriptor n, Word 0 (MPU_RGD5_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D454	Region Descriptor n, Word 1 (MPU_RGD5_WORD1)	32	R/W	See section	8.4.5/124
4000_D458	Region Descriptor n, Word 2 (MPU_RGD5_WORD2)	32	R/W	See section	8.4.6/124
4000_D45C	Region Descriptor n, Word 3 (MPU_RGD5_WORD3)	32	R/W	See section	8.4.7/127
4000_D460	Region Descriptor n, Word 0 (MPU_RGD6_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D464	Region Descriptor n, Word 1 (MPU_RGD6_WORD1)	32	R/W	See section	8.4.5/124
4000_D468	Region Descriptor n, Word 2 (MPU_RGD6_WORD2)	32	R/W	See section	8.4.6/124
4000_D46C	Region Descriptor n, Word 3 (MPU_RGD6_WORD3)	32	R/W	See section	8.4.7/127
4000_D470	Region Descriptor n, Word 0 (MPU_RGD7_WORD0)	32	R/W	0000_0000h	8.4.4/123
4000_D474	Region Descriptor n, Word 1 (MPU_RGD7_WORD1)	32	R/W	See section	8.4.5/124
4000_D478	Region Descriptor n, Word 2 (MPU_RGD7_WORD2)	32	R/W	See section	8.4.6/124
4000_D47C	Region Descriptor n, Word 3 (MPU_RGD7_WORD3)	32	R/W	See section	8.4.7/127
4000_D800	Region Descriptor Alternate Access Control n (MPU_RGDAAC0)	32	R/W	See section	8.4.8/128
4000_D804	Region Descriptor Alternate Access Control n (MPU_RGDAAC1)	32	R/W	See section	8.4.8/128
4000_D808	Region Descriptor Alternate Access Control n (MPU_RGDAAC2)	32	R/W	See section	8.4.8/128
4000_D80C	Region Descriptor Alternate Access Control n (MPU_RGDAAC3)	32	R/W	See section	8.4.8/128

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D810	Region Descriptor Alternate Access Control n (MPU_RGDAAC4)	32	R/W	See section	8.4.8/128
4000_D814	Region Descriptor Alternate Access Control n (MPU_RGDAAC5)	32	R/W	See section	8.4.8/128
4000_D818	Region Descriptor Alternate Access Control n (MPU_RGDAAC6)	32	R/W	See section	8.4.8/128
4000_D81C	Region Descriptor Alternate Access Control n (MPU_RGDAAC7)	32	R/W	See section	8.4.8/128

8.4.1 Control/Error Status Register (MPU_CESR)

Address: 4000_D000h base + 0h offset = 4000_D000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SPERR				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	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1

MPU_CESR field descriptions

Field	Description
31–28 SPERR	Slave Port n Error Indicates a captured error in EARn and EDRn. This bit is set when the hardware detects an error and records the faulting address and attributes. It is cleared by writing one to it. If another error is captured at the exact same cycle as the write, the flag remains set. A find-first-one instruction or equivalent can detect the presence of a captured error. The following shows the correspondence between the bit number and slave port number: • Bit 31 corresponds to slave port 0. • Bit 30 corresponds to slave port 1. • Bit 29 corresponds to slave port 2. • Bit 28 corresponds to slave port 3. 0 No error has occurred for slave port n. 1 An error has occurred for slave port n.
27–24 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.

Table continues on the next page...

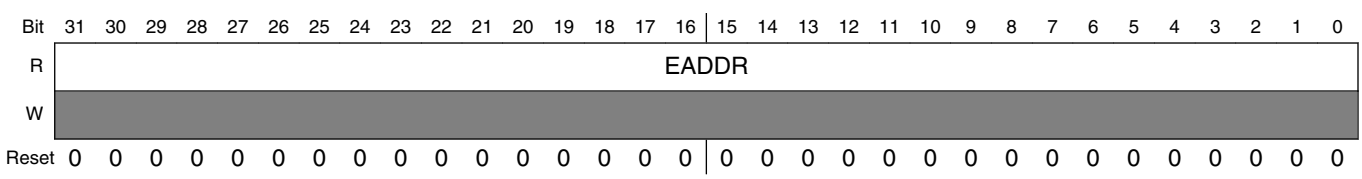
MPU_CESR field descriptions (continued)

Field	Description
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

8.4.2 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[SPERRn] is 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[SPERRn], as the error registers are always loaded upon the occurrence of each protection violation.

Address: 4000_D000h base + 10h offset + (8d × i), where i=0d to 3d



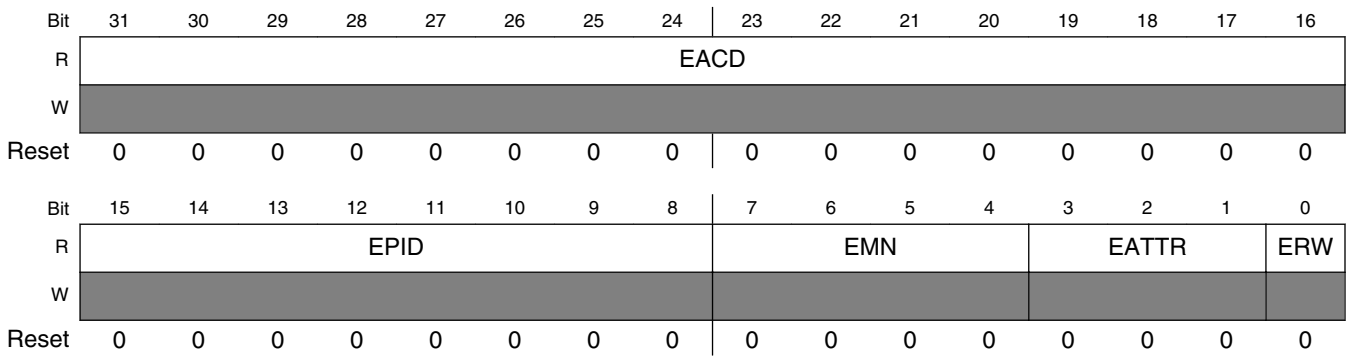
MPU_EAR*n* field descriptions

Field	Description
EADDR	Error Address Indicates the reference address from slave port n that generated the access error

8.4.3 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[SPERRn] 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[SPERRn] as the error registers are always loaded upon the occurrence of each protection violation.

Address: 4000_D000h base + 14h offset + (8d × i), where i=0d to 3d



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.
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

Table continues on the next page...

MPU_EDR*n* field descriptions (continued)

Field	Description
0 ERW	Error Read/Write Indicates the access type of the faulting reference. 0 Read 1 Write

8.4.4 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]).

Address: 4000_D000h 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.

8.4.5 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]).

Address: 4000_D000h 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.

8.4.6 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. Each of these bus masters optionally includes a process identification field (if implemented for the master) 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_D000h 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 0061_F7DFh

  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. In M3UM[2:0]: M3UM[2] controls read permissions, M3UM[1] controls write permissions, and M3UM[0] controls execute 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.

Table continues on the next page...

MPU_RGD*n*_WORD2 field descriptions (continued)

Field	Description
10–9 M1SM	Bus Master 1 Supervisor Mode Access Control See M3SM description.
8–6 M1UM	Bus Master 1 User Mode Access Control See M3UM description.
5 M0PE	Bus Master 0 Process Identifier enable See M0PE description.
4–3 M0SM	Bus Master 0 Supervisor Mode Access Control See M3SM description.
M0UM	Bus Master 0 User Mode Access Control See M3UM description.

8.4.7 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.

Address: 4000_D000h 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.
23–16 PIDMASK	Process Identifier Mask

Table continues on the next page...

MPU_RGD*n*_WORD3 field descriptions (continued)

Field	Description
	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

8.4.8 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_D000h 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 0061_F7DFh

  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. In M3UM[2:0]: M3UM[2] controls read permissions, M3UM[1] controls write permissions, and M3UM[0] controls execute permissions.

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.

8.5 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.

8.5.1 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.

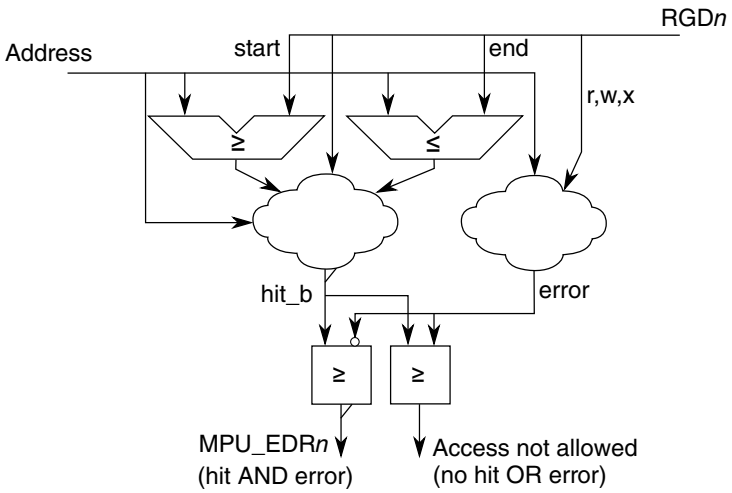


Figure 8-2. MPU access evaluation macro

8.5.1.1 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_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.

8.5.1.2 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 8-3. 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

8.5.2 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.

8.5.3 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.

8.6 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.

8.7 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 8-4. 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.

8.8 Usage Guide

8.8.1 MPU Access Violation Indications

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

Table 8-5. Access Violation Indications

Bus Master	Core Indication
Core	Bus fault (interrupt vector #5)

Table continues on the next page...

Table 8-5. Access Violation Indications (continued)

Bus Master	Core Indication
	NOTE: To enable bus faults, set the core's System Handler Control and State Register's BUSFAULTENA bit. If this bit is not set, MPU violations result in a hard fault (interrupt vector #3).
Debugger	The STICKY ERROR flag is set in the Debug Port Control/Status Register, see JTAG status and control registers.
DMA	Interrupt vector #32

8.8.2 Reset Values for RGD0 Registers

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

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

Table 8-6. Reset Values for RGD0 Registers

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

8.8.3 Write Access Restrictions for RGD0 Registers

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

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

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

Table 8-7. Write Access to RGD0 Registers

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

Chapter 9
Peripheral Bridge (AIPS-Lite)

9.1 Chip-specific information for this module

9.1.1 Peripheral slot assignment

The peripheral bridge is used to access the registers of most of the modules on this device. See Peripheral Bridge (AIPS-Lite) Memory Map for the memory slot assignment.

The following table shows the mapping between on/off-platform Peripheral Access Control Register and Peripheral bridge slot assignment. For example, AIPS_OPACR0 is correspoding to slot 32 (Flash memory).

Peripheral Access Control Register	Peripheral bridge slot number
AIPS_PACR*n*	*n*
AIPS_OPACR*n*	*n*+32

9.2 Introduction

The peripheral bridge converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

The peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB. (It might be possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

9.2.1 Features

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Programming model provides memory protection functionality

9.2.2 General operation

The slave devices connected to the peripheral bridge are modules which contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map. Two global external module enables are available for the remaining address space to allow for customization and expansion of addressed peripheral devices.

9.3 Memory map/register definition

The 32-bit peripheral bridge registers can be accessed only in supervisor mode by trusted bus masters. 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_0000	Master Privilege Register A (AIPS_MPRA)	32	R/W	See section	9.3.1/141
4000_0020	Peripheral Access Control Register (AIPS_PACRA)	32	R/W	See section	9.3.2/144
4000_0024	Peripheral Access Control Register (AIPS_PACRB)	32	R/W	See section	9.3.2/144
4000_0028	Peripheral Access Control Register (AIPS_PACRC)	32	R/W	See section	9.3.2/144
4000_002C	Peripheral Access Control Register (AIPS_PACRD)	32	R/W	See section	9.3.2/144
4000_0040	Off-Platform Peripheral Access Control Register (AIPS_OPACRA)	32	R/W	See section	9.3.3/149
4000_0044	Off-Platform Peripheral Access Control Register (AIPS_OPACRB)	32	R/W	See section	9.3.3/149
4000_0048	Off-Platform Peripheral Access Control Register (AIPS_OPACRC)	32	R/W	See section	9.3.3/149

Table continues on the next page...

AIPS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_004C	Off-Platform Peripheral Access Control Register (AIPS_OPACRD)	32	R/W	See section	9.3.3/149
4000_0050	Off-Platform Peripheral Access Control Register (AIPS_OPACRE)	32	R/W	See section	9.3.3/149
4000_0054	Off-Platform Peripheral Access Control Register (AIPS_OPACRF)	32	R/W	See section	9.3.3/149
4000_0058	Off-Platform Peripheral Access Control Register (AIPS_OPACRG)	32	R/W	See section	9.3.3/149
4000_005C	Off-Platform Peripheral Access Control Register (AIPS_OPACRH)	32	R/W	See section	9.3.3/149
4000_0060	Off-Platform Peripheral Access Control Register (AIPS_OPACRI)	32	R/W	See section	9.3.3/149
4000_0064	Off-Platform Peripheral Access Control Register (AIPS_OPACRJ)	32	R/W	See section	9.3.3/149
4000_0068	Off-Platform Peripheral Access Control Register (AIPS_OPACRK)	32	R/W	See section	9.3.3/149
4000_006C	Off-Platform Peripheral Access Control Register (AIPS_OPACRL)	32	R/W	See section	9.3.3/149
4000_0080	Peripheral Access Control Register (AIPS_PACRU)	32	R/W	See section	9.3.4/154

9.3.1 Master Privilege Register A (AIPS_MPRA)

The MPRA specifies identical 4-bit fields defining the access-privilege level associated with a bus master to various peripherals on the chip. The register provides one field per bus master.

NOTE

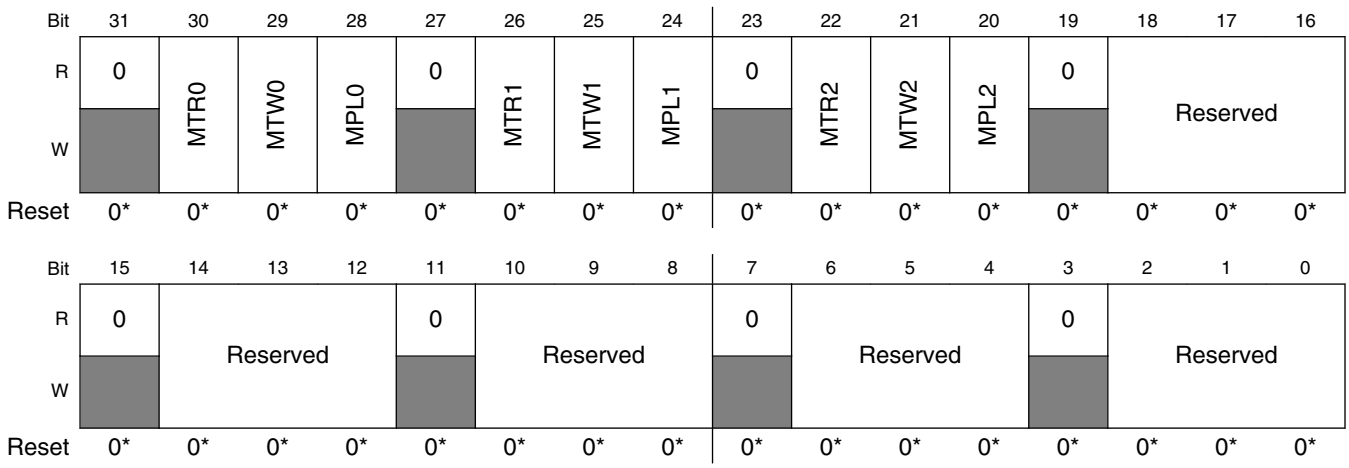
At reset, the default value loaded into the MPRA fields is chip-specific. See the chip configuration details for the value of a particular device.

A register field that maps to an unimplemented master or peripheral behaves as read-only-zero.

Each master is assigned a logical ID from 0 to 15. See the master logical ID assignment table in the chip-specific AIPS information.

Memory map/register definition

Address: 4000_0000h base + 0h offset = 4000_0000h



* Notes:
- The reset value is chip-dependent and can be found in the chip-specific AIPS information.

AIPS_MPRA field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 MTR0	Master 0 Trusted For Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
29 MTW0	Master 0 Trusted For Writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
28 MPL0	Master 0 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 MTR1	Master 1 Trusted for Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
25 MTW1	Master 1 Trusted for Writes Determines whether the master is trusted for write accesses.

Table continues on the next page...

AIPS_MPRA field descriptions (continued)

Field	Description
	0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
24 MPL1	Master 1 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 MTR2	Master 2 Trusted For Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
21 MTW2	Master 2 Trusted For Writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
20 MPL2	Master 2 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 Reserved	This field is reserved.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 Reserved	This field is reserved.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 Reserved	This field is reserved.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved.

9.3.2 Peripheral Access Control Register (AIPS_PACR*n*)

Each PACR register consists of eight 4-bit PACR fields. Each PACR field defines the access levels for a particular on-platform peripheral. The peripheral assignment to each PACR field is defined by the memory map slot of the peripheral. See the chip-specific AIPS information for the field assignment of a particular peripheral.

Every PACR field to which no peripheral is assigned is reserved. Reads to reserved locations return zeros, and writes are ignored.

The following table shows the location of each peripheral slot's PACR field in the PACR registers.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x20	PACRA	PACR0	PACR1	PACR2	PACR3	PACR4	PACR5	PACR6	PACR7
0x24	PACRB	PACR8	PACR9	PACR10	PACR11	PACR12	PACR13	PACR14	PACR15
0x28	PACRC	PACR16	PACR17	PACR18	PACR19	PACR20	PACR21	PACR22	PACR23
0x2C	PACRD	PACR24	PACR25	PACR26	PACR27	PACR28	PACR29	PACR30	PACR31

Address: 4000_0000h base + 20h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3
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	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
W																
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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SP0	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SP1	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 SP2	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 SP3	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
17 WP3	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SP4	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 SP5	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
9 WP5	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 SP6	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SP7	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
1 WP7	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

9.3.3 Off-Platform Peripheral Access Control Register (AIPS_OPACR*n*)

Each OPACR register consists of eight 4-bit OPACR fields. Each OPACR field defines the access levels for a particular off-platform peripheral. The peripheral assignment to each OPACR field is defined by the memory map slot of the peripheral. See the chip-specific AIPS information for the field assignment of a particular peripheral.

Every OPACR field to which no peripheral is assigned is reserved. Reads to reserved locations return zeros, and writes are ignored.

The following table shows the location of each peripheral slot's OPACR field in the OPACR registers.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x40	OPACRA	OPACR0	OPACR1	OPACR2	OPACR3	OPACR4	OPACR5	OPACR6	OPACR7
0x44	OPACRB	OPACR8	OPACR9	OPACR10	OPACR11	OPACR12	OPACR13	OPACR14	OPACR15
0x48	OPACRC	OPACR16	OPACR17	OPACR18	OPACR19	OPACR20	OPACR21	OPACR22	OPACR23
0x4C	OPACRD	OPACR24	OPACR25	OPACR26	OPACR27	OPACR28	OPACR29	OPACR30	OPACR31
0x50	OPACRE	OPACR32	OPACR33	OPACR34	OPACR35	OPACR36	OPACR37	OPACR38	OPACR39
0x54	OPACRF	OPACR40	OPACR41	OPACR42	OPACR43	OPACR44	OPACR45	OPACR46	OPACR47
0x58	OPACRG	OPACR48	OPACR49	OPACR50	OPACR51	OPACR52	OPACR53	OPACR54	OPACR55
0x5C	OPACRH	OPACR56	OPACR57	OPACR58	OPACR59	OPACR60	OPACR61	OPACR62	OPACR63
0x60	OPACRI	OPACR64	OPACR65	OPACR66	OPACR67	OPACR68	OPACR69	OPACR70	OPACR71
0x64	OPACRJ	OPACR72	OPACR73	OPACR74	OPACR75	OPACR76	OPACR77	OPACR78	OPACR79
0x68	OPACRK	OPACR80	OPACR81	OPACR82	OPACR83	OPACR84	OPACR85	OPACR86	OPACR87
0x6C	OPACRL	OPACR88	OPACR89	OPACR90	OPACR91	OPACR92	OPACR93	OPACR94	OPACR95

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	0	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3
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	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
W																
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_OPACR*n* field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SP0	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SP1	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

Table continues on the next page...

AIPS_OPACR*n* field descriptions (continued)

Field	Description
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 SP2	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 SP3	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
17 WP3	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

Table continues on the next page...

AIPS_OPACR*n* field descriptions (continued)

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SP4	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 SP5	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
9 WP5	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

Table continues on the next page...

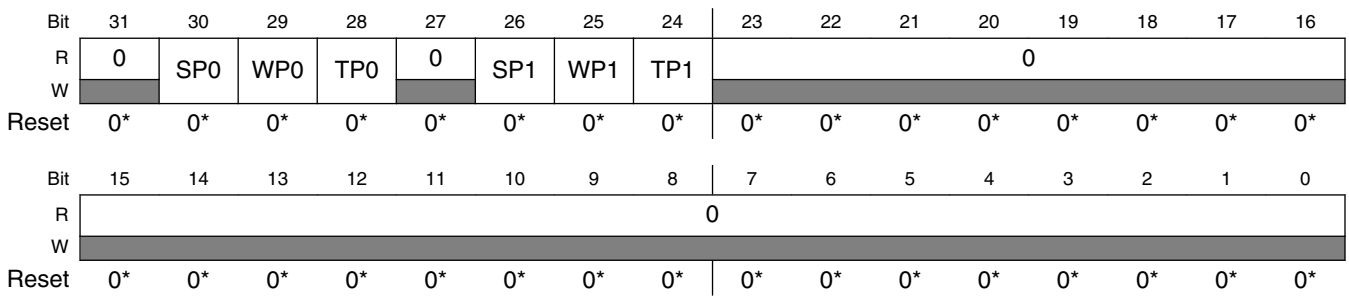
AIPS_OPACR*n* field descriptions (continued)

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 SP6	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SP7	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
1 WP7	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

9.3.4 Peripheral Access Control Register (AIPS_PACRU)

PACRU defines the access levels for the two global spaces.

Address: 4000_0000h base + 80h offset = 4000_0080h



* Notes:
- The reset value is chip-dependent and can be found in the AIPS chip-specific information.

AIPS_PACRU field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SP0	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

AIPS_PACRU field descriptions (continued)

Field	Description
26 SP1	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

9.4 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

9.4.1 Access support

Aligned and misaligned 32-bit, 16-bit, and byte accesses are supported for 32-bit peripherals. Misaligned accesses are supported to allow memory to be placed on the slave peripheral bus. Peripheral registers must not be misaligned, although no explicit checking is performed by the peripheral bridge. All accesses are performed with a single transfer.

All accesses to the peripheral slots must be sized less than or equal to the designated peripheral slot size. If an access is attempted that is larger than the targeted port, an error response is generated.

Chapter 10
Trigger MUX Control (TRGMUX)

10.1 Chip-specific information for this module

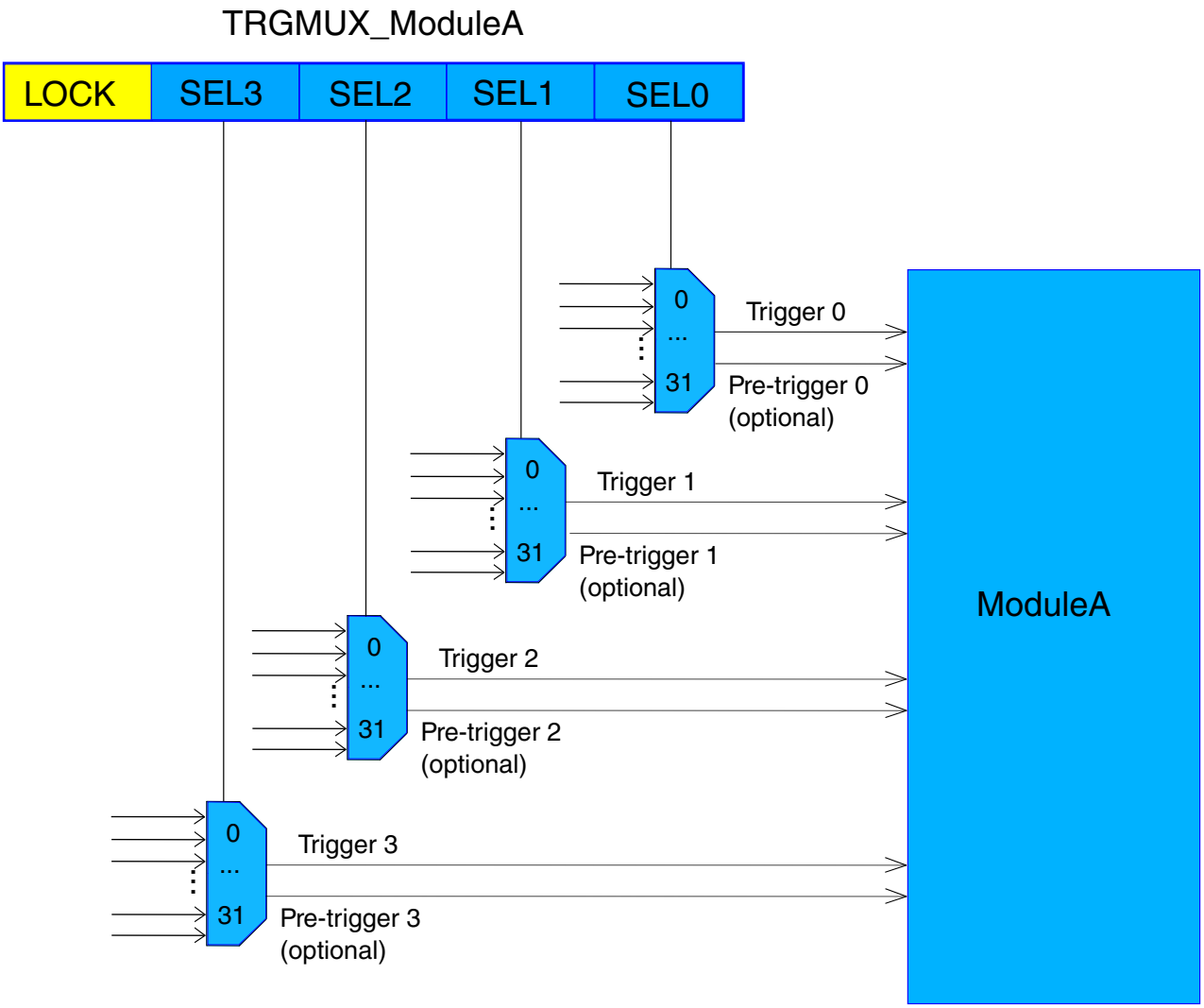
10.1.1 Module Interconnectivity

The module interconnectivity scheme is based on the TRGMUX. The TRGMUX introduces an extremely flexible methodology for connecting various trigger sources to multiple pins/peripherals. This TRGMUX design has removed some trigger inputs, and added one pre-stage trigger source TRGMUX1 for the TRGMUX0. TRGMUX1 supports up to 32 trigger sources and has 8 outputs. These 8 outputs will be the trigger inputs of TRGMUX0. TRGMUX0 supports up to 32 input sources, and its output will be the target modules.

With the TRGMUX, each peripheral which accepts external triggers will usually have one specific 32-bit trigger control register. Each control register supports up to 4 triggers, and each trigger can be selected from up to 32 inputs.

For some trigger sources, there is optional pre-trigger. The trigger and the pre-trigger are 1-1 paired up, and are both selected by the same trigger control register. Not every module has pre-trigger input, please refer to the respective module chapter for details.

Following is the main structure of TRGMUX, and take ModuleA as an example.



NOTE

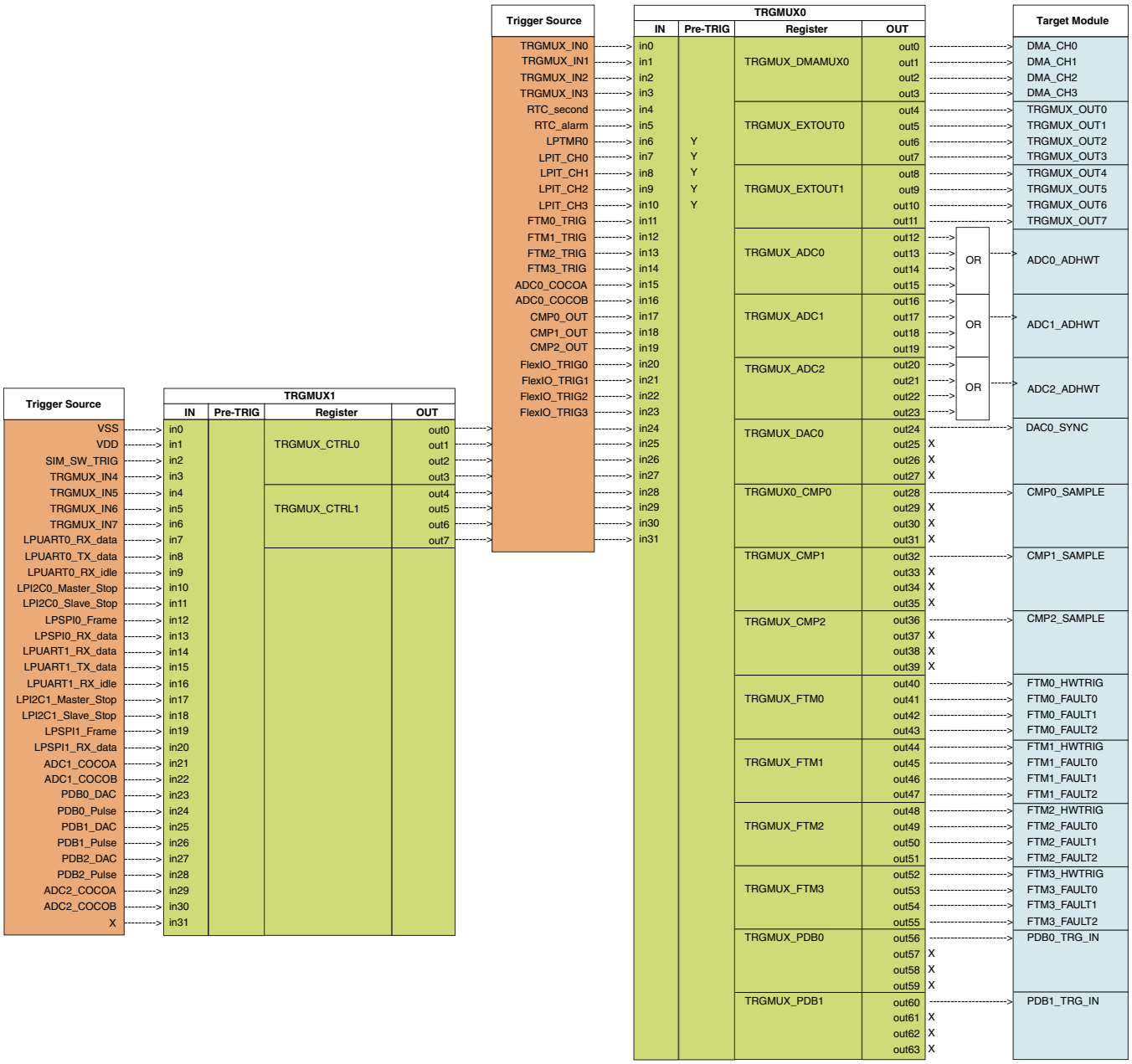
Each TRGMUX control register supports up to 4 trigger channels, but it's not necessary for each module to implement all of the 4 triggers. For those modules (e.g. external output, etc.) which needs more than 4 trigger inputs, multiple control registers are created to support that.

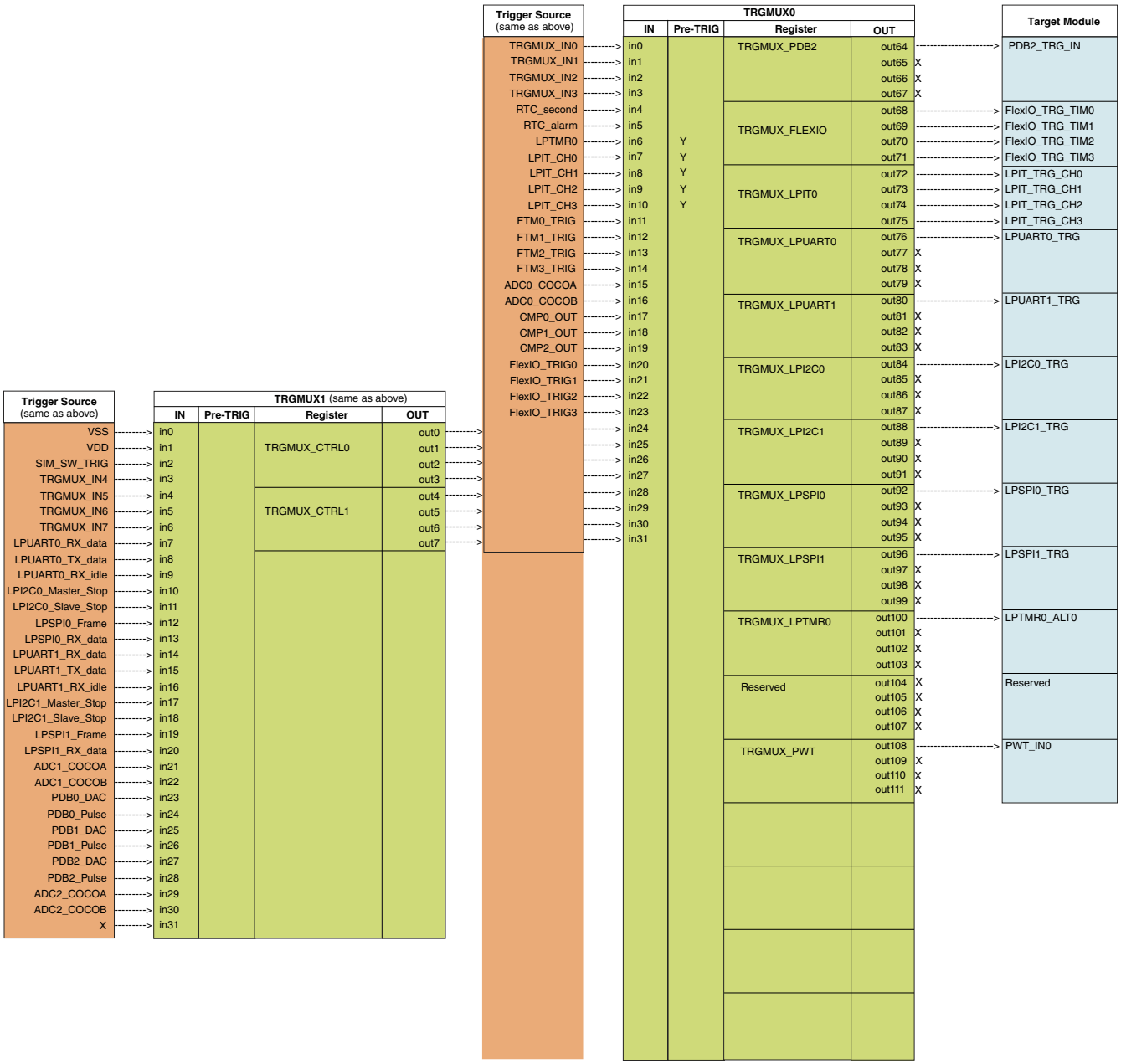
The trigger input and peripheral trigger control are assigned as the following figure indication.

Trigger source	Explanation
VSS	VSS trigger
VDD	VDD trigger
SIM_SW_TRG	Software trigger controlled by SIM module
TRGMUX_INx	TRGMUX external trigger input x
LPUARTx_RX_data	LPUARTx receive end of word trigger

Table continues on the next page...

LPUARTx_TX_data	LPUARTx transmit end of word trigger
LPUARTx_RX_idle	LPUARTx receive idle detected trigger
LPI2Cx_Master_Stop	LPI2Cx master stop or repeated start trigger
LPI2Cx_Slave_Stop	LPI2Cx slave stop or repeated start trigger
LPSPIx_Frame	LPSPIx end of frame trigger
LPSPIx_RX_data	LPSPIx receive data trigger
ADCx_COCOA	ADCx conversion complete trigger for data result A
ADCx_COCOB	ADCx conversion complete trigger for data result B
PDBx_Pulse0	PDBx pulse0 trigger
RTC_second	RTC second trigger
RTC_alarm	RTC alarm trigger
LPTMRx	LPTMRx timer counter match trigger
LPIT_CHx	LPIT channel x timer counter match trigger
FTMx_TRIG	FTMx timer counter match trigger
CMPx_OUT	CMPx output trigger
FlexIO_TRIGx	FlexIO timer x counter match trigger


Trigger Source
VSS
VDD
SIM_SW_TRIG
TRGMUX_IN4
TRGMUX_IN5
TRGMUX_IN6
TRGMUX_IN7
LPUART0_RX_data
LPUART0_TX_data
LPUART0_RX_idle
LPI2C0_Master_Stop
LPI2C0_Slave_Stop
LPSPI0_Frame
LPSPI0_RX_data
LPUART1_RX_data
LPUART1_TX_data
LPUART1_RX_idle
LPI2C1_Master_Stop
LPI2C1_Slave_Stop
LPSPI1_Frame
LPSPI1_RX_data
ADC1_COCOA
ADC1_COCOB
PDB0_DAC
PDB0_Pulse
PDB1_DAC
PDB1_Pulse
PDB2_DAC
PDB2_Pulse
ADC2_COCOA
ADC2_COCOB
X
TRGMUX1
IN
Pre-TRIG
Register
OUT
in0 in1 in2 in3 in4 in5 in6 in7 in8 in9 in10 in11 in12 in13 in14 in15 in16 in17 in18 in19 in20 in21 in22 in23 in24 in25 in26 in27 in28 in29 in30 in31
TRGMUX_CTRL0
TRGMUX_CTRL1
out0 out1 out2 out3 out4 out5 out6 out7
Trigger Source
TRGMUX_IN0
TRGMUX_IN1
TRGMUX_IN2
TRGMUX_IN3
RTC_second
RTC_alarm
LPTMR0
LPIT_CH0
LPIT_CH1
LPIT_CH2
LPIT_CH3
FTM0_TRIG
FTM1_TRIG
FTM2_TRIG
FTM3_TRIG
ADC0_COCOA
ADC0_COCOB
CMP0_OUT
CMP1_OUT
CMP2_OUT
FlexIO_TRIG0
FlexIO_TRIG1
FlexIO_TRIG2
FlexIO_TRIG3
TRGMUX0
IN
Pre-TRIG
Register
OUT
in0 in1 in2 in3 in4 in5 in6 in7 in8 in9 in10 in11 in12 in13 in14 in15 in16 in17 in18 in19 in20 in21 in22 in23 in24 in25 in26 in27 in28 in29 in30 in31
Y Y Y Y Y
TRGMUX_DMAMUX0
TRGMUX_EXTOUT0
TRGMUX_EXTOUT1
TRGMUX_ADC0
TRGMUX_ADC1
TRGMUX_ADC2
TRGMUX_DAC0
TRGMUX0_CMP0
TRGMUX_CMP1
TRGMUX_CMP2
TRGMUX_FTM0
TRGMUX_FTM1
TRGMUX_FTM2
TRGMUX_FTM3
TRGMUX_PDB0
TRGMUX_PDB1
out0 out1 out2 out3 out4 out5 out6 out7 out8 out9 out10 out11 out12 out13 out14 out15 out16 out17 out18 out19 out20 out21 out22 out23 out24 out25 out26 out27 out28 out29 out30 out31 out32 out33 out34 out35 out36 out37 out38 out39 out40 out41 out42 out43 out44 out45 out46 out47 out48 out49 out50 out51 out52 out53 out54 out55 out56 out57 out58 out59 out60 out61 out62 out63
X
OR
OR
OR
Target Module
DMA_CH0
DMA_CH1
DMA_CH2
DMA_CH3
TRGMUX_OUT0
TRGMUX_OUT1
TRGMUX_OUT2
TRGMUX_OUT3
TRGMUX_OUT4
TRGMUX_OUT5
TRGMUX_OUT6
TRGMUX_OUT7
ADC0_ADHWT
ADC1_ADHWT
ADC2_ADHWT
DAC0_SYNC
CMP0_SAMPLE
CMP1_SAMPLE
CMP2_SAMPLE
FTM0_HWTRIG
FTM0_FAULT0
FTM0_FAULT1
FTM0_FAULT2
FTM1_HWTRIG
FTM1_FAULT0
FTM1_FAULT1
FTM1_FAULT2
FTM2_HWTRIG
FTM2_FAULT0
FTM2_FAULT1
FTM2_FAULT2
FTM3_HWTRIG
FTM3_FAULT0
FTM3_FAULT1
FTM3_FAULT2
PDB0_TRG_IN
PDB1_TRG_IN



NOTE

When using the TRGMUX to trigger DMA, DMAMUX must be configured (in the DMAMUX_CHCFG register) with ENBL, TRIG bit set, meanwhile SOURCE bits must be !=0 .

NOTE

For each ADC, the four triggers are OR'ed together to provide a flexible trigger scheme for the hardware trigger of each ADC, while the pre-triggers are not OR'ed. The LPIT pre-triggers can be pre-triggers for each ADC. There is another PDB pre-trigger scheme existing on this device, which is not through

TRGMUX. Please refer to ADC section for details on ADC trigger implementation on this device.

10.2 Introduction

The trigger MUX module (TRGMUX) allows software to configure the trigger inputs for various peripherals.

10.2.1 Features

The Trigger MUX module allows software to configure the trigger inputs for various peripherals.

- Trigger MUX select

10.3 Functional description

The Trigger MUX module allows software to configure the trigger inputs for various peripherals.

Each peripheral has its own unique TRGMUX register that is used to select the trigger source for peripheral.

See each peripheral's TRGMUX register for details.

10.4 Memory map and register definition

The TRGMUX module contains register fields for selecting the trigger input for peripheral modules.

10.4.1 TRGMUX0 Register Descriptions

10.4.1.1 TRGMUX0 Memory Map

Table 10-1. Select Bit Fields

Field	Function
5-0 SEL	This read/write bit field is used to configure the MUX select for the peripheral trigger inputs. 000_0000 - (0x00) TRGMUX_IN0 input is selected. 000_0001 - (0x01) TRGMUX_IN1 input is selected. 000_0010 - (0x02) TRGMUX_IN2 input is selected. 000_0011 - (0x03) TRGMUX_IN3 input is selected. 000_0100 - (0x04) RTC_second input is selected. 000_0101 - (0x05) RTC_alarm input is selected. 000_0110 - (0x06) LPTMR0 input is selected. 000_0111 - (0x07) LPIT_CH0 is selected. 000_1000 - (0x08) LPIT_CH1 is selected. 000_1001 - (0x09) LPIT_CH2 is selected. 000_1010 - (0x0A) LPIT_CH3 is selected. 000_1011 - (0x0B) FTM0_TRIG is selected. 000_1100 - (0x0C) FTM1_TRIG is selected. 000_1101 - (0x0D) FTM2_TRIG is selected. 000_1110 - (0x0E) FTM3_TRIG is selected. 000_1111 - (0x0F) ADC0_COCOA is selected. 001_0000 - (0x10) ADC0_COCOB is selected. 001_0001 - (0x11) CMP0_OUT is selected. 001_0010 - (0x12) CMP1_OUT is selected. 001_0011 - (0x13) CMP2_OUT is selected. 001_0100 - (0x14) FLEXIO_TRIG0 is selected. 001_0101 - (0x15) FLEXIO_TRIG1 is selected. 001_0110 - (0x16) FLEXIO_TRIG2 is selected. 001_0111 - (0x17) FLEXIO_TRIG3 is selected. 001_1000 - (0x18) Unused. 001_1001 - (0x19) Unused. 001_1010 - (0x1A) Unused. 001_1011 - (0x1B) Unused. 001_1100 - (0x1C) Unused. 001_1101 - (0x1D) Unused. 001_1110 - (0x1E) Unused. 001_1111 - (0x1F) Unused. 010_0000 - (0x20) Unused 010_0001 - (0x21) Unused 010_0010 - (0x22) Unused 010_0011 - (0x23) Unused

Table 10-1. Select Bit Fields

Field	Function
	010_0100 - (0x24) Unused
	010_0101 - (0x25) Unused
	010_0110 - (0x26) Unused
	010_0111 - (0x27) Unused
	010_1000 - (0x28) Unused
	010_1001 - (0x29) Unused
	010_1010 - (0x2A) Unused
	010_1011 - (0x2B) Unused
	010_1100 - (0x2C) Unused
	010_1101 - (0x2D) Unused
	010_1110 - (0x2E) Unused
	010_1111 - (0x2F) Unused
	011_0000 - (0x30) Unused
	011_0001 - (0x31) Unused
	011_0010 - (0x32) Unused
	011_0011 - (0x33) Unused
	011_0100 - (0x34) Unused
	011_0101 - (0x35) Unused
	011_0110 - (0x36) Unused
	011_0111 - (0x37) Unused
	011_1000 - (0x38) Unused
	011_1001 - (0x39) Unused
	011_1010 - (0x3A) Unused
	011_1011 - (0x3B) Unused
	011_1100 - (0x3C) Unused
	011_1101 - (0x3D) Unused
	011_1110 - (0x3E) Unused
	011_1111 - (0x3F) Unused
	100_0000 - (0x40) Unused
	100_0001 - (0x41) Unused
	100_0010 - (0x42) Unused
	100_0011 - (0x43) Unused
	100_0100 - (0x44) Unused
	100_0101 - (0x45) Unused
	100_0110 - (0x46) Unused
	100_0111 - (0x47) Unused
	100_1000 - (0x48) Unused
	100_1001 - (0x49) Unused

Table 10-1. Select Bit Fields

Field	Function
	100_1010 - (0x4A) Unused
	100_1011 - (0x4B) Unused
	100_1100 - (0x4C) Unused
	100_1101 - (0x4D) Unused
	100_1110 - (0x4E) Unused
	100_1111 - (0x4F) Unused
	101_0000 - (0x50) Unused
	101_0001 - (0x51) Unused
	101_0010 - (0x52) Unused
	101_0011 - (0x53) Unused
	101_0100 - (0x54) Unused
	101_0101 - (0x55) Unused
	101_0110 - (0x56) Unused
	101_0111 - (0x57) Unused
	101_1000 - (0x58) Unused
	101_1001 - (0x59) Unused
	101_1010 - (0x5A) Unused
	101_1011 - (0x5B) Unused
	101_1100 - (0x5C) Unused
	101_1101 - (0x5D) Unused
	101_1110 - (0x5E) Unused
	101_1111 - (0x5F) Unused
	110_0000 - (0x60) Unused
	110_0001 - (0x61) Unused
	110_0010 - (0x62) Unused
	110_0011 - (0x63) Unused
	110_0100 - (0x64) Unused
	110_0101 - (0x65) Unused
	110_0110 - (0x66) Unused
	110_0111 - (0x67) Unused
	110_1000 - (0x68) Unused
	110_1001 - (0x69) Unused
	110_1010 - (0x6A) Unused
	110_1011 - (0x6B) Unused
	110_1100 - (0x6C) Unused
	110_1101 - (0x6D) Unused
	110_1110 - (0x6E) Unused
	110_1111 - (0x6F) Unused

Table 10-1. Select Bit Fields

Field	Function
	111_0000 - (0x70) Unused 111_0001 - (0x71) Unused 111_0010 - (0x72) Unused 111_0011 - (0x73) Unused 111_0100 - (0x74) Unused 111_0101 - (0x75) Unused 111_0110 - (0x76) Unused 111_0111 - (0x77) Unused 111_1000 - (0x78) Unused 111_1001 - (0x79) Unused 111_1010 - (0x7A) Unused 111_1011 - (0x7B) Unused 111_1100 - (0x7C) Unused 111_1101 - (0x7D) Unused 111_1110 - (0x7E) Unused 111_1111 - (0x7F) Unused

Absolute address	Register	Width (In bits)	Access	Reset value
40062000h	TRGMUX DMAMUX0 (TRGMUX_DMAMUX0)	32	RW	00000000h
40062004h	TRGMUX EXTOUT0 (TRGMUX_EXTOUT0)	32	RW	00000000h
40062008h	TRGMUX EXTOUT1 (TRGMUX_EXTOUT1)	32	RW	00000000h
4006200Ch	TRGMUX ADC0 (TRGMUX_ADC0)	32	RW	00000000h
40062010h	TRGMUX ADC1 (TRGMUX_ADC1)	32	RW	00000000h
40062014h	TRGMUX ADC2 (TRGMUX_ADC2)	32	RW	00000000h
40062018h	TRGMUX DAC0 (TRGMUX_DAC0)	32	RW	00000000h
4006201Ch	TRGMUX CMP0 (TRGMUX_CMP0)	32	RW	00000000h
40062020h	TRGMUX CMP1 (TRGMUX_CMP1)	32	RW	00000000h
40062024h	TRGMUX CMP2 (TRGMUX_CMP2)	32	RW	00000000h
40062028h	TRGMUX FTM0 (TRGMUX_FTM0)	32	RW	00000000h
4006202Ch	TRGMUX FTM1 (TRGMUX_FTM1)	32	RW	00000000h
40062030h	TRGMUX FTM2 (TRGMUX_FTM2)	32	RW	00000000h
40062034h	TRGMUX FTM3 (TRGMUX_FTM3)	32	RW	00000000h
40062038h	TRGMUX PDB0 (TRGMUX_PDB0)	32	RW	00000000h
4006203Ch	TRGMUX PDB1 (TRGMUX_PDB1)	32	RW	00000000h
40062040h	TRGMUX PDB2 (TRGMUX_PDB2)	32	RW	00000000h
40062044h	TRGMUX FLEXIO (TRGMUX_FLEXIO)	32	RW	00000000h
40062048h	TRGMUX LPIT0 (TRGMUX_LPIT0)	32	RW	00000000h

Table continues on the next page...

Absolute address	Register	Width (In bits)	Access	Reset value
4006204Ch	TRGMUX LPUART0 (TRGMUX_LPUART0)	32	RW	00000000h
40062050h	TRGMUX LPUART1 (TRGMUX_LPUART1)	32	RW	00000000h
40062054h	TRGMUX LPI2C0 (TRGMUX_LPI2C0)	32	RW	00000000h
40062058h	TRGMUX LPI2C1 (TRGMUX_LPI2C1)	32	RW	00000000h
4006205Ch	TRGMUX LPSPI0 (TRGMUX_LPSPI0)	32	RW	00000000h
40062060h	TRGMUX LPSPI1 (TRGMUX_LPSPI1)	32	RW	00000000h
40062064h	TRGMUX LPTMR0 (TRGMUX_LPTMR0)	32	RW	00000000h
40062068h	TRGMUX Reserved (TRGMUX_Reserved)	32	RO	00000000h
4006206Ch	TRGMUX PWT (TRGMUX_PWT)	32	RW	00000000h

10.4.1.2 TRGMUX DMAMUX0 (TRGMUX_DMAMUX0)

10.4.1.2.1 Address

Register	Offset
TRGMUX_DMAMUX0	40062000h

10.4.1.2.2 Function

TRGMUX Register

10.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.2.4 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.3 TRGMUX EXTOUT0 (TRGMUX_EXTOUT0)

10.4.1.3.1 Address

Register	Offset
TRGMUX_EXTOUT0	40062004h

TRGMUX Register

10.4.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.3.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.4 TRGMUX EXTOUT1 (TRGMUX_EXTOUT1)

10.4.1.4.1 Address

Register	Offset
TRGMUX_EXTOUT1	40062008h

TRGMUX Register

10.4.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.4.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.5 TRGMUX ADC0 (TRGMUX_ADC0)

10.4.1.5.1 Address

Register	Offset
TRGMUX_ADC0	4006200Ch

TRGMUX Register

10.4.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.5.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.6 TRGMUX ADC1 (TRGMUX_ADC1)

10.4.1.6.1 Address

Register	Offset
TRGMUX_ADC1	40062010h

TRGMUX Register

10.4.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.6.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

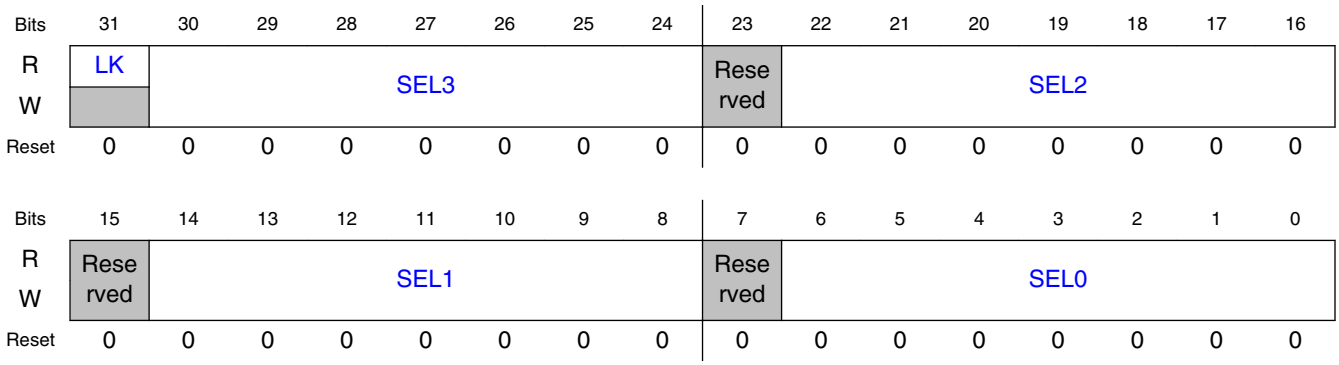
10.4.1.7 TRGMUX ADC2 (TRGMUX_ADC2)

10.4.1.7.1 Address

Register	Offset
TRGMUX_ADC2	40062014h

TRGMUX Register

10.4.1.7.2 Diagram



10.4.1.7.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.8 TRGMUX DAC0 (TRGMUX_DAC0)

10.4.1.8.1 Address

Register	Offset
TRGMUX_DAC0	40062018h

TRGMUX Register

10.4.1.8.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.8.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.9 TRGMUX CMP0 (TRGMUX_CMP0)

10.4.1.9.1 Address

Register	Offset
TRGMUX_CMP0	4006201Ch

TRGMUX Register

10.4.1.9.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.9.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.10 TRGMUX CMP1 (TRGMUX_CMP1)

10.4.1.10.1 Address

Register	Offset
TRGMUX_CMP1	40062020h

TRGMUX Register

10.4.1.10.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.10.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.11 TRGMUX CMP2 (TRGMUX_CMP2)

10.4.1.11.1 Address

Register	Offset
TRGMUX_CMP2	40062024h

TRGMUX Register

10.4.1.11.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.11.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.12 TRGMUX FTM0 (TRGMUX_FTM0)

10.4.1.12.1 Address

Register	Offset
TRGMUX_FTM0	40062028h

TRGMUX Register

10.4.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.12.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.13 TRGMUX FTM1 (TRGMUX_FTM1)

10.4.1.13.1 Address

Register	Offset
TRGMUX_FTM1	4006202Ch

TRGMUX Register

10.4.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.13.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.14 TRGMUX FTM2 (TRGMUX_FTM2)

10.4.1.14.1 Address

Register	Offset
TRGMUX_FTM2	40062030h

TRGMUX Register

10.4.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.14.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.15 TRGMUX FTM3 (TRGMUX_FTM3)

10.4.1.15.1 Address

Register	Offset
TRGMUX_FTM3	40062034h

TRGMUX Register

10.4.1.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.15.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.16 TRGMUX PDB0 (TRGMUX_PDB0)

10.4.1.16.1 Address

Register	Offset
TRGMUX_PDB0	40062038h

TRGMUX Register

10.4.1.16.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.16.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.17 TRGMUX PDB1 (TRGMUX_PDB1)

10.4.1.17.1 Address

Register	Offset
TRGMUX_PDB1	4006203Ch

TRGMUX Register

10.4.1.17.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.17.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.18 TRGMUX PDB2 (TRGMUX_PDB2)

10.4.1.18.1 Address

Register	Offset
TRGMUX_PDB2	40062040h

TRGMUX Register

10.4.1.18.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.18.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.19 TRGMUX FLEXIO (TRGMUX_FLEXIO)

10.4.1.19.1 Address

Register	Offset
TRGMUX_FLEXIO	40062044h

TRGMUX Register

10.4.1.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.19.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24	Trigger MUX Input 3 Source Select

Table continues on the next page...

Field	Function
SEL3	This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.20 TRGMUX LPIT0 (TRGMUX_LPIT0)

10.4.1.20.1 Address

Register	Offset
TRGMUX_LPIT0	40062048h

TRGMUX Register

10.4.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.20.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.21 TRGMUX LPUART0 (TRGMUX_LPUART0)

10.4.1.21.1 Address

Register	Offset
TRGMUX_LPUART0	4006204Ch

TRGMUX Register

10.4.1.21.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.21.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.22 TRGMUX LPUART1 (TRGMUX_LPUART1)

10.4.1.22.1 Address

Register	Offset
TRGMUX_LPUART1	40062050h

TRGMUX Register

10.4.1.22.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.22.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

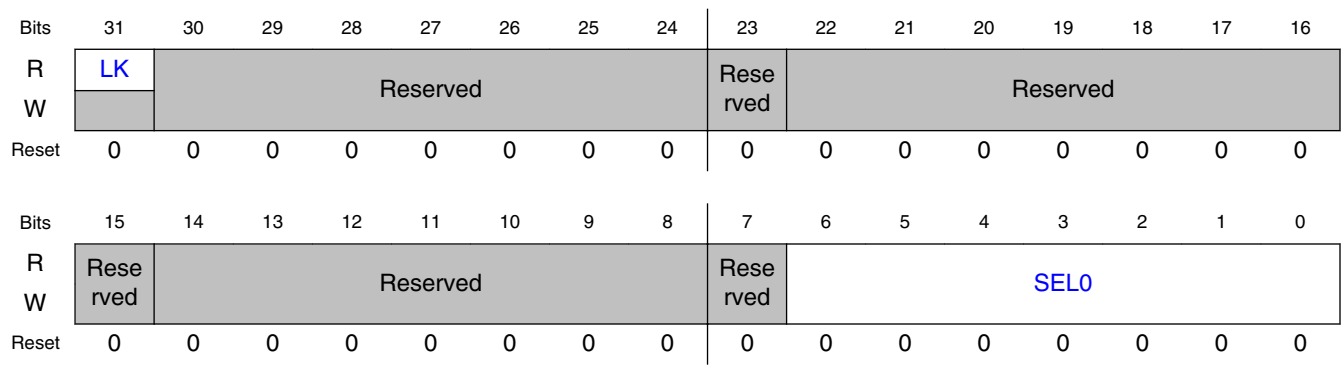
10.4.1.23 TRGMUX LPI2C0 (TRGMUX_LPI2C0)

10.4.1.23.1 Address

Register	Offset
TRGMUX_LPI2C0	40062054h

TRGMUX Register

10.4.1.23.2 Diagram



10.4.1.23.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.24 TRGMUX LPI2C1 (TRGMUX_LPI2C1)

10.4.1.24.1 Address

Register	Offset
TRGMUX_LPI2C1	40062058h

TRGMUX Register

10.4.1.24.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.24.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.25 TRGMUX LPSPI0 (TRGMUX_LPSPI0)

10.4.1.25.1 Address

Register	Offset
TRGMUX_LPSPI0	4006205Ch

TRGMUX Register

10.4.1.25.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.25.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.26 TRGMUX LPSPI1 (TRGMUX_LPSPI1)

10.4.1.26.1 Address

Register	Offset
TRGMUX_LPSPI1	40062060h

TRGMUX Register

10.4.1.26.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.26.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.27 TRGMUX LPTMR0 (TRGMUX_LPTMR0)

10.4.1.27.1 Address

Register	Offset
TRGMUX_LPTMR0	40062064h

TRGMUX Register

10.4.1.27.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.27.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.1.28 TRGMUX Reserved (TRGMUX_Reserved)

10.4.1.28.1 Address

Register	Offset
TRGMUX_Reserved	40062068h

TRGMUX Register

10.4.1.28.2 Diagram

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

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

10.4.1.28.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
7 —	This read-only bit field is reserved and always has the value 0.
6-0 —	This read-only bit field is reserved and always has the value 0.

10.4.1.29 TRGMUX PWT (TRGMUX_PWT)

10.4.1.29.1 Address

Register	Offset
TRGMUX_PWT	4006206Ch

TRGMUX Register

10.4.1.29.2 Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.1.29.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.4.2 TRGMUX1 Register Descriptions

10.4.2.1 TRGMUX1 Memory Map

Table 10-2. Select Bit Fields

Field	Function
5-0 SEL	This read/write bit field is used to configure the MUX select for the peripheral trigger inputs. 000_0000 - (0x00) Trigger function is disabled. 000_0001 - (0x01) VDD is selected. 000_0010 - (0x02) SIM Software trigger is selected. 000_0011 - (0x03) TRGMUX_IN4 input is selected. 000_0100 - (0x04) TRGMUX_IN5 input is selected. 000_0101 - (0x05) TRGMUX_IN6 input is selected. 000_0110 - (0x06) TRGMUX_IN7 input is selected. 000_0111 - (0x07) LPUART0 RX Data is selected. 000_1000 - (0x08) LPUART0 TX Data is selected. 000_1001 - (0x09) LPUART0 RX Idle is selected. 000_1010 - (0x0A) LPI2C0 Master STOP is selected. 000_1011 - (0x0B) LPI2C0 Slave STOP is selected. 000_1100 - (0x0C) LPSPI0 Frame is selected. 000_1101 - (0x0D) LPSPI0 RX data is selected. 000_1110 - (0x0E) LPUART1 RX Data is selected.

Table 10-2. Select Bit Fields

Field	Function
	000_1111 - (0x0F) LPUART1 TX Data is selected. 001_0000 - (0x10) LPUART1 RX Idle is selected. 001_0001 - (0x11) LPI2C1 Master STOP is selected. 001_0010 - (0x12) LPI2C1 Slave STOP is selected. 001_0011 - (0x13) LPSPI1 Frame is selected. 001_0100 - (0x14) LPSPI1 RX data is selected. 001_0101 - (0x15) ADC1_COCOA is selected. 001_0110 - (0x16) ADC1_COCOB is selected. 001_0111 - (0x17) PDB0_DAC is selected. 001_1000 - (0x18) PDB0_Pulse is selected. 001_1001 - (0x19) PDB1_DAC is selected. 001_1010 - (0x1A) PDB1_Pulse is selected. 001_1011 - (0x1B) PDB2_DAC is selected. 001_1100 - (0x1C) PDB2_Pulse is selected. 001_1101 - (0x1D) ADC2_COCOA is selected. 001_1110 - (0x1E) ADC2_COCOB is selected. 001_1111 - (0x1F) Unused. 010_0000 - (0x20) Unused 010_0001 - (0x21) Unused 010_0010 - (0x22) Unused 010_0011 - (0x23) Unused 010_0100 - (0x24) Unused 010_0101 - (0x25) Unused 010_0110 - (0x26) Unused 010_0111 - (0x27) Unused 010_1000 - (0x28) Unused 010_1001 - (0x29) Unused 010_1010 - (0x2A) Unused 010_1011 - (0x2B) Unused 010_1100 - (0x2C) Unused 010_1101 - (0x2D) Unused 010_1110 - (0x2E) Unused 010_1111 - (0x2F) Unused 011_0000 - (0x30) Unused 011_0001 - (0x31) Unused 011_0010 - (0x32) Unused 011_0011 - (0x33) Unused 011_0100 - (0x34) Unused

Table 10-2. Select Bit Fields

Field	Function
	011_0101 - (0x35) Unused
	011_0110 - (0x36) Unused
	011_0111 - (0x37) Unused
	011_1000 - (0x38) Unused
	011_1001 - (0x39) Unused
	011_1010 - (0x3A) Unused
	011_1011 - (0x3B) Unused
	011_1100 - (0x3C) Unused
	011_1101 - (0x3D) Unused
	011_1110 - (0x3E) Unused
	011_1111 - (0x3F) Unused
	100_0000 - (0x40) Unused
	100_0001 - (0x41) Unused
	100_0010 - (0x42) Unused
	100_0011 - (0x43) Unused
	100_0100 - (0x44) Unused
	100_0101 - (0x45) Unused
	100_0110 - (0x46) Unused
	100_0111 - (0x47) Unused
	100_1000 - (0x48) Unused
	100_1001 - (0x49) Unused
	100_1010 - (0x4A) Unused
	100_1011 - (0x4B) Unused
	100_1100 - (0x4C) Unused
	100_1101 - (0x4D) Unused
	100_1110 - (0x4E) Unused
	100_1111 - (0x4F) Unused
	101_0000 - (0x50) Unused
	101_0001 - (0x51) Unused
	101_0010 - (0x52) Unused
	101_0011 - (0x53) Unused
	101_0100 - (0x54) Unused
	101_0101 - (0x55) Unused
	101_0110 - (0x56) Unused
	101_0111 - (0x57) Unused
	101_1000 - (0x58) Unused
	101_1001 - (0x59) Unused
	101_1010 - (0x5A) Unused

Table 10-2. Select Bit Fields

Field	Function
	101_1011 - (0x5B) Unused
	101_1100 - (0x5C) Unused
	101_1101 - (0x5D) Unused
	101_1110 - (0x5E) Unused
	101_1111 - (0x5F) Unused
	110_0000 - (0x60) Unused
	110_0001 - (0x61) Unused
	110_0010 - (0x62) Unused
	110_0011 - (0x63) Unused
	110_0100 - (0x64) Unused
	110_0101 - (0x65) Unused
	110_0110 - (0x66) Unused
	110_0111 - (0x67) Unused
	110_1000 - (0x68) Unused
	110_1001 - (0x69) Unused
	110_1010 - (0x6A) Unused
	110_1011 - (0x6B) Unused
	110_1100 - (0x6C) Unused
	110_1101 - (0x6D) Unused
	110_1110 - (0x6E) Unused
	110_1111 - (0x6F) Unused
	111_0000 - (0x70) Unused
	111_0001 - (0x71) Unused
	111_0010 - (0x72) Unused
	111_0011 - (0x73) Unused
	111_0100 - (0x74) Unused
	111_0101 - (0x75) Unused
	111_0110 - (0x76) Unused
	111_0111 - (0x77) Unused
	111_1000 - (0x78) Unused
	111_1001 - (0x79) Unused
	111_1010 - (0x7A) Unused
	111_1011 - (0x7B) Unused
	111_1100 - (0x7C) Unused
	111_1101 - (0x7D) Unused
	111_1110 - (0x7E) Unused
	111_1111 - (0x7F) Unused

Absolute address	Register	Width (In bits)	Access	Reset value
40063000h	TRGMUX CTRL0 (TRGMUX_CTRL0)	32	RW	00000000h
40063004h	TRGMUX CTRL1 (TRGMUX_CTRL1)	32	RW	00000000h

10.4.2.2 TRGMUX CTRL0 (TRGMUX_CTRL0)

10.4.2.2.1 Address

Register	Offset
TRGMUX_CTRL0	40063000h

10.4.2.2.2 Function

TRGMUX Register

10.4.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							Reserved	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SEL1							Reserved	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.4.2.2.4 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select

Table continues on the next page...

Field	Function
	This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

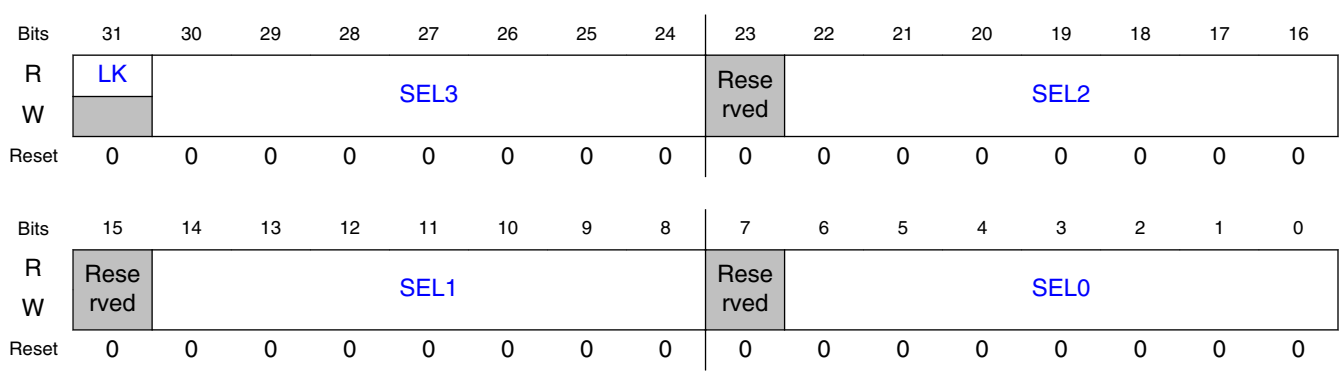
10.4.2.3 TRGMUX CTRL1 (TRGMUX_CTRL1)

10.4.2.3.1 Address

Register	Offset
TRGMUX_CTRL1	40063004h

TRGMUX Register

10.4.2.3.2 Diagram



10.4.2.3.3 Fields

Field	Function
31 LK	Enable This bit shows whether the register can be written or not. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

10.5 Usage Guide

The TRGMUX is an extremely flexible module interconnectivity scheme. The trigger source could be from various peripherals and external input pins, to multiple pins/peripherals. The module level interconnections and trigger scheme offload the intervention of CPU, which is also useful when CPU is in WAIT/STOP mode. The following are some typical use-cases for TRGMUX.

10.5.1 ADC Trigger Source

The following triggers are via the TRGMUX:

- CMP out to trigger each ADC
- LPIT capable to trigger each ADC

- RTC capable to trigger each ADC
- LPTMR capable to trigger each ADC

For details, please refer to “ADC Trigger Sources” section.

FTM module support counter init trigger and channel match trigger, these triggers couldbe used as trigger input of PDB, PDB then be used to trigger other modules like ADC.

For details, please refer to “ADC Trigger Concept – Use Case ” section.

10.5.2 CMP Window/Sample Input

PDB and LPIT could be used to generate pulse output which can be used as sampling windows of CMP block via TRGMUX.

For details, please refer to “Window Mode” section in the CMP chapter.

10.5.3 FTM Fault Detection Input / Hardware Triggers and Synchronization

Please refer to the FTM chapter for more details.

Chapter 11
Direct Memory Access Multiplexer (DMAMUX)

11.1 Chip-specific information for this module

11.1.1 Instantiation Information

11.1.1.1 DMAMUX request sources

This device includes a DMA request mux that allows up to 63 DMA request signals to be mapped to any of the 16 DMA channels. The DMA request sources could be peripheral DMA requests or always-on slots. Because of the mux, there is not a hard correlation between any of the DMA request sources and a specific DMA channel.

Some of the modules support Asynchronous DMA operation as indicated by the last column in the following DMA source assignment table. Asynchronous DMA requests can be used to activate a DMA channel in WAIT or STOP mode.

Table 11-1. DMA request sources - MUX 0

Source number	Source module	Source description	Async DMA capable
0	—	Channel disabled[1]	
1	Reserved	—	
2	LPUART0	Receive	Yes
3	LPUART0	Transmit	Yes
4	LPUART1	Receive	Yes
5	LPUART1	Transmit	Yes
6	LPUART2	Receive	Yes
7	LPUART2	Transmit	Yes
8	Reserved	—	

Table continues on the next page...

Table 11-1. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description	Async DMA capable
9	Reserved	—	
10	FlexIO	Shifter0	Yes
11	FlexIO	Shifter1	Yes
12	FlexIO	Shifter2	Yes
13	FlexIO	Shifter3	Yes
14	LPSPI0	Receive	Yes
15	LPSPI0	Transmit	Yes
16	LPSPI1	Receive	Yes
17	LPSPI1	Transmit	Yes
18	LPI^{2}C0	Receive	Yes
19	LPI^{2}C0	Transmit	Yes
20	FTM0	Channel 0	
21	FTM0	Channel 1	
22	FTM0	Channel 2	
23	FTM0	Channel 3	
24	FTM0	Channel 4	
25	FTM0	Channel 5	
26	FTM0	Channel 6	
27	FTM0	Channel 7	
28	FTM1	Channel 0	
29	FTM1	Channel 1	
30	FTM2	Channel 0	
31	FTM2	Channel 1	
32	LPI^{2}C1 or FTM3	LPI^{2}C1 Receiveor FTM3 Channel 0	Yes for LPI^{2}C1
33	LPI^{2}C1or FTM3	LPI^{2}C1 Transmitor FTM3 Channel 1	Yes for LPI^{2}C1
34	FTM3	Channel 2	
35	FTM3	Channel 3	
36	FTM3	Channel 4	
37	FTM3	Channel 5	
38	FTM3	Channel 6	
39	FTM3	Channel 7	
40	ADC0	ADC0 COCO	Yes
41	ADC1	ADC1 COCO	Yes
42	ADC2	ADC2 COCO	Yes
43	CMP0	—	Yes
44	CMP1	—	Yes
45	CMP2	—	Yes

Table continues on the next page...

Table 11-1. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description	Async DMA capable
46	PDB0	—	
47	PDB1	—	
48	PDB2	—	
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	FlexCAN0[2]	FlexCAN0	
55	FlexCAN1[2]	FlexCAN1	
56	DAC0	DAC0	Yes
57	FTM1	OR of ch2-ch7	
58	FTM2	OR of ch2-ch7	
59	LPTMR0	—	Yes
60	DMAMUX	Always enabled	
61	DMAMUX	Always enabled	
62	DMAMUX	Always enabled	
63	DMAMUX	Always enabled	

1. Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.
2. The several FlexCAN DMA Rx will be OR'd with the DAC.

11.1.1.2 DMA trigger sources

The DMAMUX on this device also supports a periodic trigger mode. The trigger sources are from TRGMUX output showed in following table. The triggers from TRGMUX module can trigger a DMA transfer on the first four DMA channels (channel 0 -3), for example, the LPIT can trigger DMA via TRGMUX.

Table 11-2. DMAMUX trigger sources

Trigger number	Trigger module	Trigger description
0	TRGMUX	TRGMUX trigger out0
1	TRGMUX	TRGMUX trigger out1
2	TRGMUX	TRGMUX trigger out2
3	TRGMUX	TRGMUX trigger out3

11.2 Introduction

11.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 16 DMA channels. This process is illustrated in the following figure.

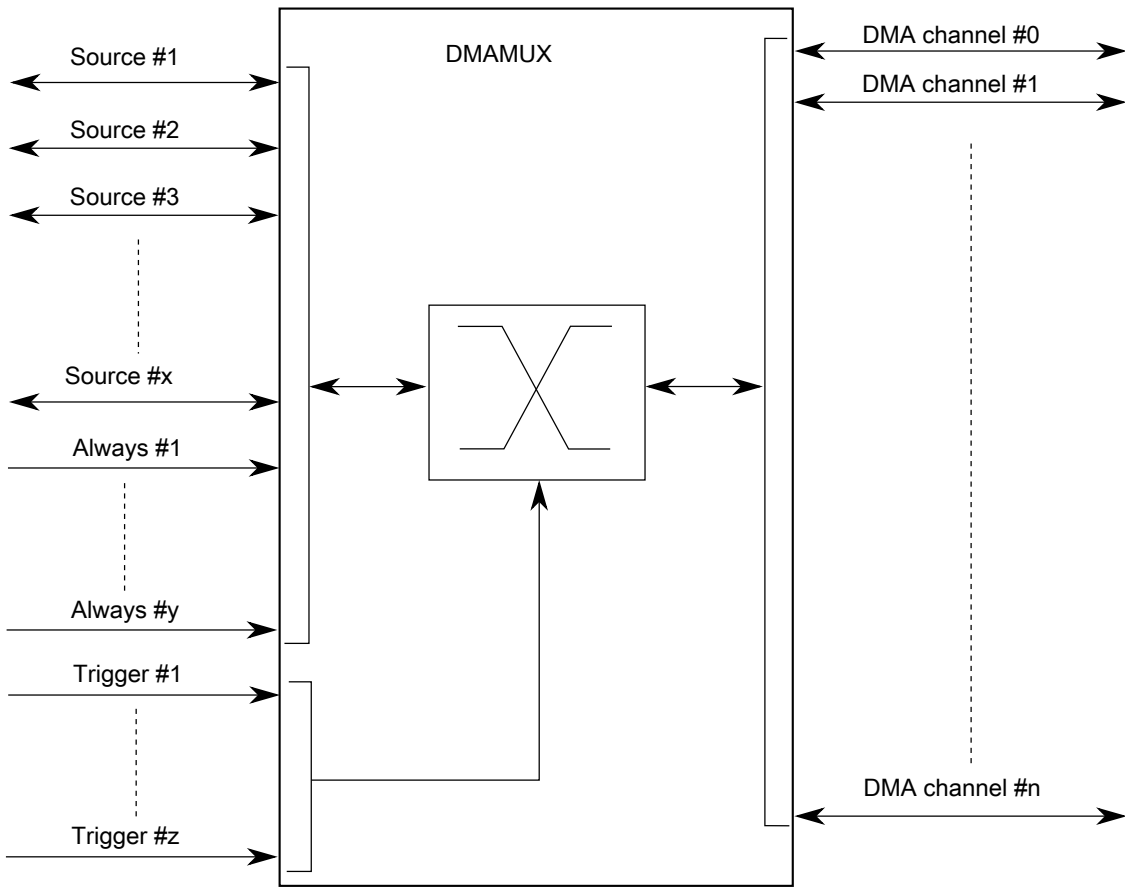


Figure 11-1. DMAMUX block diagram

11.2.2 Features

The DMAMUX module provides these features:

- Up to 63 peripheral slots and up to two always-on slots can be routed to 16 channels.
- 16 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.

11.2.3 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 (LPIT). This mode is available only for channels 0–3.

11.3 External signal description

The DMAMUX has no external pins.

11.4 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

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	11.4.1/214
4002_1001	Channel Configuration register (DMAMUX_CHCFG1)	8	R/W	00h	11.4.1/214
4002_1002	Channel Configuration register (DMAMUX_CHCFG2)	8	R/W	00h	11.4.1/214
4002_1003	Channel Configuration register (DMAMUX_CHCFG3)	8	R/W	00h	11.4.1/214
4002_1004	Channel Configuration register (DMAMUX_CHCFG4)	8	R/W	00h	11.4.1/214
4002_1005	Channel Configuration register (DMAMUX_CHCFG5)	8	R/W	00h	11.4.1/214
4002_1006	Channel Configuration register (DMAMUX_CHCFG6)	8	R/W	00h	11.4.1/214
4002_1007	Channel Configuration register (DMAMUX_CHCFG7)	8	R/W	00h	11.4.1/214
4002_1008	Channel Configuration register (DMAMUX_CHCFG8)	8	R/W	00h	11.4.1/214
4002_1009	Channel Configuration register (DMAMUX_CHCFG9)	8	R/W	00h	11.4.1/214
4002_100A	Channel Configuration register (DMAMUX_CHCFG10)	8	R/W	00h	11.4.1/214
4002_100B	Channel Configuration register (DMAMUX_CHCFG11)	8	R/W	00h	11.4.1/214
4002_100C	Channel Configuration register (DMAMUX_CHCFG12)	8	R/W	00h	11.4.1/214
4002_100D	Channel Configuration register (DMAMUX_CHCFG13)	8	R/W	00h	11.4.1/214
4002_100E	Channel Configuration register (DMAMUX_CHCFG14)	8	R/W	00h	11.4.1/214
4002_100F	Channel Configuration register (DMAMUX_CHCFG15)	8	R/W	00h	11.4.1/214

11.4.1 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 15d

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 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.

11.5 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

11.5.1 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 (LPIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the LPIT. 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.

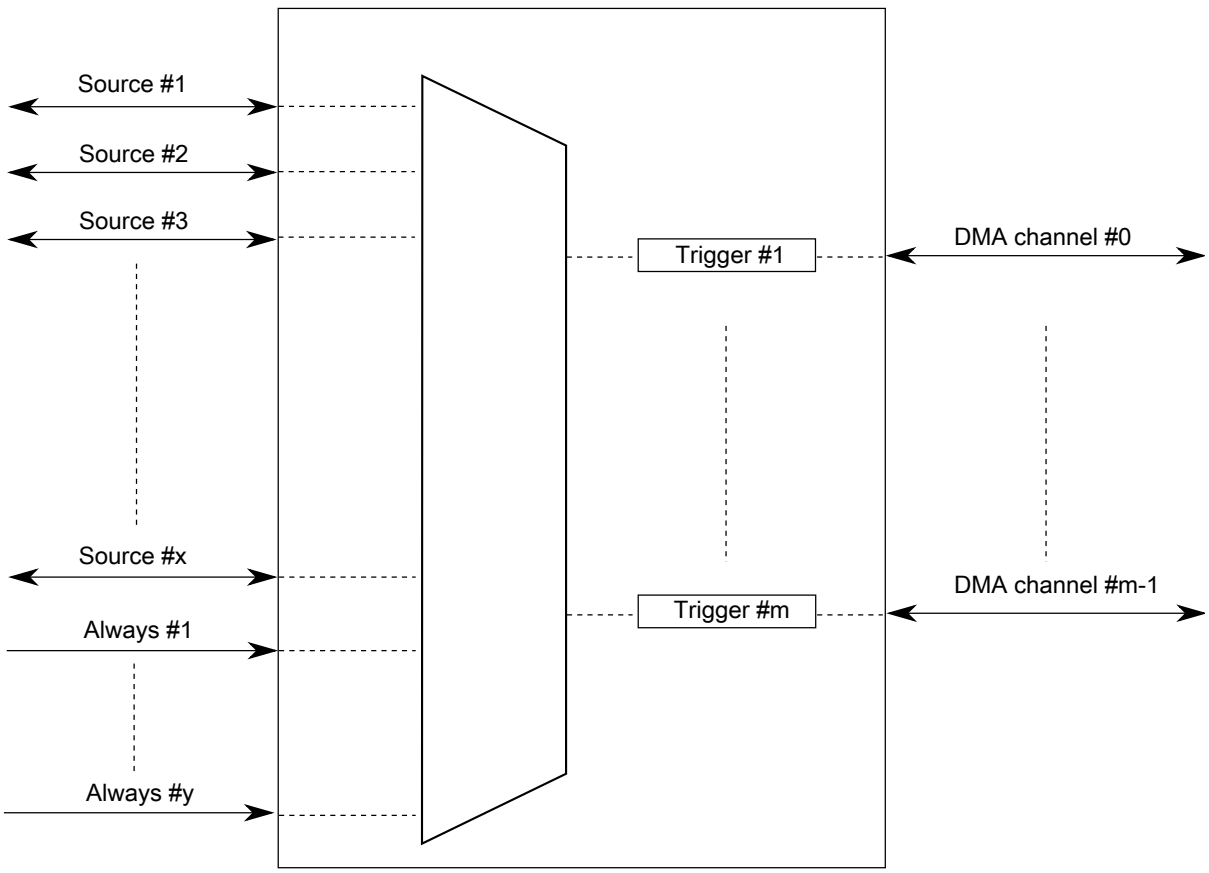


Figure 11-2. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

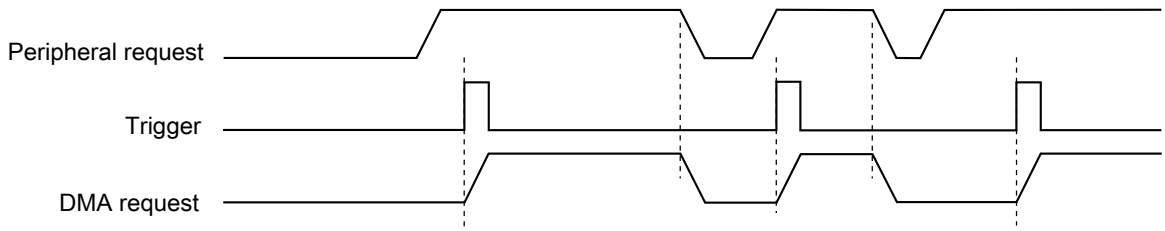


Figure 11-3. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

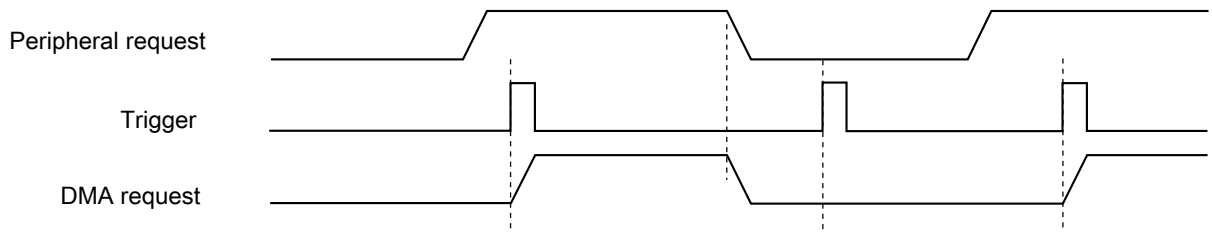


Figure 11-4. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

  As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

  By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

11.5.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in Modes of operation.

11.5.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are two 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.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

  By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

  In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

  In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

11.6 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

11.6.1 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.

11.6.2 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.
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.

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.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes 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.
3. Write 0x87 to CHCFG8. (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes 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 12
Enhanced Direct Memory Access (eDMA)

12.1 Introduction

The enhanced direct memory access (eDMA) controller is a second-generation module capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 16 channels

12.1.1 eDMA system block diagram

Figure 12-1 illustrates the components of the eDMA system, including the eDMA module ("engine").

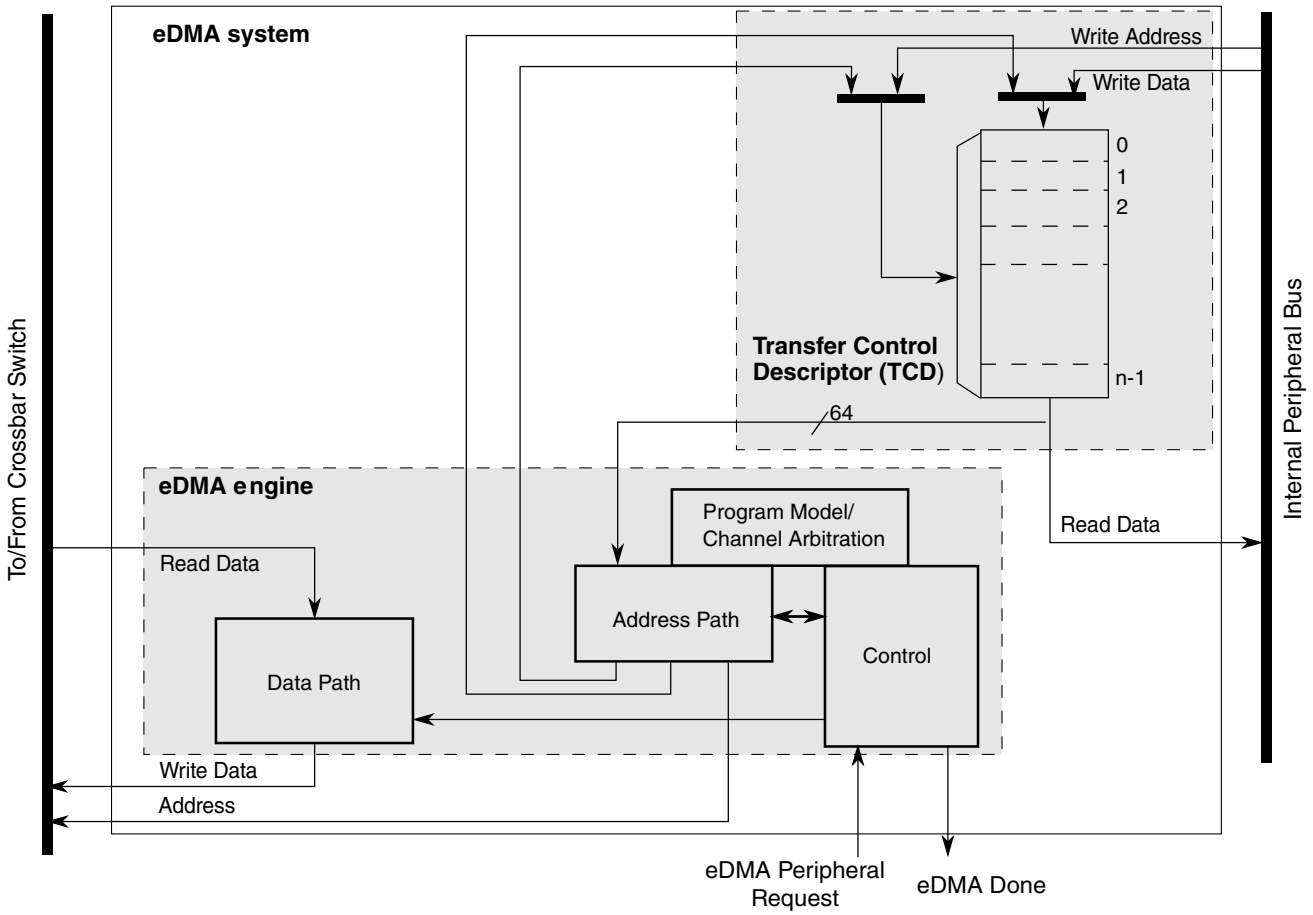


Figure 12-1. eDMA system block diagram

12.1.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 12-1. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, channel x and channel y, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI*n*[ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes

Table continues on the next page...

Table 12-1. eDMA engine submodules (continued)

Submodule	Function
	the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/ destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

Table 12-2. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

12.1.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for enhanced addressing modes

- 16-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage to support 16- and 32-byte transfers
  - Connections to the crossbar switch for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

12.2 Modes of operation

The eDMA operates in the following modes:

Table 12-3. Modes of operation

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	DMA operation is configurable in Debug mode via the control register: • If CR[EDBG] is cleared, the DMA continues to operate. • If CR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, the DMA attempts to complete its current transfer. After the transfer completes, the device enters Wait mode.

12.3 Memory map/register definition

The eDMA's programming model is partitioned into two regions:

- The first region defines a number of registers providing control functions
- The second region corresponds to the local transfer control descriptor (TCD) memory

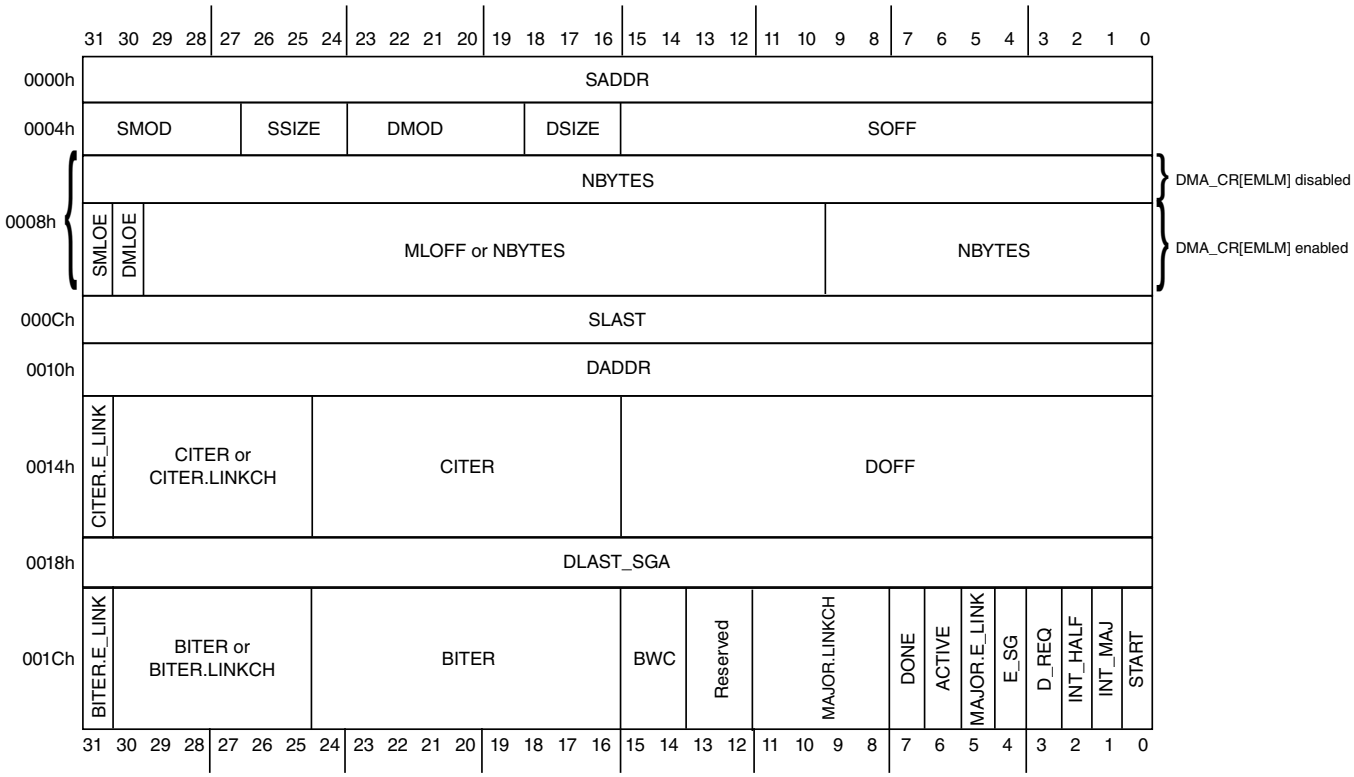
12.3.1 TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 15. Each TCD*n* definition is presented as 11 registers of 16 or 32 bits.

12.3.2 TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

12.3.3 TCD structure



12.3.4 Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_8000	Control Register (DMA_CR)	32	R/W	0000_0000h	12.3.5/239
4000_8004	Error Status Register (DMA_ES)	32	R	0000_0000h	12.3.6/242
4000_800C	Enable Request Register (DMA_ERQ)	32	R/W	0000_0000h	12.3.7/244
4000_8014	Enable Error Interrupt Register (DMA_EEI)	32	R/W	0000_0000h	12.3.8/246
4000_8018	Clear Enable Error Interrupt Register (DMA_CEEI)	8	W (always reads 0)	00h	12.3.9/249
4000_8019	Set Enable Error Interrupt Register (DMA_SEEI)	8	W (always reads 0)	00h	12.3.10/250

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_801A	Clear Enable Request Register (DMA_CERQ)	8	W (always reads 0)	00h	12.3.11/250
4000_801B	Set Enable Request Register (DMA_SERQ)	8	W (always reads 0)	00h	12.3.12/251
4000_801C	Clear DONE Status Bit Register (DMA_CDNE)	8	W (always reads 0)	00h	12.3.13/252
4000_801D	Set START Bit Register (DMA_SSRT)	8	W (always reads 0)	00h	12.3.14/253
4000_801E	Clear Error Register (DMA_CERR)	8	W (always reads 0)	00h	12.3.15/254
4000_801F	Clear Interrupt Request Register (DMA_CINT)	8	W (always reads 0)	00h	12.3.16/255
4000_8024	Interrupt Request Register (DMA_INT)	32	R/W	0000_0000h	12.3.17/256
4000_802C	Error Register (DMA_ERR)	32	R/W	0000_0000h	12.3.18/258
4000_8034	Hardware Request Status Register (DMA_HRS)	32	R	0000_0000h	12.3.19/261
4000_8044	Enable Asynchronous Request in Stop Register (DMA_EARS)	32	R/W	0000_0000h	12.3.20/264
4000_8100	Channel n Priority Register (DMA_DCHPRI3)	8	R/W	See section	12.3.21/266
4000_8101	Channel n Priority Register (DMA_DCHPRI2)	8	R/W	See section	12.3.21/266
4000_8102	Channel n Priority Register (DMA_DCHPRI1)	8	R/W	See section	12.3.21/266
4000_8103	Channel n Priority Register (DMA_DCHPRI0)	8	R/W	See section	12.3.21/266
4000_8104	Channel n Priority Register (DMA_DCHPRI7)	8	R/W	See section	12.3.21/266
4000_8105	Channel n Priority Register (DMA_DCHPRI6)	8	R/W	See section	12.3.21/266
4000_8106	Channel n Priority Register (DMA_DCHPRI5)	8	R/W	See section	12.3.21/266
4000_8107	Channel n Priority Register (DMA_DCHPRI4)	8	R/W	See section	12.3.21/266
4000_8108	Channel n Priority Register (DMA_DCHPRI11)	8	R/W	See section	12.3.21/266
4000_8109	Channel n Priority Register (DMA_DCHPRI10)	8	R/W	See section	12.3.21/266
4000_810A	Channel n Priority Register (DMA_DCHPRI9)	8	R/W	See section	12.3.21/266
4000_810B	Channel n Priority Register (DMA_DCHPRI8)	8	R/W	See section	12.3.21/266
4000_810C	Channel n Priority Register (DMA_DCHPRI15)	8	R/W	See section	12.3.21/266
4000_810D	Channel n Priority Register (DMA_DCHPRI14)	8	R/W	See section	12.3.21/266
4000_810E	Channel n Priority Register (DMA_DCHPRI13)	8	R/W	See section	12.3.21/266
4000_810F	Channel n Priority Register (DMA_DCHPRI12)	8	R/W	See section	12.3.21/266
4000_9000	TCD Source Address (DMA_TCD0_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9004	TCD Signed Source Address Offset (DMA_TCD0_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9006	TCD Transfer Attributes (DMA_TCD0_ATTR)	16	R/W	Undefined	12.3.24/268

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9008	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD0_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD0_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD0_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_900C	TCD Last Source Address Adjustment (DMA_TCD0_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9010	TCD Destination Address (DMA_TCD0_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9014	TCD Signed Destination Address Offset (DMA_TCD0_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9016	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9016	DMA_TCD0_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9018	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD0_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_901C	TCD Control and Status (DMA_TCD0_CSR)	16	R/W	Undefined	12.3.34/277
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD0_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9020	TCD Source Address (DMA_TCD1_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9024	TCD Signed Source Address Offset (DMA_TCD1_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9026	TCD Transfer Attributes (DMA_TCD1_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9028	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD1_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD1_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD1_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_902C	TCD Last Source Address Adjustment (DMA_TCD1_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9030	TCD Destination Address (DMA_TCD1_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9034	TCD Signed Destination Address Offset (DMA_TCD1_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9036	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9036	DMA_TCD1_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9038	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD1_DLASTSGA)	32	R/W	Undefined	12.3.33/276

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_903C	TCD Control and Status (DMA_TCD1_CSR)	16	R/W	Undefined	12.3.34/277
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD1_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9040	TCD Source Address (DMA_TCD2_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9044	TCD Signed Source Address Offset (DMA_TCD2_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9046	TCD Transfer Attributes (DMA_TCD2_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9048	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD2_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD2_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD2_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_904C	TCD Last Source Address Adjustment (DMA_TCD2_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9050	TCD Destination Address (DMA_TCD2_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9054	TCD Signed Destination Address Offset (DMA_TCD2_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9056	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9056	DMA_TCD2_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9058	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD2_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_905C	TCD Control and Status (DMA_TCD2_CSR)	16	R/W	Undefined	12.3.34/277
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD2_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9060	TCD Source Address (DMA_TCD3_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9064	TCD Signed Source Address Offset (DMA_TCD3_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9066	TCD Transfer Attributes (DMA_TCD3_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9068	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD3_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD3_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD3_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_906C	TCD Last Source Address Adjustment (DMA_TCD3_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9070	TCD Destination Address (DMA_TCD3_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9074	TCD Signed Destination Address Offset (DMA_TCD3_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9076	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9076	DMA_TCD3_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9078	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD3_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_907C	TCD Control and Status (DMA_TCD3_CSR)	16	R/W	Undefined	12.3.34/277
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD3_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9080	TCD Source Address (DMA_TCD4_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9084	TCD Signed Source Address Offset (DMA_TCD4_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9086	TCD Transfer Attributes (DMA_TCD4_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9088	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD4_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD4_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD4_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_908C	TCD Last Source Address Adjustment (DMA_TCD4_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9090	TCD Destination Address (DMA_TCD4_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9094	TCD Signed Destination Address Offset (DMA_TCD4_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9096	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9096	DMA_TCD4_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9098	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD4_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_909C	TCD Control and Status (DMA_TCD4_CSR)	16	R/W	Undefined	12.3.34/277
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD4_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_90A0	TCD Source Address (DMA_TCD5_SADDR)	32	R/W	Undefined	12.3.22/267

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90A4	TCD Signed Source Address Offset (DMA_TCD5_SOFF)	16	R/W	Undefined	12.3.23/267
4000_90A6	TCD Transfer Attributes (DMA_TCD5_ATTR)	16	R/W	Undefined	12.3.24/268
4000_90A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD5_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD5_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD5_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_90AC	TCD Last Source Address Adjustment (DMA_TCD5_SLAST)	32	R/W	Undefined	12.3.28/272
4000_90B0	TCD Destination Address (DMA_TCD5_DADDR)	32	R/W	Undefined	12.3.29/273
4000_90B4	TCD Signed Destination Address Offset (DMA_TCD5_DOFF)	16	R/W	Undefined	12.3.30/273
4000_90B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_90B6	DMA_TCD5_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_90B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD5_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_90BC	TCD Control and Status (DMA_TCD5_CSR)	16	R/W	Undefined	12.3.34/277
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD5_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_90C0	TCD Source Address (DMA_TCD6_SADDR)	32	R/W	Undefined	12.3.22/267
4000_90C4	TCD Signed Source Address Offset (DMA_TCD6_SOFF)	16	R/W	Undefined	12.3.23/267
4000_90C6	TCD Transfer Attributes (DMA_TCD6_ATTR)	16	R/W	Undefined	12.3.24/268
4000_90C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD6_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD6_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD6_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_90CC	TCD Last Source Address Adjustment (DMA_TCD6_SLAST)	32	R/W	Undefined	12.3.28/272
4000_90D0	TCD Destination Address (DMA_TCD6_DADDR)	32	R/W	Undefined	12.3.29/273
4000_90D4	TCD Signed Destination Address Offset (DMA_TCD6_DOFF)	16	R/W	Undefined	12.3.30/273
4000_90D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_90D6	DMA_TCD6_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD6_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_90DC	TCD Control and Status (DMA_TCD6_CSR)	16	R/W	Undefined	12.3.34/277
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD6_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_90E0	TCD Source Address (DMA_TCD7_SADDR)	32	R/W	Undefined	12.3.22/267
4000_90E4	TCD Signed Source Address Offset (DMA_TCD7_SOFF)	16	R/W	Undefined	12.3.23/267
4000_90E6	TCD Transfer Attributes (DMA_TCD7_ATTR)	16	R/W	Undefined	12.3.24/268
4000_90E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD7_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD7_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD7_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_90EC	TCD Last Source Address Adjustment (DMA_TCD7_SLAST)	32	R/W	Undefined	12.3.28/272
4000_90F0	TCD Destination Address (DMA_TCD7_DADDR)	32	R/W	Undefined	12.3.29/273
4000_90F4	TCD Signed Destination Address Offset (DMA_TCD7_DOFF)	16	R/W	Undefined	12.3.30/273
4000_90F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_90F6	DMA_TCD7_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_90F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD7_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_90FC	TCD Control and Status (DMA_TCD7_CSR)	16	R/W	Undefined	12.3.34/277
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD7_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9100	TCD Source Address (DMA_TCD8_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9104	TCD Signed Source Address Offset (DMA_TCD8_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9106	TCD Transfer Attributes (DMA_TCD8_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9108	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD8_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9108	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD8_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9108	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD8_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_910C	TCD Last Source Address Adjustment (DMA_TCD8_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9110	TCD Destination Address (DMA_TCD8_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9114	TCD Signed Destination Address Offset (DMA_TCD8_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9116	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9116	DMA_TCD8_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9118	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD8_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_911C	TCD Control and Status (DMA_TCD8_CSR)	16	R/W	Undefined	12.3.34/277
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD8_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9120	TCD Source Address (DMA_TCD9_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9124	TCD Signed Source Address Offset (DMA_TCD9_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9126	TCD Transfer Attributes (DMA_TCD9_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9128	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD9_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9128	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD9_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9128	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD9_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_912C	TCD Last Source Address Adjustment (DMA_TCD9_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9130	TCD Destination Address (DMA_TCD9_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9134	TCD Signed Destination Address Offset (DMA_TCD9_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9136	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9136	DMA_TCD9_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9138	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD9_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_913C	TCD Control and Status (DMA_TCD9_CSR)	16	R/W	Undefined	12.3.34/277
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD9_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9140	TCD Source Address (DMA_TCD10_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9144	TCD Signed Source Address Offset (DMA_TCD10_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9146	TCD Transfer Attributes (DMA_TCD10_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9148	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD10_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9148	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD10_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9148	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD10_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_914C	TCD Last Source Address Adjustment (DMA_TCD10_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9150	TCD Destination Address (DMA_TCD10_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9154	TCD Signed Destination Address Offset (DMA_TCD10_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9156	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9156	DMA_TCD10_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9158	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD10_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_915C	TCD Control and Status (DMA_TCD10_CSR)	16	R/W	Undefined	12.3.34/277
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD10_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9160	TCD Source Address (DMA_TCD11_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9164	TCD Signed Source Address Offset (DMA_TCD11_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9166	TCD Transfer Attributes (DMA_TCD11_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9168	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD11_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9168	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD11_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9168	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD11_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_916C	TCD Last Source Address Adjustment (DMA_TCD11_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9170	TCD Destination Address (DMA_TCD11_DADDR)	32	R/W	Undefined	12.3.29/273

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9174	TCD Signed Destination Address Offset (DMA_TCD11_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9176	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9176	DMA_TCD11_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9178	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD11_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_917C	TCD Control and Status (DMA_TCD11_CSR)	16	R/W	Undefined	12.3.34/277
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD11_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_9180	TCD Source Address (DMA_TCD12_SADDR)	32	R/W	Undefined	12.3.22/267
4000_9184	TCD Signed Source Address Offset (DMA_TCD12_SOFF)	16	R/W	Undefined	12.3.23/267
4000_9186	TCD Transfer Attributes (DMA_TCD12_ATTR)	16	R/W	Undefined	12.3.24/268
4000_9188	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD12_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_9188	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD12_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_9188	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD12_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_918C	TCD Last Source Address Adjustment (DMA_TCD12_SLAST)	32	R/W	Undefined	12.3.28/272
4000_9190	TCD Destination Address (DMA_TCD12_DADDR)	32	R/W	Undefined	12.3.29/273
4000_9194	TCD Signed Destination Address Offset (DMA_TCD12_DOFF)	16	R/W	Undefined	12.3.30/273
4000_9196	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_9196	DMA_TCD12_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_9198	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD12_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_919C	TCD Control and Status (DMA_TCD12_CSR)	16	R/W	Undefined	12.3.34/277
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD12_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_91A0	TCD Source Address (DMA_TCD13_SADDR)	32	R/W	Undefined	12.3.22/267
4000_91A4	TCD Signed Source Address Offset (DMA_TCD13_SOFF)	16	R/W	Undefined	12.3.23/267

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91A6	TCD Transfer Attributes (DMA_TCD13_ATTR)	16	R/W	Undefined	12.3.24/268
4000_91A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD13_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD13_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD13_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_91AC	TCD Last Source Address Adjustment (DMA_TCD13_SLAST)	32	R/W	Undefined	12.3.28/272
4000_91B0	TCD Destination Address (DMA_TCD13_DADDR)	32	R/W	Undefined	12.3.29/273
4000_91B4	TCD Signed Destination Address Offset (DMA_TCD13_DOFF)	16	R/W	Undefined	12.3.30/273
4000_91B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_91B6	DMA_TCD13_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_91B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD13_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_91BC	TCD Control and Status (DMA_TCD13_CSR)	16	R/W	Undefined	12.3.34/277
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD13_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_91C0	TCD Source Address (DMA_TCD14_SADDR)	32	R/W	Undefined	12.3.22/267
4000_91C4	TCD Signed Source Address Offset (DMA_TCD14_SOFF)	16	R/W	Undefined	12.3.23/267
4000_91C6	TCD Transfer Attributes (DMA_TCD14_ATTR)	16	R/W	Undefined	12.3.24/268
4000_91C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD14_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD14_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD14_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_91CC	TCD Last Source Address Adjustment (DMA_TCD14_SLAST)	32	R/W	Undefined	12.3.28/272
4000_91D0	TCD Destination Address (DMA_TCD14_DADDR)	32	R/W	Undefined	12.3.29/273
4000_91D4	TCD Signed Destination Address Offset (DMA_TCD14_DOFF)	16	R/W	Undefined	12.3.30/273
4000_91D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_91D6	DMA_TCD14_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD14_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_91DC	TCD Control and Status (DMA_TCD14_CSR)	16	R/W	Undefined	12.3.34/277
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD14_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280
4000_91E0	TCD Source Address (DMA_TCD15_SADDR)	32	R/W	Undefined	12.3.22/267
4000_91E4	TCD Signed Source Address Offset (DMA_TCD15_SOFF)	16	R/W	Undefined	12.3.23/267
4000_91E6	TCD Transfer Attributes (DMA_TCD15_ATTR)	16	R/W	Undefined	12.3.24/268
4000_91E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD15_NBYTES_MLNO)	32	R/W	Undefined	12.3.25/269
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD15_NBYTES_MLOFFNO)	32	R/W	Undefined	12.3.26/270
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD15_NBYTES_MLOFFYES)	32	R/W	Undefined	12.3.27/271
4000_91EC	TCD Last Source Address Adjustment (DMA_TCD15_SLAST)	32	R/W	Undefined	12.3.28/272
4000_91F0	TCD Destination Address (DMA_TCD15_DADDR)	32	R/W	Undefined	12.3.29/273
4000_91F4	TCD Signed Destination Address Offset (DMA_TCD15_DOFF)	16	R/W	Undefined	12.3.30/273
4000_91F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_CITER_ELINKYES)	16	R/W	Undefined	12.3.31/274
4000_91F6	DMA_TCD15_CITER_ELINKNO	16	R/W	Undefined	12.3.32/275
4000_91F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD15_DLASTSGA)	32	R/W	Undefined	12.3.33/276
4000_91FC	TCD Control and Status (DMA_TCD15_CSR)	16	R/W	Undefined	12.3.34/277
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_BITER_ELINKYES)	16	R/W	Undefined	12.3.35/279
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD15_BITER_ELINKNO)	16	R/W	Undefined	12.3.36/280

12.3.5 Control Register (DMA_CR)

The CR defines the basic operating configuration of the DMA.

Arbitration can be configured to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, the highest priority channel requesting service is selected to execute. The channel priority registers assign the priorities; see the DCHPRIn registers. For round-robin arbitration, the channel priorities are ignored and channels are cycled through (from high to low channel number) without regard to priority.

NOTE

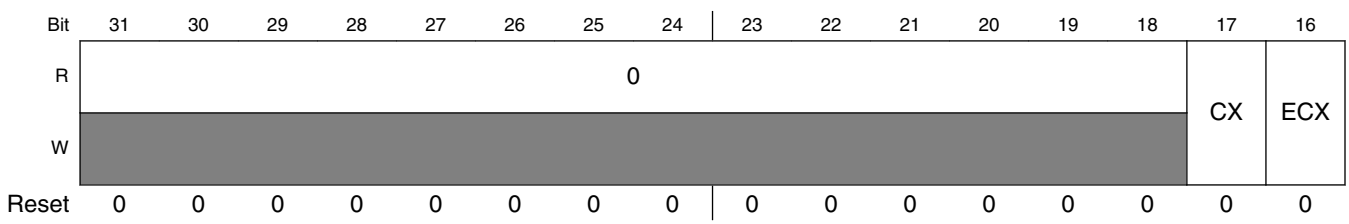
For correct operation, writes to the CR register must be performed only when the DMA channels are inactive; that is, when TCDn_CSR[ACTIVE] bits are cleared.

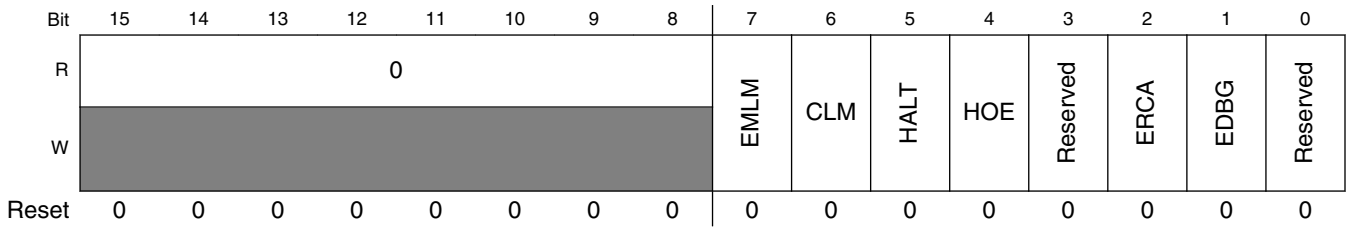
Minor loop offsets are address offset values added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. When minor loop offsets are enabled, the minor loop offset (MLOFF) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both prior to the addresses being written back into the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (EMLM is 1), TCDn word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

When minor loop mapping is disabled (EMLM is 0), all 32 bits of TCDn word2 are assigned to the NBYTES field.

Address: 4000_8000h base + 0h offset = 4000_8000h





DMA_CR field descriptions

Field	Description
31–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 CX	Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The CX bit clears itself after the cancel has been honored. This cancel retires the channel normally as if the minor loop was completed.
16 ECX	Error Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer in the same fashion as the CX bit. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The ECX bit clears itself after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating the Error Status register (DMAx_ES) and generating an optional error interrupt.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 EMLM	Enable Minor Loop Mapping 0 Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1 Enabled. TCDn.word2 is redefined to include individual enable fields, an offset field, and the NBYTES field. The individual enable fields allow the minor loop offset to be applied to the source address, the destination address, or both. The NBYTES field is reduced when either offset is enabled.
6 CLM	Continuous Link Mode **NOTE:** Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, e.g., if the channel's NBYTES value is the same as either the source or destination size. The same data transfer profile can be achieved by simply increasing the NBYTES value, which provides more efficient, faster processing. 0 A minor loop channel link made to itself goes through channel arbitration before being activated again. 1 A minor loop channel link made to itself does not go through channel arbitration before being activated again. Upon minor loop completion, the channel activates again if that channel has a minor loop channel link enabled and the link channel is itself. This effectively applies the minor loop offsets and restarts the next minor loop.
5 HALT	Halt DMA Operations 0 Normal operation 1 Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.

Table continues on the next page...

DMA_CR field descriptions (continued)

Field	Description
4 HOE	Halt On Error 0 Normal operation 1 Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
3 Reserved	This field is reserved. Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0 Fixed priority arbitration is used for channel selection . 1 Round robin arbitration is used for channel selection .
1 EDBG	Enable Debug 0 When in debug mode, the DMA continues to operate. 1 When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
0 Reserved	This field is reserved. Reserved

12.3.6 Error Status Register (DMA_ES)

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor, or
  - An illegal priority register setting in fixed-arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error bit that will be set when a transfer is canceled via the corresponding cancel transfer control bit

See Fault reporting and handling for more details.

Address: 4000_8000h base + 4h offset = 4000_8004h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CPE	0		ERRCHN				SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_ES field descriptions

Field	Description
31 VLD	Logical OR of all ERR status bits 0 No ERR bits are set. 1 At least one ERR bit is set indicating a valid error exists that has not been cleared.
30–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ECX	Transfer Canceled 0 No canceled transfers 1 The last recorded entry was a canceled transfer by the error cancel transfer input
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 CPE	Channel Priority Error 0 No channel priority error 1 The last recorded error was a configuration error in the channel priorities . Channel priorities are not unique.
13–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error, excluding CPE errors, or last recorded error canceled transfer.
7 SAE	Source Address Error 0 No source address configuration error. 1 The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0 No source offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error 0 No destination address configuration error 1 The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0 No destination offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0 No NBYTES/CITER configuration error 1 The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or

Table continues on the next page...

DMA_ES field descriptions (continued)

Field	Description
	• TCDn_CITER[CITER] is equal to zero, or • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0 No scatter/gather configuration error 1 The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0 No source bus error 1 The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0 No destination bus error 1 The last recorded error was a bus error on a destination write

12.3.7 Enable Request Register (DMA_ERQ)

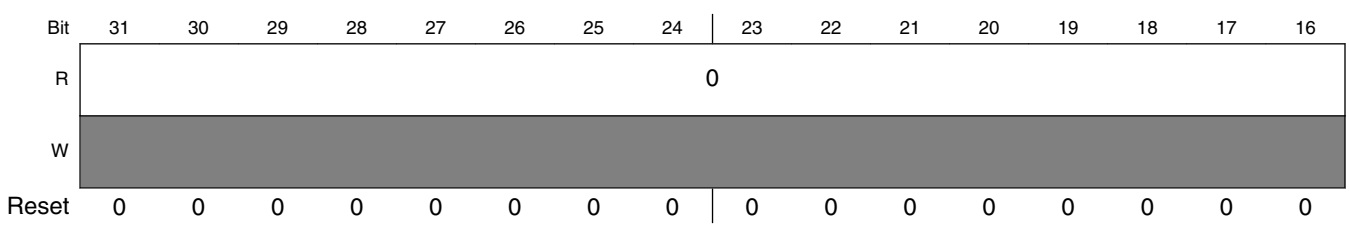
The ERQ register provides a bit map for the 16 channels to enable the request signal for each channel. The state of any given channel enable is directly affected by writes to this register; it is also affected by writes to the SERQ and CERQ registers. These registers are provided so the request enable for a single channel can easily be modified without needing to perform a read-modify-write sequence to the ERQ.

DMA request input signals and this enable request flag must be asserted before a channel's hardware service request is accepted. The state of the DMA enable request flag does not affect a channel service request made explicitly through software or a linked channel request.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 4000_8000h base + Ch offset = 4000_800Ch



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ERQ15	ERQ14	ERQ13	ERQ12	ERQ11	ERQ10	ERQ9	ERQ8	ERQ7	ERQ6	ERQ5	ERQ4	ERQ3	ERQ2	ERQ1	ERQ0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_ERQ field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 ERQ15	Enable DMA Request 15 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
14 ERQ14	Enable DMA Request 14 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
13 ERQ13	Enable DMA Request 13 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
12 ERQ12	Enable DMA Request 12 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
11 ERQ11	Enable DMA Request 11 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
10 ERQ10	Enable DMA Request 10 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
9 ERQ9	Enable DMA Request 9 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
8 ERQ8	Enable DMA Request 8 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
7 ERQ7	Enable DMA Request 7 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
6 ERQ6	Enable DMA Request 6

Table continues on the next page...

DMA_ERQ field descriptions (continued)

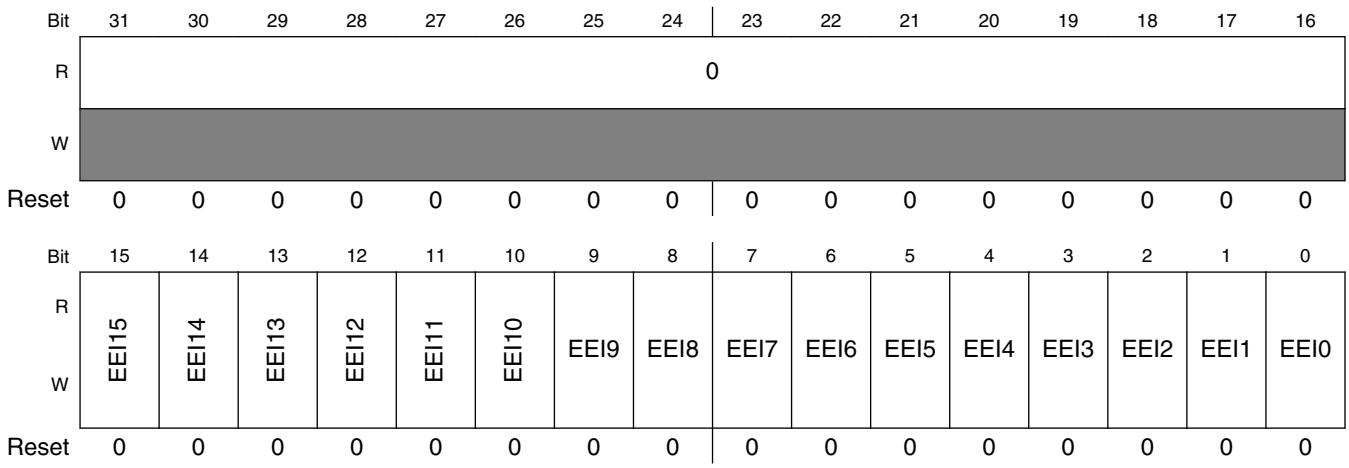
Field	Description
	0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
5 ERQ5	Enable DMA Request 5 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
4 ERQ4	Enable DMA Request 4 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
3 ERQ3	Enable DMA Request 3 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
2 ERQ2	Enable DMA Request 2 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
1 ERQ1	Enable DMA Request 1 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
0 ERQ0	Enable DMA Request 0 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

12.3.8 Enable Error Interrupt Register (DMA_EEI)

The EEI register provides a bit map for the 16 channels to enable the error interrupt signal for each channel. The state of any given channel's error interrupt enable is directly affected by writes to this register; it is also affected by writes to the SEEI and CEEI. These registers are provided so that the error interrupt enable for a single channel can easily be modified without the need to perform a read-modify-write sequence to the EEI register.

The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

Address: 4000_8000h base + 14h offset = 4000_8014h



DMA_EEI field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 EEI15	Enable Error Interrupt 15 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
14 EEI14	Enable Error Interrupt 14 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
13 EEI13	Enable Error Interrupt 13 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
12 EEI12	Enable Error Interrupt 12 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
11 EEI11	Enable Error Interrupt 11 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
10 EEI10	Enable Error Interrupt 10 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
9 EEI9	Enable Error Interrupt 9 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
8 EEI8	Enable Error Interrupt 8

Table continues on the next page...

DMA_EEI field descriptions (continued)

Field	Description
	0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
7 EEI7	Enable Error Interrupt 7 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
2 EEI2	Enable Error Interrupt 2 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

12.3.9 Clear Enable Error Interrupt Register (DMA_CEEI)

The CEEI provides a simple memory-mapped mechanism to clear a given bit in the EEI to disable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be cleared. Setting the CAEE bit provides a global clear function, forcing the EEI contents to be cleared, disabling all DMA request inputs. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 18h offset = 4000_8018h

Bit	7	6	5	4	3	2	1	0
Read	0	0	0					
Write	NOP	CAEE	0		CEEI			
Reset	0	0	0	0	0	0	0	0

DMA_CEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEE	Clear All Enable Error Interrupts 0 Clear only the EEI bit specified in the CEEI field 1 Clear all bits in EEI
5–4 Reserved	This field is reserved.
CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

12.3.10 Set Enable Error Interrupt Register (DMA_SEEI)

The SEEI provides a simple memory-mapped mechanism to set a given bit in the EEI to enable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be set. Setting the SAEE bit provides a global set function, forcing the entire EEI contents to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 19h offset = 4000_8019h

Bit	7	6	5	4	3	2	1	0
Read	0	0	0					
Write	NOP	SAEE	0		SEEI			
Reset	0	0	0	0	0	0	0	0

DMA_SEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAEE	Sets All Enable Error Interrupts 0 Set only the EEI bit specified in the SEEI field. 1 Sets all bits in EEI
5–4 Reserved	This field is reserved.
SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

12.3.11 Clear Enable Request Register (DMA_CERQ)

The CERQ provides a simple memory-mapped mechanism to clear a given bit in the ERQ to disable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be cleared. Setting the CAER bit provides a global clear function, forcing the entire contents of the ERQ to be cleared, disabling all DMA request inputs. If NOP is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 4000_8000h base + 1Ah offset = 4000_801Ah

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CAER	0		CERQ			
Reset	0	0	0	0	0	0	0	0

DMA_CERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAER	Clear All Enable Requests 0 Clear only the ERQ bit specified in the CERQ field 1 Clear all bits in ERQ
5–4 Reserved	This field is reserved.
CERQ	Clear Enable Request Clears the corresponding bit in ERQ.

12.3.12 Set Enable Request Register (DMA_SERQ)

The SERQ provides a simple memory-mapped mechanism to set a given bit in the ERQ to enable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be set. Setting the SAER bit provides a global set function, forcing the entire contents of ERQ to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Bh offset = 4000_801Bh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	SAER	0		SERQ			
Reset	0	0	0	0	0	0	0	0

DMA_SERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAER	Set All Enable Requests 0 Set only the ERQ bit specified in the SERQ field 1 Set all bits in ERQ
5–4 Reserved	This field is reserved.
SERQ	Set Enable Request Sets the corresponding bit in ERQ.

12.3.13 Clear DONE Status Bit Register (DMA_CDNE)

The CDNE provides a simple memory-mapped mechanism to clear the DONE bit in the TCD of the given channel. The data value on a register write causes the DONE bit in the corresponding transfer control descriptor to be cleared. Setting the CADN bit provides a global clear function, forcing all DONE bits to be cleared. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Ch offset = 4000_801Ch

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CADN	0		CDNE			
Reset	0	0	0	0	0	0	0	0

DMA_CDNE field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CADN	Clears All DONE Bits 0 Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1 Clears all bits in TCDn_CSR[DONE]
5–4 Reserved	This field is reserved.

Table continues on the next page...

DMA_CDNE field descriptions (continued)

Field	Description
CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

12.3.14 Set START Bit Register (DMA_SSRT)

The SSRT provides a simple memory-mapped mechanism to set the START bit in the TCD of the given channel. The data value on a register write causes the START bit in the corresponding transfer control descriptor to be set. Setting the SAST bit provides a global set function, forcing all START bits to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Dh offset = 4000_801Dh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	SAST	0		SSRT			
Reset	0	0	0	0	0	0	0	0

DMA_SSRT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAST	Set All START Bits (activates all channels) 0 Set only the TCDn_CSR[START] bit specified in the SSRT field 1 Set all bits in TCDn_CSR[START]
5–4 Reserved	This field is reserved.
SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

12.3.15 Clear Error Register (DMA_CERR)

The CERR provides a simple memory-mapped mechanism to clear a given bit in the ERR to disable the error condition flag for a given channel. The given value on a register write causes the corresponding bit in the ERR to be cleared. Setting the CAEI bit provides a global clear function, forcing the ERR contents to be cleared, clearing all channel error indicators. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Eh offset = 4000_801Eh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CAEI	0		CERR			
Reset	0	0	0	0	0	0	0	0

DMA_CERR field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEI	Clear All Error Indicators 0 Clear only the ERR bit specified in the CERR field 1 Clear all bits in ERR
5–4 Reserved	This field is reserved.
CERR	Clear Error Indicator Clears the corresponding bit in ERR

12.3.16 Clear Interrupt Request Register (DMA_CINT)

The CINT provides a simple, memory-mapped mechanism to clear a given bit in the INT to disable the interrupt request for a given channel. The given value on a register write causes the corresponding bit in the INT to be cleared. Setting the CAIR bit provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Fh offset = 4000_801Fh

Bit	7	6	5	4	3	2	1	0
Read	0	0	0					
Write	NOP	CAIR	0		CINT			
Reset	0	0	0	0	0	0	0	0

DMA_CINT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAIR	Clear All Interrupt Requests 0 Clear only the INT bit specified in the CINT field 1 Clear all bits in INT
5–4 Reserved	This field is reserved.
CINT	Clear Interrupt Request Clears the corresponding bit in INT

12.3.17 Interrupt Request Register (DMA_INT)

The INT register provides a bit map for the 16 channels signaling the presence of an interrupt request for each channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion. The outputs of this register are directly routed to the interrupt controller. During the interrupt-service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. Typically, a write to the CINT register in the interrupt service routine is used for this purpose.

The state of any given channel's interrupt request is directly affected by writes to this register; it is also affected by writes to the CINT register. On writes to INT, a 1 in any bit position clears the corresponding channel's interrupt request. A zero in any bit position has no affect on the corresponding channel's current interrupt status. The CINT register is provided so the interrupt request for a single channel can easily be cleared without the need to perform a read-modify-write sequence to the INT register.

Address: 4000_8000h base + 24h offset = 4000_8024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	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	INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_INT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
15 INT15	Interrupt Request 15 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
14 INT14	Interrupt Request 14 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
13 INT13	Interrupt Request 13 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
12 INT12	Interrupt Request 12 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
11 INT11	Interrupt Request 11 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
10 INT10	Interrupt Request 10 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
9 INT9	Interrupt Request 9 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
8 INT8	Interrupt Request 8 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
7 INT7	Interrupt Request 7 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
6 INT6	Interrupt Request 6 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
5 INT5	Interrupt Request 5 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
4 INT4	Interrupt Request 4 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

Table continues on the next page...

DMA_INT field descriptions (continued)

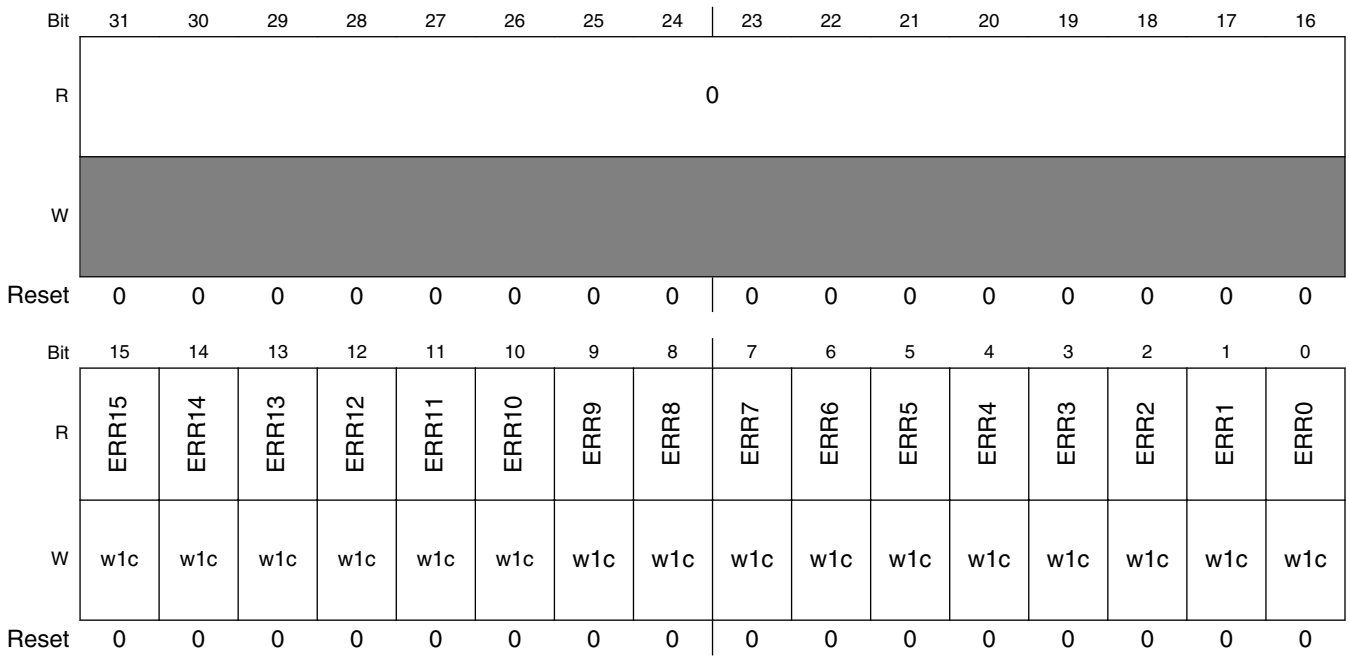
Field	Description
3 INT3	Interrupt Request 3 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
2 INT2	Interrupt Request 2 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
1 INT1	Interrupt Request 1 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
0 INT0	Interrupt Request 0 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

12.3.18 Error Register (DMA_ERR)

The ERR provides a bit map for the 16 channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI, and then routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. Typically, a write to the CERR in the interrupt-service routine is used for this purpose. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected.

The contents of this register can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI. The state of any given channel's error indicators is affected by writes to this register; it is also affected by writes to the CERR. On writes to the ERR, a one in any bit position clears the corresponding channel's error status. A zero in any bit position has no affect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be cleared.

Address: 4000_8000h base + 2Ch offset = 4000_802Ch



DMA_ERR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 ERR15	Error In Channel 15 0 An error in this channel has not occurred 1 An error in this channel has occurred
14 ERR14	Error In Channel 14 0 An error in this channel has not occurred 1 An error in this channel has occurred
13 ERR13	Error In Channel 13 0 An error in this channel has not occurred 1 An error in this channel has occurred
12 ERR12	Error In Channel 12 0 An error in this channel has not occurred 1 An error in this channel has occurred
11 ERR11	Error In Channel 11 0 An error in this channel has not occurred 1 An error in this channel has occurred
10 ERR10	Error In Channel 10 0 An error in this channel has not occurred 1 An error in this channel has occurred

Table continues on the next page...

DMA_ERR field descriptions (continued)

Field	Description
9 ERR9	Error In Channel 9 0 An error in this channel has not occurred 1 An error in this channel has occurred
8 ERR8	Error In Channel 8 0 An error in this channel has not occurred 1 An error in this channel has occurred
7 ERR7	Error In Channel 7 0 An error in this channel has not occurred 1 An error in this channel has occurred
6 ERR6	Error In Channel 6 0 An error in this channel has not occurred 1 An error in this channel has occurred
5 ERR5	Error In Channel 5 0 An error in this channel has not occurred 1 An error in this channel has occurred
4 ERR4	Error In Channel 4 0 An error in this channel has not occurred 1 An error in this channel has occurred
3 ERR3	Error In Channel 3 0 An error in this channel has not occurred 1 An error in this channel has occurred
2 ERR2	Error In Channel 2 0 An error in this channel has not occurred 1 An error in this channel has occurred
1 ERR1	Error In Channel 1 0 An error in this channel has not occurred 1 An error in this channel has occurred
0 ERR0	Error In Channel 0 0 An error in this channel has not occurred 1 An error in this channel has occurred

12.3.19 Hardware Request Status Register (DMA_HRS)

The HRS register provides a bit map for the DMA channels, signaling the presence of a hardware request for each channel. The hardware request status bits reflect the current state of the register and qualified (via the ERQ fields) DMA request signals as seen by the DMA's arbitration logic. This view into the hardware request signals may be used for debug purposes.

NOTE

These bits reflect the state of the request as seen by the arbitration logic. Therefore, this status is affected by the ERQ bits.

Address: 4000_8000h base + 34h offset = 4000_8034h

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	HRS15	HRS14	HRS13	HRS12	HRS11	HRS10	HRS9	HRS8	HRS7	HRS6	HRS5	HRS4	HRS3	HRS2	HRS1	HRS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_HRS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 HRS15	Hardware Request Status Channel 15

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 15 is not present 1 A hardware service request for channel 15 is present
14 HRS14	Hardware Request Status Channel 14 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 14 is not present 1 A hardware service request for channel 14 is present
13 HRS13	Hardware Request Status Channel 13 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 13 is not present 1 A hardware service request for channel 13 is present
12 HRS12	Hardware Request Status Channel 12 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 12 is not present 1 A hardware service request for channel 12 is present
11 HRS11	Hardware Request Status Channel 11 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 11 is not present 1 A hardware service request for channel 11 is present
10 HRS10	Hardware Request Status Channel 10 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 10 is not present 1 A hardware service request for channel 10 is present
9 HRS9	Hardware Request Status Channel 9 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	0 A hardware service request for channel 9 is not present 1 A hardware service request for channel 9 is present
8 HRS8	Hardware Request Status Channel 8 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 8 is not present 1 A hardware service request for channel 8 is present
7 HRS7	Hardware Request Status Channel 7 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 7 is not present 1 A hardware service request for channel 7 is present
6 HRS6	Hardware Request Status Channel 6 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 6 is not present 1 A hardware service request for channel 6 is present
5 HRS5	Hardware Request Status Channel 5 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 5 is not present 1 A hardware service request for channel 5 is present
4 HRS4	Hardware Request Status Channel 4 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 4 is not present 1 A hardware service request for channel 4 is present
3 HRS3	Hardware Request Status Channel 3 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 3 is not present 1 A hardware service request for channel 3 is present
2 HRS2	Hardware Request Status Channel 2

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 2 is not present 1 A hardware service request for channel 2 is present
1 HRS1	Hardware Request Status Channel 1 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 1 is not present 1 A hardware service request for channel 1 is present
0 HRS0	Hardware Request Status Channel 0 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 0 is not present 1 A hardware service request for channel 0 is present

12.3.20 Enable Asynchronous Request in Stop Register (DMA_EARS)

Address: 4000_8000h base + 44h offset = 4000_8044h

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	EDREQ_15	EDREQ_14	EDREQ_13	EDREQ_12	EDREQ_11	EDREQ_10	EDREQ_9	EDREQ_8	EDREQ_7	EDREQ_6	EDREQ_5	EDREQ_4	EDREQ_3	EDREQ_2	EDREQ_1	EDREQ_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_EARS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

DMA_EARS field descriptions (continued)

Field	Description
15 EDREQ_15	Enable asynchronous DMA request in stop mode for channel 15 0 Disable asynchronous DMA request for channel 15. 1 Enable asynchronous DMA request for channel 15.
14 EDREQ_14	Enable asynchronous DMA request in stop mode for channel 14 0 Disable asynchronous DMA request for channel 14. 1 Enable asynchronous DMA request for channel 14.
13 EDREQ_13	Enable asynchronous DMA request in stop mode for channel 13 0 Disable asynchronous DMA request for channel 13. 1 Enable asynchronous DMA request for channel 13.
12 EDREQ_12	Enable asynchronous DMA request in stop mode for channel 12 0 Disable asynchronous DMA request for channel 12. 1 Enable asynchronous DMA request for channel 12.
11 EDREQ_11	Enable asynchronous DMA request in stop mode for channel 11 0 Disable asynchronous DMA request for channel 11. 1 Enable asynchronous DMA request for channel 11.
10 EDREQ_10	Enable asynchronous DMA request in stop mode for channel 10 0 Disable asynchronous DMA request for channel 10. 1 Enable asynchronous DMA request for channel 10.
9 EDREQ_9	Enable asynchronous DMA request in stop mode for channel 9 0 Disable asynchronous DMA request for channel 9. 1 Enable asynchronous DMA request for channel 9.
8 EDREQ_8	Enable asynchronous DMA request in stop mode for channel 8 0 Disable asynchronous DMA request for channel 8. 1 Enable asynchronous DMA request for channel 8.
7 EDREQ_7	Enable asynchronous DMA request in stop mode for channel 7 0 Disable asynchronous DMA request for channel 7. 1 Enable asynchronous DMA request for channel 7.
6 EDREQ_6	Enable asynchronous DMA request in stop mode for channel 6 0 Disable asynchronous DMA request for channel 6. 1 Enable asynchronous DMA request for channel 6.
5 EDREQ_5	Enable asynchronous DMA request in stop mode for channel 5 0 Disable asynchronous DMA request for channel 5. 1 Enable asynchronous DMA request for channel 5.
4 EDREQ_4	Enable asynchronous DMA request in stop mode for channel 4 0 Disable asynchronous DMA request for channel 4. 1 Enable asynchronous DMA request for channel 4.

Table continues on the next page...

DMA_EARS field descriptions (continued)

Field	Description
3 EDREQ_3	Enable asynchronous DMA request in stop mode for channel 3. 0 Disable asynchronous DMA request for channel 3. 1 Enable asynchronous DMA request for channel 3.
2 EDREQ_2	Enable asynchronous DMA request in stop mode for channel 2. 0 Disable asynchronous DMA request for channel 2. 1 Enable asynchronous DMA request for channel 2.
1 EDREQ_1	Enable asynchronous DMA request in stop mode for channel 1. 0 Disable asynchronous DMA request for channel 1 1 Enable asynchronous DMA request for channel 1.
0 EDREQ_0	Enable asynchronous DMA request in stop mode for channel 0. 0 Disable asynchronous DMA request for channel 0. 1 Enable asynchronous DMA request for channel 0.

12.3.21 Channel n Priority Register (DMA_DCHPRI*n*)

When fixed-priority channel arbitration is enabled (CR[ERCA] = 0), the contents of these registers define the unique priorities associated with each channel. The channel priorities are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, a configuration error is reported. The range of the priority value is limited to the values of 0 through 15.

Address: 4000_8000h base + 100h offset + (1d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	ECP	DPA	0		CHPRI			
Write								
Reset	0	0	0	0	*	*	*	*

* Notes:
- CHPRI field: See bit field description.

DMA_DCHPRI*n* field descriptions

Field	Description
7 ECP	Enable Channel Preemption. 0 Channel n cannot be suspended by a higher priority channel's service request. 1 Channel n can be temporarily suspended by the service request of a higher priority channel.

Table continues on the next page...

DMA_DCHPRI*n* field descriptions (continued)

Field	Description
6 DPA	Disable Preempt Ability. 0 Channel n can suspend a lower priority channel. 1 Channel n cannot suspend any channel, regardless of channel priority.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled **NOTE:** Reset value for the channel priority field, CHPRI, is equal to the corresponding channel number for each priority register, that is, DCHPRI15[CHPRI] = 0b1111.

12.3.22 TCD Source Address (DMA_TCD*n*_SADDR)

Address: 4000_8000h base + 1000h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_SADDR field descriptions

Field	Description
SADDR	Source Address Memory address pointing to the source data.

12.3.23 TCD Signed Source Address Offset (DMA_TCD*n*_SOFF)

Address: 4000_8000h base + 1004h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_SOFF field descriptions

Field	Description
SOFF	Source address signed offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

12.3.24 TCD Transfer Attributes (DMA_TCD*n*_ATTR)

Address: 4000_8000h base + 1006h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SMOD					SSIZE			DMOD					DSIZE		
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_ATTR field descriptions

Field	Description
15–11 SMOD	Source Address Modulo 0 Source address modulo feature is disabled ≠0 This value defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range.
10–8 SSIZE	Source data transfer size **NOTE:** Using a Reserved value causes a configuration error. The eDMA defaults to privileged data access for all transactions. 000 8-bit 001 16-bit 010 32-bit 011 Reserved 100 16-byte burst 101 32-byte burst 110 Reserved 111 Reserved
7–3 DMOD	Destination Address Modulo See the SMOD definition

Table continues on the next page...

DMA_TCD*n*_ATTR field descriptions (continued)

Field	Description
DSIZE	Destination data transfer size See the SSIZE definition

12.3.25 TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD*n*_NBYTES_MLNO)

This register, or one of the next two registers (TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is disabled (CR[EMLM] = 0)

If minor loop mapping is enabled, see the TCD_NBYTES_MLOFFNO and TCD_NBYTES_MLOFFYES register descriptions for the definition of TCD word 2.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLNO field descriptions

Field	Description
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed. **NOTE:** An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.

12.3.26 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD*n*_NBYTES_MLOFFNO)

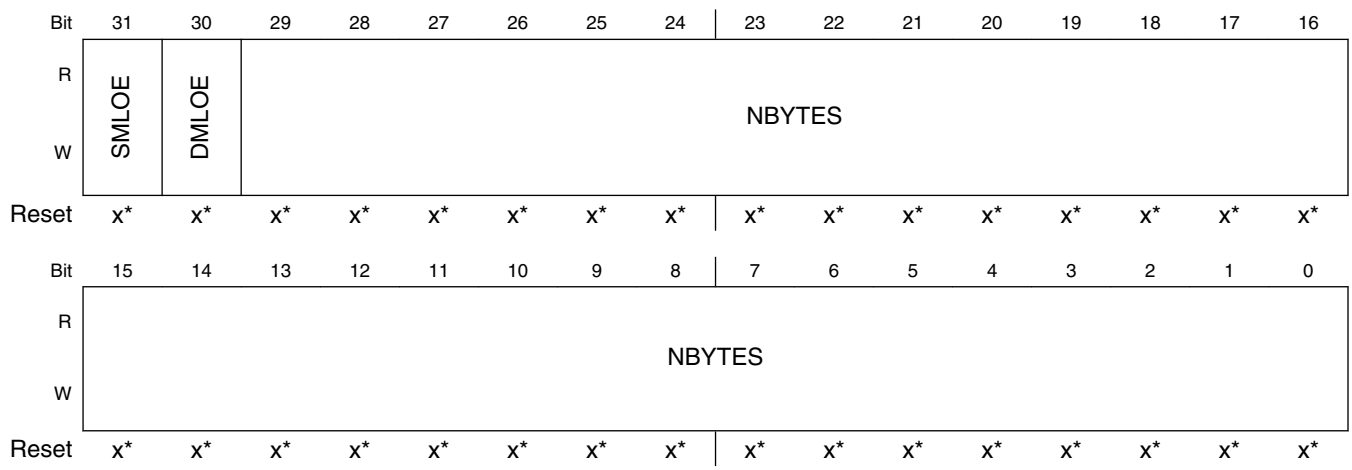
One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- SMLOE = 0 and DMLOE = 0

If minor loop mapping is enabled and SMLOE or DMLOE is set, then refer to the TCD_NBYTES_MLOFFYES register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d



* Notes:
- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLOFFNO field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion.

Table continues on the next page...

DMA_TCD*n*_NBYTES_MLOFFNO field descriptions (continued)

Field	Description
	0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

12.3.27 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD*n*_NBYTES_MLOFFYES)

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFNO), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset is enabled (SMLOE or DMLOE = 1)

If minor loop mapping is enabled and SMLOE and DMLOE are cleared, then refer to the TCD_NBYTES_MLOFFNO register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d

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

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLOFFYES field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
29–10 MLOFF	If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

12.3.28 TCD Last Source Address Adjustment (DMA_TCD*n*_SLAST)

Address: 4000_8000h base + 100Ch offset + (32d × i), where i=0d to 15d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_SLAST field descriptions

Field	Description
SLAST	Last Source Address Adjustment Adjustment value added to the source address at the completion of the major iteration count. This value can be applied to restore the source address to the initial value, or adjust the address to reference the next data structure. This register uses two's complement notation; the overflow bit is discarded.

12.3.29 TCD Destination Address (DMA_TCD*n*_DADDR)

Address: 4000_8000h base + 1010h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DADDR field descriptions

Field	Description
DADDR	Destination Address Memory address pointing to the destination data.

12.3.30 TCD Signed Destination Address Offset (DMA_TCD*n*_DOFF)

Address: 4000_8000h base + 1014h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DOFF field descriptions

Field	Description
DOFF	Destination Address Signed Offset Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

12.3.31 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD*n*_CITER_ELINKYES)

If TCDn_CITER[ELINK] is set, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			CITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–13 Reserved	This field is reserved.
12–9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.

Table continues on the next page...

DMA_TCD*n*_CITER_ELINKYES field descriptions (continued)

Field	Description
	NOTE: When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

12.3.32 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD*n*_CITER_ELINKNO)

If TCDn_CITER[ELINK] is cleared, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	CITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for

Table continues on the next page...

DMA_TCD*n*_CITER_ELINKNO field descriptions (continued)

Field	Description
	example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

12.3.33 TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD*n*_DLASTSGA)

Address: 4000_8000h base + 1018h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DLASTSGA field descriptions

Field	Description
DLASTSGA	Destination last address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.

12.3.34 TCD Control and Status (DMA_TCD*n*_CSR)

Address: 4000_8000h base + 101Ch offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read	BWC				MAJORLINKCH			
Write			0					
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read	DONE	ACTIVE	MAJORELINK	ESG	DREQ	INTHALF	INTMAJOR	START
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CSR field descriptions

Field	Description
15–14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces the eDMA to stall after the completion of each read/write access to control the bus request bandwidth seen by the crossbar switch. **NOTE:** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00 No eDMA engine stalls. 01 Reserved 10 eDMA engine stalls for 4 cycles after each R/W. 11 eDMA engine stalls for 8 cycles after each R/W.
13–12 Reserved	This field is reserved.
11–8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7 DONE	Channel Done This flag indicates the eDMA has completed the major loop. The eDMA engine sets it as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated. The access of this field is W0C, write-zero-to-clear.

Table continues on the next page...

DMA_TCD*n*_CSR field descriptions (continued)

Field	Description
	NOTE: This bit must be cleared to write the MAJORELINK or ESG bits.
6 ACTIVE	Channel Active This flag signals the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.
5 MAJORELINK	Enable channel-to-channel linking on major loop complete As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. **NOTE:** To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The channel-to-channel linking is disabled. 1 The channel-to-channel linking is enabled.
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory. **NOTE:** To support the dynamic scatter/gather coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The current channel's TCD is normal format. 1 The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.
3 DREQ	Disable Request If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero. 0 The channel's ERQ bit is not affected. 1 The channel's ERQ bit is cleared when the major loop is complete.
2 INTHALF	Enable an interrupt when major counter is half complete. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER >> 1)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE:** If BITER = 1, do not use INTHALF. Use INTMAJOR instead. 0 The half-point interrupt is disabled. 1 The half-point interrupt is enabled.
1 INTMAJOR	Enable an interrupt when major iteration count completes. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count reaches zero. 0 The end-of-major loop interrupt is disabled. 1 The end-of-major loop interrupt is enabled.

Table continues on the next page...

DMA_TCD*n*_CSR field descriptions (continued)

Field	Description
0 START	Channel Start If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution. 0 The channel is not explicitly started. 1 The channel is explicitly started via a software initiated service request.

12.3.35 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD*n*_BITER_ELINKYES)

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			BITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–13 Reserved	This field is reserved.
12–9 LINKCH	Link Channel Number

Table continues on the next page...

DMA_TCD*n*_BITER_ELINKYES field descriptions (continued)

Field	Description
	If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.
BITER	Starting major iteration count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

12.3.36 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD*n*_BITER_ELINKNO)

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	BITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.

Table continues on the next page...

DMA_TCD*n*_BITER_ELINKNO field descriptions (continued)

Field	Description
	NOTE: When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

12.4 Functional description

The operation of the eDMA is described in the following subsections.

12.4.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

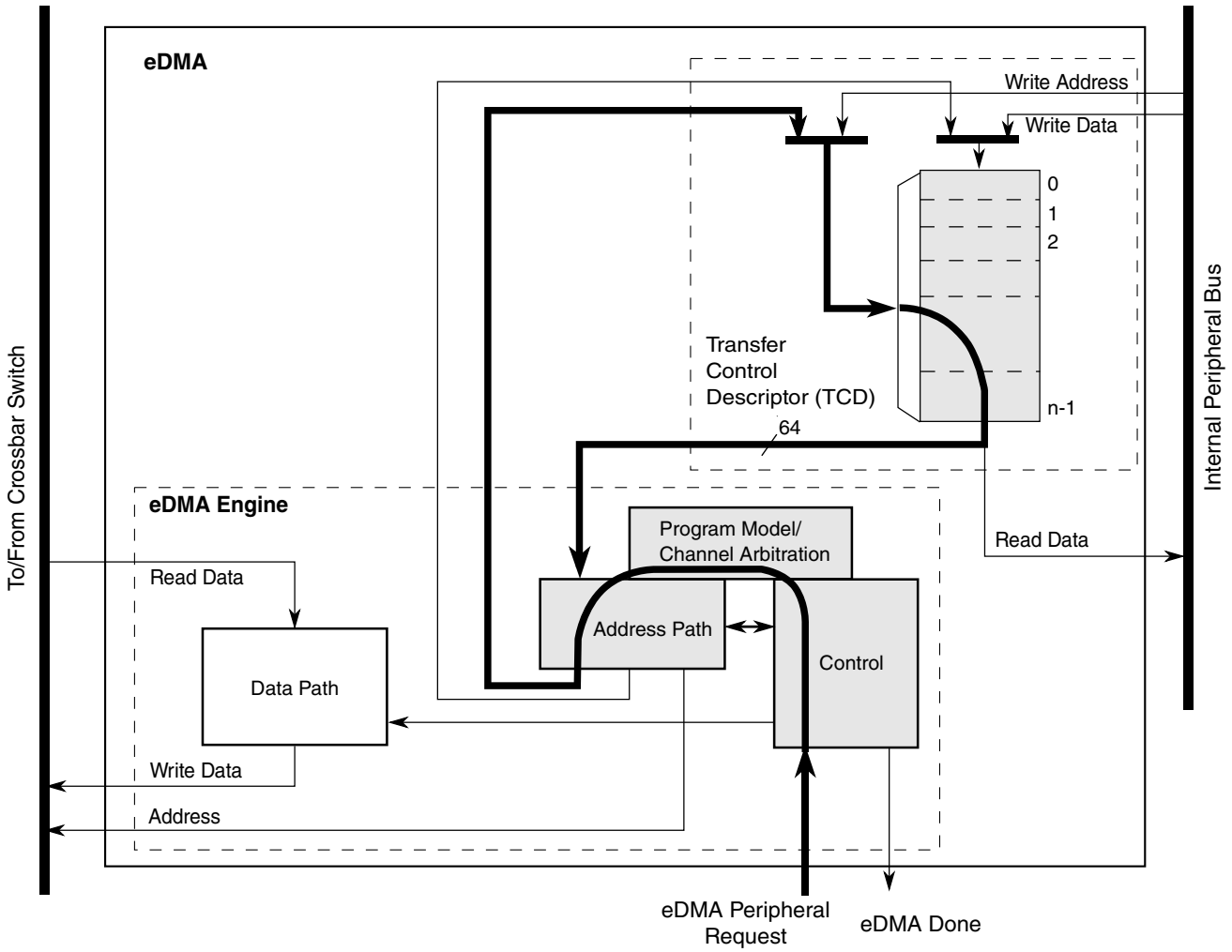


Figure 12-2. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration. In the next cycle, the channel arbitration performs, using the fixed-priority or round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path channel x or y registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path channel x or y registers.

The following diagram illustrates the second part of the basic data flow:

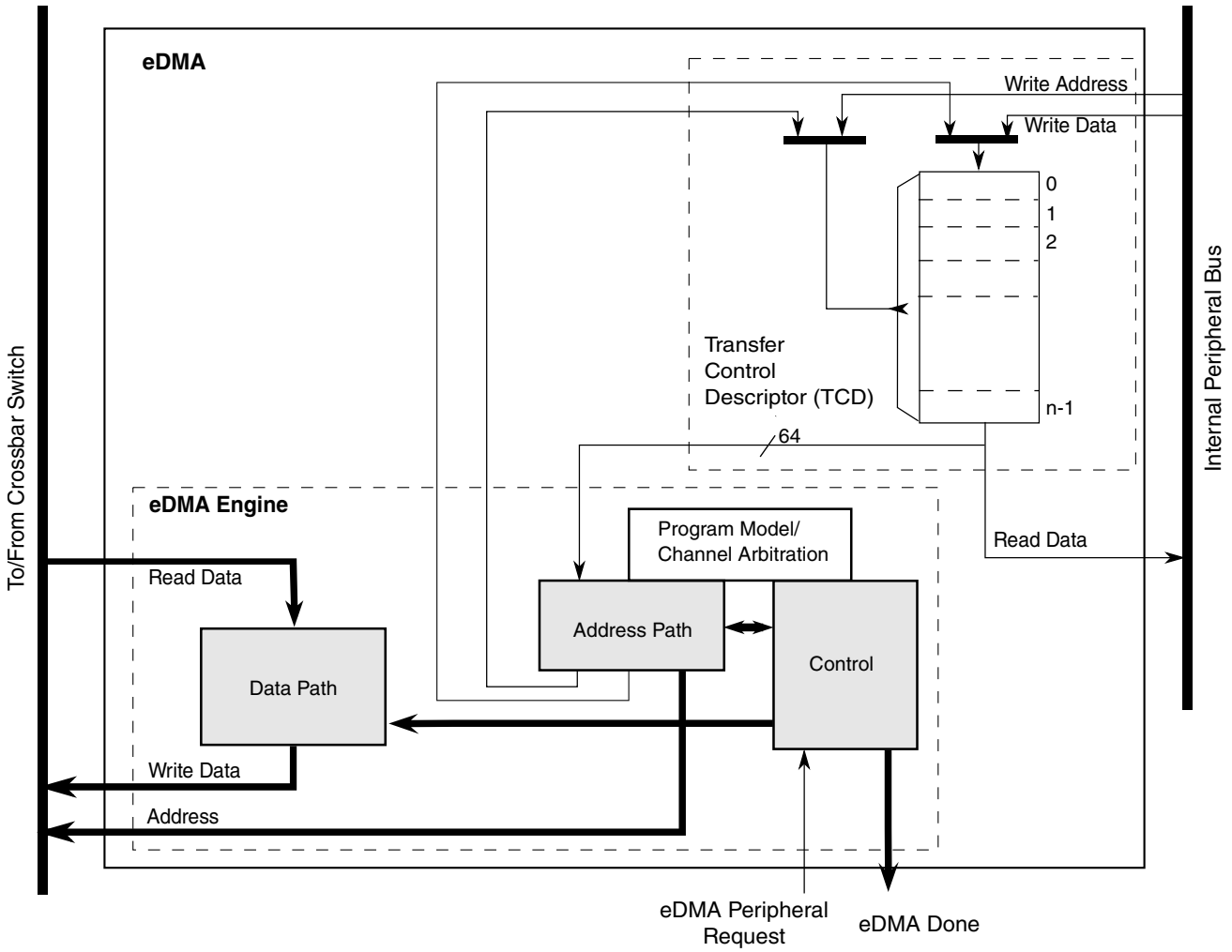


Figure 12-3. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the minor byte count has transferred.

After the minor byte count has moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

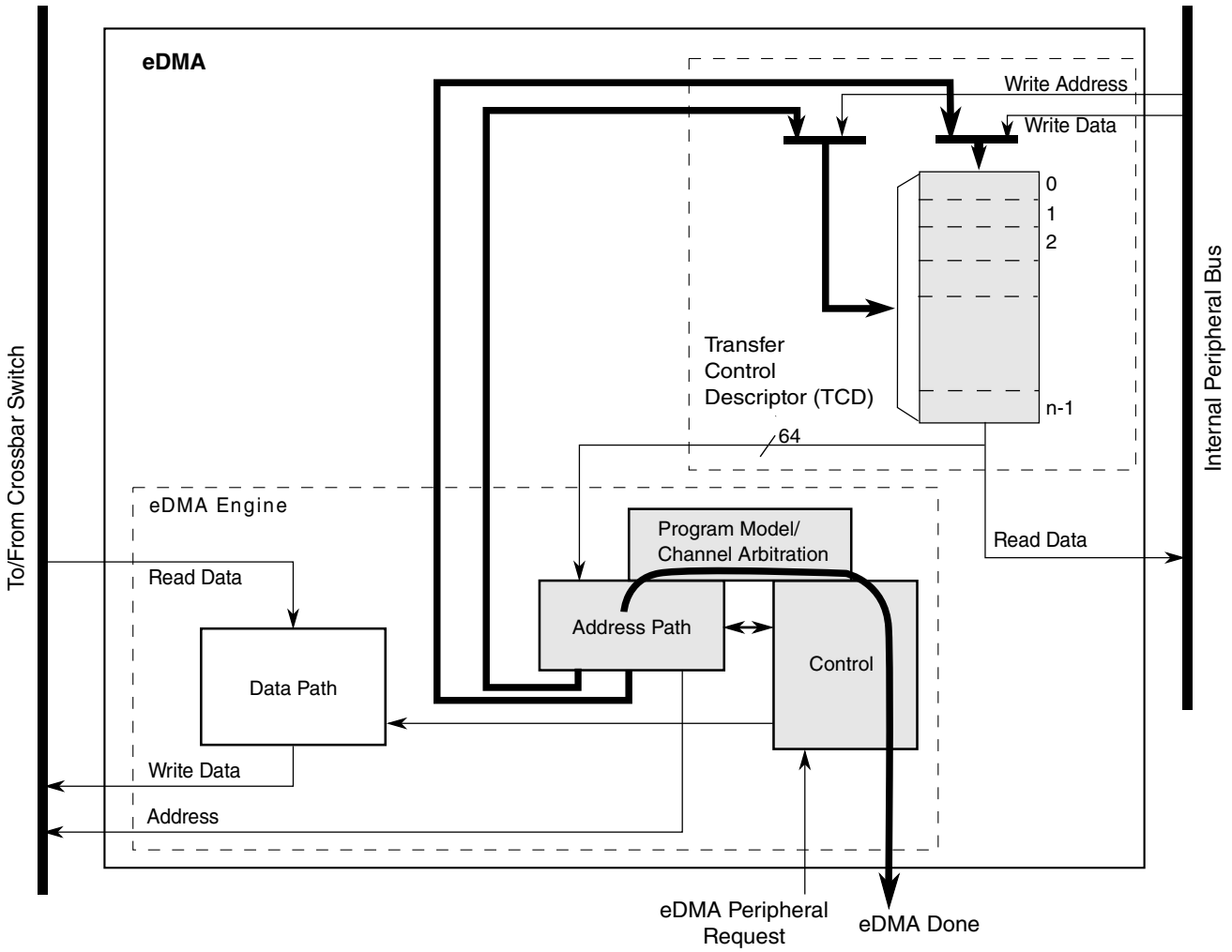


Figure 12-4. eDMA operation, part 3

12.4.2 Fault reporting and handling

Channel errors are reported in the Error Status register (DMAx_ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor or an illegal priority register setting in Fixed-Arbitration mode, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.

- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.
- In fixed arbitration mode, a configuration error is caused by any two channel priorities being equal. All channel priority levels must be unique when fixed arbitration mode is enabled.

NOTE

When two channels have the same priority, a channel priority error exists and will be reported in the Error Status register. However, the channel number will not be reported in the Error Status register. When all of the channel priorities within a group are not unique, the channel number selected by arbitration is undetermined.

To aid in Channel Priority Error (CPE) debug, set the Halt On Error bit in the DMA's Control Register. If all of the channel priorities within a group are not unique, the DMA will be halted after the CPE error is recorded. The DMA will remain halted and will not process any channel service requests. Once all of the channel priorities are set to unique numbers, the DMA may be enabled again by clearing the Halt bit.

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[E_LINK] bit does not equal the TCDn_BITER[E_LINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error

occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

A transfer may be cancelled by software with the CR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. The current read-write sequence is allowed to finish. If the cancel occurs on the last read-write sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (DMAx_ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into DMA_ES[ERRCHN] and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

NOTE

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed. The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The application software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (DMAx_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

12.4.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the DCHPRIn[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted. Preemption is available only when fixed arbitration is selected.

A channel's ability to preempt another channel can be disabled by setting DCHPRIn[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel.

12.4.4 Performance

This section addresses the performance of the eDMA module, focusing on two separate metrics:

- In the traditional data movement context, performance is best expressed as the peak data transfer rates achieved using the eDMA. In most implementations, this transfer rate is limited by the speed of the source and destination address spaces.
- In a second context where device-paced movement of single data values to/from peripherals is dominant, a measure of the requests that can be serviced in a fixed time is a more relevant metric. In this environment, the speed of the source and destination address spaces remains important. However, the microarchitecture of the eDMA also factors significantly into the resulting metric.

12.4.4.1 Peak transfer rates

The peak transfer rates for several different source and destination transfers are shown in the following tables. These tables assume:

- Internal SRAM can be accessed with zero wait-states when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states, when viewed from the system bus data phase
- All internal peripheral bus accesses are 32-bits in size

NOTE

All architectures will not meet the assumptions listed above. See the SRAM configuration section for more information.

This table compares peak transfer rates based on different possible system speeds. Specific chips/devices may not support all system speeds listed.

Table 12-4. eDMA peak transfer rates (Mbytes/sec)

System Speed, Width	Internal SRAM-to-Internal SRAM	32 bit internal peripheral bus-to-Internal SRAM	Internal SRAM-to-32 bit internal peripheral bus
48 MHz, 32 bit	96.0	48.0	38.4
66.7 MHz, 32 bit	133.3	66.7	53.3
83.3 MHz, 32 bit	166.7	83.3	66.7
100.0 MHz, 32 bit	200.0	100.0	80.0
133.3 MHz, 32 bit	266.7	133.3	106.7
150.0 MHz, 32 bit	300.0	150.0	120.0

Internal-SRAM-to-internal-SRAM transfers occur at the core's datapath width. For all transfers involving the internal peripheral bus, 32-bit transfer sizes are used. In all cases, the transfer rate includes the time to read the source plus the time to write the destination.

12.4.4.2 Peak request rates

The second performance metric is a measure of the number of DMA requests that can be serviced in a given amount of time. For this metric, assume that the peripheral request causes the channel to move a single internal peripheral bus-mapped operand to/from internal SRAM. The same timing assumptions used in the previous example apply to this calculation. In particular, this metric also reflects the time required to activate the channel.

The eDMA design supports the following hardware service request sequence. Note that the exact timing from Cycle 7 is a function of the response times for the channel's read and write accesses. In the case of an internal peripheral bus read and internal SRAM write, the combined data phase time is 4 cycles. For an SRAM read and internal peripheral bus write, it is 5 cycles.

Table 12-5. Hardware service request process

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
1		eDMA peripheral request is asserted.
2		The eDMA peripheral request is registered locally in the eDMA module and qualified. TCD*n*_CSR[START] bit initiated requests start at this point with the registering of the user write to TCD*n* word 7.
3		Channel arbitration begins.
4		Channel arbitration completes. The transfer control descriptor local memory read is initiated.
5–6		The first two parts of the activated channel's TCD is read from the local memory. The memory width to the eDMA engine is 64 bits, so the entire descriptor can be accessed in four cycles
7		The first system bus read cycle is initiated, as the third part of the channel's TCD is read from the local memory. Depending on the state of the crossbar switch, arbitration at the system bus may insert an additional cycle of delay here.
8–11	8–12	The last part of the TCD is read in. This cycle represents the first data phase for the read, and the address phase for the destination write.
12	13	This cycle represents the data phase of the last destination write.
13	14	The eDMA engine completes the execution of the inner minor loop and prepares to write back the required TCD*n* fields into the local memory. The TCD*n* word 7 is read and checked for channel linking or scatter/gather requests.
14	15	The appropriate fields in the first part of the TCD*n* are written back into the local memory.
15	16	The fields in the second part of the TCD*n* are written back into the local memory. This cycle coincides with the next channel arbitration cycle start.
16	17	The next channel to be activated performs the read of the first part of its TCD from the local memory. This is equivalent to Cycle 4 for the first channel's service request.

Assuming zero wait states on the system bus, DMA requests can be processed every 9 cycles. Assuming an average of the access times associated with internal peripheral bus-to-SRAM (4 cycles) and SRAM-to-internal peripheral bus (5 cycles), DMA requests can

be processed every 11.5 cycles (4 + (4+5)/2 + 3). This is the time from Cycle 4 to Cycle x +5. The resulting peak request rate, as a function of the system frequency, is shown in the following table.

Table 12-6. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
48.0	5.3	4.2
66.6	7.4	5.8
83.3	9.2	7.2
100.0	11.1	8.7
133.3	14.8	11.6
150.0	16.6	13.0

A general formula to compute the peak request rate with overlapping requests is:

PEAKreq = freq / [entry + (1 + read_ws) + (1 + write_ws) + exit]

where:

Table 12-7. Peak request formula operands

Operand	Description
PEAKreq	Peak request rate
freq	System frequency
entry	Channel startup (4 cycles)
read_ws	Wait states seen during the system bus read data phase
write_ws	Wait states seen during the system bus write data phase
exit	Channel shutdown (3 cycles)

12.4.4.3 eDMA performance example

Consider a system with the following characteristics:

- Internal SRAM can be accessed with one wait-state when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states viewed from the system bus data phase
- System operates at 150 MHz

For an SRAM to internal peripheral bus transfer,

PEAKreq = 150 MHz / [4 + (1 + 1) + (1 + 3) + 3] cycles = 11.5 Mreq/sec

For an internal peripheral bus to SRAM transfer,

PEAKreq = 150 MHz / [4 + (1 + 2) + (1 + 1) + 3] cycles = 12.5 Mreq/sec

Assuming an even distribution of the two transfer types, the average peak request rate would be:

PEAKreq = (11.5 Mreq/sec + 12.5 Mreq/sec) / 2 = 12.0 Mreq/sec

The minimum number of cycles to perform a single read/write, zero wait states on the system bus, from a cold start where no channel is executing and eDMA is idle are:

- 11 cycles for a software, that is, a TCD*n*_CSR[START] bit, request
- 12 cycles for a hardware, that is, an eDMA peripheral request signal, request

Two cycles account for the arbitration pipeline and one extra cycle on the hardware request resulting from the internal registering of the eDMA peripheral request signals. For the peak request rate calculations above, the arbitration and request registering is absorbed in or overlaps the previous executing channel.

Note

When channel linking or scatter/gather is enabled, a two cycle delay is imposed on the next channel selection and startup. This allows the link channel or the scatter/gather channel to be eligible and considered in the arbitration pool for next channel selection.

12.5 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

12.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CR if a configuration other than the default is desired.
2. Write the channel priority levels to the DCHPRI*n* registers if a configuration other than the default is desired.

3. Enable error interrupts in the EEI register if so desired.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the ERQ register.
6. Request channel service via either:
   - Software: setting the TCD*n*_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by TCD*n*_SADDR, to the destination, as defined by TCD*n*_DADDR, continue until the number of bytes specified by TCD*n*_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 12-8. TCD Control and Status fields

TCD*n*_CSR field name	Description
START	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
ACTIVE	Status bit indicating the channel is currently in execution
DONE	Status bit indicating major loop completion (cleared by software when using a software initiated DMA service)
D_REQ	Control bit to disable DMA request at end of major loop completion when using a hardware initiated DMA service
BWC	Control bits for throttling bandwidth control of a channel
E_SG	Control bit to enable scatter-gather feature
INT_HALF	Control bit to enable interrupt when major loop is half complete
INT_MAJ	Control bit to enable interrupt when major loop completes

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

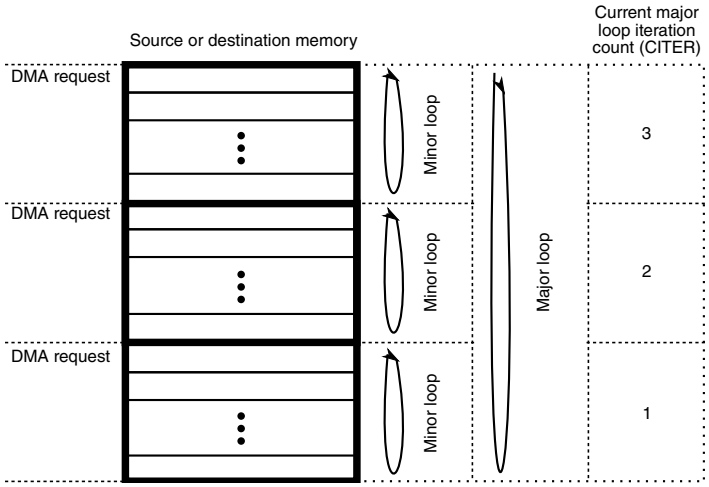


Figure 12-5. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

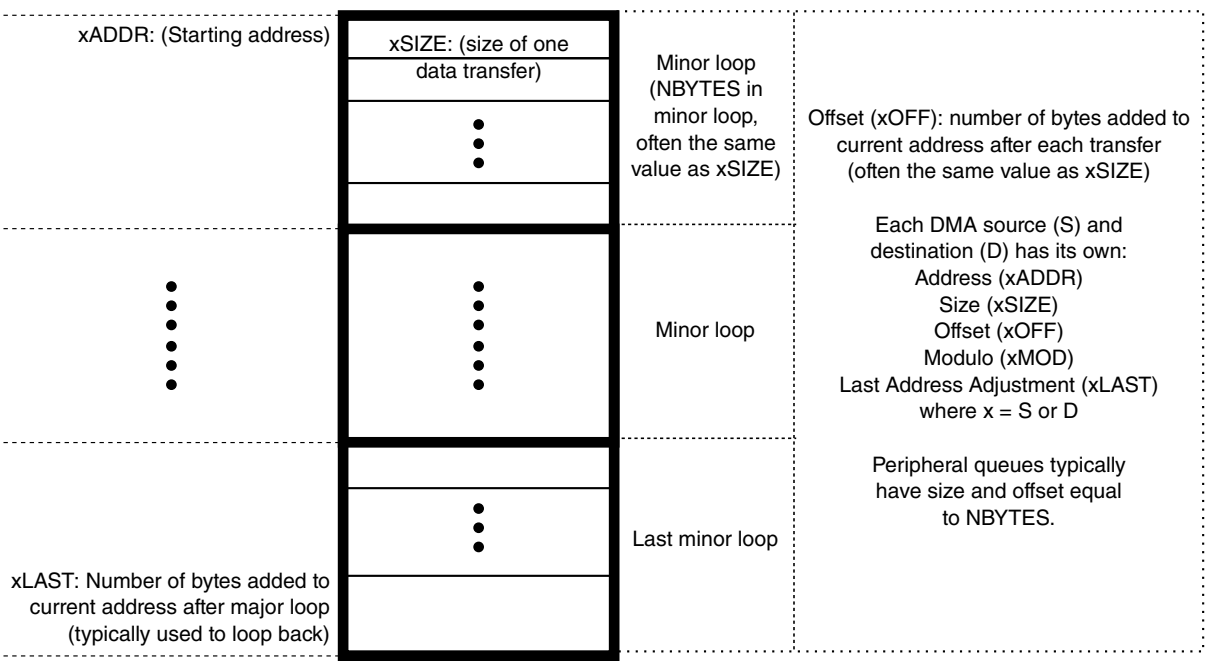


Figure 12-6. Memory array terms

12.5.2 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data. Most programming errors are reported on a per channel basis with the exception of channel priority error (ES[CPE]).

For all error types other than channel priority error, the channel number causing the error is recorded in the Error Status register (DMAx_ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again.

If priority levels are not unique, when any channel requests service, a channel priority error is reported. The highest channel priority with an active request is selected, but the lowest numbered channel with that priority is selected by arbitration and executed by the eDMA engine. The hardware service request handshake signals, error interrupts, and error reporting is associated with the selected channel.

12.5.3 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

12.5.3.1 Fixed channel arbitration

In this mode, the channel service request from the highest priority channel is selected to execute.

12.5.3.2 Round-robin channel arbitration

Channels are serviced starting with the highest channel number and rotating through to the lowest channel number without regard to the channel priority levels.

12.5.4 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

12.5.4.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the TCD*n*_CSR[DONE] bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] bit requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
    b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
    c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
    d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
    e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
    f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
    g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop → major loop complete.

6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[*n*] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

12.5.4.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled in the ERQ register, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.

f. Write 32-bits to location 0x2008 → third iteration of the minor loop.

g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0.
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:

    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.

    b. Write 32-bits to location 0x2010 → first iteration of the minor loop.

    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.

    d. Write 32-bits to location 0x2014 → second iteration of the minor loop.

    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.

    f. Write 32-bits to location 0x2018 → third iteration of the minor loop.

    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.

    h. Write 32-bits to location 0x201C → last iteration of the minor loop → major loop complete.

14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).

15. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[n] = 1.
16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

12.5.4.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567*x*) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a 2^4 byte (16-byte) size queue.

Table 12-9. Modulo example

Transfer Number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

12.5.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

12.5.5.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the TCD*n*_CITER field and test for a change. Another method may be extracted from the sequence shown below. The second method is

to test the TCD*n*_CSR[START] bit and the TCD*n*_CSR[ACTIVE] bit. The minor-loop-complete condition is indicated by both bits reading zero after the TCD*n*_CSR[START] was set. Polling the TCD*n*_CSR[ACTIVE] bit may be inconclusive, because the active status may be missed if the channel execution is short in duration.

The TCD status bits execute the following sequence for a software activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD*n*_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the TCD*n*_CSR[DONE] bit.

The TCD*n*_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

12.5.5.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_NBYTES values if read while a channel executes. The true values of the SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file and not the values in the TCD local memory for that channel. The addresses, SADDR and

DADDR, and NBYTES, which decrement to zero as the transfer progresses, can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

12.5.5.3 Checking channel preemption status

Preemption is available only when fixed arbitration is selected as the channel arbitration mode. A preemptive situation is one in which a preempt-enabled channel runs and a higher priority request becomes active. When the eDMA engine is not operating in fixed channel arbitration mode, the determination of the actively running relative priority outstanding requests become undefined. Channel priorities are treated as equal, that is, constantly rotating, when Round-Robin Arbitration mode is selected.

The TCD*n*_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two TCD*n*_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

12.5.6 Channel Linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the EDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD*n*_CITER[E_LINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER[E_LINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJOR_E_LINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] bit

2. Minor loop done → set TCD12_CSR[START] bit
3. Minor loop done → set TCD12_CSR[START] bit
4. Minor loop done, major loop done→ set TCD7_CSR[START] bit

When minor loop linking is enabled (TCD*n*_CITER[E_LINK] = 1), the TCD*n*_CITER[CITER] field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled (TCD*n*_CITER[E_LINK] = 0), the TCD*n*_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCD*n*_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

Note

The TCD*n*_CITER[E_LINK] bit and the TCD*n*_BITER[E_LINK] bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

Table 12-10. Channel Linking Parameters

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	CITER[E_LINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	CSR[MAJOR_E_LINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

12.5.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

12.5.7.1 Dynamically changing the channel priority

The following two options are recommended for dynamically changing channel priority levels:

1. Switch to Round-Robin Channel Arbitration mode, change the channel priorities, then switch back to Fixed Arbitration mode,
2. Disable all the channels, change the channel priorities, then enable the appropriate channels.

12.5.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCD.major.e_link bit during channel execution (see the diagram in TCD structure). This bit is read from the TCD local memory at the end of channel execution, thus allowing the user to enable the feature during channel execution.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCD.major.e_link bit at the same time the eDMA engine is retiring the channel. The TCD.major.e_link would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCD.major.e_link bit.
2. Read back the TCD.major.e_link bit.
3. Test the TCD.major.e_link request status:
   - If TCD.major.e_link = 1, the dynamic link attempt was successful.
   - If TCD.major.e_link = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCD.major.e_link bit to zero on any writes to a channel's TCD.word7 after that channel's TCD.done bit is set, indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link bit. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

12.5.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources.When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in eDMA programmer's model, thus replacing the current descriptor.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic scatter/gather operation by enabling the TCD.e_sg bit at the same time the eDMA engine is retiring the channel. The TCD.e_sg would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods for this coherency model are shown in the following subsections. Method 1 has the advantage of reading the major.linkch field and the e_sg bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD.major.e_link and TCD.e_sg bits to zero on any writes to a channel's TCD.word7 if that channel's TCD.done bit is set indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link or TCD.e_sg bits. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

12.5.7.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCD.major.e_link bit is zero, the TCD.major.linkch field is not used by the eDMA. In this case, the TCD.major.linkch bits may be used for other purposes. This method uses the TCD.major.linkch field as a TCD indentification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCD.major.linkch field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to the TCD.d_req bit.

Should a dynamic scatter/gather attempt fail, setting the TCD.d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

3. Write the TCD.dlast_sga field with the scatter/gather address.
4. Write 1b to the TCD.e_sg bit.
5. Read back the 16 bit TCD control/status field.
6. Test the TCD.e_sg request status and TCD.major.linkch value:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b and the major.linkch (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If e_sg = 0b and the major.linkch (ID) changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

12.5.7.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCD.dlast_sga field as a TCD indentification (ID).

1. Write 1b to the TCD.d_req bit.

   Should a dynamic scatter/gather attempt fail, setting the d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

2. Write theTCD.dlast_sga field with the scatter/gather address.
3. Write 1b to the TCD.e_sg bit.
4. Read back the TCD.e_sg bit.
5. Test the TCD.e_sg request status:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b, read the 32 bit TCD dlast_sga field.

   If e_sg = 0b and the dlast_sga did not change, the attempted dynamic link did not succeed (the channel was already retiring).

If e_sg = 0b and the dlast_sga changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

12.5.8 Suspend/resume a DMA channel with active hardware service requests

The DMA allows the user to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. Once the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), ADC, or other module.

12.5.8.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check Hardware Request Status Register (DMA_HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on the appropriate DMA channel.

12.5.8.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ bit.
2. Enable the DMA service request at the peripheral.

For example, assume a SPI module is set as a master for transmitting data via a DMA service request when the SPI TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If the user needs to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to SPI_RSER[TFFF_RE]. Confirm that SPI_RSER[TFFF_RE] is 0.

2. Ensure there is no DMA service request from the SPI by verifying that DMA_HRS[HRS*n*] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.

12.6 Usage Guide

NOTE

User should configure DMA_TCD*n*_CSR[BWC] (bit 15-14) as 10 when another DMA channel is active.

Related application notes on this DMA module are as follows.

- Using DMA for pulse counting on Kinetis
- Using DMA and GPIO to emulate timer functionality on Kinetis Family devices
- Using DMA to Emulate ADC Flexible Scan Mode on Kinetis K Series

Chapter 13
Memory and memory map

13.1 Introduction

This device contains various memories and memory-mapped peripherals which are located in one 4G bytes (32-bit address) contiguous memory space. This chapter describes the memory and peripheral locations within that memory space.

The following figure shows the system memory and peripheral locations.

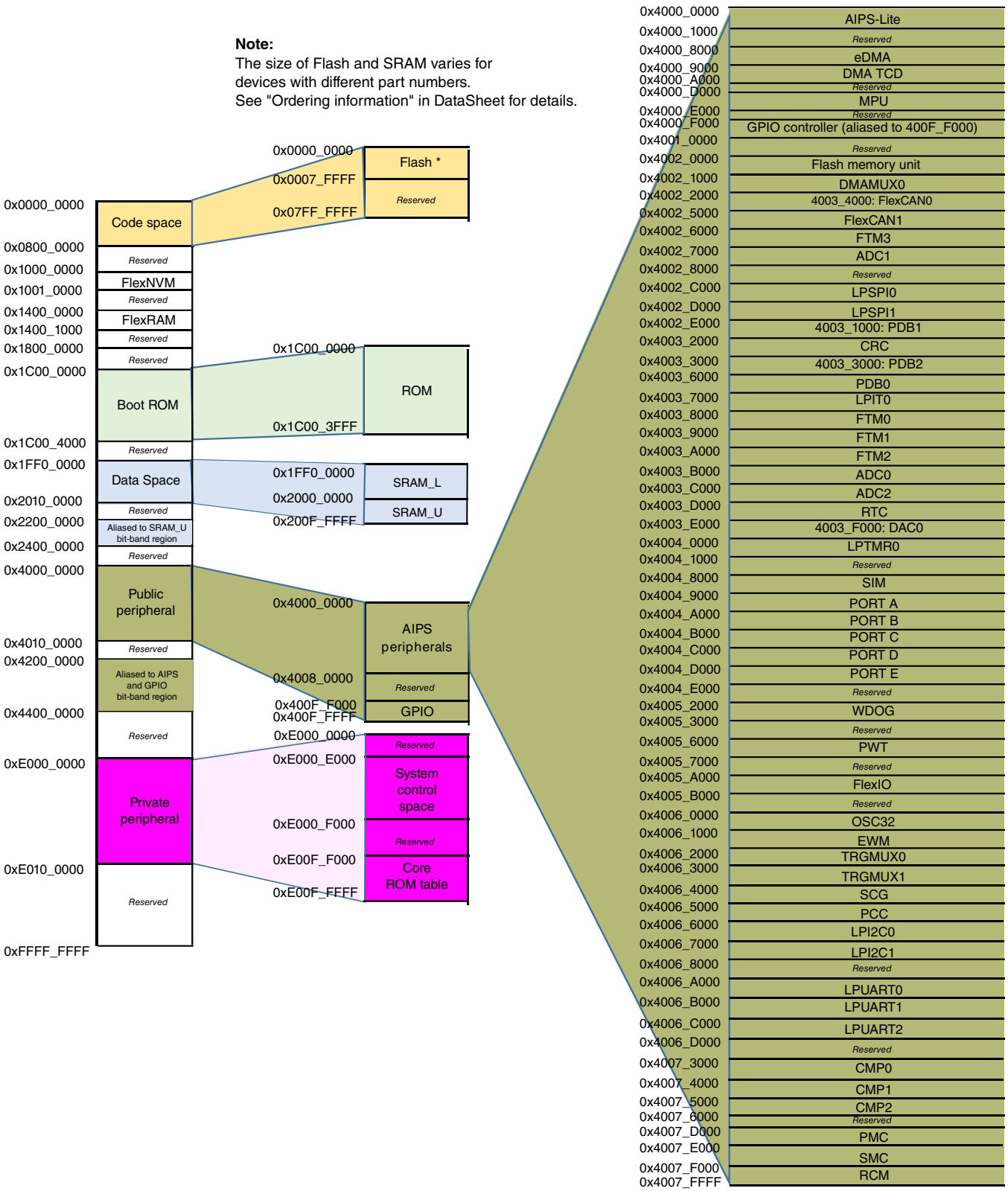


Figure 13-1. Memory map

13.2 Flash memory

13.2.1 Flash memory types

This device contains the following types of flash memory:

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

13.2.2 Flash Memory Sizes

The devices covered in this document contain:

- 2 blocks (256 KB each) of program flash consisting of 4 KB sectors
- 1 block (64 KB) of FlexNVM consisting of 2 KB sectors
- 1 block (4 KB) of FlexRAM

The amounts of flash memory and the address range for the devices is shown in following table.

Device	Program flash (KB)	FlexNVM (KB)	FlexRAM (KB)	Address range
KE1xF512VLL15	512	64	4	0x0000_0000–0x0007_FFFF (P-Flash) 0x1000_0000–0x1000_FFFF (FlexNVM) 0x1400_0000–0x1400_0FFF (FlexRAM)
KE1xF512VLH15	512	64	4	0x0000_0000–0x0007_FFFF (P-Flash) 0x1000_0000–0x1000_FFFF (FlexNVM) 0x1400_0000–0x1400_0FFF (FlexRAM)
KE1xF256VLL15	256	64	4	0x0000_0000–0x0003_FFFF (P-Flash) 0x1000_0000–0x1000_FFFF (FlexNVM) 0x1400_0000–0x1400_0FFF (FlexRAM)
KE1xF256VLH15	256	64	4	0x0000_0000–0x0003_FFFF (P-Flash) 0x1000_0000–0x1000_FFFF (FlexNVM) 0x1400_0000–0x1400_0FFF (FlexRAM)

13.3 SRAM memory

13.3.1 SRAM sizes

This device contains SRAM tightly coupled to the ARM Cortex-M4 core. The on-chip SRAM is split into SRAM_L and SRAM_U regions where the SRAM_L and SRAM_U ranges form a contiguous block in the memory map anchored at address 0x2000_0000. As such:

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

NOTE

Misaligned accesses across the 0x2000_0000 boundary are not supported in the ARM Cortex-M4 architecture.

NOTE

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

The amount of SRAM for the devices covered in this document is shown in the following table.

Device	SRAM_L size (KB)	SRAM_U size (KB)	Total SRAM (KB)	Address Range
MKE1xF512VLL15	32	32	64	0x1FFF_8000-0x2000_7FFF
MKE1xF512VLH15	32	32	64	0x1FFF_8000-0x2000_7FFF
MKE1xF256VLL15	16	16	32	0x1FFF_C000-0x2000_3FFF
MKE1xF256VLH15	16	16	32	0x1FFF_C000-0x2000_3FFF

13.3.2 SRAM retention in low power modes

The SRAM is retained power on to all power modes on this device.

13.3.3 SRAM accesses

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

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

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

The following figure illustrates the SRAM accesses within the device.

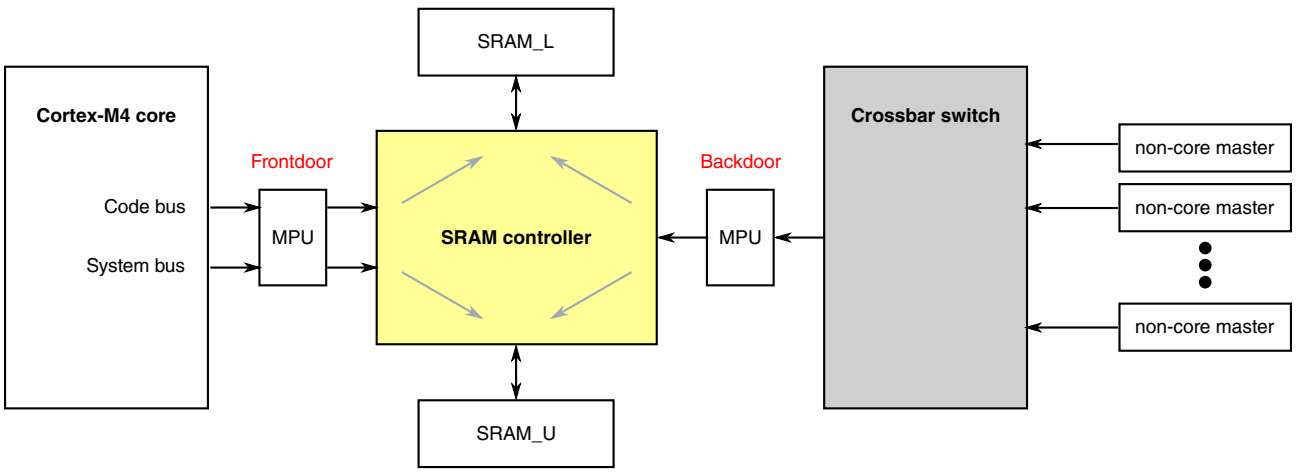


Figure 13-2. SRAM access diagram

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

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

NOTE

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

13.3.4 SRAM arbitration and priority control

The MCM_CPCR register controls the arbitration and priority schemes for the two SRAM arrays.

13.4 System memory map

The following table shows the high-level device memory map. This map provides the complete architectural address space definition for the various sections. Based on the physical sizes of the memories and peripherals, the actual address regions used may be smaller.

The system memory map includes address spaces that are intended for specific purposes.

- Flash region aliasing is specifically intended for references to read-only data coefficients in the flash while still preserving a full Harvard memory organization in the processor core supporting concurrent instruction fetches (for example, from RAM) and data accesses (from flash via the aliased space).
- The bitbanding functionality supported by the processor core uses aliased regions that map to the basic RAM and peripheral address spaces. This functionality maps each 32-bit word of the aliased address space to a unique bit in the underlying RAM or peripheral address space to support single-bit insert and extract operations from the processor.

Table 13-1. System memory map

System 32-bit Address Range	Destination Slave	Access
0x0000_0000–0x07FF_FFFF [1]	Program flash and read-only data (Includes exception vectors in first 1024 bytes)	All masters
0x0800_0000–0x0FFF_FFFF	Reserved	–
0x1000_0000–0x13FF_FFFF	FlexNVM	All masters
0x1400_0000–0x17FF_FFFF	FlexRAM	All masters
0x1800_0000–0x1BFF_FFFF	Reserved	–
0x1C00_0000–0x1C00_3FFF	Boot ROM	All masters
0x1C00_4000–0x1FEF_FFFF	Reserved	–
0x1FF0_0000–0x1FFF_FFFF [2]	SRAM_L: Lower SRAM (ICODE/DCODE)	All masters
0x2000_0000–0x200F_FFFF [2]	SRAM_U: Upper SRAM bitband region	All masters
0x2010_0000–0x201F_FFFF	Reserved	–
0x2020_0000–0x21FF_FFFF	Reserved	–
0x2200_0000–0x23FF_FFFF	Aliased to TCMU SRAM_U bitband	Cortex-M4 core only
0x2400_0000-0x2FFF_FFFF	Reserved	–
0x3000_0000-0x33FF_FFFF [1]	Reserved	–
0x3400_0000-0x3FFF_FFFF	Reserved	–
0x4000_0000–0x4007_FFFF	Bitband region for peripheral bridge (AIPS-Lite)	Cortex-M4 core & DMA
0x4008_0000–0x400F_EFFF	Reserved	–

Table continues on the next page...

Table 13-1. System memory map (continued)

System 32-bit Address Range	Destination Slave	Access
0x400F_F000–0x400F_FFFF	Bitband region for general purpose input/output (GPIO)	Cortex-M4 core & DMA
0x4010_0000–0x41FF_FFFF	Reserved	–
0x4200_0000–0x43FF_FFFF	Aliased to peripheral bridge (AIPS-Lite) and general purpose input/output (GPIO) bitband	Cortex-M4 core only
0x4400_0000–0xDFFF_FFFF	Reserved	–
0xE000_0000–0xE00F_FFFF	Private peripherals	Cortex-M4 core only
0xE010_0000–0xFFFF_FFFF	Reserved	–

1. This map provides the complete architectural address space definition for the flash. Based on the physical sizes of the memories implemented for a particular device, the actual address regions used may be smaller. See Flash Memory Sizes for details.
2. This range varies depending on amount of SRAM implemented for a particular device. See SRAM sizes for details.

NOTE

1. Access rights to AIPS-Lite peripheral bridge and general purpose input/output (GPIO) module address space is limited to the core, DMA.
2. ARM Cortex-M4 core access privileges also includes accesses via the debug interface.
3. The SRAM on this device could be accessed through normal way with 32-bit operation, and also could be accessed with bit operation through aliased bit-band region.

13.4.1 Aliased bit-band regions

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

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

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

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

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

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

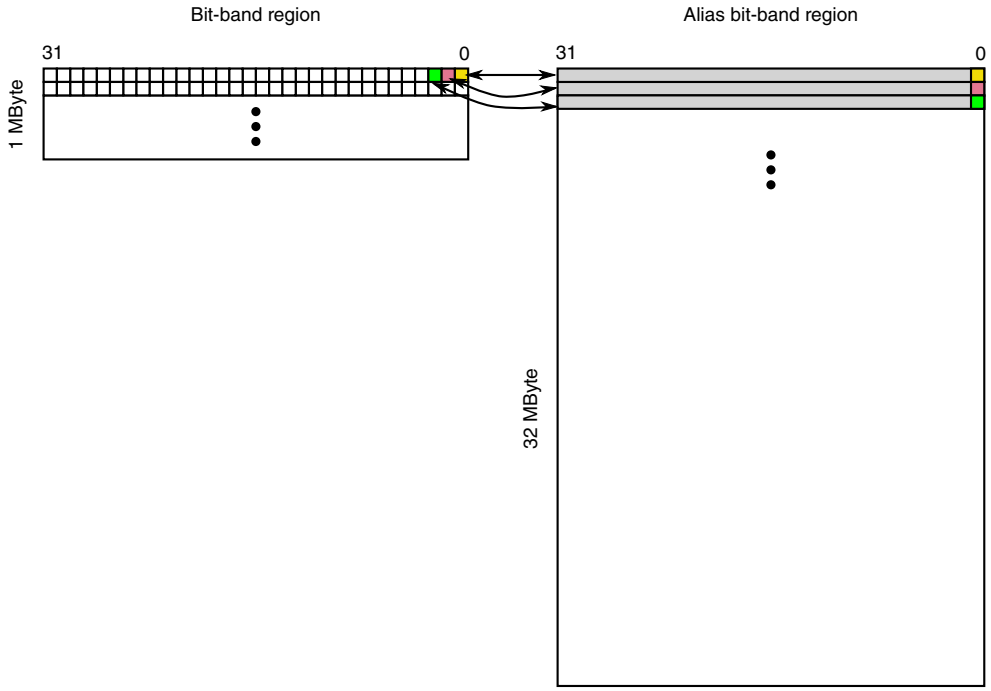


Figure 13-3. Alias bit-band mapping

NOTE

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

13.5 Peripheral memory map

The peripheral memory map is accessible via a crossbar slave port and the AIPS peripheral bridge. The peripheral bridge converts register access from AHB bus domain to peripheral bus domain.

For peripherals that have clock gating control bits (CGC bit) in PCC module, the associated peripherals could be enabled/disabled by these control bits. Access to a disabled peripheral or unimplemented AIPS 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.

13.5.1 Peripheral Bridge (AIPS-Lite) Memory Map

Table 13-2. Peripheral bridge slot assignments

System 32-bit base address	Slot number	Module
0x4000_0000	0	Peripheral bridge (AIPS-Lite)
0x4000_1000	1	MSCM
0x4000_2000	2	—
0x4000_3000	3	—
0x4000_4000	4	—
0x4000_5000	5	—
0x4000_6000	6	—
0x4000_7000	7	—
0x4000_8000	8	DMA controller
0x4000_9000	9	DMA controller transfer control descriptors
0x4000_A000	10	—
0x4000_B000	11	—
0x4000_C000	12	—
0x4000_D000	13	MPU
0x4000_E000	14	—
0x4000_F000	15	RGPIO controller (aliased to 0x400F_F000)
0x4001_0000	16	—
0x4001_1000	17	—
0x4001_2000	18	—
0x4001_3000	19	—
0x4001_4000	20	—
0x4001_5000	21	—
0x4001_6000	22	—
0x4001_7000	23	—
0x4001_8000	24	—
0x4001_9000	25	—
0x4001_A000	26	—
0x4001_B000	27	—
0x4001_C000	28	—
0x4001_D000	29	—
0x4001_E000	30	—
0x4001_F000	31	—
0x4002_0000	32	Flash memory
0x4002_1000	33	DMA channel mutiplexer 0
0x4002_2000	34	—

Table continues on the next page...

Table 13-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4002_3000	35	—
0x4002_4000	36	FlexCAN 0
0x4002_5000	37	FlexCAN 1
0x4002_6000	38	FlexTimer (FTM) 3
0x4002_7000	39	Analog-to-digital converter (ADC) 1
0x4002_8000	40	—
0x4002_9000	41	—
0x4002_A000	42	—
0x4002_B000	43	
0x4002_C000	44	Low Power SPI (LPSPI) 0
0x4002_D000	45	Low Power SPI (LPSPI) 1
0x4002_E000	46	—
0x4002_F000	47	—
0x4003_0000	48	—
0x4003_1000	49	Programmable delay block (PDB) 1
0x4003_2000	50	CRC
0x4003_3000	51	Programmable delay block (PDB) 2
0x4003_4000	52	—
0x4003_5000	53	—
0x4003_6000	54	Programmable delay block (PDB) 0
0x4003_7000	55	Low-power Periodic interrupt timer (LPIT0)
0x4003_8000	56	FlexTimer (FTM) 0
0x4003_9000	57	FlexTimer (FTM) 1
0x4003_A000	58	FlexTimer (FTM) 2
0x4003_B000	59	Analog-to-digital converter (ADC) 0
0x4003_C000	60	Analog-to-digital converter (ADC) 2
0x4003_D000	61	Real-time clock (RTC)
0x4003_E000	62	—
0x4003_F000	63	DAC0
0x4004_0000	64	Low-power timer (LPTMR0)
0x4004_1000	65	—
0x4004_2000	66	—
0x4004_3000	67	—
0x4004_4000	68	—
0x4004_5000	69	—
0x4004_6000	70	—
0x4004_7000	71	—
0x4004_8000	72	System integration module (SIM)
0x4004_9000	73	Port A multiplexing control

Table continues on the next page...

Table 13-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4004_A000	74	Port B multiplexing control
0x4004_B000	75	Port C multiplexing control
0x4004_C000	76	Port D multiplexing control
0x4004_D000	77	Port E multiplexing control
0x4004_E000	78	—
0x4004_F000	79	—
0x4005_0000	80	—
0x4005_1000	81	—
0x4005_2000	82	Software watchdog (WDOG)
0x4005_3000	83	—
0x4005_4000	84	—
0x4005_5000	85	—
0x4005_6000	86	Pulse Width Timer (PWT)
0x4005_7000	87	—
0x4005_8000	88	—
0x4005_9000	89	—
0x4005_A000	90	Flexible IO (FlexIO)
0x4005_B000	91	—
0x4005_C000	92	—
0x4005_D000	93	—
0x4005_E000	94	—
0x4005_F000	95	—
0x4006_0000	96	OSC32
0x4006_1000	97	External watchdog (EWM)
0x4006_2000	98	Trigger Multiplexing Control (TRGMUX 0)
0x4006_3000	99	Trigger Multiplexing Control (TRGMUX 1)
0x4006_4000	100	System Clock Generator (SCG)
0x4006_5000	101	Peripheral Clock Control (PCC)
0x4006_6000	102	Low Power I^2C (LPI^{2}C 0)
0x4006_7000	103	Low Power I^2C (LPI^{2}C 1)
0x4006_8000	104	—
0x4006_9000	105	—
0x4006_A000	106	Low Power UART (LPUART 0)
0x4006_B000	107	Low Power UART (LPUART 1)
0x4006_C000	108	Low Power UART (LPUART 2)
0x4006_D000	109	—
0x4006_E000	110	—
0x4006_F000	111	—
0x4007_0000	112	—

Table continues on the next page...

Table 13-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4007_1000	113	—
0x4007_2000	114	—
0x4007_3000	115	Analog comparator (CMP 0)
0x4007_4000	116	Analog comparator (CMP 1)
0x4007_5000	117	Analog comparator (CMP 2)
0x4007_6000	118	—
0x4007_7000	119	—
0x4007_8000	120	—
0x4007_9000	121	—
0x4007_A000	122	—
0x4007_B000	123	—
0x4007_C000	124	—
0x4007_D000	125	Power management controller (PMC)
0x4007_E000	126	System Mode controller (SMC)
0x4007_F000	127	Reset Control Module (RCM)
0x400F_F000		GPIO controller

13.6 Private Peripheral Bus (PPB) memory map

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

Table 13-3. PPB memory map

System 32-bit Address Range	Resource
0xE000_0000–0xE000_0FFF	Instrumentation Trace Macrocell (ITM)
0xE000_1000–0xE000_1FFF	Data Watchpoint and Trace (DWT)
0xE000_2000–0xE000_2FFF	Flash Patch and Breakpoint (FPB)
0xE000_3000–0xE000_DFFF	Reserved
0xE000_E000–0xE000_EFFF	System Control Space (SCS) (for NVIC and FPU)
0xE000_F000–0xE003_FFFF	Reserved
0xE004_0000–0xE004_0FFF	Trace Port Interface Unit (TPIU)
0xE004_1000–0xE004_1FFF	Reserved
0xE004_2000–0xE004_2FFF	Reserved
0xE004_3000–0xE004_3FFF	Reserved
0xE004_4000–0xE007_FFFF	Reserved

Table continues on the next page...

Table 13-3. PPB memory map (continued)

System 32-bit Address Range	Resource
0xE008_0000–0xE008_0FFF	Miscellaneous Control Module (MCM)
0xE008_1000–0xE008_1FFF	Reserved
0xE008_2000–0xE008_2FFF	Cache Controller (LMEM)
0xE008_3000–0xE00F_EFFF	Reserved
0xE00F_F000–0xE00F_FFFF	ARM Core ROM Table[1] - allows auto-detection of debug components

1. The ARM Core ROM table is optionally required by ARM CoreSight debug infrastructure to discover the components on the chip. This ROM table has no any relationship with the MCU Boot ROM.

Chapter 14
Local Memory Controller (LMEM)

14.1 Chip-specific information for this module

14.1.1 Local memory controller region assignment

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

Table 14-1. Cache regions

Address range	Destination slave	Region number	Available cache modes
0x0000_0000–0x07FF_FFFF	Program flash and read-only data	R0	Write-through and cacheable [1]
0x0800_0000–0x0FFF_FFFF	Reserved region	R1	
0x1000_0000–0x13FF_FFFF	FlexNVM	R2	Write-through and cacheable [1]
0x1400_0000–0x17FF_FFFF	FlexRAM		Non-cacheable
0x1800_0000–0x1BFF_FFFF	Reserved region	R3	–
0x1C00_0000–0x1C00_3FFF	Boot ROM		Write-through (always)
0x1C00_4000–0x1FEF_FFFF	Reserved region		–
0x1FF0_0000–0x1FFF_FFFF	SRAM_L: Lower SRAM (ICODE/ DCODE)	R4	Non-cacheable
0x2000_0000–0x200F_FFFF	SRAM_U: Upper SRAM	R5	Non-cacheable
0x6000_0000–0x6FFF_FFFF	Reserved region	R6	–
0x7000_0000–0x7FFF_FFFF	Reserved region	R7	–
0x8000_0000–0x8FFF_FFFF	Reserved region	R8	–
0x9000_0000–0x9FFF_FFFF	Reserved region	R9	–

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

14.2 Introduction

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

14.2.1 Block Diagram

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

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

The local memory controller includes three memory controllers and their attached memories:

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

The local memory controller has the following AMBA_ AHB buses:

- Two inputs are for the core's modified Harvard busses – the Processor Code (PC) and the Processor System (PS) buses.
- One input is for the "backdoor" port used by all other bus masters to access the SRAM controller space.
- Two output ports are the CCM (Core Code Master) bus used for PC accesses that do not hit the PC cache or SRAM_L or are non-cacheable and the CSM (Core System Master) bus used for PS references that do not hit the SRAM_U.

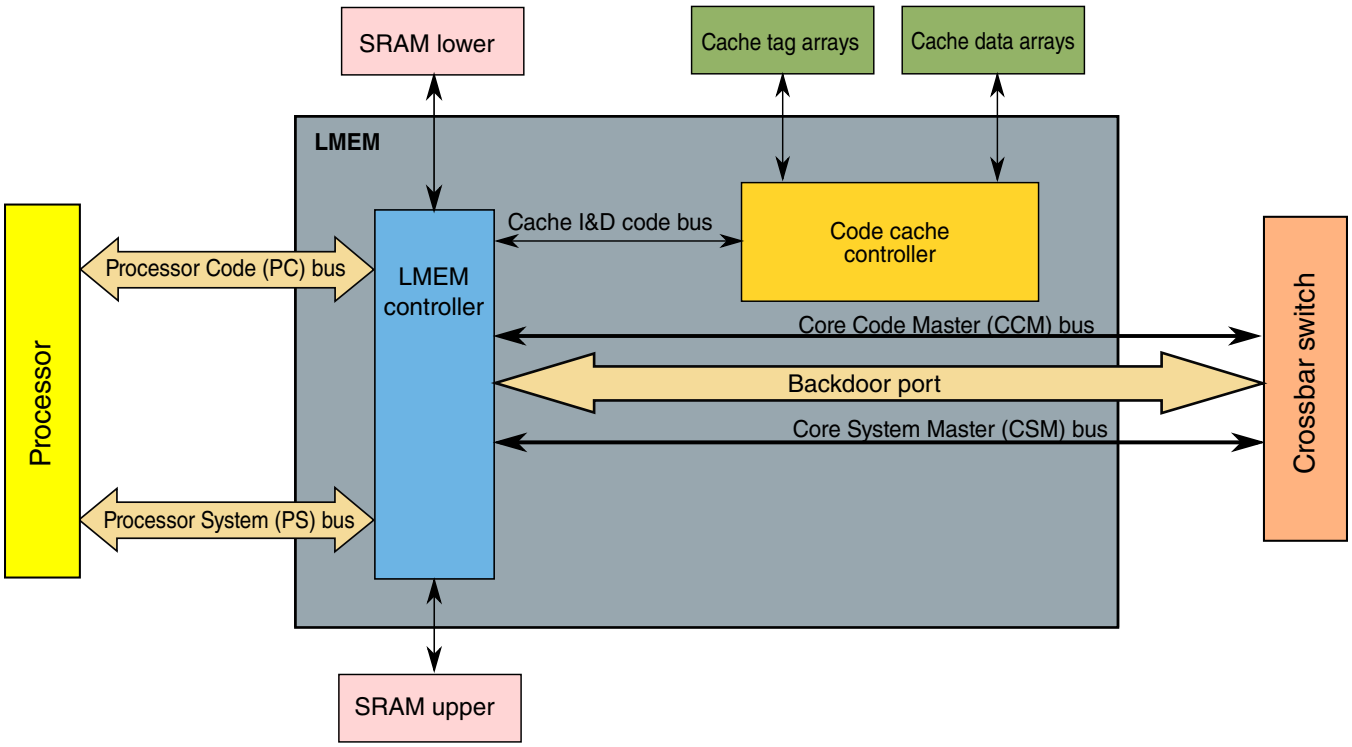


Figure 14-1. Local memory controller block diagram

NOTE

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

The LMEM contains address decode logic for the PC and PS buses. This logic routes the core's accesses to the various system resources. The address spaces are device-specific. See the chip-specific LMEM information for the address space decode details.

14.2.2 Cache features

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

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

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

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

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

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

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

The local memory controller supports the following modes of operation:

1. Write-through — access to address spaces with this cache mode are cacheable.
   - A write-through read miss on the input bus causes a line read on the output bus of a 16-byte-aligned memory address containing the desired address. This miss data is loaded into the cache and is marked as valid and not modified.
   - A write-through read hit to a valid cache location returns data from the cache with no output bus access.
   - A write-through write miss bypasses the cache and writes to the output bus (no allocate on write miss policy for write-through mode spaces).
   - A write-through write hit updates the cache hit data and writes to the output bus.
   - The caches are processor-local and do not support hardware cache coherency. If the processor has accessed write-through regions and an external bus master (such as DMA) then needs update these regions, software must first perform explicit cache clears to any needed cache memory range to ensure all modified cache lines update their associated memories before being modified by external masters and subsequent processor accesses will get the updated memory.
2. Non-cacheable — access to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the output bus.

14.3 Memory Map/Register Definition

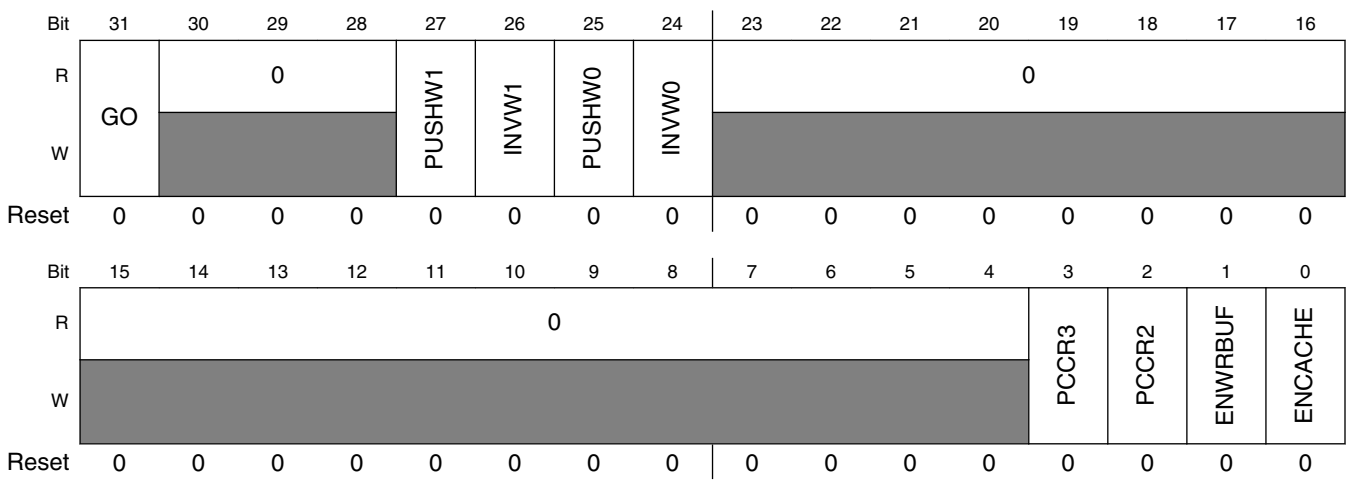
The cache programmer's model provides a variety of registers for configuring and controlling the cache, as well as indirect access paths to all cache tag and data storage.

LMEM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_2000	Cache control register (LMEM_PCCCR)	32	R/W	0000_0000h	14.3.1/325
E008_2004	Cache line control register (LMEM_PCCLCR)	32	R/W	0000_0000h	14.3.2/326
E008_2008	Cache search address register (LMEM_PCCSAR)	32	R/W	0000_0000h	14.3.3/329
E008_200C	Cache read/write value register (LMEM_PCCCVR)	32	R/W	0000_0000h	14.3.4/330
E008_2020	Cache regions mode register (LMEM_PCCRMR)	32	R/W	AA0F_A000h	14.3.5/330

14.3.1 Cache control register (LMEM_PCCCR)

Address: E008_2000h base + 0h offset = E008_2000h



LMEM_PCCCR field descriptions

Field	Description
31 GO	Initiate Cache Command Setting this bit initiates the cache command indicated by bits 27-24. Reading this bit indicates if a command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. 0 Write: no effect. Read: no cache command active. 1 Write: initiate command indicated by bits 27-24. Read: cache command active.

Table continues on the next page...

LMEM_PCCCR field descriptions (continued)

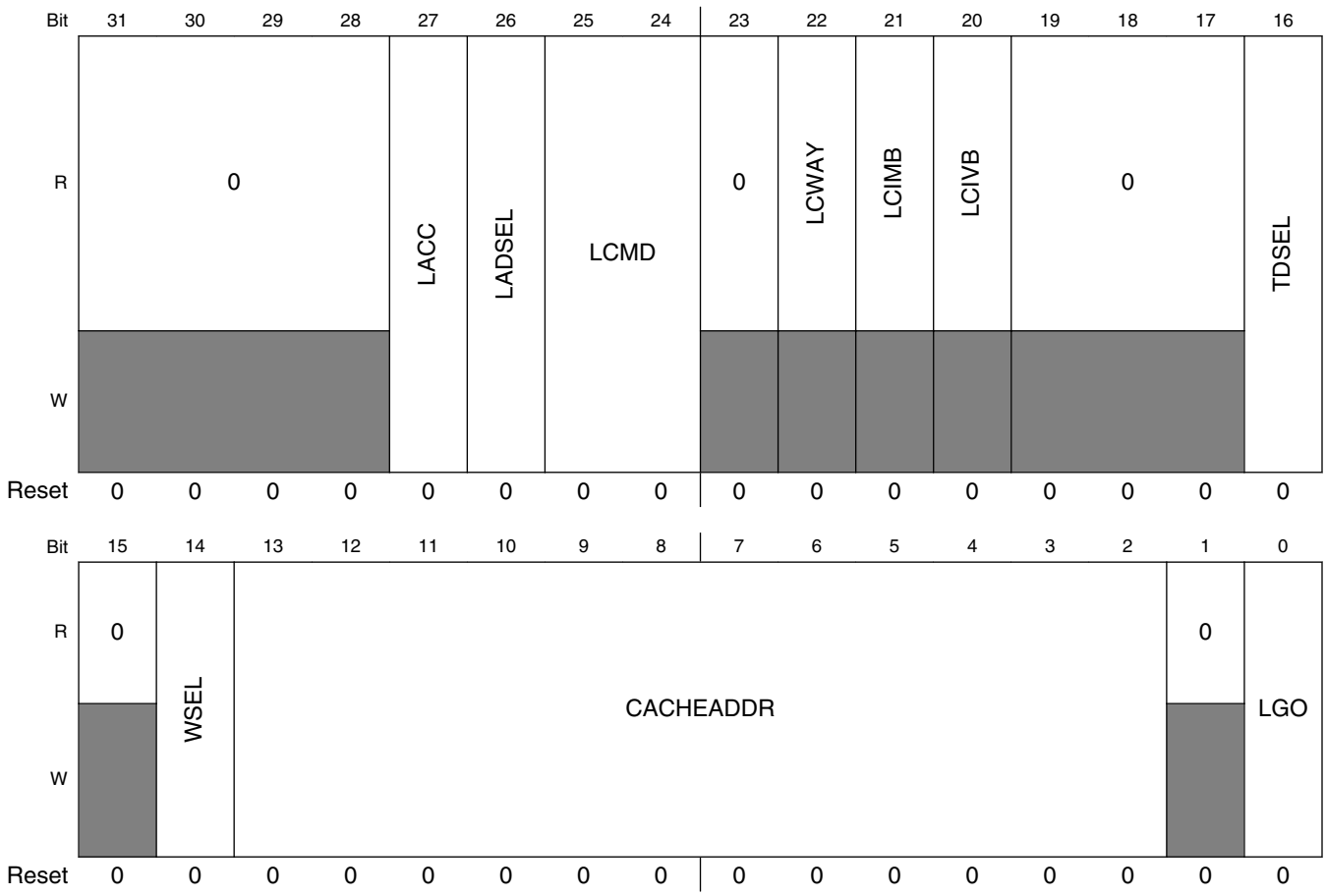
Field	Description
30–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 PUSHW1	Push Way 1 0 No operation 1 When setting the GO bit, push all modified lines in way 1
26 INVW1	Invalidate Way 1 **NOTE:** If the PUSHW1 and INVW1 bits are set, then after setting the GO bit, push all modified lines in way 1 and invalidate all lines in way 1 (clear way 1). 0 No operation 1 When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0 No operation 1 When setting the GO bit, push all modified lines in way 0
24 INVW0	Invalidate Way 0 **NOTE:** If the PUSHW0 and INVW0 bits are set, then after setting the GO bit, push all modified lines in way 0 and invalidate all lines in way 0 (clear way 0). 0 No operation 1 When setting the GO bit, invalidate all lines in way 0.
23–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 PCCR3	Forces no allocation on cache misses (must also have PCCR2 asserted)
2 PCCR2	Forces all cacheable spaces to write through
1 ENWRBUF	Enable Write Buffer 0 Write buffer disabled 1 Write buffer enabled
0 ENCACHE	Cache enable 0 Cache disabled 1 Cache enabled

14.3.2 Cache line control register (LMEM_PCCLCR)

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

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

Address: E008_2000h base + 4h offset = E008_2004h



LMEM_PCCLCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 LACC	Line access type 0 Read 1 Write
26 LADSEL	Line Address Select When using the cache address, the way must also be specified in CLCR[WSEL]. When using the physical address, both ways are searched and the command is performed only if a hit. 0 Cache address 1 Physical address
25–24 LCMD	Line Command 00 Search and read or write 01 Invalidate 10 Push 11 Clear

Table continues on the next page...

LMEM_PCCLCR field descriptions (continued)

Field	Description
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 LCWAY	Line Command Way Indicates the way used by the line command. If the command used cache address and way, then LCWAY will equal the way it was searched. If the command is a physical address search, then the LCWAY value is only valid if LCIVB=1.
21 LCIMB	Line Command Initial Modified Bit If command used cache address and way, then this bit shows the initial state of the modified bit If command used physical address and a hit, then this bit shows the initial state of the modified bit. If a miss, this bit reads zero.
20 LCIVB	Line Command Initial Valid Bit If command used cache address and way, then this bit shows the initial state of the valid bit If command used physical address and a hit, then this bit shows the initial state of the valid bit. If a miss, this bit reads zero.
19–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0 Data 1 Tag
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 WSEL	Way select Selects the way for line commands. 0 Way 0 1 Way 1
13–2 CACHEADDR	Cache address CLCR[11:4] bits are used to access the tag arrays CLCR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CSAR[LGO] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits 27-24. Read: line command active.

14.3.3 Cache search address register (LMEM_PCCSAR)

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

Address: E008_2000h base + 8h offset = E008_2008h

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

LMEM_PCCSAR field descriptions

Field	Description
31–2 PHYADDR	Physical Address PHYADDR represents bits [31:2] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:4] bits are used to access the tag arrays CSAR[11:2] bits are used to access the data arrays
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CLCR[LGO] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

14.3.4 Cache read/write value register (LMEM_PCCCVR)

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

Address: E008_2000h base + Ch offset = E008_200Ch

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

LMEM_PCCCVR field descriptions

Field	Description
DATA	Cache read/write Data For tag search, read or write: • CCVR[31:12] bits are used for tag array R/W value • CCVR[11:4] bits are used for tag set address on reads; unused on writes • CCVR[3:2] bits are reserved • CCVR[1] tag modify bit • CCVR[0] tag valid bit For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

14.3.5 Cache regions mode register (LMEM_PCCRMR)

The CRMR register allows you to demote the cache mode of various subregions within the device's memory map. Demoting the cache mode reduces the cache function applied to a memory region from write-back to write-through to non-cacheable. After a region is demoted, its cache mode can only be raised by a reset, which returns it to its default state.

To maintain cache coherency, changes to the cache mode should be completed while the address space being changed is not being accessed or the cache is disabled. Before a cache mode change, complete a cache clear all command to push and invalidate any cache entries that may have changed.

NOTE

The address/module assignment of the 16 subregions is device-specific. See the chip-specific LMEM information for these details. Some of the regions may not be used (non-cacheable), and some regions may not be capable of write-back.

Address: E008_2000h base + 20h offset = E008_2020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	R0		R1		R2		R3		R4		R5		R6		R7		R8		R9		R10		R11		R12		R13		R14		R15	
Reset	1	0	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PCCRMR field descriptions

Field	Description
31–30 R0	Region 0 mode Controls the cache mode for region 0 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
29–28 R1	Region 1 mode Controls the cache mode for region 1 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
27–26 R2	Region 2 mode Controls the cache mode for region 2 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
25–24 R3	Region 3 mode Controls the cache mode for region 3 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
23–22 R4	Region 4 mode Controls the cache mode for region 4 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
21–20 R5	Region 5 mode Controls the cache mode for region 5 00 Non-cacheable 01 Non-cacheable

Table continues on the next page...

LMEM_PCCRMR field descriptions (continued)

Field	Description
	10 Write-through 11 Write-back
19–18 R6	Region 6 mode Controls the cache mode for region 6 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
17–16 R7	Region 7 mode Controls the cache mode for region 7 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
15–14 R8	Region 8 mode Controls the cache mode for region 8 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
13–12 R9	Region 9 mode Controls the cache mode for region 9 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
11–10 R10	Region 10 mode Controls the cache mode for region 10 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
9–8 R11	Region 11 mode Controls the cache mode for region 11 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back

Table continues on the next page...

LMEM_PCCRMR field descriptions (continued)

Field	Description
7–6 R12	Region 12 mode Controls the cache mode for region 12 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
5–4 R13	Region 13 mode Controls the cache mode for region 13 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
3–2 R14	Region 14 mode Controls the cache mode for region 14 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
R15	Region 15 mode Controls the cache mode for region 15 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back

14.4 Functional Description

14.4.1 LMEM Function

The Local Memory Controller receives the following requests:

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

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

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

14.4.1.1 Processor Code accesses

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

14.4.1.2 Processor System accesses

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

14.4.1.3 Backdoor port accesses

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

14.4.2 SRAM Function

14.4.2.1 SRAM Configuration

The figure below shows how the SRAM controller is configured. See chip-specific memory information for on-chip SRAM size details.

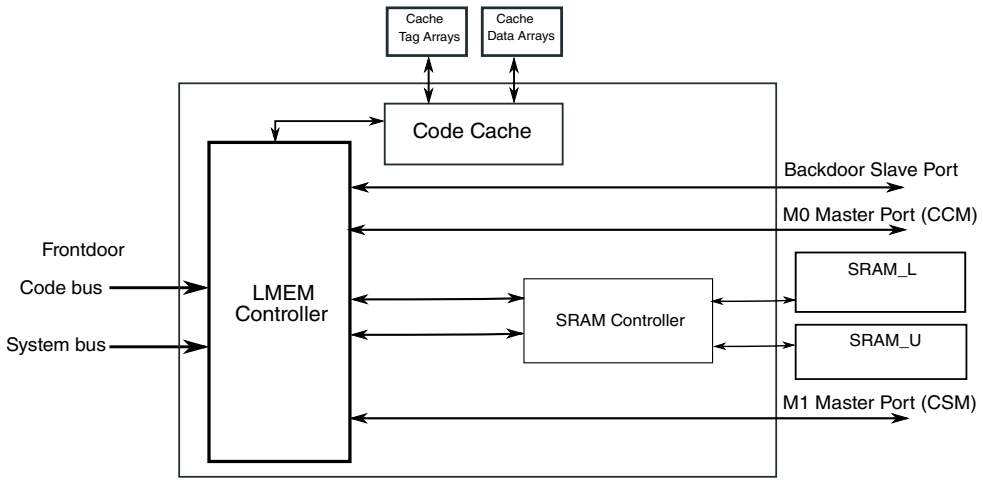


Figure 14-2. LMEM Interface to SRAM

14.4.2.2 SRAM Arrays

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

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

- SRAM_L = (0x20000_0000 - SRAML_SIZE) - 0x1fff_ffff
- SRAM_U = 0x2000_0000 - (0x2000_0000 + SRAMU_SIZE)

SRAML_SIZE and SRAMU_SIZE do not have to be equal.

For example, if SRAM_L size is 32 KBytes and SRAM_U size is 64 KBytes. Valid address ranges for SRAM_L and SRAM_U are then defined as:

- SRAM_L = (0x20000_0000 - 32 KBytes) - 0x1fff_ffff
- SRAM_U = 0x2000_0000 - (0x2000_0000 + 64 KBytes)

14.4.2.3 SRAM accesses

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

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

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

Figure 14-3 illustrates the SRAM accesses within the device.

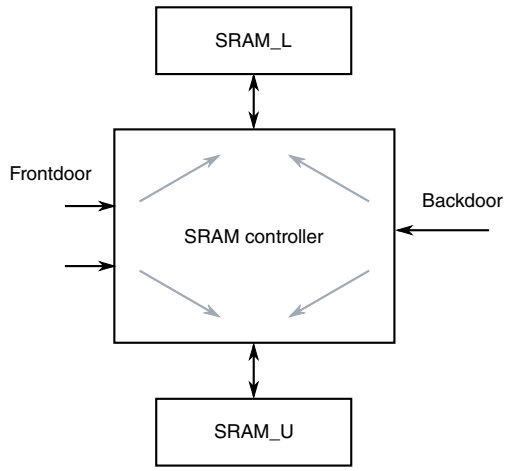


Figure 14-3. SRAM access diagram

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

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

NOTE

Two non-core masters cannot access SRAM simultaneously. The required arbitration and serialization is provided by the crossbar switch. The SRAM_{L,U} arbitration is controlled by the SRAM controller based on the configuration bits in the MCM module.

NOTE

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

14.4.3 Cache Function

The cache on this device is structured as follows. The cache has a 2-way set-associative cache structure with a total size of 8 KB. The cache has 32-bit address and data paths and a 16-byte line size. The cache tags and data storage use single-port, synchronous RAMs.

For this 8 KB cache, each cache TAG function uses two 256 x 22-bit RAM arrays and the cache DATA function uses two 1024 x 32-bit RAM arrays. The cache TAG entries store 20 bits of upper address as well as a modified and valid bit per cache line. The cache DATA entries store four bytes of code or data.

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

CACHE - 8 KB size = (256 sets) x (16-byte lines) x (2-way set-associative)

TAG:

- address[31:12] used in tag for compare (hit) logic
- address[11:4] used to select 1 of 256 sets
- address[3:0] not used

DATA

- address[31:12] not used
- address[11:4] used to select one of 256 sets
- address[3:2] used to select one of four 32-bit words within a set
- address[1:0] used to select the byte within the 32-bit word

14.4.4 Cache Control

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

14.4.4.1 Cache set commands

The cache set commands may operate on:

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

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

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

Supported cache set commands are given in Table 14-2. Set commands work as follows:

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

Table 14-2. Cache Set Commands

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

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

14.4.4.2 Cache line commands

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

- A cache address consists of a set address and a way select. The line command acts on the specified cache line.
- Cache line commands with physical addresses first search both ways of the cache set specified by bits [11:4] of the physical address. If they hit, the commands perform their action on the hit way.

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

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

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

Table 14-3. Cache Line Commands

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

14.4.4.2.1 Executing a series of line commands using cache addresses

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

- Place the command in CLCR[27:24],
- Set the way (CLCR[WSEL]) and tag/data (CLCR[TDSEL]) controls as needed,

- Place the cache address in CLCR[CACHEADDR], and
- Set the line command go bit (CLCR[LGO]).

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

- Increment the cache address (at bit 2 to step through data or at bit 4 to step through lines), and
- Set the line command go bit (CLCR[LGO]).

14.4.4.2.2 Executing a series of line commands using physical addresses

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

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

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

- Increment the physical address (at bit 2 to step through data or at bit 4 to step through lines), and
- Set the line command go bit (CSAR[LGO]).

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

14.4.4.2.3 Line command results

At completion of a line command, the CLCR register contains information on the initial state of the line targeted by the command. For line commands with cache addresses, this information is read before the line command action is performed from the targeted cache line. For line commands with physical addresses, this information is read on a hit before the line command action is performed from the hit cache line or has initial valid bit cleared if the command misses. In general, if the valid indicator (CLCR[LCIVB] is cleared, the targeted line was invalid at the start of the line command and no line operation was performed.

Table 14-4. Line command results

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

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

Chapter 15
Miscellaneous System Control Module (MSCM)

15.1 Overview

The Miscellaneous System Control Module (MSCM) contains CPU configuration registers and on-chip memory controller registers.

15.2 Chip Configuration and Boot

The device configuration is defined by flash test bits, supported memory sizes and packing options. Collectively, these configuration bits define an RCON (reset configuration) value.

Once the core has fetched the needed reset vector(s), it is expected they read core and system configuration information from a globally-accessible slave peripheral that properly converts the information into more appropriate values. More specifically, the core accesses configuration information from a common set of peripheral addresses and the chip configuration logic properly evaluates based on the requesting processor and returns the appropriate value for the given processor, including core identification.

As an example, there is a single 32-bit read-only location for the core identification. A 32-bit read from this location returns a four character ASCII string: 0x43_4D_34 ("Cortex-M4").

The programming model associated with the core configuration information is included as part of the Miscellaneous System Control Module (MSCM). It specifically includes multiple views of the processor configuration; one that is available generically to the core and others that are available to any bus masters in the system.

15.3 MSCM Memory Map/Register Definition

15.3.1 CPU Configuration Memory Map and Registers

The CPU configuration portion of the MSCM module provides a set of memory-mapped read-only addresses defining the processor setup. This portion of the MSCM programming model can only be accessed with privileged mode 32-bit read references; any other access type or size are terminated with an error. If the processor is logically not included in the chip configuration, reads of its configuration registers return zeroes.

The CPU Configuration registers are organized based on the logical processor number (not any type of physical port number) and partitioned into several equal sections:

- Offset addresses 0x000 - 0x01F define the generic processor "x" configuration. This region is only accessible to the processor core(s); reads by non-core bus masters are treated as RAZ (read as zero) accesses.
- Offset addresses 0x020 - 0x03F define the configuration information for processor 0 (CP0). This region is accessible to any bus master.

Reads from any other bus master return all zeroes. Attempted user mode or write accesses are terminated with an error.

MSCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_1000	Processor X Type Register (MSCM_CPxTYPE)	32	R	See section	15.3.2/345
4000_1004	Processor X Number Register (MSCM_CPxNUM)	32	R	See section	15.3.3/346
4000_1008	Processor X Master Register (MSCM_CPxMASTER)	32	R	See section	15.3.4/346
4000_100C	Processor X Count Register (MSCM_CPxCOUNT)	32	R	See section	15.3.5/347
4000_1010	Processor X Configuration Register (MSCM_CPxCFG0)	32	R	See section	15.3.6/348
4000_1014	Processor X Configuration Register (MSCM_CPxCFG1)	32	R	See section	15.3.6/348
4000_1018	Processor X Configuration Register (MSCM_CPxCFG2)	32	R	See section	15.3.6/348
4000_101C	Processor X Configuration Register (MSCM_CPxCFG3)	32	R	See section	15.3.6/348
4000_1020	Processor 0 Type Register (MSCM_CP0TYPE)	32	R	See section	15.3.7/349
4000_1024	Processor 0 Number Register (MSCM_CP0NUM)	32	R	See section	15.3.8/350
4000_1028	Processor 0 Master Register (MSCM_CP0MASTER)	32	R	See section	15.3.9/350
4000_102C	Processor 0 Count Register (MSCM_CP0COUNT)	32	R	See section	15.3.10/351
4000_1030	Processor 0 Configuration Register (MSCM_CP0CFG0)	32	R	See section	15.3.11/352
4000_1034	Processor 0 Configuration Register (MSCM_CP0CFG1)	32	R	See section	15.3.11/352
4000_1038	Processor 0 Configuration Register (MSCM_CP0CFG2)	32	R	See section	15.3.11/352

Table continues on the next page...

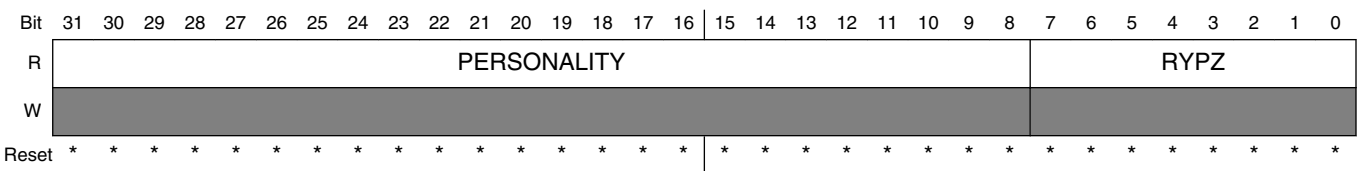
MSCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_103C	Processor 0 Configuration Register (MSCM_CP0CFG3)	32	R	See section	15.3.11/352
4000_1400	On-Chip Memory Descriptor Register (MSCM_OCMDR0)	32	R/W	See section	15.3.12/353
4000_1404	On-Chip Memory Descriptor Register (MSCM_OCMDR1)	32	R/W	See section	15.3.12/353
4000_1408	On-Chip Memory Descriptor Register (MSCM_OCMDR2)	32	R/W	See section	15.3.12/353
4000_140C	On-Chip Memory Descriptor Register (MSCM_OCMDR3)	32	R/W	See section	15.3.12/353

15.3.2 Processor X Type Register (MSCM_CPxTYPE)

The register provides a CPU-specific response indicating the personality of the core making the access. The 32 bit response includes 3 ASCII characters defining the CPU type along with a byte defining the logical revision number. The logical revision number follows ARM's rYpZ nomenclature.

Address: 4000_1000h base + 0h offset = 4000_1000h



* Notes:
- PERSONALITY field: See bit field description
- RYPZ field: See bit field description

MSCM_CPxTYPE field descriptions

Field	Description
31–8 PERSONALITY	Processor x Personality This read-only field defines the processor personality for CPx if CPx = Cortex-M4, then PERSONALITY = 0x43_4D_34 ("CM4").
RYPZ	Processor x Revision This read-only field defines the processor revision for CPx: 0x01 corresponds to the r0p1 core release. ...

15.3.3 Processor X Number Register (MSCM_CPxNUM)

The register provides a CPU-specific response indicating the logical processor number of the core making the access. In single processor configurations, the logical processor number is always zero.

Address: 4000_1000h base + 4h offset = 4000_1004h

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															CPN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0*

* Notes:
- CPN field: See bit field description

MSCM_CPxNUM field descriptions

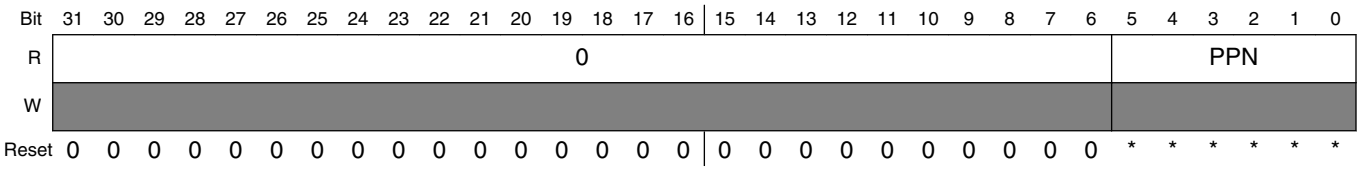
Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CPN	Processor x Number This zero-filled word defines the logical processor number for CPx **NOTE:** If single core configuration, then CPN = 0.

15.3.4 Processor X Master Register (MSCM_CPxMASTER)

The register provides a CPU-specific response indicating the physical bus master number the core making the access. The 32 bit response defines the physical master number for processor x.

A privileged read from the CM4 returns the appropriate processor information. Reads from any other bus master return all zeroes. Attempted user mode or write accesses are terminated with an error.

Address: 4000_1000h base + 8h offset = 4000_1008h



* Notes:
- PPN field: See the bit field description.

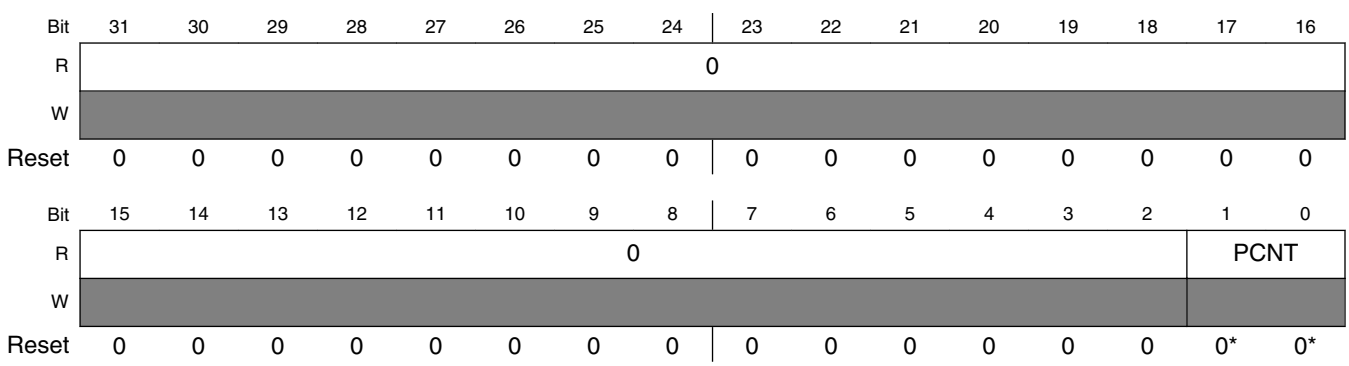
MSCM_CPxMASTER field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PPN	Processor x Physical Port Number This read-only field defines the physical port number for CPUx. **NOTE:** For single core (CPU0), PPN = 0x00.

15.3.5 Processor X Count Register (MSCM_CPxCOUNT)

The register provides a CPU-specific response indicating the total number of processor cores in the chip configuration.

Address: 4000_1000h base + Ch offset = 4000_100Ch



* Notes:
- PCNT field: See bit field description

MSCM_CPxCOUNT field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MSCM_CPxCOUNT field descriptions (continued)

Field	Description
PCNT	Processor Count This read-only field defines the processor count for the chip configuration: **NOTE:** If single core configuration, then PCNT = 00.

15.3.6 Processor X Configuration Register (MSCM_CPxCFG*n*)

The register provides information on the Level 1 caches (if present).

Address: 4000_1000h base + 10h 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	ICSZ								0								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:
- The reset values are different for the individual CPxCFG registers. CPxCFG0: 0x0000_0000; CPxCFG1: 0x0000_0000; CPxCFG2: 0x0701_0801; CPxCFG3: 0x0000_0000. x = Undefined at reset.

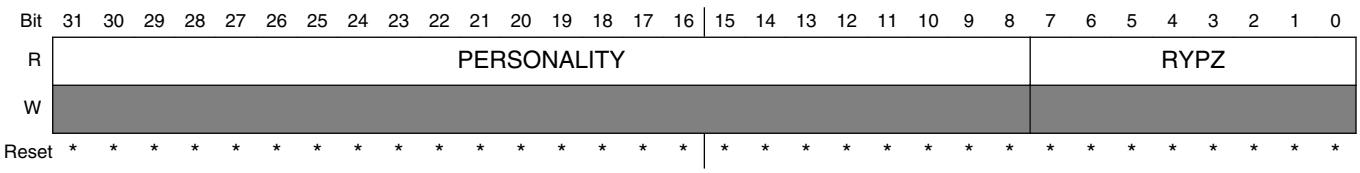
MSCM_CPxCFG*n* field descriptions

Field	Description
31–24 ICSZ	Level 1 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+SZ)}$, where SZ is non-zero; a SZ = 0 indicates the memory is not present. if no Instruction Cache, then ICSZ = 0x00 if a 4 Kbyte Instruction Cache, then ICSZ = 0x03 if an 8 Kbyte Instruction Cache, then ICSZ = 0x04 if a 16 Kbyte Instruction Cache, then ICSZ = 0x05 if a 32 Kbyte Instruction Cache, then ICSZ = 0x06 if a 64 Kbyte Instruction Cache, then ICSZ = 0x07 if a 128 Kbyte Instruction Cache, then ICSZ = 0x08 if a 256 Kbyte Instruction Cache, then ICSZ = 0x09 if a 512 Kbyte Instruction Cache, then ICSZ = 0x0A
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

15.3.7 Processor 0 Type Register (MSCM_CP0TYPE)

The register provides a CPU-specific response indicating the personality of the core making the access. The 32 bit response includes 3 ASCII characters defining the CPU type along with a byte defining the logical revision number. The logical revision number follows ARM's rYpZ nomenclature.

Address: 4000_1000h base + 20h offset = 4000_1020h



* Notes:
- PERSONALITY field: See bit field description
- RYPZ field: See bit field description

MSCM_CP0TYPE field descriptions

Field	Description
31–8 PERSONALITY	Processor x Personality This read-only field defines the processor personality for CPx if CPx = Cortex-M4, then PERSONALITY = 0x43_4D_34 ("CM4").
RYPZ	Processor x Revision This read-only field defines the processor revision for CPx: 0x01 corresponds to the r0p1 core release. ...

15.3.8 Processor 0 Number Register (MSCM_CP0NUM)

The register provides a CPU-specific response indicating the logical processor number of the core making the access. In single processor configurations, the logical processor number is always zero.

Address: 4000_1000h base + 24h offset = 4000_1024h

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															CPN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*

* Notes:
- CPN field: See bit field description

MSCM_CP0NUM field descriptions

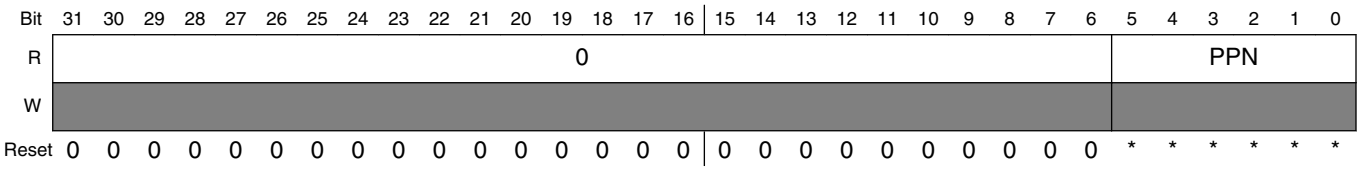
Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CPN	Processor x Number This zero-filled word defines the logical processor number for CPx If single core configuration, then CPN = 0

15.3.9 Processor 0 Master Register (MSCM_CP0MASTER)

The register provides a CPU-specific response indicating the physical bus master number the core making the access. The 32 bit response defines the physical master number for processor x.

A privileged read from the CA5 or the CM4 returns the appropriate processor information. Reads from any other bus master return all zeroes. Attempted user mode or write accesses are terminated with an error.

Address: 4000_1000h base + 28h offset = 4000_1028h



* Notes:
- PPN field: See the bit field description.

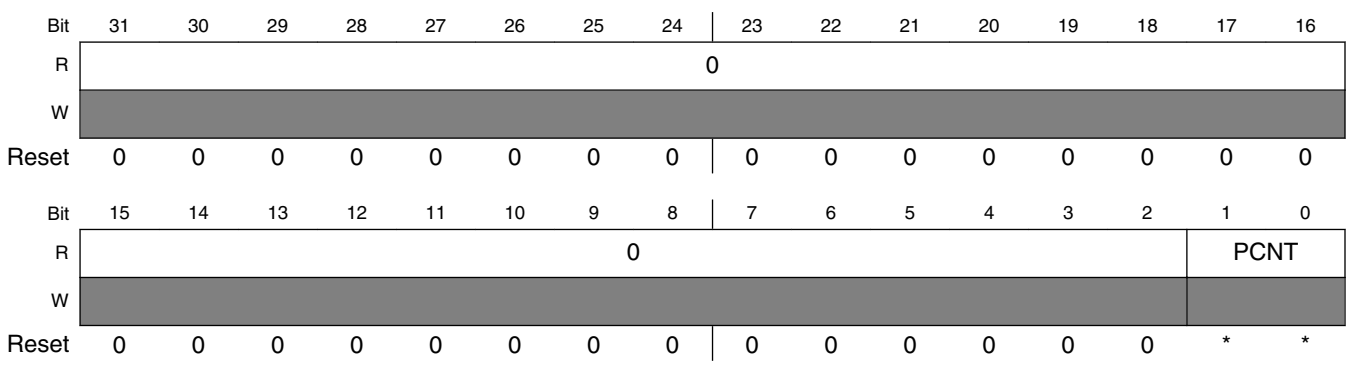
MSCM_CP0MASTER field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PPN	Processor x Physical Port Number This read-only field defines the physical port number for CPUx. For CPU0, PPN = 0x00

15.3.10 Processor 0 Count Register (MSCM_CP0COUNT)

The register provides a CPU-specific response indicating the total number of processor cores in the chip configuration.

Address: 4000_1000h base + 2Ch offset = 4000_102Ch



* Notes:
- PCNT field: See bit field description

MSCM_CP0COUNT field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MSCM_CP0COUNT field descriptions (continued)

Field	Description
PCNT	Processor Count This read-only field defines the processor count for the chip configuration: If single core configuration, then PCNT = 00

15.3.11 Processor 0 Configuration Register (MSCM_CP0CFG*n*)

The register provides information on the Level 1 caches (if present).

Address: 4000_1000h base + 30h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ICSZ								0								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:
- The reset values are different for the individual CP0CFG registers. CP0CFG0: 0x0000_0000; CP0CFG1: 0x0000_0000; CP0CFG2: 0x0701_0801; CP0CFG3: 0x0000_0000. x = Undefined at reset.

MSCM_CP0CFG*n* field descriptions

Field	Description
31–24 ICSZ	Level 1 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+SZ)}$, where SZ is non-zero; a SZ = 0 indicates the memory is not present. if no Instruction Cache, then ICSZ = 0x00 if a 4 Kbyte Instruction Cache, then ICSZ = 0x03 if an 8 Kbyte Instruction Cache, then ICSZ = 0x04 if a 16 Kbyte Instruction Cache, then ICSZ = 0x05 if a 32 Kbyte Instruction Cache, then ICSZ = 0x06 if a 64 Kbyte Instruction Cache, then ICSZ = 0x07 if a 128 Kbyte Instruction Cache, then ICSZ = 0x08 if a 256 Kbyte Instruction Cache, then ICSZ = 0x09 if a 512 Kbyte Instruction Cache, then ICSZ = 0x0A
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

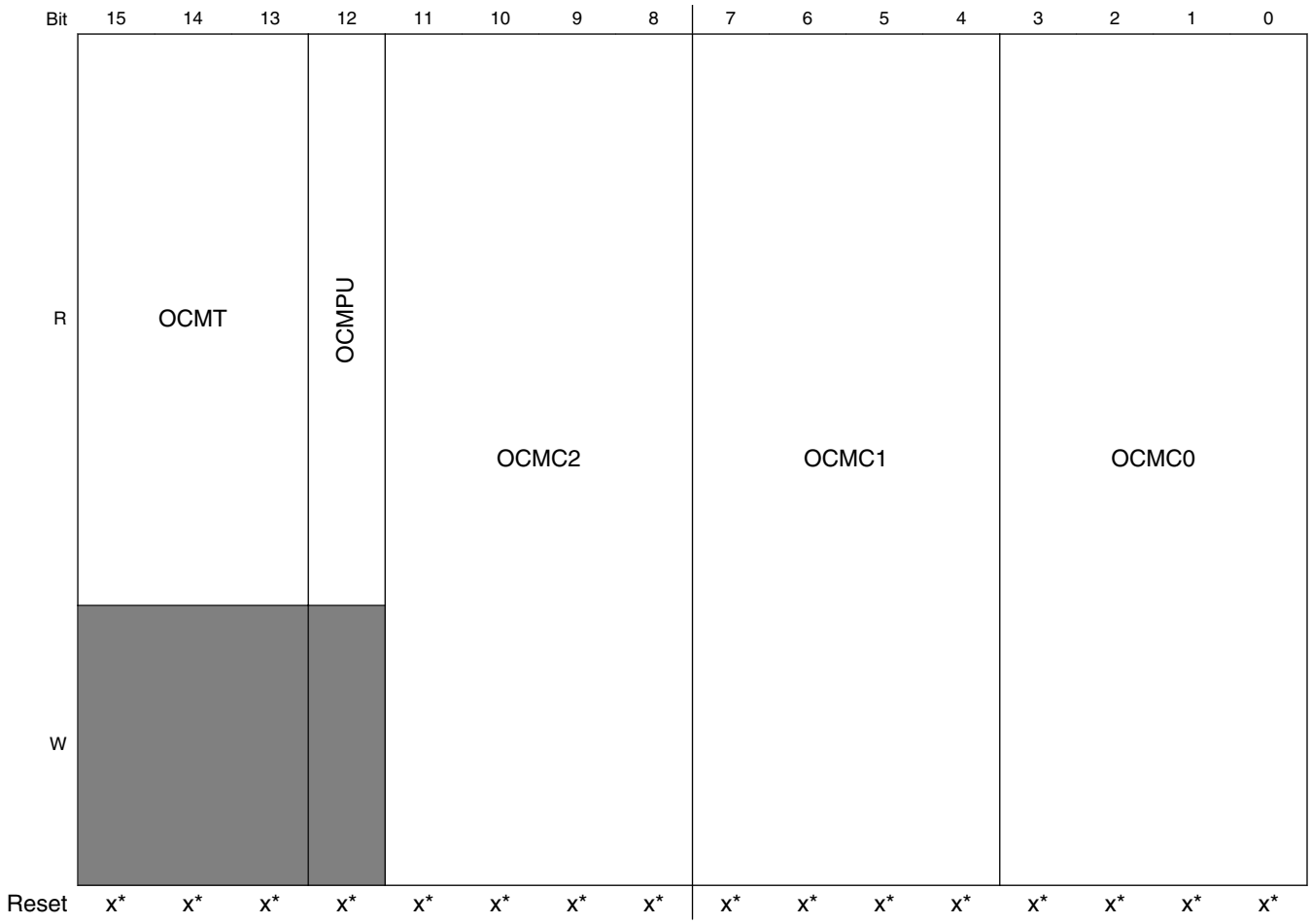
15.3.12 On-Chip Memory Descriptor Register (MSCM_OCMDR*n*)

This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information on the attached memories as well as configurable controls (where appropriate).

Privileged 32-bit reads from a processor core or the debugger return the appropriate processor information. Reads from any other bus master return all zeroes. Privileged writes from a processor core or the debugger to writeable registers update the appropriate fields. Privileged writes from other bus masters are ignored. Attempted user mode accesses or any access with a size other than 32 bits are terminated with an error.

Address: 4000_1000h base + 400h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	V	Reserved	Reserved	OCMSZH	OCMSZ				Reserved				OCMW			RO
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*



* Notes:
- The reset values are different for the individual OCMDR registers. OCMDR0: 0xCA08_9000; OCMDR1: 0xC706_B000; OCMDR2: 0xC304_D000; OCMDR3: 0xC504_7000. x = Undefined at reset.

MSCM_OCMDR*n* field descriptions

Field	Description
31 V	OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory 0 OCMEMn is not present. 1 OCMEMn is present.
30 Reserved	This field is reserved.
29 Reserved	This field is reserved.
28 OCMSZH	OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0 OCMEMn is a power-of-2 capacity. 1 OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.

Table continues on the next page...

MSCM_OCMDR*n* field descriptions (continued)

Field	Description
27–24 OCMSZ	OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+SZ)}$ where SZ is non-zero; a SZ = 0 indicates the memory is not present. 0000 no OCMEMn 0011 4KB OCMEMn 0100 8KB OCMEMn 0101 16KB OCMEMn 0110 32KB OCMEMn 0111 64KB OCMEMn 1000 128KB OCMEMn 1001 256KB OCMEMn 1010 512KB OCMEMn 1011 1024KB OCMEMn 1100 2048KB OCMEMn 1101 4096KB OCMEMn 1110 8192KB OCMEMn 1111 16384KB OCMEMn
23–20 Reserved	This field is reserved.
19–17 OCMW	OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000-001 Reserved 010 OCMEMn 32-bits wide 011 OCMEMn 64-bits wide 100 OCMEMn 128-bits wide 101 OCMEMn 256-bits wide 110-111 Reserved
16 RO	Read-Only. This register bit provides a mechanism to “lock” the configuration state defined by OCMDRn[11:0]. Once asserted, attempted writes to the OCMDRn[11:0] register are ignored until the next reset clears the flag. 0 writes to the OCMDRn[11:0] are allowed 1 writes to the OCMDRn[11:0] are ignored
15–13 OCMT	OCMEM Type. This field defines the type of the on-chip memory: 000 Reserved 001 Reserved 010 Reserved 011 OCMEMn is a ROM. 100 OCMEMn is a program flash. 101 OCMEMn is a data flash. 110 OCMEMn is an EEE. 111 Reserved
12 OCMPU	OCMEM Memory Protection Unit. This read-only field identifies a memory protected by a Memory Protection Unit. 0 OCMEMn is not protected by an MPU. 1 OCMEMn is protected by an MPU.

Table continues on the next page...

MSCM_OCMDR*n* field descriptions (continued)

Field	Description
11–8 OCMC2	OCMEM Control Field 2. This 4-bit field (if used) defines the configuration of the on-chip memory. The field's functionality is dependent on the OCMT value.
7–4 OCMC1	OCMEM Control Field 1. This 4-bit field (if used) defines the configuration of the on-chip memory. The field's functionality is dependent on the OCMT value. OCMDR0[5] or OCMDR1[5] bit controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches or data references. • OCMDR bit 4: data prefetch. Value 0 means enable and value 1 means disable. • OCMDR bit 5: flash speculate. Value 0 means enable and value 1 means disable. **NOTE:** The control bit is only applicable when OCMT = 100 (program flash) or 101 (data flash). In other cases, it is unused.
OCMC0	OCMEM Control Field 0. This 4-bit field (if used) defines the configuration of the on-chip memory. The field's functionality is dependent on the OCMT value.

Chapter 16
Flash Acceleration Unit (FAU)

16.1 Flash Acceleration Unit (FAU)

16.1.1 Introduction

The Flash Acceleration Unit (FAU) is a memory acceleration unit. It includes a buffer that can accelerate program flash memory data transfers. In addition, this module provides two separate mechanisms for accelerating the interface between bus masters and program flash memory. A 128-bit speculation buffer can prefetch the next 128-bit flash memory location.

16.1.2 Modes of operation

The FAU operates only when a bus master accesses the program flash memory or FlexMemory.

In terms of chip power modes:

- The FAU operates only in Run and Wait modes, including VLPR and VLPW modes.
- For any power mode where the program flash memory or FlexMemory cannot be accessed, the FAU is disabled.

16.1.3 External signal description

The FAU has no external (off-chip) signals.

16.1.4 Functional description

The FAU is a flash acceleration unit with flexible buffers for user configuration.

Whenever a hit occurs for the prefetch speculation buffer, or the single-entry buffer, the requested data is transferred within a single system clock.

Default configuration

Upon system reset, the FAU is configured to provide a significant level of buffering for transfers from the flash memory or FlexMemory:

- For bank 0 (Program Flash) and bank 1 (Data Flash):
  - Prefetch support for data and instructions is enabled.
  - The single-entry buffer is enabled.

Speculative reads

The FAU has a single buffer that reads ahead to the next word in the flash memory if there is an idle cycle. Speculative prefetching is programmable for each bank for instruction and/or data accesses using MSCM_OCMDR0[5] and MSCM_OCMDR1[5] (value 0 means "enables prefetches (or speculative accesses)"). Because many code accesses are sequential, using the speculative prefetch buffer improves performance in most cases.

When speculative reads are enabled, the FAU immediately requests the next sequential address after a read completes. By requesting the next word immediately, speculative reads can help to reduce or even eliminate wait states when accessing sequential code and/or data.

For example, consider the following scenario:

- Assume a system with a 4:1 core-to-flash clock ratio and with speculative reads enabled.
- The core requests four (for Data Flash, bank 1) or eight (for Program Flash, bank 0) sequential longwords in back-to-back requests, meaning there are no core cycle delays except for stalls waiting for flash memory data to be returned.
- None of the data is already stored in the speculation buffer.

In this scenario, the sequence of events for accessing the four (for Data Flash, bank 1) or eight (for Program Flash, bank 0) longwords is as follows:

1. The first longword read requires 4 to 7 core clocks.
2. Due to the 128-bit data bus of the flash memory, the second longword read takes only 1 core clock because the data is already available inside the FAU. For the same reason, the third and fourth longword reads each take only 1 core clock.
3. Per 64-bit for Data Flash (bank 1), accessing the third longword requires 3 core clock cycles. The flash memory read itself takes 4 clocks, but the first clock overlaps with the second longword read.

4. Per 128-bit for Program Flash (bank 0), accessing the fifth longword requires 1 core clock cycle. The flash memory read itself takes 4 clocks, but the access starts immediately after the first read. As a result, 3 clocks for this access overlap with the second, third, and fourth longword reads from the core.
5. Per 64-bit for Data Flash (bank 1), reading the fourth longword, like the second longword, takes only 1 clock due to the 64-bit flash memory data bus.
6. Per 128-bit for Program Flash (bank 0), reading the sixth, seventh, and eighth longwords takes only 1 clock each because the data is already available inside the FAU.

16.2 Usage Guide

For many systems the on-chip flash is the main memory. The Flash Acceleration Unit (FAU) is the interface between the flash memory blocks and the system. In a typical configuration, the core and system bus clock speeds are clock significantly faster than the flash memory clock. The FAU includes features designed to accelerate flash accesses.

For more detailed information, refer to the FMC (same module as FAU) section in AN4745: Optimizing Performance on Kinetis K-series MCUs.

Chapter 17
Flash Memory Module (FTFE)

17.1 Chip-specific Information for this Module

The chip-specific Flash information is as below. See the "Ordering information" section and the cover-page "Memory and memory interfaces" feature list in DataSheet for more information.

- Program Flash = 512 or 256 KB
- SRAM = 64 or 32 KB
- FlexNVM = 64 KB
- FlexRAM = 4 KB

NOTE

For device with BootROM, the flash driver is exported from ROM for customer use. Please visit http://www.nxp.com/kboot for more information.

17.2 Introduction

The FTFE module includes the following accessible memory regions:

- Program flash memory for vector space and code store
- FlexNVM for data store and additional code store
- FlexRAM for high-endurance data store or traditional RAM

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The FTFE module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

17.2.1 Features

The FTFE module includes the following features.

NOTE

See Memories and Memory Interfaces chapter for the exact amount of flash memory available on your device.

17.2.1.1 Program Flash Memory Features

- Sector size of 4 Kbytes
- Program flash protection scheme prevents accidental program or erase of stored data
- Program flash access control scheme prevents unauthorized access to selected code segments
- Automated, built-in, program and erase algorithms with verify
- Section programming for faster bulk programming times
- Read access to the program flash block is possible while programming or erasing data in the data flash block or FlexRAM

17.2.1.2 FlexNVM memory features

When FlexNVM is partitioned for data flash memory:

- Sector size of 2 Kbytes
- Protection scheme prevents accidental program or erase of stored data
- Automated, built-in program and erase algorithms with verify
- Section programming for faster bulk programming times
- Read access to the data flash block possible while programming or erasing data in the program flash block

17.2.1.3 FlexRAM features

- Memory that can be used as traditional RAM or as high-endurance EEPROM storage
- Up to 4 Kbytes of FlexRAM configured for EEPROM or traditional RAM operations
- When configured for EEPROM:
  - Protection scheme prevents accidental program or erase of data written for EEPROM
  - Built-in hardware emulation scheme to automate EEPROM record maintenance functions
  - Programmable EEPROM data set size and FlexNVM partition code facilitating EEPROM memory endurance trade-offs
  - Supports FlexRAM aligned writes of 1, 2, or 4 bytes at a time
  - Read access to FlexRAM possible while programming or erasing data in the program or data flash memory
- When configured for traditional RAM:
  - Read and write access possible to the FlexRAM while programming or erasing data in the program or data flash memory

17.2.1.4 Other FTFE module features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

17.2.2 Block diagram

The block diagram of the FTFE module is shown in the following figure.

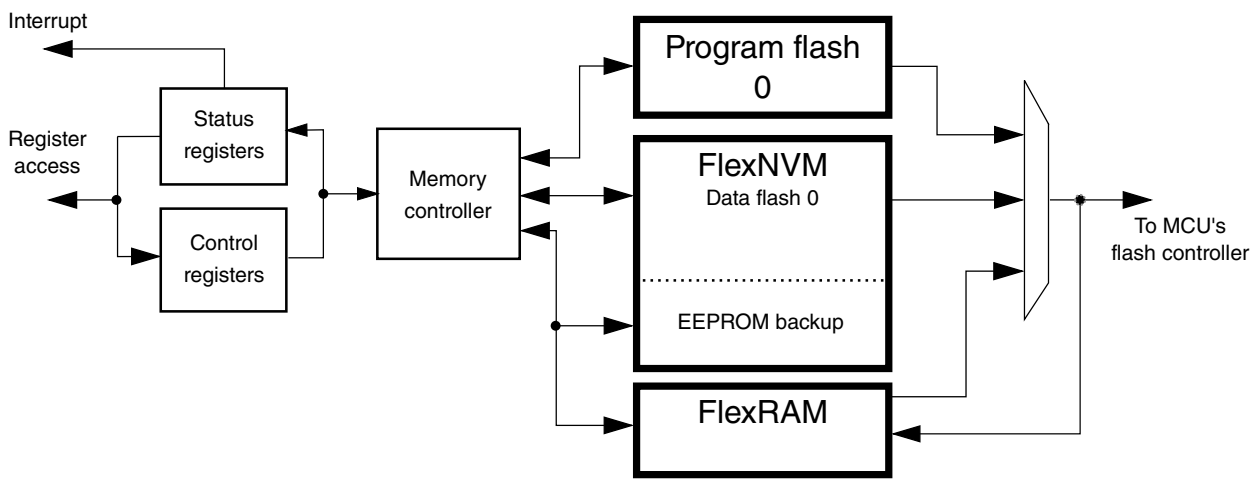


Figure 17-1. FTFE block diagram

17.2.3 Glossary

Command write sequence — A series of MCU writes to the Flash FCCOB register group that initiates and controls the execution of Flash algorithms that are built into the FTFE module.

Data flash memory — Partitioned from the FlexNVM block, the data flash memory provides nonvolatile storage for user data, boot code, and additional code store.

Data flash sector — The data flash sector is the smallest portion of the data flash memory that can be erased.

EEPROM — Using a built-in filing system, the FTFE module emulates the characteristics of an EEPROM by effectively providing a high-endurance, byte-writeable (program and erase) NVM.

EEPROM backup data header — The EEPROM backup data header is comprised of a 64-bit field found in EEPROM backup data memory which contains information used by the EEPROM filing system to determine the status of a specific EEPROM backup flash sector.

EEPROM backup data record — The EEPROM backup data record is comprised of a 7-bit status field, a 13-bit address field, and a 32-bit data field found in EEPROM backup data memory which is used by the EEPROM filing system. If the status field indicates a record is valid, the data field is mirrored in the FlexRAM at a location determined by the address field.

EEPROM backup data memory — Partitioned from the FlexNVM block, EEPROM backup data memory provides nonvolatile storage for the EEPROM filing system representing data written to the FlexRAM requiring highest endurance.

EEPROM backup data sector — The EEPROM backup data sector contains one EEPROM header and up to 255 EEPROM backup data records, which are used by the EEPROM filing system.

Endurance — The number of times that a flash memory location can be erased and reprogrammed.

FCCOB (Flash Common Command Object) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the FTFE module.

Flash block — A macro within the FTFE module which provides the nonvolatile memory storage.

FlexMemory — FTFE configuration that supports data flash, EEPROM, and FlexRAM.

FlexNVM Block — The FlexNVM block can be configured to be used as data flash memory, EEPROM backup flash memory, or a combination of both.

FlexRAM — The FlexRAM refers to a RAM, dedicated to the FTFE module, that can be configured to store EEPROM data or as traditional RAM. When configured for EEPROM, valid writes to the FlexRAM generates a new EEPROM backup data record stored in the EEPROM backup flash memory.

FTFE Module — All flash blocks plus 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 FTFE 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.

NVM — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

NVM Normal Mode — An NVM mode that provides basic user access to FTFE resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the FTFE module.

Double-Phrase — 128 bits of data with an aligned double-phrase having byte-address[3:0] = 0000.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

Program flash — The program flash memory provides nonvolatile storage for vectors and code store.

Program flash sector — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

Retention — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

RWW— Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

Section program buffer — Lower quarter of the FlexRAM allocated for storing large amounts of data for programming via the Program Section command.

Secure — An MCU state conveyed to the FTFE module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

17.3 External signal description

The FTFE module contains no signals that connect off-chip.

17.4 Memory map and registers

This section describes the memory map and registers for the FTFE module. Data read from unimplemented memory space in the FTFE module is undefined. Writes to unimplemented or reserved memory space (registers) in the FTFE module are ignored.

17.4.1 Flash configuration field description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the FTFE module.

NOTE

The flash configuration field offset addresses are relative byte addresses. Check your device specific memory map for the location of the program flash memory.

Flash Configuration Field Offset Address	Size (Bytes)	Field Description
0x0_0400 - 0x0_0407	8	Backdoor Comparison Key. Refer to Verify Backdoor Access Key command and Unsecuring the MCU Using Backdoor Key Access.
0x0_0408 - 0x0_040B	4	Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).
0x0_040F	1	Data flash protection byte. Refer to the description of the Data Flash Protection Register (FDPROT).
0x0_040E	1	EEPROM protection byte. Refer to the description of the EEPROM Protection Register (FEPROT).
0x0_040D	1	Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).
0x0_040C	1	Flash security byte. Refer to the description of the Flash Security Register (FSEC).

17.4.2 Program flash 0 IFR map

The program flash 0 IFR is a 1 Kbyte nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see Read Resource command, Read Once command, Program Once command). The program flash 0 IFR is located within the program flash 0 memory block. The contents of the program flash 0 IFR are summarized in the following table.

Offset Address Range	Size (Bytes)	Field Description
0x000 – 0x39F	928	Reserved
0x3A0 – 0x3A3	4	Program Once XACCH-1 Field (index = 0x08)
0x3A4 – 0x3A7	4	Program Once XACCL-1 Field (index = 0x08)
0x3A8 – 0x3AB	4	Program Once XACCH-2 Field (index = 0x09)
0x3AC – 0x3AF	4	Program Once XACCL-2 Field (index = 0x09)
0x3B0 – 0x3B3	4	Program Once SACCH-1 Field (index = 0x0A)
0x3B4 – 0x3B7	4	Program Once SACCL-1 Field (index = 0x0A)
0x3B8 – 0x3BB	4	Program Once SACCH-2 Field (index = 0x0B)
0x3BC – 0x3BF	4	Program Once SACCL-2 Field (index = 0x0B)
0x3C0 – 0x3FF	64	Program Once ID Field (index = 0x00 - 0x07)

17.4.2.1 Program Once field

The Program Once field in the program flash 0 IFR provides 96 bytes of user data storage separate from the program flash 0 main array. The user can program the Program Once field one time only as there is no erase mechanism available for the program flash 0 IFR. The Program Once field can be read any number of times. This section of the program flash 0 IFR is accessed in 8 byte records using the Read Once command and Program Once command.

17.4.3 Data flash 0 IFR map

The data flash 0 IFR is a 1 Kbyte nonvolatile information memory that can be read and erased, but the user has limited program capabilities in the data flash 0 IFR (see Read Resource command, Erase All Blocks command, Erase All Blocks Unsecure command, Program Partition command). The data flash 0 IFR is located within the data flash 0 memory block. The contents of the data flash 0 IFR are summarized in the following table.

Offset Address Range	Size (Bytes)	Field Description
0x00 – 0x3FB, 0x3FE – 0x3FF	1022	Reserved
0x3FD	1	EEPROM Data Set Size
0x3FC	1	FlexNVM Partition Code

17.4.3.1 EEPROM Data Set Size

The EEPROM data set size byte in the data flash 0 IFR supplies information which determines the amount of FlexRAM used in each of the available EEPROM subsystems and indicates whether the FlexRAM is loaded with valid EEPROM data during the flash reset sequence. To program the EEERST, EEESIZE value, see the Program Partition command described in Program Partition command.

Table 17-1. EEPROM Data Set Size

Data flash IFR: 0x03FD							
7	6	5	4	3	2	1	0
1	EEERST	EEESPLIT		EEESIZE			
	= Unimplemented or Reserved						

Table 17-2. EEPROM Data Set Size Field Description

Field	Description
7 Reserved	This read-only bitfield is reserved and must always be written as one.
6 EEERST	**EEPROM Load on Reset** — Determines whether the flash reset sequence takes time to load the FlexRAM with valid EEPROM data. '0' = FlexRAM is not loaded with valid EEPROM data during the flash reset sequence (see the Set FlexRAM Function command described in Set FlexRAM Function command to load the FlexRAM with valid EEPROM data) '1' = FlexRAM is loaded with valid EEPROM data during the flash reset sequence
5-4 EEESPLIT	This read-only bitfield is reserved and each bit will always read as one.

Table continues on the next page...

Table 17-2. EEPROM Data Set Size Field Description (continued)

Field	Description
3-0 EEESIZE	**EEPROM Size** — Encoding of the total available FlexRAM for EEPROM use. **NOTE:** EEESIZE must be 0 bytes (1111b) when the FlexNVM partition code (FlexNVM partition code) is set to 'No EEPROM'. '0000' = Reserved '0001' = Reserved '0010' = 4,096 Bytes '0011' = 2,048 Bytes '0100' = 1,024 Bytes '0101' = 512 Bytes '0110' = 256 Bytes '0111' = 128 Bytes '1000' = 64 Bytes '1001' = 32 Bytes '1010' = Reserved '1011' = Reserved '1100' = Reserved '1101' = Reserved '1110' = Reserved '1111' = 0 Bytes

17.4.3.2 FlexNVM partition code

The FlexNVM partition code byte in the data flash 0 IFR supplies a code which specifies how to split the FlexNVM block between data flash memory and EEPROM backup memory supporting EEPROM functions. To program the DEPART value, see the Program Partition command described in Program Partition command.

Table 17-3. FlexNVM partition code

Data Flash IFR: 0x03FC							
7	6	5	4	3	2	1	0
1	1	1	1	DEPART			
	= Unimplemented or Reserved						

Table 17-4. FlexNVM partition code field description

Field	Description
7-4	This read-only bitfield is reserved and must always be written as one.

Table continues on the next page...

Table 17-4. FlexNVM partition code field description (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Reserved</td><td></td></tr>
<tr><td>3-0
DEPART</td><td>FlexNVM Partition Code — Encoding of the data flash / EEPROM backup split within the FlexNVM memory block. FlexNVM memory not partitioned for data flash is used to store EEPROM records.
<table>
<tr><th>DEPART</th><th>Data flash (KByte)</th><th>EEPROM backup (KByte)</th></tr>
<tr><td>0000</td><td>64</td><td>0</td></tr>
<tr><td>0001</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0010</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0011</td><td>32</td><td>32</td></tr>
<tr><td>0100</td><td>0</td><td>64</td></tr>
<tr><td>0101</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0110</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0111</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1000</td><td>0</td><td>64</td></tr>
<tr><td>1001</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1010</td><td>16</td><td>48</td></tr>
<tr><td>1011</td><td>32</td><td>32</td></tr>
<tr><td>1100</td><td>64</td><td>0</td></tr>
<tr><td>1101</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1110</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1111</td><td>64</td><td>0</td></tr>
</table></td></tr>
</table>

17.4.4 Register descriptions

The FTFE module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

FTFE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0000	Flash Status Register (FTFE_FSTAT)	8	R/W	00h	17.4.4.1/ 373
4002_0001	Flash Configuration Register (FTFE_FCNFG)	8	R/W	00h	17.4.4.2/ 375
4002_0002	Flash Security Register (FTFE_FSEC)	8	R	Undefined	17.4.4.3/ 377
4002_0003	Flash Option Register (FTFE_FOPT)	8	R	Undefined	17.4.4.4/ 378
4002_0004	Flash Common Command Object Registers (FTFE_FCCOB3)	8	R/W	00h	17.4.4.5/ 379
4002_0005	Flash Common Command Object Registers (FTFE_FCCOB2)	8	R/W	00h	17.4.4.5/ 379
4002_0006	Flash Common Command Object Registers (FTFE_FCCOB1)	8	R/W	00h	17.4.4.5/ 379
4002_0007	Flash Common Command Object Registers (FTFE_FCCOB0)	8	R/W	00h	17.4.4.5/ 379
4002_0008	Flash Common Command Object Registers (FTFE_FCCOB7)	8	R/W	00h	17.4.4.5/ 379
4002_0009	Flash Common Command Object Registers (FTFE_FCCOB6)	8	R/W	00h	17.4.4.5/ 379
4002_000A	Flash Common Command Object Registers (FTFE_FCCOB5)	8	R/W	00h	17.4.4.5/ 379
4002_000B	Flash Common Command Object Registers (FTFE_FCCOB4)	8	R/W	00h	17.4.4.5/ 379
4002_000C	Flash Common Command Object Registers (FTFE_FCCOBB)	8	R/W	00h	17.4.4.5/ 379
4002_000D	Flash Common Command Object Registers (FTFE_FCCOBA)	8	R/W	00h	17.4.4.5/ 379
4002_000E	Flash Common Command Object Registers (FTFE_FCCOB9)	8	R/W	00h	17.4.4.5/ 379
4002_000F	Flash Common Command Object Registers (FTFE_FCCOB8)	8	R/W	00h	17.4.4.5/ 379
4002_0010	Program Flash Protection Registers (FTFE_FPROT3)	8	R/W	Undefined	17.4.4.6/ 380
4002_0011	Program Flash Protection Registers (FTFE_FPROT2)	8	R/W	Undefined	17.4.4.6/ 380
4002_0012	Program Flash Protection Registers (FTFE_FPROT1)	8	R/W	Undefined	17.4.4.6/ 380
4002_0013	Program Flash Protection Registers (FTFE_FPROT0)	8	R/W	Undefined	17.4.4.6/ 380
4002_0016	EEPROM Protection Register (FTFE_FEPROT)	8	R/W	Undefined	17.4.4.7/ 382
4002_0017	Data Flash Protection Register (FTFE_FDPROT)	8	R/W	Undefined	17.4.4.8/ 383

Table continues on the next page...

FTFE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0018	Execute-only Access Registers (FTFE_XACCH3)	8	R	Undefined	17.4.4.9/ 384
4002_0019	Execute-only Access Registers (FTFE_XACCH2)	8	R	Undefined	17.4.4.9/ 384
4002_001A	Execute-only Access Registers (FTFE_XACCH1)	8	R	Undefined	17.4.4.9/ 384
4002_001B	Execute-only Access Registers (FTFE_XACCH0)	8	R	Undefined	17.4.4.9/ 384
4002_001C	Execute-only Access Registers (FTFE_XACCL3)	8	R	Undefined	17.4.4.9/ 384
4002_001D	Execute-only Access Registers (FTFE_XACCL2)	8	R	Undefined	17.4.4.9/ 384
4002_001E	Execute-only Access Registers (FTFE_XACCL1)	8	R	Undefined	17.4.4.9/ 384
4002_001F	Execute-only Access Registers (FTFE_XACCL0)	8	R	Undefined	17.4.4.9/ 384
4002_0020	Supervisor-only Access Registers (FTFE_SACCH3)	8	R	Undefined	17.4.4.10/ 385
4002_0021	Supervisor-only Access Registers (FTFE_SACCH2)	8	R	Undefined	17.4.4.10/ 385
4002_0022	Supervisor-only Access Registers (FTFE_SACCH1)	8	R	Undefined	17.4.4.10/ 385
4002_0023	Supervisor-only Access Registers (FTFE_SACCH0)	8	R	Undefined	17.4.4.10/ 385
4002_0024	Supervisor-only Access Registers (FTFE_SACCL3)	8	R	Undefined	17.4.4.10/ 385
4002_0025	Supervisor-only Access Registers (FTFE_SACCL2)	8	R	Undefined	17.4.4.10/ 385
4002_0026	Supervisor-only Access Registers (FTFE_SACCL1)	8	R	Undefined	17.4.4.10/ 385
4002_0027	Supervisor-only Access Registers (FTFE_SACCL0)	8	R	Undefined	17.4.4.10/ 385
4002_0028	Flash Access Segment Size Register (FTFE_FACSS)	8	R	Undefined	17.4.4.11/ 386
4002_002B	Flash Access Segment Number Register (FTFE_FACSN)	8	R	Undefined	17.4.4.12/ 387
4002_002E	Flash Error Status Register (FTFE_FERSTAT)	8	R/W	00h	17.4.4.13/ 388
4002_002F	Flash Error Configuration Register (FTFE_FERCNFG)	8	R/W	00h	17.4.4.14/ 388

17.4.4.1 Flash Status Register (FTFE_FSTAT)

The FSTAT register reports the operational status of the FTFE module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

NOTE

When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands or writes to the FlexRAM (when EEERDY is set) until the flag is cleared (by writing a one to it).

Address: 4002_0000h base + 0h offset = 4002_0000h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
Write	w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0

FTFE_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag The CCIF flag indicates that a FTFE command or EEPROM file system operation has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command, and CCIF stays low until command completion or command violation. The CCIF flag is also cleared by a successful write to FlexRAM while enabled for EEPROM operations, and CCIF stays low until the EEPROM file system has created the associated EEPROM data record. The CCIF bit is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value. 0 FTFE command or EEPROM file system operation in progress 1 FTFE command or EEPROM file system operation has completed
6 RDCOLERR	FTFE Read Collision Error Flag The RDCOLERR error bit indicates that the MCU attempted a read from an FTFE resource that was being manipulated by an FTFE command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect. 0 No collision error detected 1 Collision error detected
5 ACCERR	Flash Access Error Flag The ACCERR error bit indicates an illegal access has occurred to an FTFE resource caused by a violation of the command write sequence or issuing an illegal FTFE command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to 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

Table continues on the next page...

FTFE_FSTAT field descriptions (continued)

Field	Description
	The FPVIOL error bit indicates an attempt was made to program or erase an address in a protected area of program flash or data flash memory during a command write sequence or a write was attempted to a protected area of the FlexRAM while enabled for EEPROM. While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to 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 an FTFE command or during the flash reset sequence. As a status flag, this bit cannot (and need not) be cleared by the user like the other error flags in this register. The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.

17.4.4.2 Flash Configuration Register (FTFE_FCNFG)

This register provides information on the current functional state of the FTFE module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. PFLSH, RAMRDY, and EEERDY are read-only status bits. The reset values for the PFLSH, RAMRDY, and EEERDY bits are determined during the reset sequence.

Address: 4002_0000h base + 1h offset = 4002_0001h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	0	PFLSH	RAMRDY	EEERDY
Write								
Reset	0	0	0	0	0	0	0	0

FTFE_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable The CCIE bit controls interrupt generation when an FTFE command completes. 0 Command complete interrupt disabled 1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable The RDCOLLIE bit controls interrupt generation when an FTFE read collision error occurs.

Table continues on the next page...

FTFE_FCNFG field descriptions (continued)

Field	Description
	0 Read collision error interrupt disabled 1 Read collision error interrupt enabled. An interrupt request is generated whenever an FTFE read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request This bit issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command. The ERSAREQ bit sets when an erase all request is triggered external to the FTFE and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the FTFE when the operation completes. 0 No request or request complete 1 Request to: 1. run the Erase All Blocks command, 2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state
4 ERSSUSP	Erase Suspend The ERSSUSP bit allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0 No suspend requested 1 Suspend the current Erase Flash Sector command execution
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 PFLSH	FTFE configuration 0 FTFE configuration supports one program flash block and one FlexNVM block 1 Reserved
1 RAMRDY	RAM Ready This flag indicates the current status of the FlexRAM. The state of the RAMRDY flag is normally controlled by the Set FlexRAM Function command. During the reset sequence, the RAMRDY flag is cleared if the FlexNVM block is partitioned for EEPROM with the option to load the FlexRAM during the reset sequence and will be set if the FlexNVM block is not partitioned for EEPROM or if the FlexNVM block is partitioned for EEPROM with the option to not load the FlexRAM during the reset sequence. The RAMRDY flag is cleared if the Program Partition command is run to partition the FlexNVM block for EEPROM. The RAMRDY flag sets after completion of the Erase All Blocks command or execution of the erase-all operation triggered external to the FTFE. 0 FlexRAM is not available for traditional RAM access 1 FlexRAM is available as traditional RAM only; writes to the FlexRAM do not trigger EEPROM operations
0 EEERDY	This flag indicates if the EEPROM backup data has been copied to the FlexRAM and is therefore available for read access. During the reset sequence, the EEERDY flag remains clear while CCIF=0 and only sets if the FlexNVM block is partitioned for EEPROM.

Table continues on the next page...

FTFE_FCNFG field descriptions (continued)

Field	Description
	0 FlexRAM is not available for EEPROM operation 1 FlexRAM is available for EEPROM operations where: • reads from the FlexRAM return data previously written to the FlexRAM in EEPROM mode and • writes launch an EEPROM operation to store the written data in the FlexRAM and EEPROM backup

17.4.4.3 Flash Security Register (FTFE_FSEC)

This read-only register holds all bits associated with the security of the MCU and FTFE module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The Flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 2h offset = 4002_0002h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		MEEN		FSLACC		SEC	
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable These bits enable and disable backdoor key access to the FTFE module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access) 10 Backdoor key access enabled 11 Backdoor key access disabled
5–4 MEEN	Mass Erase Enable Bits Enables and disables mass erase capability of the FTFE module. When the SEC field is set to unsecure, the MEEN setting does not matter. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled

Table continues on the next page...

FTFE_FSEC field descriptions (continued)

Field	Description
3–2 FSLACC	Factory Security Level Access Code These bits enable or disable access to the flash memory contents during returned part failure analysis at NXP. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by NXP 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), NXP factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when the SEC bits are set to secure. When the SEC field is set to unsecure, the FSLACC setting does not matter. 00 Factory access granted 01 Factory access denied 10 Factory access denied 11 Factory access granted
SEC	Flash Security These bits define the security state of the MCU. In the secure state, the MCU limits access to FTFE module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the FTFE module is unsecured using backdoor key access, the SEC bits are forced to 10b. 00 MCU security status is secure 01 MCU security status is secure 10 MCU security status is unsecure (The standard shipping condition of the FTFE is unsecure.) 11 MCU security status is secure

17.4.4.4 Flash Option Register (FTFE_FOPT)

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only.

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 3h offset = 4002_0003h

Bit	7	6	5	4	3	2	1	0
Read	OPT							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FOPT field descriptions

Field	Description
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.

17.4.4.5 Flash Common Command Object Registers (FTFE_FCCOB*n*)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Address: 4002_0000h base + 4h offset + (1d × i), where i=0d to 11d

Bit	7	6	5	4	3	2	1	0
Read Write	CCOBn							
Reset	0	0	0	0	0	0	0	0

FTFE_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. The following table shows a generic FTFE command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific FTFE command, typically an address and/or data values. **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[1]	Typical Command Parameter Contents [7:0]
0	FCMD (a code that defines the FTFE command)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0]
4	Data Byte 0

FTFE_FCCOB*n* field descriptions (continued)

Field	Description
	FCCOB Number[1] / **Typical Command Parameter Contents [7:0]** 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 and Multi-Byte Access:** The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number. The FCCOB register group may be read and written as individual bytes, aligned words (2 bytes) or aligned longwords (4 bytes).

1. Refers to FCCOB register name, not register address

17.4.4.6 Program Flash Protection Registers (FTFE_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 FTFE command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions of equal memory size.

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

Table continues on the next page...

Program flash protection register	Flash Configuration Field offset address
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.

FTFE_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 to the protected state. 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. 0 Program flash region is protected. 1 Program flash region is not protected

17.4.4.7 EEPROM Protection Register (FTFE_FEPROT)

The FEPROT register defines which EEPROM regions of the FlexRAM are protected against program and erase operations. Protected EEPROM regions cannot have their content changed by writing to it. Unprotected regions can be changed by writing to the FlexRAM.

Address: 4002_0000h base + 16h offset = 4002_0016h

Bit	7	6	5	4	3	2	1	0
Read Write	EPROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FEPROT field descriptions

Field	Description
EPROT	EEPROM Region Protect Individual EEPROM regions can be protected from alteration by setting the associated EPROT bit to the protected state. The EPROT bits are not used when the FlexNVM Partition Code is set to data flash only. When the FlexNVM Partition Code is set to data flash and EEPROM or EEPROM only, each EPROT bit covers one-eighth of the configured EEPROM data (see the EEPROM Data Set Size parameter description). The protection can only be increased. This means that currently-unprotected memory can be protected, but currently-protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FEPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **Restriction:** Never write to the FEPROT register while a command is running (CCIF=0). **Reset:** During the reset sequence, the FEPROT register is loaded with the contents of the EEPROM protection byte in the Flash Configuration Field located in program flash. The flash basis for the reset values is signified by X in the register diagram. To change the EEPROM protection that will be loaded during the reset sequence, the sector of program flash that contains the Flash Configuration Field must be unprotected; then the EEPROM protection byte must be erased and reprogrammed. Trying to alter data by writing to any protected area in the EEPROM results in a protection violation error and sets the FSTAT[FPVIOL] bit. 0 EEPROM region is protected 1 EEPROM region is not protected

17.4.4.8 Data Flash Protection Register (FTFE_FDPROT)

The FDPROT register defines which data flash regions are protected against program and erase operations. Protected Flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any FTFE command. Unprotected regions can be changed by both program and erase operations.

Address: 4002_0000h base + 17h offset = 4002_0017h

Bit	7	6	5	4	3	2	1	0
Read Write	DPROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FDPROT field descriptions

Field	Description
DPROT	Data Flash Region Protect Individual data flash regions can be protected from program and erase operations by setting the associated DPROT bit to the protected state. Each DPROT bit protects one-eighth of the partitioned data flash memory space for 2^n sizes. The granularity of data flash protection cannot be less than the data flash sector size. If an unused DPROT bit is set to the protected state, the Erase all Blocks command does not execute and sets the FSTAT[FPVIOL] bit. See the Flash Protection section for more information. The protection can only be increased, meaning that currently unprotected memory can be protected but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FDPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **Restriction:** The user must never write to the FDPROT register while a command is running (CCIF=0). **Reset:** During the reset sequence, the FDPROT register is loaded with the contents of the data flash protection byte in the Flash Configuration Field located in program flash memory. The flash basis for the reset values is signified by X in the register diagram. To change the data flash protection that will be loaded during the reset sequence, unprotect the sector of program flash that contains the Flash Configuration Field. Then, erase and reprogram the data flash protection byte. Trying to alter data with the program and erase commands in any protected area in the data flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A block erase of any data flash memory block (see the Erase Flash Block command description) is not possible if the data flash block contains any protected region or if the FlexNVM memory has been partitioned for EEPROM. 0 Data Flash region is protected 1 Data Flash region is not protected

17.4.4.9 Execute-only Access Registers (FTFE_XACC*n*)

The XACC registers define which program flash segments are restricted to data read or execute only or both data and instruction fetches.

The eight XACC registers allow up to 64 restricted segments of equal memory size.

Execute-only access register	Program flash execute-only access bits
XACCH0	XA[63:56]
XACCH1	XA[55:48]
XACCH2	XA[47:40]
XACCH3	XA[39:32]
XACCL0	XA[31:24]
XACCL1	XA[23:16]
XACCL2	XA[15:8]
XACCL3	XA[7:0]

During the reset sequence, the XACC registers are loaded with the logical AND of Program Flash IFR addresses A and B as indicated in the following table.

Execute-only access register	Program Flash IFR address A	Program Flash IFR address B
XACCH0	0x03A3	0x03AB
XACCH1	0x03A2	0x03AA
XACCH2	0x03A1	0x03A9
XACCH3	0x03A0	0x03A8
XACCL0	0x03A7	0x03AF
XACCL1	0x03A6	0x03AE
XACCL2	0x03A5	0x03AD
XACCL3	0x03A4	0x03AC

Use the Program Once command to program the execute-only access control fields that are loaded during the reset sequence.

Address: 4002_0000h base + 18h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read	XA							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_XACC*n* field descriptions

Field	Description
XA	Execute-only access control 0 Associated segment is accessible in execute mode only (as an instruction fetch) 1 Associated segment is accessible as data or in execute mode

17.4.4.10 Supervisor-only Access Registers (FTFE_SACC*n*)

The SACC registers define which program flash segments are restricted to supervisor only or user and supervisor access.

The eight SACC registers allow up to 64 restricted segments of equal memory size.

Supervisor-only access register	Program flash supervisor-only access bits
SACCH0	SA[63:56]
SACCH1	SA[55:48]
SACCH2	SA[47:40]
SACCH3	SA[39:32]
SACCL0	SA[31:24]
SACCL1	SA[23:16]
SACCL2	SA[15:8]
SACCL3	SA[7:0]

During the reset sequence, the SACC registers are loaded with the logical AND of Program Flash IFR addresses A and B as indicated in the following table.

Supervisor-only access register	Program Flash IFR address A	Program Flash IFR address B
SACCH0	0x03B3	0x03BB
SACCH1	0x03B2	0x03BA
SACCH2	0x03B1	0x03B9
SACCH3	0x03B0	0x03B8
SACCL0	0x03B7	0x03BF
SACCL1	0x03B6	0x03BE
SACCL2	0x03B5	0x03BD
SACCL3	0x03B4	0x03BC

Use the Program Once command to program the supervisor-only access control fields that are loaded during the reset sequence.

Address: 4002_0000h base + 20h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read	SA							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_SACC*n* field descriptions

Field	Description
SA	Supervisor-only access control 0 Associated segment is accessible in supervisor mode only 1 Associated segment is accessible in user or supervisor mode

17.4.4.11 Flash Access Segment Size Register (FTFE_FACSS)

The flash access segment size register determines which bits in the address are used to index into the SACC and XACC bitmaps to get the appropriate permission flags.

All bits in the register are read-only.

The contents of this register are loaded during the reset sequence.

Address: 4002_0000h base + 28h offset = 4002_0028h

Bit	7	6	5	4	3	2	1	0
Read	SGSIZE							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FACSS field descriptions

Field	Description
SGSIZE	Segment Size The segment size is a fixed value based on the available program flash size divided by NUMSG. Flash Size / Segment Size / Segment Size Encoding 256 KBytes / 4 KBytes / 0x4 512 KBytes / 8 KBytes / 0x5 768 KBytes / 16 KBytes / 0x6

FTFE_FACSS field descriptions (continued)

Field	Description		
	1 MByte	16 KBytes	0x6
	1.5 MBytes	32 KBytes	0x7
	2 MBytes	32 KBytes	0x7

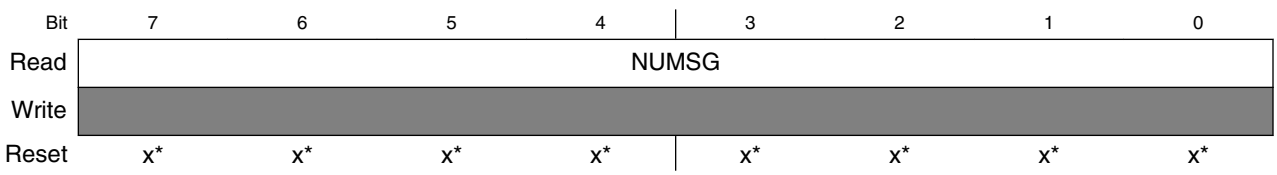
17.4.4.12 Flash Access Segment Number Register (FTFE_FACSN)

The flash access segment number register provides the number of program flash segments that are available for XACC and SACC permissions.

All bits in the register are read-only.

The contents of this register are loaded during the reset sequence.

Address: 4002_0000h base + 2Bh offset = 4002_002Bh



* Notes:
- x = Undefined at reset.

FTFE_FACSN field descriptions

Field	Description
NUMSG	Number of Segments Indicator The NUMSG field indicates the number of equal-sized segments in the program flash. 0x20 32 segments 0x24 36 segments 0x28 40 segments 0x2C 44 segments 0x30 48 segments 0x38 56 segments 0x3C 60 segments 0x40 64 segments

17.4.4.13 Flash Error Status Register (FTFE_FERSTAT)

This register reports the detection of uncorrected ECC errors during read access to the FTFE module.

The DFDIF flag is readable and writable. The unassigned bits read 0 and are not writable.

Address: 4002_0000h base + 2Eh offset = 4002_002Eh

Bit	7	6	5	4	3	2	1	0
Read	0						DFDIF	0
Write							w1c	
Reset	0	0	0	0	0	0	0	0

FTFE_FERSTAT field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DFDIF	Double Bit Fault Detect Interrupt Flag The DFDIF flag indicates an uncorrectable ECC fault was detected during a valid flash read access from the platform flash controller. The DFDIF flag is cleared by writing a 1 to it. Writing a 0 to DFDIF has no effect. 0 Double bit fault not detected during a valid flash read access from the platform flash controller 1 Double bit fault detected (or FERCNFG[FDFD] is set) during a valid flash read access from the platform flash controller
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

17.4.4.14 Flash Error Configuration Register (FTFE_FERCNFG)

This register enables the force and interrupt of uncorrected ECC errors detected during read access to the FTFE module.

The FDFD and DFDIE bits are readable and writable. The unassigned bits read 0 and are not writable.

Address: 4002_0000h base + 2Fh offset = 4002_002Fh

Bit	7	6	5	4	3	2	1	0
Read	0		FDFD	0			DFDIE	0
Write								
Reset	0	0	0	0	0	0	0	0

FTFE_FERCNFG field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 FDFD	Force Double Bit Fault Detect The FDFD bit enables the user to emulate the setting of the FERSTAT[DFDIF] flag to check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. 0 FERSTAT[DFDIF] sets only if a double bit fault is detected during read access from the platform flash controller 1 FERSTAT[DFDIF] sets during any valid flash read access from the platform flash controller. An interrupt request is generated if the DFDIE bit is set.
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DFDIE	Double Bit Fault Detect Interrupt Enable The DFDIE bit controls interrupt generation when an uncorrectable ECC fault is detected during a valid flash read access from the platform flash controller. 0 Double bit fault detect interrupt disabled 1 Double bit fault detect interrupt enabled. An interrupt request is generated whenever the FERSTAT[DFDIF] flag is set.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

17.5 Functional Description

The following sections describe functional details of the FTFE module.

17.5.1 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* — Four registers protect 32 regions of the program flash memory as shown in the following figure

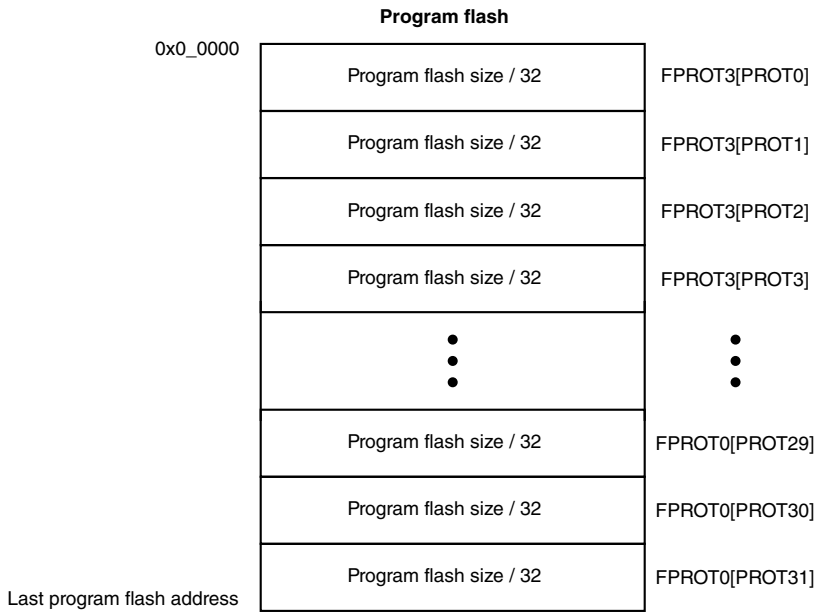


Figure 17-2. Program flash protection

- FDPROT —
  - For 2^n data flash sizes, protects eight regions of the data flash memory as shown in the following figure

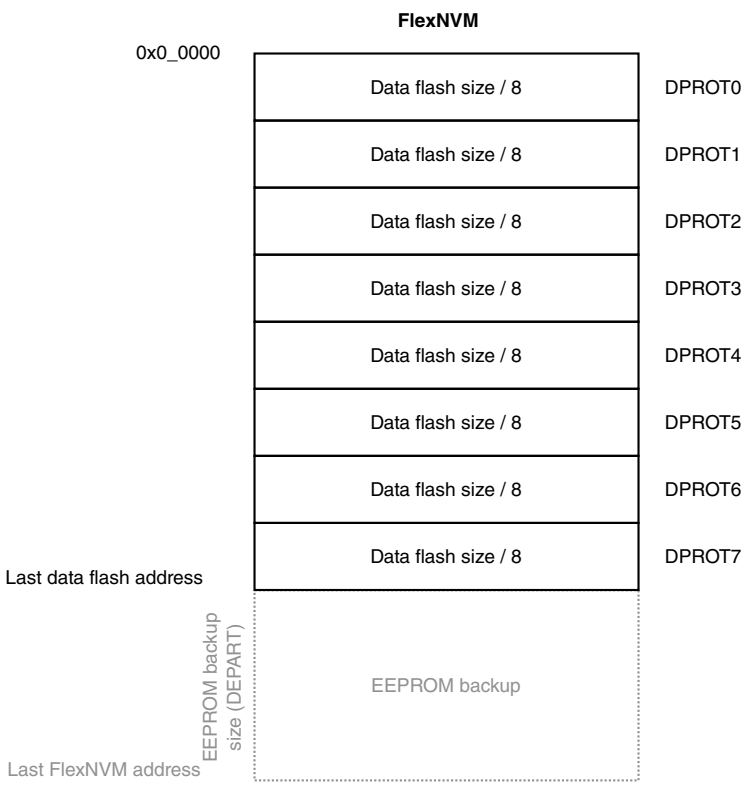


Figure 17-3. Data flash protection (2^n data flash sizes)

- FEPROT — Protects eight regions of the EEPROM memory as shown in the following figure

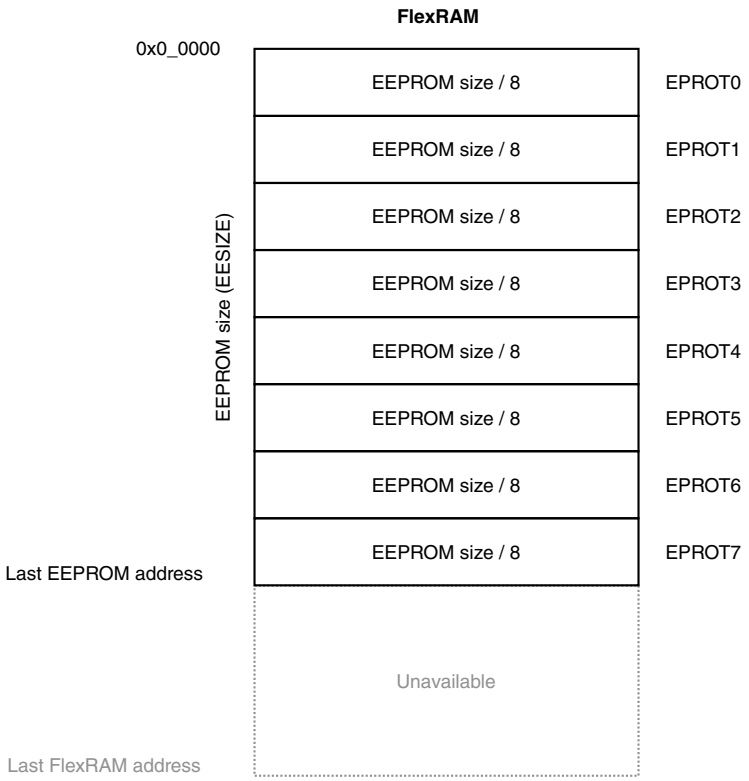


Figure 17-4. EEPROM protection

NOTE

Flash protection features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Some features described in the application note may not be available on this device.

17.5.2 Flash Access Protection

Individual segments within the program flash memory can be designated for restricted access. Specific flash commands (Program Check, Program Phrase, Erase Flash Block, Erase Flash Sector) monitor FXACC contents to protect program flash memory but the FSACC contents do not impact flash command operation.

See AN5112: Using the Kinetis Flash Execute-Only Access Control Feature for further details.

Access is controlled by the following registers:

- FXACC —
  - eight registers control 64 segments of the program flash memory as shown in the following figure

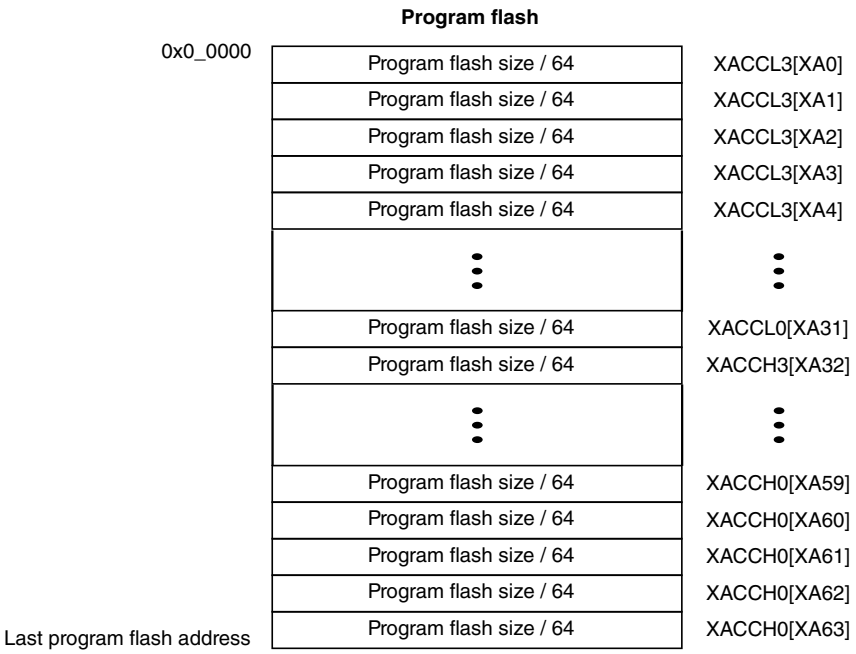


Figure 17-5. Program flash execute-only access control

- FSACC —
  - eight registers control 64 segments of the program flash memory as shown in the following figure

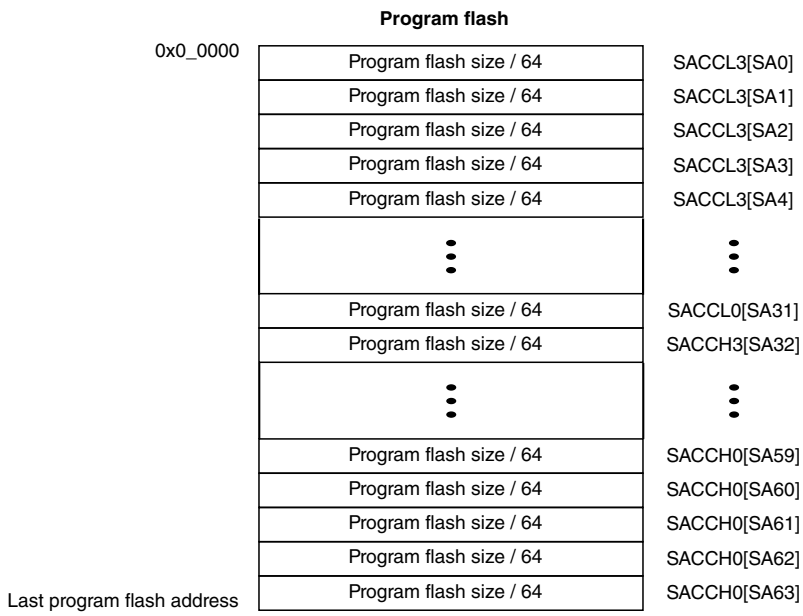


Figure 17-6. Program flash supervisor access control

17.5.3 FlexNVM Description

This section describes the FlexNVM memory.

17.5.3.1 FlexNVM Block Partitioning for FlexRAM

The user can configure the FlexNVM block as either:

- Basic data flash,
- EEPROM flash records to support the built-in EEPROM feature, or
- A combination of both.

The user's FlexNVM configuration choice is specified using the Program Partition command described in Program Partition command.

CAUTION

While different partitions of the FlexNVM block are available, the intention is that a single partition choice is used throughout the entire lifetime of a given application. The FlexNVM partition code choices affect the endurance and data retention characteristics of the device.

17.5.3.2 EEPROM User Perspective

The EEPROM system is shown in the following figure.

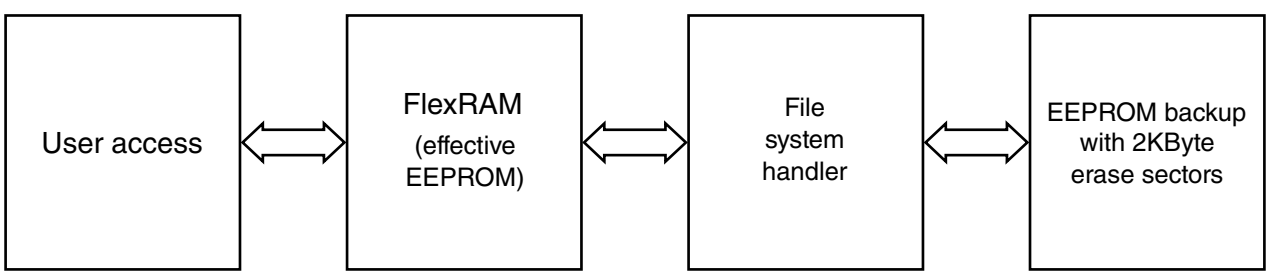


Figure 17-7. Top Level EEPROM Architecture

To handle varying customer requirements, the FlexRAM and FlexNVM blocks can be split into partitions as shown in the figure below.

1. **EEPROM partition** (EEESIZE) — The amount of FlexRAM used for EEPROM can be set from 0 Bytes (no EEPROM) to the maximum FlexRAM size (see Table 17-2). The remainder of the FlexRAM not used for EEPROM is not accessible while the FlexRAM is configured for EEPROM (see Set FlexRAM Function command). The EEPROM partition grows upward from the bottom of the FlexRAM address space.

2. **Data flash partition** (DEPART) — The amount of FlexNVM memory used for data flash can be programmed from 0 bytes (all of the FlexNVM block is available for EEPROM backup) to the maximum size of the FlexNVM block (see Table 17-4).
3. **FlexNVM EEPROM partition** — The amount of FlexNVM memory used for EEPROM backup, which is equal to the FlexNVM block size minus the data flash memory partition size. The EEPROM backup size must be at least 16 times the EEPROM partition size in FlexRAM.

The partition information (EEESIZE, DEPART) is stored in the data flash IFR and is programmed using the Program Partition command (see Program Partition command). Typically, the Program Partition command is executed only once in the lifetime of the device.

Data flash memory is useful for applications that need to quickly store large amounts of data or store data that is static. The EEPROM partition in FlexRAM is useful for storing smaller amounts of data that will be changed often.

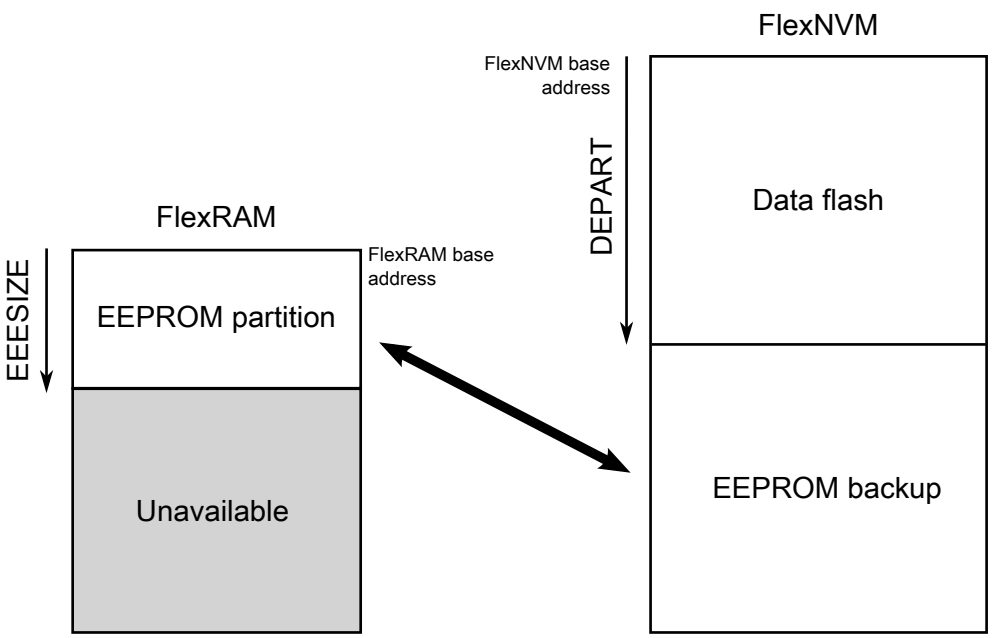


Figure 17-8. FlexRAM to FlexNVM Memory Mapping for EEPROM

17.5.3.3 EEPROM implementation overview

Out of reset with the FSTAT[CCIF] bit clear, the partition settings (EEESIZE, DEPART) are read from the data flash IFR and the EEPROM file system is initialized accordingly. The EEPROM file system locates all valid EEPROM data records in EEPROM backup and copies the newest data to FlexRAM. The FSTAT[CCIF] and FCNFG[EEERDY] bits are set after data from all valid EEPROM data records is copied to the FlexRAM. After the CCIF bit is set, the FlexRAM is available for read or write access.

When configured for EEPROM use, writes to an unprotected location in FlexRAM invokes the EEPROM file system to program a new EEPROM data record in the EEPROM backup memory in a round-robin fashion. As needed, the EEPROM file system identifies the EEPROM backup sector that is being erased for future use and partially erases that EEPROM backup sector. After a write to the FlexRAM, the FlexRAM is not accessible until the FSTAT[CCIF] bit is set. The FCNFG[EEERDY] bit will also be set. If enabled, the interrupt associated with the FSTAT[CCIF] bit can be used to determine when the FlexRAM is available for read or write access.

When configured for EEPROM use, attempts to write to the FlexRAM are ignored in VLP mode. Attempts to write to the FlexRAM in HSRUN mode will be trapped with the ACCERR flag being set.

After a sector in EEPROM backup is full of EEPROM data records, EEPROM data records from the sector holding the oldest data are gradually copied over to a previously-erased EEPROM backup sector. When the sector copy completes, the EEPROM backup sector holding the oldest data is tagged for erase.

17.5.3.4 Write endurance to FlexRAM for EEPROM

When the FlexNVM partition code is not set to full data flash, the EEPROM data set size can be set to any of several non-zero values.

The bytes not assigned to data flash via the FlexNVM partition code are used by the FTFE to obtain an effective endurance increase for the EEPROM data. The built-in EEPROM record management system raises the number of program/erase cycles that can be attained prior to device wear-out by cycling the EEPROM data through a larger EEPROM NVM storage space.

While different partitions of the FlexNVM are available, the intention is that a single choice for the FlexNVM partition code and EEPROM data set size is used throughout the entire lifetime of a given application. The EEPROM endurance equation and graph shown below assume that only one configuration is ever used.

$$\text{Writes\_FlexRAM} = \frac{\text{EEPROM} - 2 \times \text{EEESIZE}}{\text{EEESIZE}} \times \text{Write\_efficiency} \times n_{\text{nvmcycee}}$$

where

- Writes_FlexRAM — minimum number of writes to each FlexRAM location
- EEPROM — allocated FlexNVM based on DEPART; entered with the Program Partition command
- EEESIZE — allocated FlexRAM based on DEPART; entered with the Program Partition command

- Write_efficiency —
  - 0.25 for 8-bit writes to FlexRAM
  - 0.50 for 16-bit or 32-bit writes to FlexRAM
- $n_{nvmcycee}$ — EEPROM-backup cycling endurance

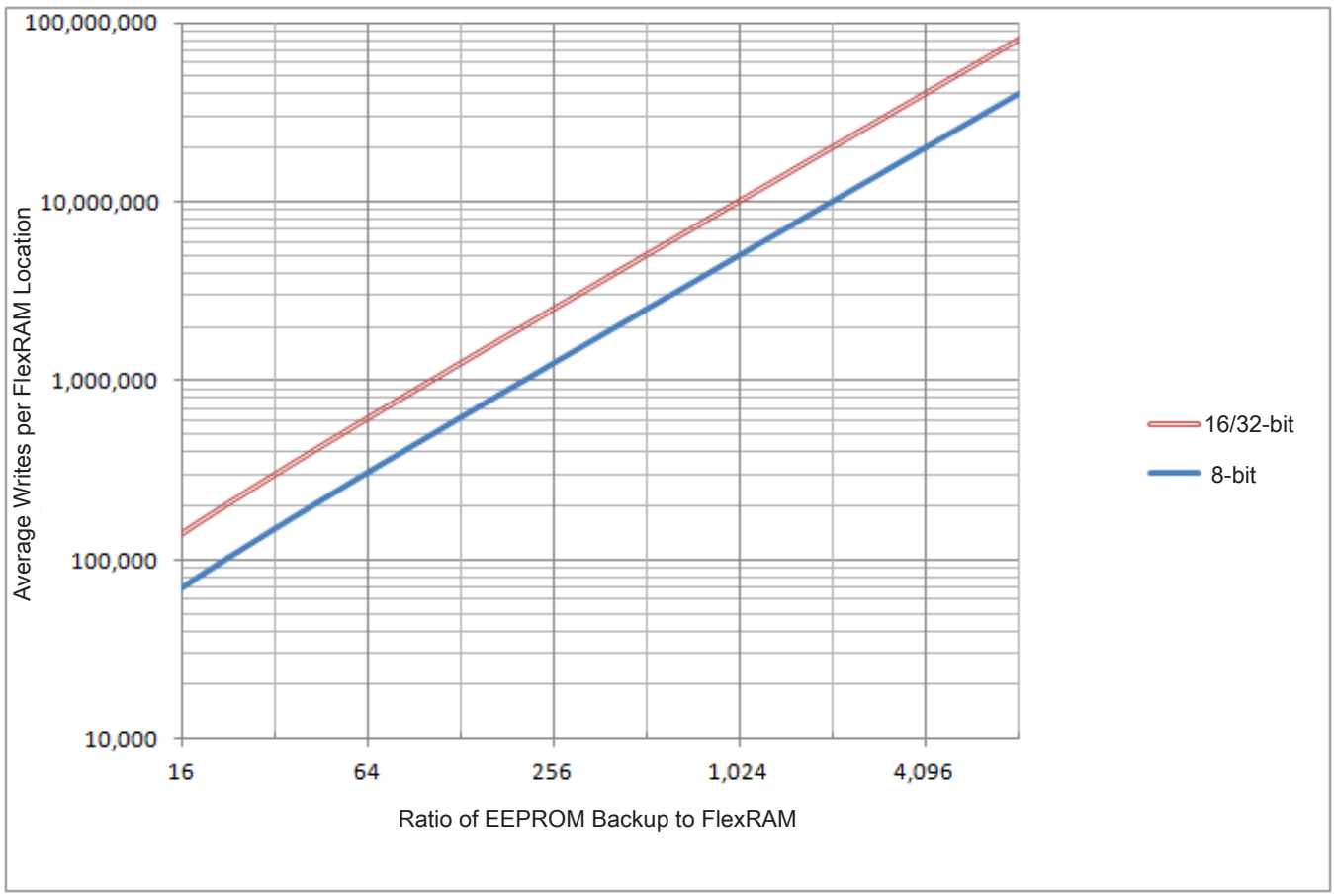


Figure 17-9. EEPROM backup writes to FlexRAM

17.5.4 Interrupts

The FTFE module can generate interrupt requests to the MCU upon the occurrence of various FTFE events. These interrupt events and their associated status and control bits are shown in the following table.

Table 17-5. FTFE Interrupt Sources

FTFE Event	Readable Status Bit	Interrupt Enable Bit
FTFE Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
FTFE Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]

Table continues on the next page...

Table 17-5. FTFE Interrupt Sources (continued)

FTFE Event	Readable Status Bit	Interrupt Enable Bit
FTFE ECC Error Detection	FERSTAT[DFDIF]	FERCNFG[DFDIE]

Note

Vector addresses and their relative interrupt priority are determined at the MCU level.

17.5.5 Flash Operation in Low-Power Modes

17.5.5.1 Wait Mode

When the MCU enters wait mode, the FTFE module is not affected. The FTFE module can recover the MCU from wait via the command complete interrupt (see Interrupts).

17.5.5.2 Stop Mode

When the MCU requests stop mode, if an FTFE command is active (CCIF = 0) the command execution completes before the MCU is allowed to enter stop mode.

CAUTION

The MCU should never enter stop mode while any FTFE command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the FTFE module does not accept flash commands.

17.5.6 Flash memory reads and ignored writes

The FTFE module requires only the flash address to execute a flash memory read. MCU read access is available to all flash memory.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

17.5.7 Read while write (RWW)

The following simultaneous accesses are allowed:

- The user may read from the program flash memory while commands (typically program and erase operations) are active in the data flash and FlexRAM memory space.
- The MCU can fetch instructions from program flash during both data flash program and erase operations and while EEPROM-backup is maintained by the EEPROM commands.
- Conversely, the user may read from data flash and FlexRAM while program and erase commands are executing on the program flash.
- When configured as traditional RAM, writes to the FlexRAM are allowed during data flash operations.

Simultaneous data flash operations and FlexRAM writes, when FlexRAM is used as EEPROM, are not possible.

Simultaneous operations are further discussed in Allowed simultaneous flash operations.

17.5.8 Flash Program and Erase

All flash functions except read require the user to setup and launch an FTFE command through a series of peripheral bus writes. The user cannot initiate any further FTFE commands until notified that the current command has completed. The FTFE command structure and operation are detailed in FTFE Command Operations.

17.5.9 FTFE Command Operations

FTFE command operations are typically used to modify flash memory contents. The next sections describe:

- The command write sequence used to set FTFE command parameters and launch execution
- A description of all FTFE commands available

17.5.9.1 Command Write Sequence

FTFE commands are specified using a command write sequence illustrated in Figure 17-10. The FTFE module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

Attempts to launch an FTFE command in VLP mode will be ignored. Attempts to launch an FTFE command in HSRUN mode will be trapped with the ACCERR flag being set.

17.5.9.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired FTFE command. The individual registers that make up the FCCOB data set can be written in any order.

17.5.9.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing the FSTAT[CCIF] bit by writing a '1' to it. The CCIF flag remains zero until the FTFE command completes.

The FSTAT register contains a blocking mechanism, which prevents a new command from launching (can't clear CCIF) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

17.5.9.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The FTFE reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

   If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. ACCERR reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

   Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, the FSTAT[FPVIOL] (protection error) flag is set.

   Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting the FSTAT[CCIF] bit.

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in the FSTAT[MGSTAT0] bit. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicable, are reported back to the user via the FCCOB and FSTAT registers.
4. The FTFE sets the FSTAT[CCIF] bit signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

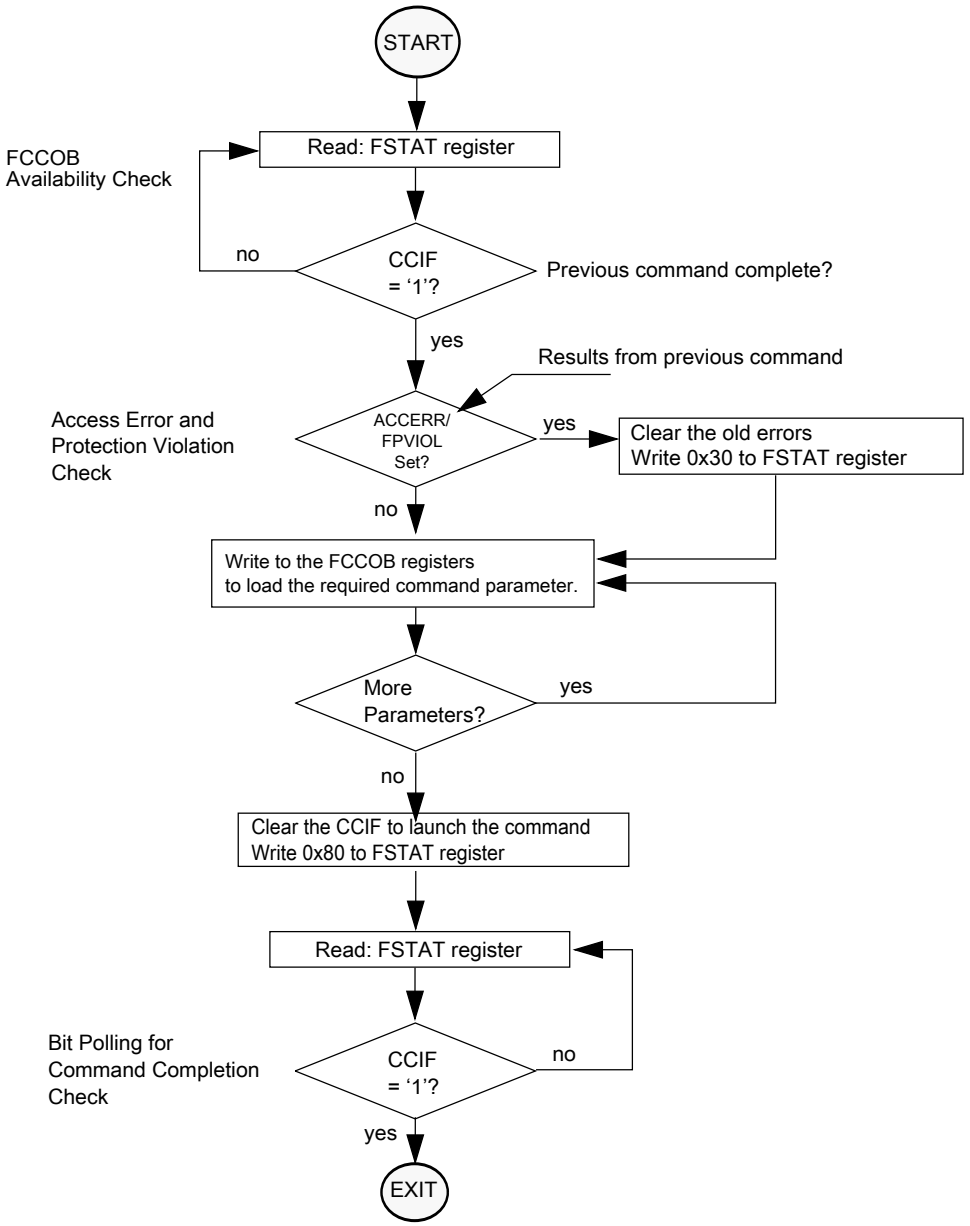


Figure 17-10. Generic Flash Command Write Sequence Flowchart

17.5.9.2 Flash commands

The following table summarizes the function of all flash commands. If any column is marked with an 'X', the flash command is relevant to that particular memory resource.

FCMD	Command	Program flash 0	Data flash	FlexRAM	Function
0x00	Read 1s Block	×	×		Verify that a program flash or data flash block is erased. FlexNVM

Table continues on the next page...

FCMD	Command	Program flash 0	Data flash	FlexRAM	Function
					block must not be partitioned for EEPROM.
0x01	Read 1s Section	×	×		Verify that a given number of program flash or data flash locations from a starting address are erased.
0x02	Program Check	×	×		Tests previously-programmed phrases at margin read levels.
0x03	Read Resource	IFR,ID	IFR		Read 8 bytes from program flash IFR, data flash IFR, or version ID.
0x07	Program Phrase	×	×		Program 8 bytes in a program flash block or a data flash block.
0x08	Erase Flash Block	×	×		Erase a program flash block or data flash block. An erase of any flash block is only possible when unprotected. FlexNVM block must not be partitioned for EEPROM.
0x09	Erase Flash Sector	×	×		Erase all bytes in a program flash or data flash sector.
0x0B	Program Section	×	×	×	Program data from the Section Program Buffer to a program flash or data flash block.
0x40	Read 1s All Blocks	×	×		Verify that all program flash, data flash blocks, EEPROM backup data records, and data flash IFR are erased then release MCU security.
0x41	Read Once	IFR			Read 8 bytes of an indexed field in the program flash 0 IFR.

Table continues on the next page...

FCMD	Command	Program flash 0	Data flash	FlexRAM	Function
0x43	Program Once	IFR			One-time program of 8 bytes of an indexed field in the program flash 0 IFR.
0x44	Erase All Blocks	×	×	×	Erase all program flash blocks, data flash blocks, FlexRAM, EEPROM backup data records, and data flash IFR. Then, verify-erase and release MCU security. **NOTE**: An erase is only possible when all memory locations are unprotected.
0x45	Verify Backdoor Access Key	×			Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.
0x49	Erase All Blocks Unsecure	×	×	×	Erase all program flash blocks, data flash blocks, FlexRAM, EEPROM backup data records, and data flash IFR. Then, verify-erase, program the security byte to the unsecure state, and release MCU security.
0x4A	Read 1s All Execute-only Segments	x			Verify that all program flash execute-only (XA) segments are erased then release flash access control.
0x4B	Erase All Execute-only Segments	x			Erase all program flash execute-only (XA) segments then release flash access control.

Table continues on the next page...

FCMD	Command	Program flash 0	Data flash	FlexRAM	Function
0x80	Program Partition		IFR, ×	×	Program the FlexNVM Partition Code and EEPROM Data Set Size into the data flash IFR. format all EEPROM backup data sectors allocated for EEPROM, initialize the FlexRAM.
0x81	Set FlexRAM Function		×	×	Switches FlexRAM function between RAM and EEPROM. When switching to EEPROM, FlexNVM is not available while valid data records are being copied from EEPROM backup to FlexRAM.

17.5.9.3 Allowed simultaneous flash operations

Only the operations marked 'OK' in the following table are permitted to run simultaneously on the program flash, data flash, and FlexRAM memories. Some operations cannot be executed simultaneously because certain hardware resources are shared by the memories. The priority has been placed on permitting program flash reads while program and erase operations execute on the FlexNVM and FlexRAM. This provides read (program flash) while write (FlexNVM, FlexRAM) functionality.

Table 17-6. Allowed Simultaneous Memory Operations

		Program flash			Data flash			FlexRAM		
		Read	Program Phrase	Erase Flash Sector[1]	Read	Program Phrase	Erase Flash Sector[1]	Read	E-Write[2]	R-Write[3]
Program flash	Read					OK	OK		OK	
	Program Phrase				OK			OK		OK
	Erase Flash Sector[1]				OK			OK		OK

Table continues on the next page...

Table 17-6. Allowed Simultaneous Memory Operations (continued)

		Program flash			Data flash			FlexRAM		
		Read	Program Phrase	Erase Flash Sector[1]	Read	Program Phrase	Erase Flash Sector[1]	Read	E-Write[2]	R-Write[3]
Data flash	Read		OK	OK						
	Program Phrase	OK						OK		OK
	Erase Flash Sector[1]	OK						OK		OK
FlexRAM	Read		OK	OK		OK	OK			
	E-Write[2]	OK								
	R-Write[3]		OK	OK		OK	OK			

1. Also applies to Erase Flash Block
2. When FlexRAM configured for EEPROM (EEERDY=1).
3. When FlexRAM configured as traditional RAM (RAMRDY=1); single cycle operation.

17.5.10 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Block, Read 1s Section, Read 1s All Execute-only Segments) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash and data flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. All simple (uncommanded) flash array reads to the MCU always use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

17.5.11 Flash command descriptions

This section describes all flash commands that can be launched by a command write sequence. The FTFE sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that the ACCERR and FPVIOL bits in the FSTAT register are cleared prior to starting the command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the FTFE is running a command (CCIF = 0) on that same block. The FTFE may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

When required by the command, address bit 23 selects between program flash memory (=0) and data flash memory (=1).

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

17.5.11.1 Read 1s Block command

The Read 1s Block command checks to see if an entire program flash or data flash block has been erased to the specified margin level. The FCCOB flash address bits determine which block is erase-verified.

Table 17-7. Read 1s Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x00 (RD1BLK)
1	Flash address [23:16] in the flash block to be verified
2	Flash address [15:8] in the flash block to be verified
3	Flash address [7:0][1] in the flash block to be verified
4	Read-1 Margin Choice

1. Must be 128-bit aligned (Flash address [3:0] = 0000) for program flash, 64-bit aligned (Flash address [2:0] = 000) for data flash.

After clearing CCIF to launch the Read 1s Block command, the FTFE sets the read margin for 1s according to Table 17-8 and then reads all locations within the selected program flash or data flash block.

When the data flash is targeted, DEPART must be set for no EEPROM, else the Read 1s Block command aborts setting the FSTAT[ACCERR] bit. If the FTFE fails to read all 1s (i.e. the flash block is not fully erased), the FSTAT[MGSTAT0] bit is set. The CCIF flag sets after the Read 1s Block operation has completed.

Table 17-8. 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 17-9. Read 1s Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]

Table continues on the next page...

Table 17-9. Read 1s Block Command Error Handling (continued)

Error Condition	Error Bit
An invalid margin choice is specified	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Data flash is selected and the address is out of data flash range	FSTAT[ACCERR]
Data flash is selected with EEPROM enabled	FSTAT[ACCERR]
Flash address is not 128-bit aligned for program flash, 64-bit aligned for data flash	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

17.5.11.2 Read 1s Section command

The Read 1s Section command checks if a section of program flash or data flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of double-phrases to be verified for program flash, phrases for data flash.

Table 17-10. Read 1s Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x01 (RD1SEC)
1	Flash address [23:16] of the first double-phrase to be verified for program flash, phrase for data flash
2	Flash address [15:8] of the first double-phrase to be verified for program flash, phrase for data flash
3	Flash address [7:0][1] of the first double-phrase to be verified for program flash, phrase for data flash
4	Number of double-phrases to be verified for program flash, phrases for data flash [15:8]
5	Number of double-phrases to be verified for program flash, phrases for data flash [7:0]
6	Read-1 Margin Choice

1. Must be 128-bit aligned (Flash address [3:0] = 0000) for program flash, 64-bit aligned (Flash address [2:0] = 000) for data flash.

Upon clearing CCIF to launch the Read 1s Section command, the FTFE sets the read margin for 1s according to Table 17-11 and then reads all locations within the specified section of flash memory.

If the FTFE fails to read all 1s (i.e. the flash section is not erased), the FSTAT(MGSTAT0) bit is set. The CCIF flag sets after the Read 1s Section operation completes.

Table 17-11. 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 17-12. Read 1s Section Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin code is supplied	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned for program flash, 64-bit aligned for data flash	FSTAT[ACCERR]
The requested section crosses a flash block boundary	FSTAT[ACCERR]
The requested number of double-phrases for program flash, phrases for data flash is zero	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

17.5.11.3 Program Check command

The Program Check command tests a previously programmed program flash or data flash longword to see if it reads correctly at the specified margin level.

Table 17-13. Program Check Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x02 (PGMCHK)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Margin Choice
8	Byte 0 expected data
9	Byte 1 expected data
A	Byte 2 expected data
B	Byte 3 expected data

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the FTFE sets the read margin for 1s based on the provided margin choice according to Table 17-14. The Program Check operation then reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, the MGSTAT0 bit is set.

The FTFE will then set the read margin for 0s based on the provided margin choice.The Program Check operation will then read the specified longword and compare the actual read data to the expected data provided by the FCCOB. If the comparison at margin-0 fails, the MGSTAT0 bit will be set. The CCIF flag will set after the Program Check operation has completed.

The starting address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 0 data is expected at the supplied 32-bit aligned address,
- Byte 1 data is expected at byte address specified + 0b01,
- Byte 2 data is expected at byte address specified + 0b10, and
- Byte 3 data is expected at byte address specified + 0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 17-14. 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 17-15. Program Check Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Flash address is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

17.5.11.4 Read Resource command

The Read Resource command is provided for the user to read data from special-purpose memory resources located within the Flash module. The special-purpose memory resources available include program flash IFR, data flash IFR space, and the Version ID field. The Version ID field contains an 8 byte code that indicates a specific FTFE implementation.

Table 17-16. Read Resource Command FCCOB Requirements

FCCOB Number	FCCOB contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Resource select code (see Table 17-17)
Returned values	
4	Read Data [64:56]
5	Read Data [55:48]
6	Read Data [47:40]
7	Read Data [39:32]
8	Read Data [31:24]
9	Read Data [23:16]
A	Read Data [15:8]
B	Read Data [7:0]

1. Must be 64-bit aligned (Flash address [2:0] = 000).

Table 17-17. Read Resource Select Codes

Resource Select Code	Description	Resource Size	Local Address Range
0x00	Program Flash 0 IFR	1024 Bytes	0x00_0000 - 0x00_03FF
0x00	Data Flash 0 IFR	1024 Bytes	0x80_0000 - 0x80_03FF
0x01	Version ID	8 Bytes	0x00_0008 - 0x00_000F

After clearing CCIF to launch the Read Resource command, eight consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag will set after the Read Resource operation has completed. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 17-18. Read Resource Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid resource code is entered	FSTAT[ACCERR]
Flash address is out-of-range for the targeted resource.	FSTAT[ACCERR]
Flash address is not 64-bit aligned	FSTAT[ACCERR]

17.5.11.5 Program Phrase command

The Program Phrase command programs eight previously-erased bytes in the program flash memory or in the data flash memory using an embedded algorithm.

CAUTION

A Flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a Flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 17-19. Program Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x07 (PGM8)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Byte 0 program value
5	Byte 1 program value
6	Byte 2 program value
7	Byte 3 program value
8	Byte 4 program value
9	Byte 5 program value
A	Byte 6 program value
B	Byte 7 program value

1. Must be 64-bit aligned (Flash address [2:0] = 000)

Upon clearing CCIF to launch the Program Phrase command, the FTFE programs the data bytes into the flash using the supplied address. The protection status is always checked. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Phrase operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in MGSTAT0. The CCIF flag is set after the Program Phrase operation completes.

The starting address must be 64-bit aligned (flash address [2:0] = 000):

- Byte 0 data is written to the starting address ('start'),
- Byte 1 data is programmed to byte address start+0b01,

- Byte 2 data is programmed to byte address start+0b10, and
- Byte 3 data is programmed to byte address start+0b11, etc.

Table 17-20. Program Phrase Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 64-bit aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Flash address is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation.	FSTAT[MGSTAT0]

17.5.11.6 Erase Flash Block command

The Erase Flash Block operation erases all addresses in a single program flash or data flash block.

Table 17-21. Erase Flash Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x08 (ERSBLK)
1	Flash address [23:16] in the flash block to be erased
2	Flash address [15:8] in the flash block to be erased
3	Flash address [7:0][1] in the flash block to be erased

1. Must be 128-bit aligned (Flash address [3:0] = 0000) for program flash, 64-bit aligned (Flash address [2:0] = 000) for data flash.

Upon clearing CCIF to launch the Erase Flash Block command, the FTFE erases the main array of the selected flash block and verifies that it is erased. When the data flash is targeted, DEPART must be set for no EEPROM (see Table 17-4) else the Erase Flash Block command aborts setting the FSTAT[ACCERR] bit. The Erase Flash Block command aborts and sets the FSTAT[FPVIOL] bit if any region within the block is protected (see the description of the program flash protection (FPROT) registers and the data flash protection (FDPROT) registers). If the erase verify fails, the MGSTAT0 bit in FSTAT is set. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 17-22. Erase Flash Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Data flash is selected and the address is out of data flash range	FSTAT[ACCERR]
Data flash is selected with EEPROM enabled	FSTAT[ACCERR]
Flash address is not 128-bit aligned for program flash, 64-bit aligned for data flash	FSTAT[ACCERR]
Any area of the selected flash block is protected	FSTAT[FPVIOL]
The selected program flash block contains an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	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.

17.5.11.7 Erase Flash Sector command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 17-23. Erase Flash Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x09 (ERSSCR)
1	Flash address [23:16] in the flash sector to be erased
2	Flash address [15:8] in the flash sector to be erased
3	Flash address [7:0][1] in the flash sector to be erased

1. Must be 128-bit aligned (Flash address [3:0] = 0000) for program flash, 64-bit aligned (Flash address [2:0] = 000) for data flash.

After clearing CCIF to launch the Erase Flash Sector command, the FTFE erases the selected program flash or data flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description of the FPROT registers). 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 17-11).

Table 17-24. Erase Flash Sector Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid Flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned for program flash, 64-bit aligned for data flash	FSTAT[ACCERR]
The selected program flash or data flash sector is protected	FSTAT[FPVIOL]

Table continues on the next page...

Table 17-24. Erase Flash Sector Command Error Handling (continued)

Error Condition	Error Bit
The selected program flash sector is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	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.

17.5.11.7.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit when CCIF, ACCERR, and FPVIOL are clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see Erase Flash Sector command), the flash samples the state of the ERSSUSP bit at convenient points. If the FTFE detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the FTFE sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the FTFE detects that a suspend request has been made, the FTFE clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the FTFE sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the FTFE has acknowledged it.

17.5.11.7.2 Resuming a Suspended Erase Flash Sector Operation

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The FTFE acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit 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.

17.5.11.7.3 Aborting a Suspended Erase Flash Sector Operation

The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the FTFE starts the new command using the new FCCOB contents.

While FCNFG[ERSSUSP] is set, a write to the FlexRAM while FCNFG[EEERDY] is set clears ERSSUSP and aborts the suspended operation. The FlexRAM write operation is executed by the FTFE.

Note

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

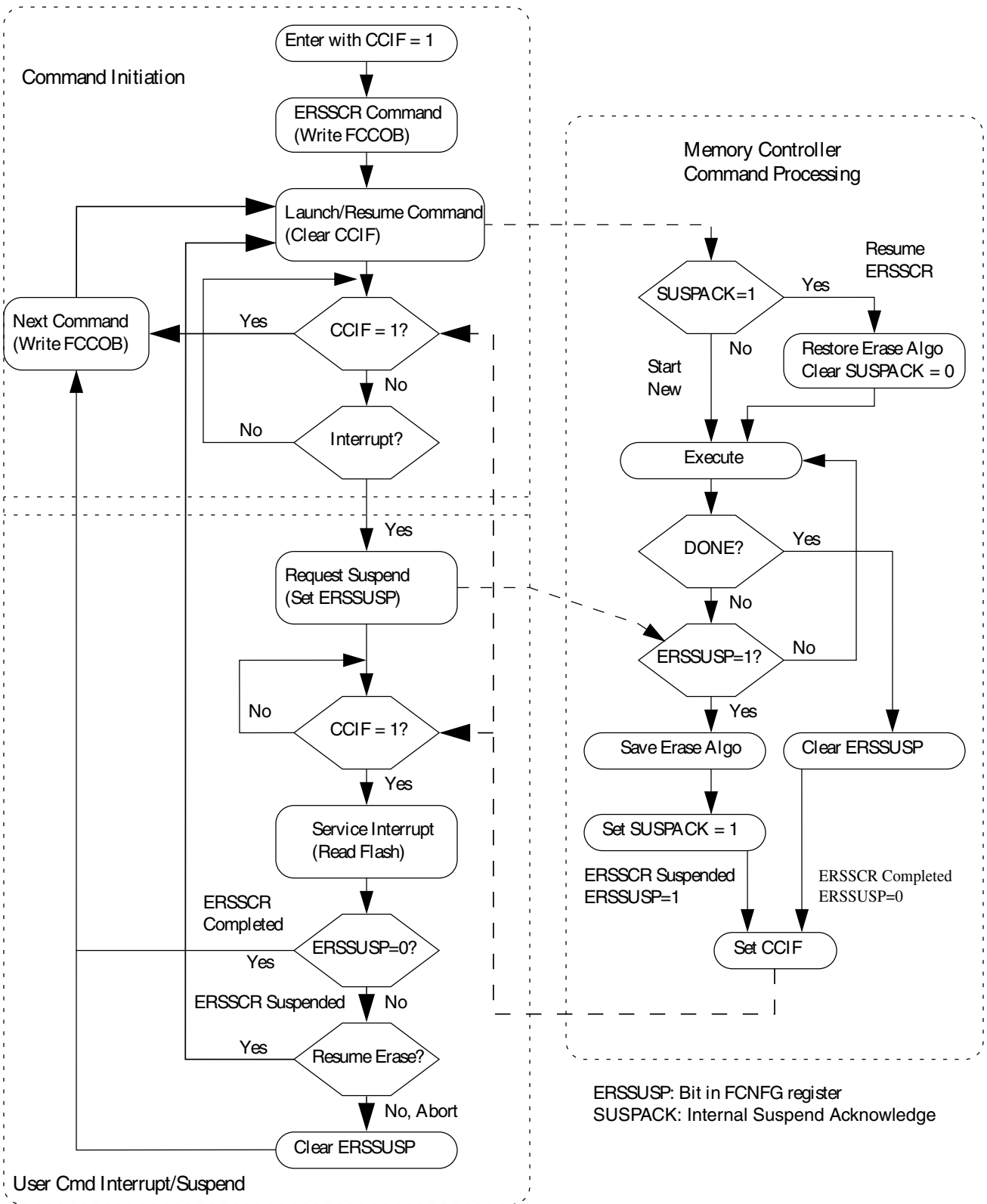


Figure 17-11. Suspend and Resume of Erase Flash Sector Operation

17.5.11.8 Program Section command

The Program Section operation programs the data found in the section program buffer to previously erased locations in the flash memory using an embedded algorithm. Data is preloaded into the section program buffer by writing to the FlexRAM while it is set to function as a programming acceleration RAM (see Flash sector programming).

The section program buffer is limited to the lower quarter of the programming acceleration RAM (relative byte addresses 0x0000-0x03FF - be sure to check your device specific memory map for the location of the programming acceleration RAM or FlexRAM). Data written to the remainder of the programming acceleration RAM is ignored and may be overwritten during Program Section command execution.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 17-25. Program Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x0B (PGMSEC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Number of double-phrases for program flash, phrases for data flash to program [15:8]
5	Number of double-phrases for program flash, phrases for data flash to program [7:0]

1. Must be 128-bit aligned (Flash address [3:0] = 0000) for program flash, 64-bit aligned (Flash address [2:0] = 000) for data flash.

After clearing CCIF to launch the Program Section command, the FTFE will block access to the FlexRAM and program the data residing in the Section Program Buffer into the flash memory starting at the flash address provided.

The starting address must be unprotected (see the description of the FPROT registers) to permit execution of the Program Section operation. Programming, which is not allowed to cross a flash sector boundary, continues until all requested double-phrases for program flash, phrases for data flash have been programmed.

After the Program Section operation has completed, the CCIF flag will set and normal access to the FlexRAM is restored. The contents of the Section Program Buffer are not changed by the Program Section operation.

Table 17-26. Program Section Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned for program flash, 64-bit aligned for data flash	FSTAT[ACCERR]
The requested section crosses a program flash sector boundary	FSTAT[ACCERR]
The requested number of double-phrases for program flash, phrases for data flash is zero	FSTAT[ACCERR]
The space required to store data for the requested number of double-phrases for program flash, phrases for data flash is more than one quarter the size of the FlexRAM	FSTAT[ACCERR]
The FlexRAM is not set to function as a traditional RAM, i.e. set if RAMRDY=0	FSTAT[ACCERR]
The flash address falls in a protected area	FSTAT[FPVIOL]
The requested flash section is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

17.5.11.8.1 Flash sector programming

The process of programming an entire flash sector using the Program Section command is as follows:

1. If required, execute the Set FlexRAM Function command to make the FlexRAM available as traditional RAM and initialize the FlexRAM to all ones.
2. Launch the Erase Flash Sector command to erase the flash sector to be programmed.
3. Beginning with the starting address of the FlexRAM, sequentially write enough data to the RAM to fill an entire flash sector, or as much data is allowed due to RAM size versus flash sector size. This area of the RAM serves as the section program buffer.

NOTE

In step 1, the section program buffer was initialized to all ones, the erased state of the flash memory.

The section program buffer can be written to while the operation launched in step 2 is executing, i.e. while CCIF = 0.

4. Execute the Program Section command to program the contents of the section program buffer into the selected flash sector.
5. Repeat steps 3 through 4 to complete the entire flash sector, if necessary.
6. To program additional flash sectors, repeat steps 2 through 5.
7. To restore EEPROM functionality, execute the Set FlexRAM Function command to make the FlexRAM available for EEPROM.

17.5.11.9 Read 1s All Blocks command

The Read 1s All Blocks command checks if the program flash blocks, data flash blocks, EEPROM backup records, and data flash IFR have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

Table 17-27. Read 1s All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x40 (RD1ALL)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Blocks command, the FTFE :

- sets the read margin for 1s according to Table 17-28,
- checks the contents of the program flash, data flash, EEPROM backup records, and data flash IFR are in the erased state.

If the FTFE confirms that these memory resources are erased, access control is disabled and security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see Flash configuration field description) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all flash memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The EEERDY and RAMRDY bits are clear during the Read 1s All Blocks operation and are restored at the end of the Read 1s All Blocks operation.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 17-28. 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 17-29. Read 1s All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

17.5.11.10 Read Once command

The Read Once command provides read access to indexed 8-byte records located in the program flash IFR (see Program flash 0 IFR map and Program Once field) . Each Program Once record index is programmed using the Program Once command.

Table 17-30. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Program Once record index (0x00 - 0x0B)
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
8	Program Once byte 4 value
9	Program Once byte 5 value
A	Program Once byte 6 value
B	Program Once byte 7 value

After clearing CCIF to launch the Read Once command, an 8-byte Program Once record is read from the program flash 0 IFR (index 0x00 to 0x0B) and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. During execution of the Read Once command, any attempt to read addresses within the flash block containing the 8-byte record returns invalid data. The Read Once command can be executed any number of times.

Table 17-31. 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]

17.5.11.11 Program Once command

The Program Once command enables programming of indexed 8-byte records located in the program flash IFR (see Program flash 0 IFR map and Program Once field) . The Program Once field can be read using the Read Once command or using the Read Resource command. Each Program Once record in program flash 0 IFR can be programmed only once since the program flash 0 IFR cannot be erased.

Table 17-32. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x0B)
2	Not Used
3	Not Used
4	Program Once Byte 0 value
5	Program Once Byte 1 value
6	Program Once Byte 2 value
7	Program Once Byte 3 value
8	Program Once Byte 4 value
9	Program Once Byte 5 value
A	Program Once Byte 6 value
B	Program Once Byte 7 value

After clearing CCIF to launch the Program Once command and verifying that the selected record is erased, the selected record is programmed using the values provided into the program flash 0 IFR (index 0x00 to 0x0B). The Program Once command also verifies that the programmed values read back correctly. Any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. The CCIF flag is set after the Program Once operation has completed. During execution of the Program Once command, any attempt to read addresses within the flash block containing the 8-byte record returns invalid data.

Table 17-33. Program Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]
The requested record has already been programmed to a non-erased value[1]	FSTAT[ACCERR]
Any errors have been encountered during the verify operation.	FSTAT[MGSTAT0]

1. If a Program Once record is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command is allowed to execute again on that same record.

17.5.11.12 Erase All Blocks command

The Erase All Blocks operation erases all flash memory, initializes the FlexRAM, verifies all memory contents, and releases MCU security.

Table 17-34. Erase All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x44 (ERSALL)

After clearing CCIF to launch the Erase All Blocks command, the FTFE erases all program flash memory, data flash memory, data flash IFR space, EEPROM backup memory, and FlexRAM, then verifies that all are erased.

If the FTFE verifies that all flash memories and the FlexRAM were properly erased, access control is disabled and security is released by setting the FSEC[SEC] field to the unsecure state and the FCNFG[RAMRDY] bit is set. The Erase All Blocks command aborts if any flash or FlexRAM region is protected. The 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.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Blocks command. While most Flash memory will be erased, the program flash 0 IFR space containing the Program Once XACC and SACC fields will not be erased and, therefore, the contents of the Program Once XACC and SACC fields will not change. The contents of the FXACC and FSACC registers will not be impacted by the execution of the Erase All Blocks command. After completion of the Erase All Blocks command, access control is disabled until the next reset of the flash module or the Read 1s All Blocks command is executed and fails (FSTAT[MGSTAT0] is set).

Table 17-35. Erase All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any region of the program flash memory, data flash memory, or FlexRAM is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s All Blocks command to verify all bits are erased.

17.5.11.12.1 Triggering an erase all external to the flash module

The functionality of the Erase All Blocks/Erase All Blocks Unsecure 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 FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory, data flash memory, data flash IFR space, EEPROM backup, and FlexRAM regardless of the state of the FSTAT[ACCERR and FPVIOL] flags or the protection settings. If the post-erase verify passes, access control determined by the contents of the FXACC registers is disabled and the routine releases security by setting the FSEC[SEC] field register to the unsecure state and the FCNFG[RAMRDY] bit sets. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting, except FPVIOL, is available as described in Erase All Blocks command/Erase All Blocks Unsecure command.

17.5.11.13 Verify Backdoor Access Key command

Execution of the Verify Backdoor Access Key command is qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash Configuration Field. The column labeled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 17-36. 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 FTFE checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the FTFE sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the FTFE compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the FTFE module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

Table 17-37. Verify Backdoor Access Key Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
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]

17.5.11.14 Erase All Blocks Unsecure command

The Erase All Blocks Unsecure operation erases all flash memory, initializes the FlexRAM, verifies all memory contents, programs the security byte in the Flash Configuration Field to the unsecure state, and releases MCU security.

Table 17-38. Erase All Blocks Unsecure Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x49 (ERSALLU)

After clearing CCIF to launch the Erase All Blocks Unsecure command, the FTFE erases all program flash memory, data flash memory, data flash IFR space, EEPROM backup memory, and FlexRAM, then verifies that all are erased.

If the FTFE verifies that all flash memories and the FlexRAM were properly erased, access control is disabled and security is released by setting the FSEC[SEC] field to the unsecure state, the security byte (see Flash configuration field description) is programmed to the unsecure state by the Erase All Blocks Unsecure command, and the

FCNFG[RAMRDY] bit is set. If the erase or program verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks Unsecure operation completes.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Blocks Unsecure command. While most Flash memory will be erased, the program flash 0 IFR space containing the Program Once XACC and SACC fields will not be erased and, therefore, the contents of the Program Once XACC and SACC fields will not change. The contents of the FXACC and FSACC registers will not be impacted by the execution of the Erase All Blocks Unsecure command. After completion of the Erase All Blocks Unsecure command, access control is disabled until the next reset of the flash module or the Read 1s All Blocks command is executed and fails (FSTAT[MGSTAT0] is set).

Table 17-39. Erase All Blocks Unsecure Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any errors have been encountered during erase or program verify operations	FSTAT[MGSTAT0]

17.5.11.15 Read 1s All Execute-only Segments command

The Read 1s All Execute-only Segments command checks if the program flash execute-only segments defined by the FXACC registers have been erased to the specified read margin level, if applicable, and releases flash access control if the readout passes, i.e. all data reads as '1'.

Table 17-40. Read 1s All Execute-only Segments Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x4A (RD1XA)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Execute-only Segments command, the flash memory module :

- sets the read margin for 1s according to Table 17-41,
- checks the contents of the program flash execute-only segments are in the erased state.

If the flash memory module confirms that these segments are erased, flash access control is disabled until the next reset or, after programming any of the execute-only segments, the Read 1s All Execute-only Segments command is executed and fails with the FSTAT[MGSTAT0] bit set. If the read fails, i.e. all segments are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Execute-only Segments operation has completed.

Table 17-41. Margin Level Choices for Read 1s All Execute-only Segments

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 17-42. Read 1s All Execute-only Segments Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

17.5.11.16 Erase All Execute-only Segments command

The Erase All Execute-only Segments operation erases all program flash execute-only segments defined by the FXACC registers, verifies all segments are erased, and releases flash access control.

Table 17-43. Erase All Execute-only Segments Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x4B (ERSXA)

After clearing CCIF to launch the Erase All Execute-only Segments command, the flash memory module erases all program flash execute-only segments, then verifies that all segments are erased.

If the flash memory module verifies that all segments were properly erased, flash access control is disabled until the next reset or, after programming any of the execute-only segments, the Read 1s All Execute-only Segments command is executed and fails with the FSTAT[MGSTAT0] bit set. The Erase All Execute-only Segments command aborts if

any XA controlled segment is protected. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Execute-only Segments operation completes.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Execute-only Segments command. While all XA controlled segments will be erased, the program flash IFR space containing the Program Once XACC fields will not be erased and, therefore, the contents of the Program Once XACC fields will not change. The contents of the FXACC registers will not be impacted by the execution of the Erase All Execute-only Segments command.

Table 17-44. Erase All Execute-only Segments Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any XA controlled segment in the program flash memory is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

17.5.11.17 Program Partition command

The Program Partition command prepares the FlexNVM block for use as data flash, EEPROM backup, or a combination of both and initializes the FlexRAM. The Program Partition command must not be launched from flash memory, since flash memory resources are not accessible during Program Partition command execution.

CAUTION

While different partitions of the FlexNVM are available, the intention is that a single partition choice is used throughout the entire lifetime of a given application. The FlexNVM Partition Code choices affect the endurance and data retention characteristics of the device.

Table 17-45. Program Partition Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x80 (PGMPART)
1	Not Used
2	Not Used
3	FlexRAM load during reset option (only bit 0 used): 0 - FlexRAM loaded with valid EEPROM data during reset sequence 1 - FlexRAM not loaded during reset sequence

Table continues on the next page...

Table 17-45. Program Partition Command FCCOB Requirements (continued)

FCCOB Number	FCCOB Contents [7:0]
4	EEPROM Data Set Size Code[1]
5	FlexNVM Partition Code[2]

1. See Valid EEPROM Data Set Size Codes and EEPROM Data Set Size
2. See Valid FlexNVM Partition Codes and FlexNVM partition code

Table 17-46. Valid EEPROM Data Set Size Codes

EEPROM Data Set Size Code (FCCOB4)[1]		EEPROM Data Set Size (Bytes)
EEESPLIT (FCCOB4[5:4)]	EEESIZE (FCCOB4[3:0])	
11	0xF	0[2]
11	0x9	32
11	0x8	64
11	0x7	128
11	0x6	256
11	0x5	512
11	0x4	1,024
11	0x3	2,048
11	0x2	4,096

1. FCCOB4[7:6] = 00
2. EEPROM Data Set Size must be set to 0 Bytes when the FlexNVM Partition Code is set for no EEPROM.

Table 17-47. Valid FlexNVM Partition Codes

FlexNVM Partition Code DEPART (FCCOB5[3:0)[1]	Data flash Size (Kbytes)	EEPROM-backup Size (Kbytes)
0000	64	0
0011	32	32
0100	0	64
1000	0	64
1010	16	48
1011	32	32
1100	64	0

1. FCCOB5[7:4] = 0000

After clearing CCIF to launch the Program Partition command, the FTFE first verifies that the EEPROM Data Set Size Code and FlexNVM Partition Code in the data flash IFR are erased. If erased, the Program Partition command erases the contents of the FlexNVM memory. If the FlexNVM is to be partitioned for EEPROM backup, the allocated EEPROM backup sectors are formatted for EEPROM use. Finally, the partition codes are

programmed into the data flash IFR using the values provided. The Program Partition command also verifies that the partition codes read back correctly after programming. The CCIF flag is set after the Program Partition operation completes.

Prior to launching the Program Partition command, the data flash IFR must be in an erased state, which can be accomplished by executing the Erase All Blocks command or by an external request (see Erase All Blocks command). The EEPROM Data Set Size Code and FlexNVM Partition Code are read using the Read Resource command (see Read Resource command).

Table 17-48. Program Partition Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
The EEPROM data size and FlexNVM partition code bytes are not initially 0xFFFF	FSTAT[ACCERR]
Invalid EEPROM Data Set Size Code is entered (see Table 17-46 for valid codes)	FSTAT[ACCERR]
Invalid FlexNVM Partition Code is entered (see Table 17-47 for valid codes)	FSTAT[ACCERR]
FlexNVM Partition Code = full data flash (no EEPROM) and EEPROM Data Set Size Code allocates FlexRAM for EEPROM	FSTAT[ACCERR]
FlexNVM Partition Code allocates space for EEPROM backup, but EEPROM Data Set Size Code allocates no FlexRAM for EEPROM	FSTAT[ACCERR]
FCCOB4[7:6] != 00	FSTAT[ACCERR]
FCCOB5[7:4] != 0000	FSTAT[ACCERR]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

17.5.11.18 Set FlexRAM Function command

The Set FlexRAM Function command changes the function of the FlexRAM:

- When not partitioned for EEPROM, the FlexRAM is typically used as traditional RAM.
- When partitioned for EEPROM, the FlexRAM is typically used to store EEPROM data.

Table 17-49. Set FlexRAM Function Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x81 (SETRAM)
1	FlexRAM Function Control Code (see Table 17-50)

Table 17-50. FlexRAM Function Control

FlexRAM Function Control Code	Action
0xFF	Make FlexRAM available as RAM: • Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags • Write a background of ones to all FlexRAM locations • Set the FCNFG[RAMRDY] flag
0x00	Make FlexRAM available for EEPROM: • Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags • Write a background of ones to all FlexRAM locations • Copy-down existing EEPROM data to FlexRAM • Set the FCNFG[EEERDY] flag

After clearing CCIF to launch the Set FlexRAM Function command, the FTFE sets the function of the FlexRAM based on the FlexRAM Function Control Code.

When making the FlexRAM available as traditional RAM, the FTFE clears the FCNFG[EEERDY] and FCNFG[RAMRDY] flags, overwrites the contents of the entire FlexRAM with a background pattern of all ones, and sets the FCNFG[RAMRDY] flag. The state of the EPROT register does not prevent the FlexRAM from being overwritten. When the FlexRAM is set to function as a RAM, normal read and write accesses to the FlexRAM are available. When large sections of flash memory need to be programmed, e.g. during factory programming, the FlexRAM can be used as the Section Program Buffer for the Program Section command (see Program Section command).

When making the FlexRAM available for EEPROM, the FTFE clears the FCNFG[RAMRDY] and FCNFG[EEERDY] flags, overwrites the contents of the FlexRAM allocated for EEPROM with a background pattern of all ones, and copies the existing EEPROM data from the EEPROM backup record space to the FlexRAM. After completion of the EEPROM copy-down, the FCNFG[EEERDY] flag is set. When the FlexRAM is set to function as EEPROM, normal read and write access to the FlexRAM is available, but writes to the FlexRAM also invoke EEPROM activity.

The CCIF flag will be set after the Set FlexRAM Function operation has completed.

Table 17-51. Set FlexRAM Function Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
FlexRAM Function Control Code is not defined	FSTAT[ACCERR]
FlexRAM Function Control Code is set to make the FlexRAM available for EEPROM, but FlexNVM is not partitioned for EEPROM	FSTAT[ACCERR]

17.5.12 Security

The FTFE module provides security information to the MCU based on contents of the FSEC security register. The MCU then limits access to FTFE resources as defined in the device's Chip Configuration details. During reset, the FTFE module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see Flash configuration field description).

The following fields are available in the FSEC register. Details of the settings are described in the FSEC register description.

Flash security features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Some features described in the application note may not be available on this device.

Table 17-52. FSEC fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Factory Security Level Access
SEC	MCU security

17.5.12.1 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 effect after the next MCU reset.

17.5.12.1.1 Unsecuring the MCU Using Backdoor Key Access

The MCU can be unsecured by using the backdoor key access feature which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see Flash configuration field description). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see Verify Backdoor Access Key command) can be run which allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the MCU. The entire 8-byte key cannot be all 0s or all 1s, i.e. 0x0000_0000_0000_0000 and 0xFFFF_FFFF_FFFF_FFFF are not accepted by the

Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN bits are in the enabled state, the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key command
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the MCU is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field (Flash configuration field description). After the next reset of the MCU, the security state of the FTFE module reverts back to the Flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured MCU has full control of the contents of the Flash Configuration Field. The MCU may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

17.6 Reset Sequence

On each system reset the FTFE module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FDPROT, FEPROT, FOPT, FSEC, FXACC, FSACC, FACSS, and FACSN registers and the FCNFG[RAMRDY, EEERDY] bits.

CCIF is cleared throughout the reset sequence. The FTFE module holds off all CPU access for a portion of the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any FTFE command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

17.7 Usage Guide

Related application notes on this FTFE module are as follows.

- Production Flash Programming Best Practices for Kinetis K- and L-series MCUs
- Using the Kinetis Security and Flash Protection Features
- Using the Kinetis Family Enhanced EEPROM Functionality
- Robust Over-the-Air Firmware Updates Using Program Flash Memory Swap on Kinetis Microcontrollers
- Using the Kinetis Flash Execute-Only Access Control Feature

Chapter 18
Clock Distribution

18.1 Introduction

This chapter presents the clock architecture overview of this device, the clock distribution and module clocks, and a clock terminology section. The clocking generation and configuration can be divided into 3 parts:

1. Clock sources generation
   - FIRC, SIRC, SOSC, PLL, all from the SCG module
   - OSC32
   - LPO from PMC
2. Peripheral Clock Controller (PCC)
3. Module level clock control (Inside specific peripherals)

The System Clock Generator (SCG) module is used on this device for main system clock generation. It generates clock sources like Fast IRC (FIRC, 48 MHz, within 1% accuracy), Slow IRC (SIRC, 2/8 MHz, within 3% accuracy), System Oscillator (SOSC) and PLL. It controls which clock source is used to derive the system clocks. The SCG also divides the selected clock source into a variety of clock domains, including the clocks for the system bus masters, system bus slaves, and flash memory .

Besides the clocks generated by SCG, there are other clock generator: OSC32, LPO from PMC.

Clock selection for most modules is controlled by the Peripheral Clock Controller (PCC) module. The PCC also implements module-specific clock gating to allow granular shutoff of modules.

Various modules have module-specific clocks that can be generated from the FIRC_CLK, SIRC_CLK, SOSC_CLK, or SCGPLLCLK clock. In addition, there are various other module-specific clocks that have other alternate sources. While clock

selection for most modules is controlled by the PCC module, some peripherals have clock source selection/divider inside the module, for details, please see the "Peripheral Clock Summary" table for more information.

18.2 High-level clocking diagram

The following diagram shows the high-level clocking architecture and various clock sources for this device.

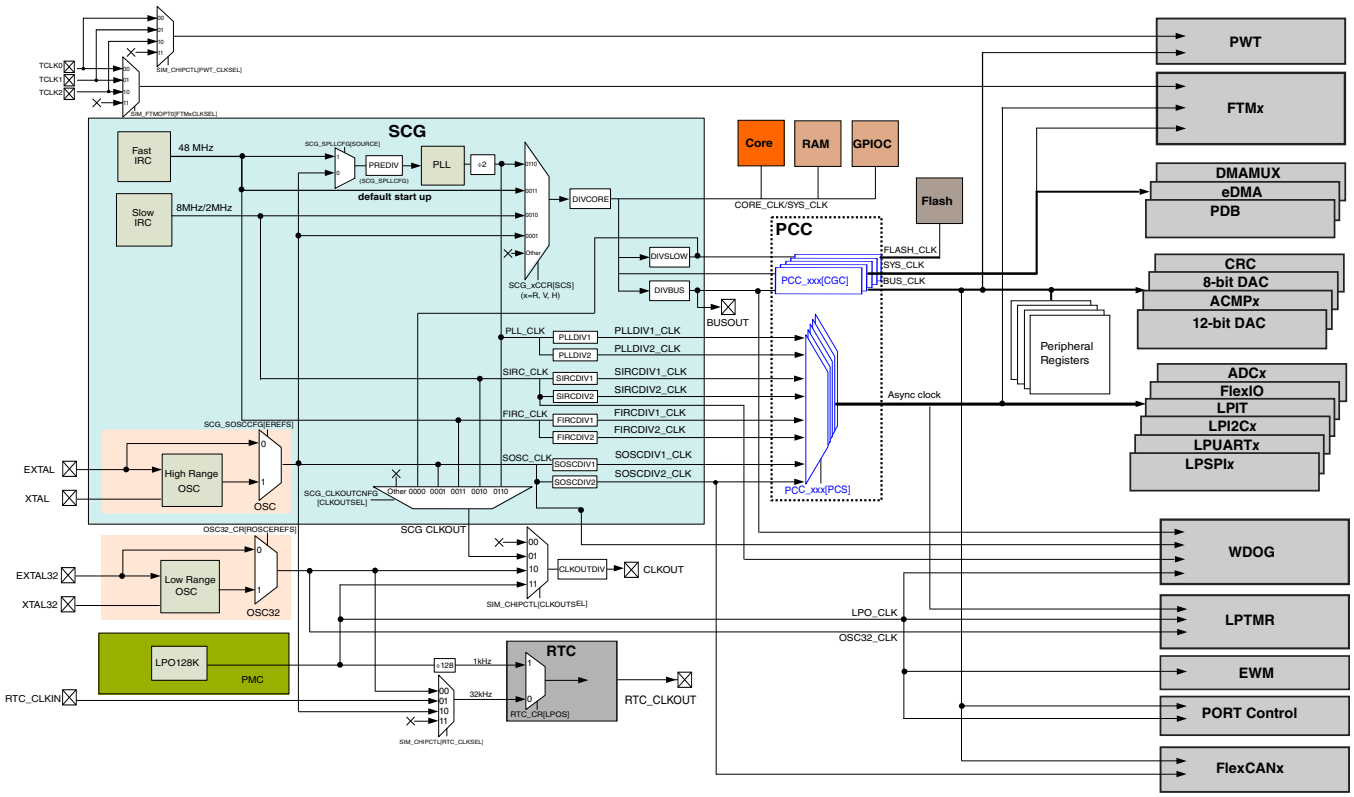


Figure 18-1. Clocking Diagram

NOTE

For clock divider DIV*x* in SCG (output to PCC), **DIV1** applies to *only* the **FTM** module, while DIV2 applies to other peripheral modules (excluding WDOG, whose clock source with no DIV*x*).

18.3 Clock definitions

The following table describes the clocks in the previous block diagram and other sections of this document.

Clock name	Description
CORE_CLK	Clocks the ARM core, divided by DIVCORE bits inside SCG
SYS_CLK	Clocks the Crossbar, NVIC, Flash controller, FTM and PDB, etc. SYS_CLK can run up to CORE_CLK and divided by DIVCORE bits inside SCG.
BUS_CLK	Clocks the Peripherals, divided by DIVBUS bits inside SCG
FLASH_CLK	Clocks the flash module, divided by DIVSLOW bits inside SCG
SPLL_CLK	Optional divided PLL source for peripherals
SIRC_CLK	Optional divided SIRC source for peripherals
FIRC_CLK	Optional divided FIRC source for peripherals
SOSC_CLK[1]	Optional divided System Oscillator clock for peripherals. **NOTE:** SOSC_CLK/ERCLK/OSCERCLK stands for the same clock source, in some module chapters.
OSC32_CLK	RTC oscillator clock for RTC and peripherals
LPO_CLK	Always on low power oscillator clock inside PMC
RTC_CLKOUT	Clock output from RTC module for both internal and external
CLKOUT	Optional output clock source for external devices
BUSOUT	Optional output bus clock through pin for external devices or diagnostics

1. • For WDOG, its SOSC_CLK is with no dividers.
   • For other peripherals (LPUART etc.), its SOSC_CLK is divided by DIVx.

18.4 Typical Clock Configuration

The clock dividers are programmed via the SCG module's clock divider registers. The following requirements must be met when configuring the clocks for this device:

1. The core and system clock frequencies must be 168 MHz or less in HSRUN mode, and 120 MHz in normal RUN mode.
2. The bus clock frequency must be programmed to 84 MHz or less in HSRUN, 60 MHz or less in RUN, and an integer divide of the core clock.
3. The flash clock frequency must be programmed to 25 MHz or less, less than or equal to the bus clock, and an integer divide of the core clock. The core clock to flash clock ratio is limited to a max value of 8.

The following are a few of the more common clock configurations for this device:

Option: High Speed RUN – (using the PLL from either the FIRC or SYS OSC)

Clock	Frequency
CORE_CLK	168 MHz

Table continues on the next page...

Clock	Frequency
SYS_CLK	168 MHz
BUS_CLK	84 MHz
FLASH_CLK	24 MHz

Option: Normal RUN (using the PLL from either the IRC or SYS OSC)

Clock	Frequency
CORE_CLK	120 MHz
SYS_CLK	120 MHz
BUS_CLK	60 MHz
FLASH_CLK	24 MHz

or

Clock	Frequency
CORE_CLK	100 MHz
SYS_CLK	100 MHz
BUS_CLK	50 MHz
FLASH_CLK	25 MHz

Option: Slow RUN (typically using the undivided FIRC)

Clock	Frequency
CORE_CLK	48 MHz
SYS_CLK	48 MHz
BUS_CLK	48 MHz
FLASH_CLK	24 MHz

Option: Very Low Power RUN, VLPR (using the SIRC or SYS OSC)

Clock	Frequency
CORE_CLK	4 MHz
SYS_CLK	4 MHz
BUS_CLK	4 MHz
FLASH_CLK	1 MHz

18.4.1 Default start-up clock

In default out of reset, the CPU is clocked from internal Fast IRC (IRC48M). The clocks, e.g. core clock and bus clock, are programmed via the SCG module. For the default reset value of divider, please refer to SCG chapter for details.

18.4.2 VLPR mode clocking

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

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

18.5 Clock Gating

The clock to most of the modules can be individually gated on and off using the PCC module. After any reset, PCC disables part of the clock to the corresponding module to conserve power. Prior to initializing a module, set the corresponding clock gating control bits in PCC 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 bus interface clock disabled (CGC=0 in PCC module) will generate a bus fault. While any bus access to a peripheral that has its functional clock disabled (PCS=0 in PCC module) will not return a fault, but the module cannot work properly.

NOTE

Changes to clock source should be done when clock is gated by PCC to avoid glitches to output clock.

18.6 Module clocks

The following table summarizes the clocks associated with each module.

Table 18-1. Peripheral Clock Summary

Module Name	Bus Interface Clock Gating	Peripheral Functional Clock [1]		Max Frequency of Clock Source
	Gated by [CGC] bit of PCC	Clocks controlled by [PCS] bits of PCC	Clocks controlled by registers inside module	
Communications				
LPUART0 – LPUART2	Yes	SIRC_CLK,FIRC_CLK,SPLL_CLK, SOSC_CLK	-	Max: 80 MHz
LPSPI0 – LPSPI1	Yes	SIRC_CLK,FIRC_CLK,SPLL_CLK, SOSC_CLK	-	Max: 80 MHz SCK clock Max: 25 MHz
LPI^{2}C0 – LPI^{2}C1	Yes	SIRC_CLK,FIRC_CLK,SPLL_CLK, SOSC_CLK	-	Max: BUS_CLK
FlexIO	Yes	SIRC_CLK,FIRC_CLK,SPLL_CLK,SOSC_CLK	-	Max: 80 MHz
FlexCAN0 – FlexCAN1	Yes	-	BUS_CLK, SOSC_CLK	Support 40 MHz from OSC
Timers				
LPTMR	Yes	SIRC_CLK,FIRC_CLK, SPLL_CLK,SOSC_CLK	LPO_CLK, OSC32_CLK	Max: BUS_CLK LPO_CLK: 128kHz
LPIT	Yes	SIRC_CLK,FIRC_CLK, SPLL_CLK,SOSC_CLK	-	Max: BUS_CLK
RTC	Yes	-	LPO_CLK, OSC32_CLK, RTC_CLKIN, SOSC_CLK	Max: BUS_CLK LPO_CLK: 1kHz
PDB0–PDB2	Yes	SYS_CLK		Max: SYS_CLK
FlexTimer0 - FlexTimer3	Yes	-	SYS_CLK, SIRC_CLK,FIRC_CLK, SPLL_CLK, SOSC_CLK, TCLKx	Max: SYS_CLK
PWT	Yes	-	BUS_CLK, TCLKx	Max: BUS_CLK
System Modules				
Watchdog	Yes	-	BUS_CLK, SIRC_CLK, LPO_CLK, SOSC_CLK,	Max: BUS_CLK LPO_CLK: 128kHz
EWM	Yes	-	LPO_CLK	LPO_CLK: 128kHz
PMC	No	-	BUS_CLK	Max: BUS_CLK
SIM	No	BUS_CLK		Max: BUS_CLK
RCM	No	-	BUS_CLK, LPO_CLK	Max: BUS_CLK
Port Control	Yes	-	BUS_CLK, LPO_CLK	Max: BUS_CLK

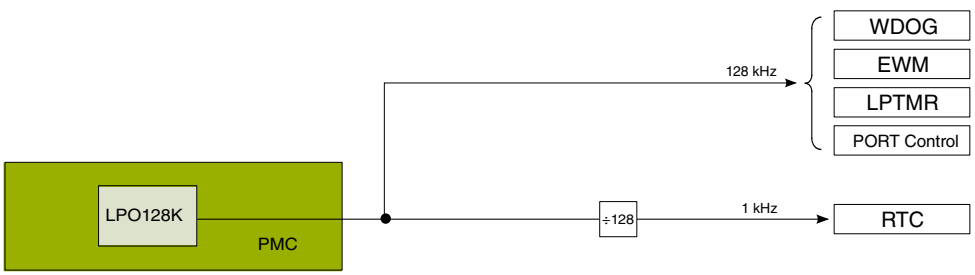
Table continues on the next page...

Table 18-1. Peripheral Clock Summary (continued)

Module Name	Bus Interface Clock Gating	Peripheral Functional Clock [1]		Max Frequency of Clock Source
	Gated by [CGC] bit of PCC	Clocks controlled by [PCS] bits of PCC	Clocks controlled by registers inside module	
CRC	Yes	BUS_CLK		Max: BUS_CLK
GPIOC	Yes	SYS_CLK		Max: SYS_CLK
DMA	Yes	SYS_CLK		Max: SYS_CLK
Memory Modules				
FTFE	Yes	FLASH_CLK		Max: FLASH_CLK
SYS RAM	No	SYS_CLK		Max: SYS_CLK
Analog Modules				
ADC0–ADC2	Yes	SIRC_CLK,FIRC_CLK,SPLL_CLK, SOSC_CLK	-	Max: 50 MHz
DAC0	Yes	BUS_CLK		Max: BUS_CLK
ACMP0–ACMP2	Yes	BUS_CLK		Max: BUS_CLK

1. The clock sources undergo clock divider DIV*x* in SCG. For details, see the "High-Level clocking diagram" section in the Clocking chapter and the "Chip-specific information" section in the respective module chapter.

18.6.1 LPO clock distribution



18.6.2 EWM clocks

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

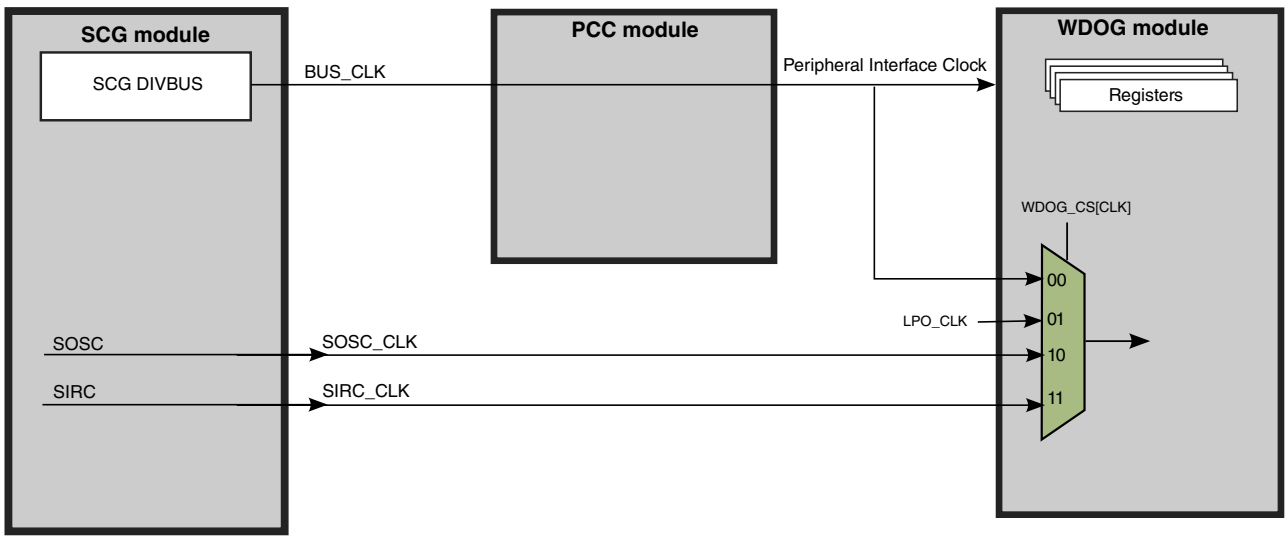
Table 18-2. EWM clock connections

Module clock	Chip clock
Low Power Clock	128 kHz LPO Clock (LPO_CLK)

18.6.3 WDOG Clocking Information

The following figure shows the input clock sources available for this module.

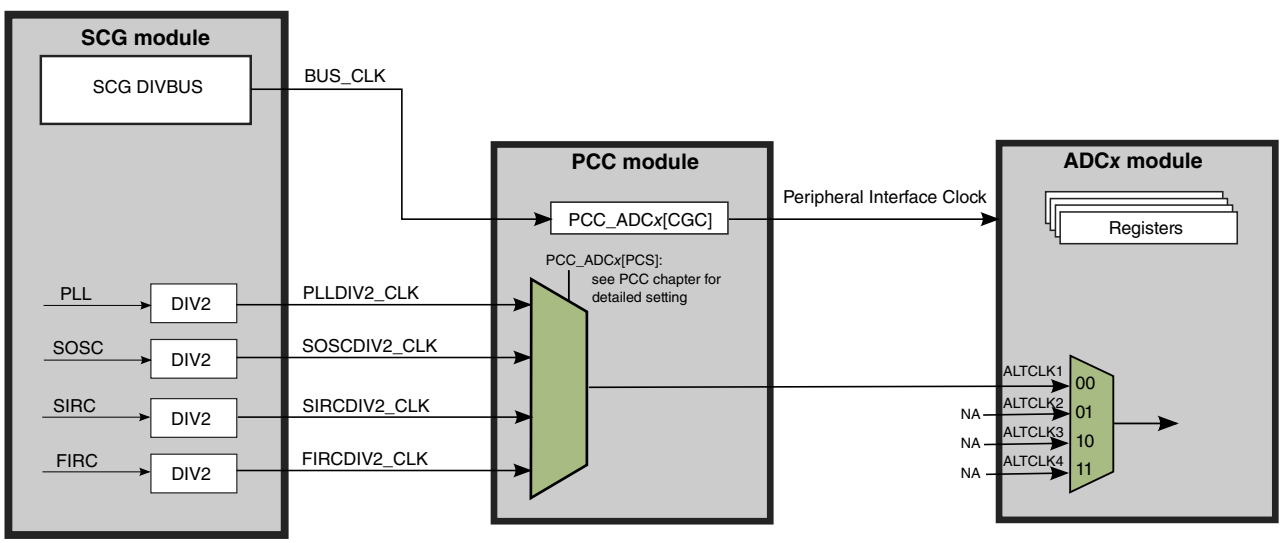
Peripheral Clocking - WDOG



18.6.4 ADC Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - ADC



NOTE

ALTCLK2~4 are not connected on this chip.

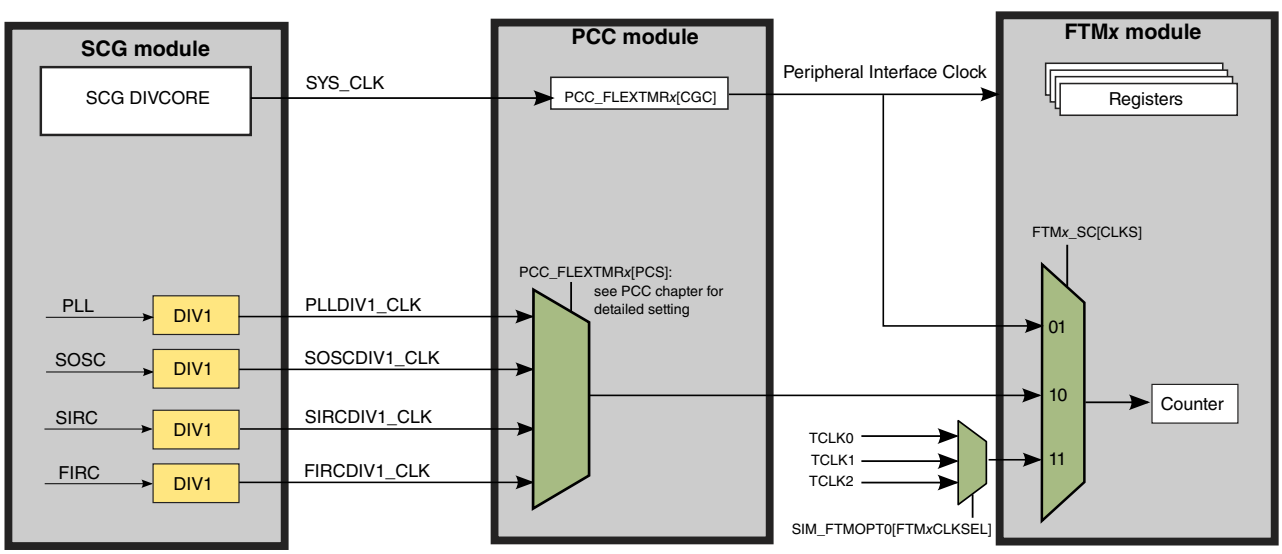
18.6.5 PDB Clock Options

The PDB module is clocked by the system clock (SYS_CLK). The SYS_CLK could run up to CPU frequency which provides higher timing resolution and more precise delay control with the PDB counter.

18.6.6 FTM Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - FTM



NOTE

Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM system clock frequency (SYS_CLK).

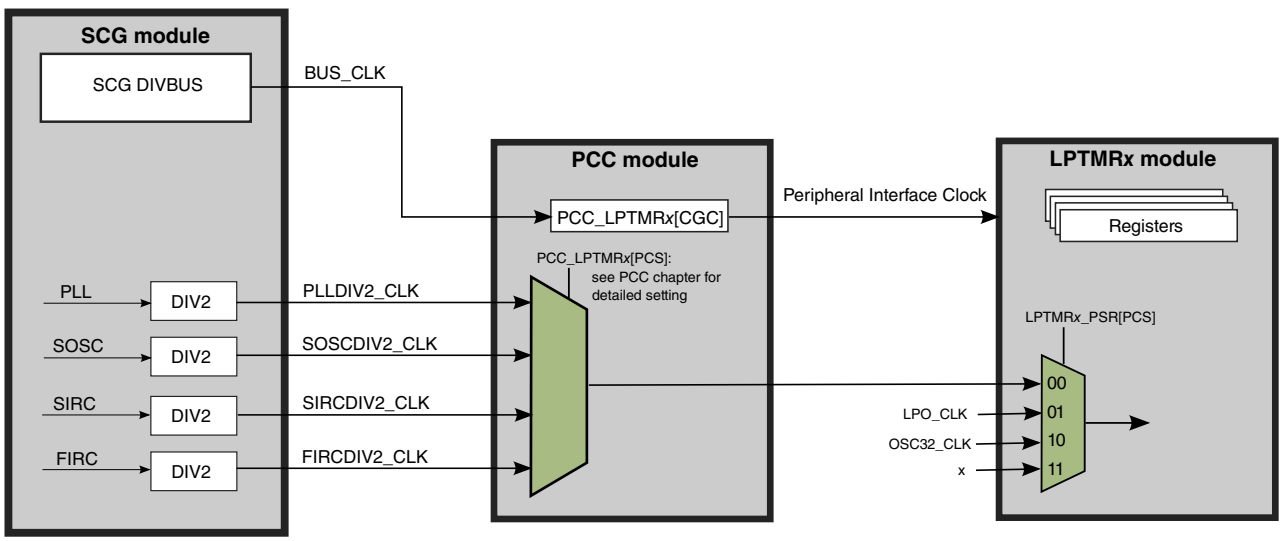
NOTE

The external clock are synchronized by FTM system clock (SYS_CLK). Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

18.6.7 LPTMR prescaler/glitch filter clocking options

The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPTMR



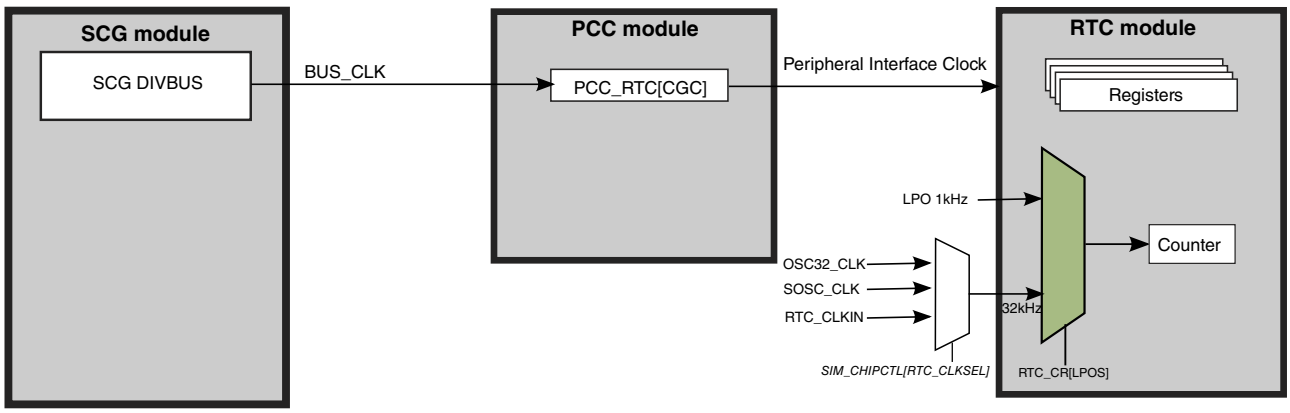
NOTE

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

18.6.8 RTC Clocking Information

The following figure shows the input clock sources available for this module.

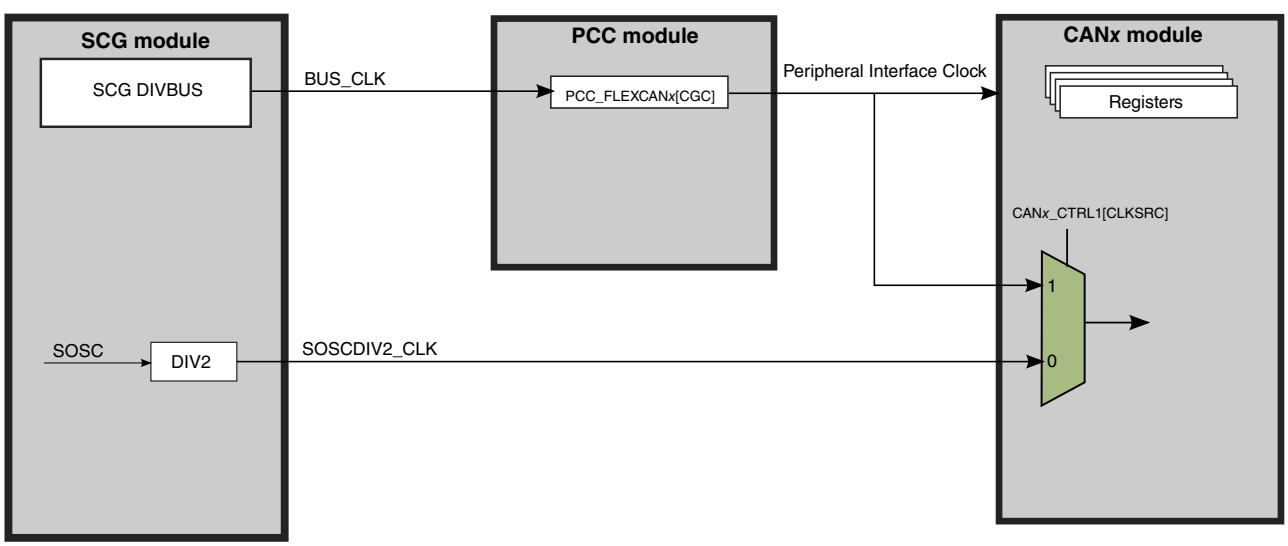
Peripheral Clocking - RTC



18.6.9 FlexCAN Clocking Information

The following figure shows the input clock sources available for this module.

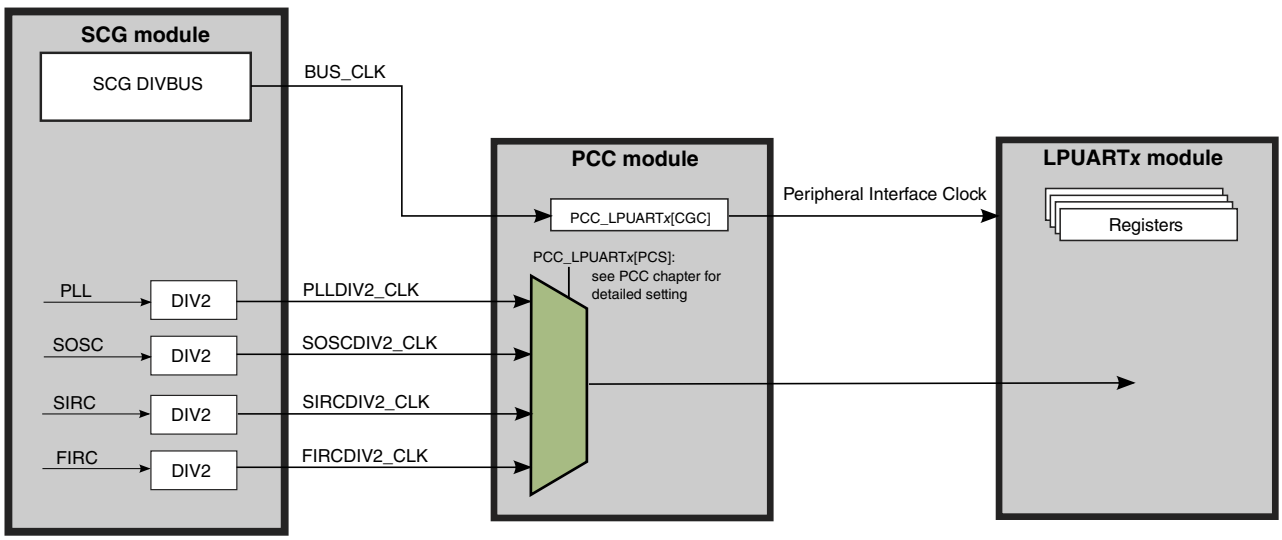
Peripheral Clocking - FlexCAN



18.6.10 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPUART Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, FlexIO and LPIT.



Chapter 19
System Clock Generator (SCG)

19.1 Chip-specific information for this module

19.1.1 Instantiation Information

19.1.1.1 Information of SCG on this device

NOTE

FIRC is trimmed to 48 MHz only in this device. Other values are reserved in SCG_FIRCCFG[RANGE].

Writing to SCG_FIRCSTAT register can cause hard fault when auto trim is disabled.

ERCLK (External Reference Clock) is either from an external pin or from the SCG internal OSC (SOSC), and configured with the SCG_SOSCCFG[EREFS] bit.

For the supported frequency range of OSC, see the "Oscillator electrical specifications" section in the Data Sheet. For this device, low frequency range: 32 kHz - 40 kHz; medium frequency range: 4 MHz - 8 MHz; high frequency range: 8 MHz - 40 MHz.

NOTE

ROSC in this chapter has the same meaning as OSC32_CLK in the Clock Distribution chapter.

19.1.1.1.1 SCG clock mode transitions

The following figure shows the valid clock mode transitions supported by SCG, for this device. For more information, see the Functional description section.

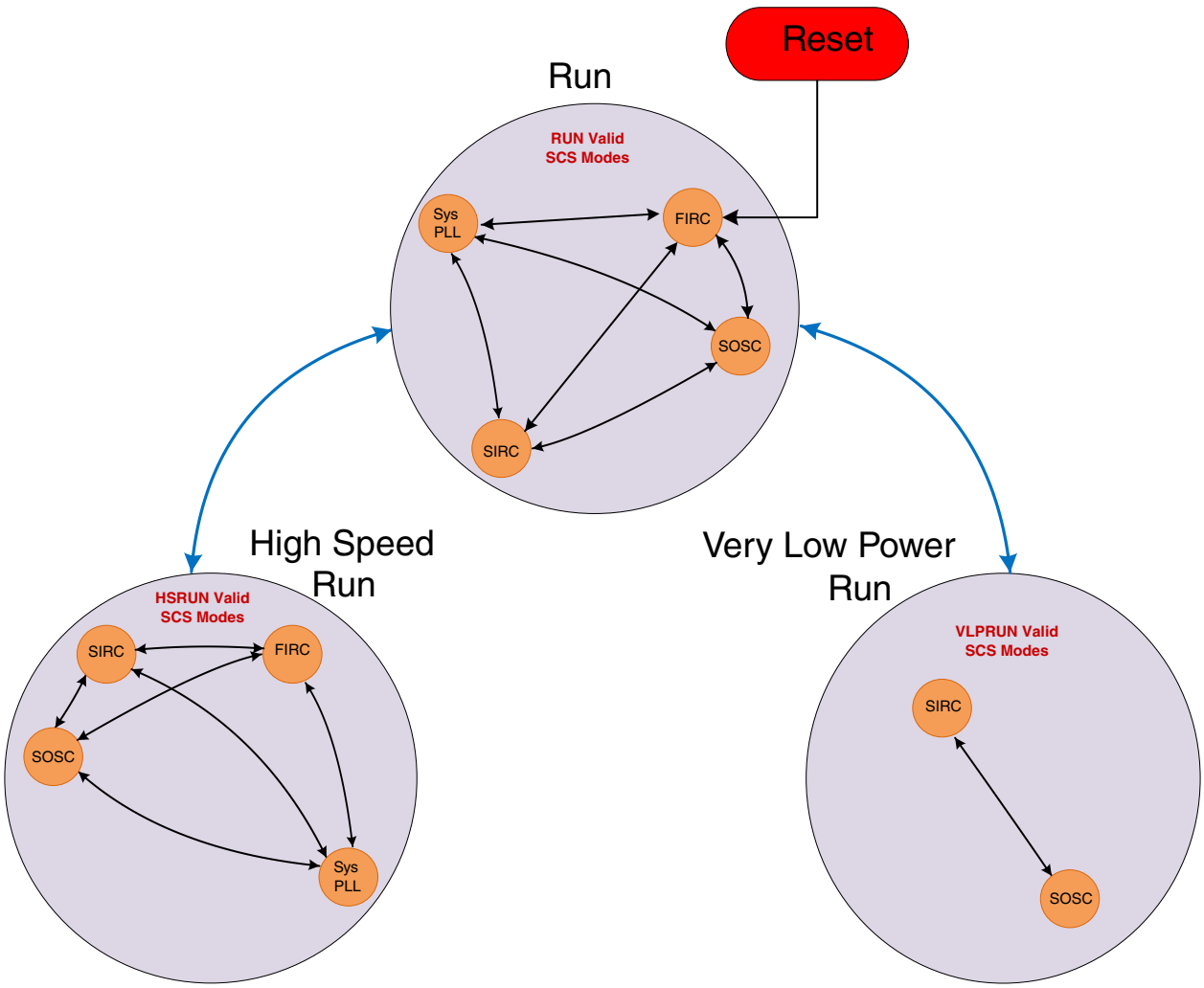


Figure 19-1. SCG Valid Mode Transition Diagram

19.2 Introduction

The system clock generator (SCG) module provides the system clocks of the MCU. The SCG contains a system phase-locked loop (SPLL), a slow internal reference clock (SIRC), a fast internal reference clock (FIRC), and the system oscillator clock (SOSC). The SPLL is sourced by either the SOSC reference clock or the FIRC reference clock. The SCG can select either the output clock of the SPLL or a SCG reference clock (SIRC, FIRC, and SOSC) as the source for the MCU system clocks. The SCG also supports

operation with crystal oscillators, which allows an external crystal, ceramic resonator, or another external clock source to produce the external reference clock (which are also available as clock sources for the MCU systems clocks).

19.2.1 Features

Key features of the SCG module are:

- System Phase-locked loop (SPLL):
  - Voltage-controlled oscillator (VCO)
  - Selectable Internal or External reference clock is used as the PLL source
  - Modulo VCO frequency divider
  - Phase/Frequency detector
  - Integrated loop filter
  - Can be selected as the clock source for the MCU system clocks
  - 2 programmable post-dividers clock outputs, which can be used as clock sources for other on-chip peripherals
- 2 Internal reference clock (IRC) generators:
  - Fast IRC clock with programmable High and Low frequency range
  - Fast clock can be used as a source for System PLL
  - Either the slow or the fast clock can be selected as the clock source for the MCU system clocks
  - 2 programmable post-divider clock outputs for each IRC, which can be used as clock sources for other on-chip peripherals
- System Crystal Oscillator:
  - Can be used as a source for the System PLL
  - Can be selected as the clock source for the MCU system clocks
- Clock monitor with reset and interrupt request capability for SPLL, SOSC, clocks
- Lock detector with interrupt request capability for use with the SPLL
- Each of the clock sources have reference dividers for clocking on-chip modules and peripherals, namely:

- SPLLDIV1_CLK / SPLLDIV2_CLK
- FIRCDIV1_CLK / SCG_FIRCDIV2_CLK
- SIRCDIV1_CLK / SIRCDIV2_CLK
- SOSCDIV1_CLK / SOSCDIV2_CLK

See the Clock Distribution Chapter for more information.

19.3 Memory Map/Register Definition

This section includes the memory map and register definition.

The SCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus transfer error. Read accesses may be performed in both supervisor and user mode.

NOTE

For any writeable SCG registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.

SCG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_4000	Version ID Register (SCG_VERID)	32	R	0100_0000h	19.3.1/451
4006_4004	Parameter Register (SCG_PARAM)	32	R	See section	19.3.2/451
4006_4010	Clock Status Register (SCG_CSR)	32	R	See section	19.3.3/452
4006_4014	Run Clock Control Register (SCG_RCCR)	32	R/W	See section	19.3.4/454
4006_4018	VLPR Clock Control Register (SCG_VCCR)	32	R/W	See section	19.3.5/457
4006_401C	HSRUN Clock Control Register (SCG_HCCR)	32	R/W	See section	19.3.6/459
4006_4020	SCG CLKOUT Configuration Register (SCG_CLKOUTCNFG)	32	R/W	0300_0000h	19.3.7/461
4006_4100	System OSC Control Status Register (SCG_SOSCCSR)	32	R/W	See section	19.3.8/462
4006_4104	System OSC Divide Register (SCG_SOSCDIV)	32	R/W	0000_0000h	19.3.9/464
4006_4108	System Oscillator Configuration Register (SCG_SOSCCFG)	32	R/W	0000_0010h	19.3.10/465
4006_4200	Slow IRC Control Status Register (SCG_SIRCCSR)	32	R/W	0100_0005h	19.3.11/467
4006_4204	Slow IRC Divide Register (SCG_SIRCDIV)	32	R/W	0000_0000h	19.3.12/468
4006_4208	Slow IRC Configuration Register (SCG_SIRCCFG)	32	R/W	0000_0001h	19.3.13/469
4006_4300	Fast IRC Control Status Register (SCG_FIRCCSR)	32	R/W	See section	19.3.14/470
4006_4304	Fast IRC Divide Register (SCG_FIRCDIV)	32	R/W	0000_0000h	19.3.15/472
4006_4308	Fast IRC Configuration Register (SCG_FIRCCFG)	32	R/W	0000_0000h	19.3.16/473
4006_430C	Fast IRC Trim Configuration Register (SCG_FIRCTCFG)	32	R/W	0000_0000h	19.3.17/474
4006_4318	Fast IRC Status Register (SCG_FIRCSTAT)	32	R/W	See section	19.3.18/475

Table continues on the next page...

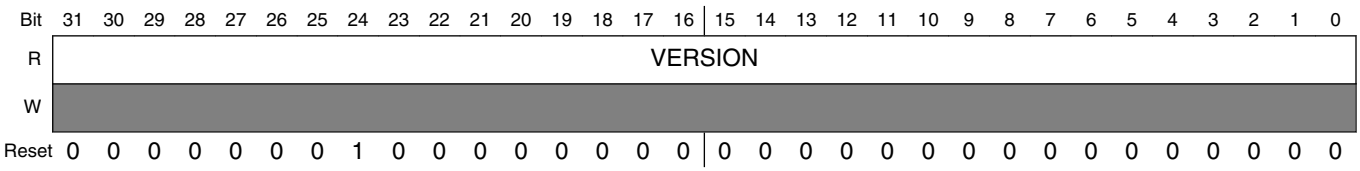
SCG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_4600	System PLL Control Status Register (SCG_SPLLCSR)	32	R/W	0000_0000h	19.3.19/476
4006_4604	System PLL Divide Register (SCG_SPLLDIV)	32	R/W	0000_0000h	19.3.20/478
4006_4608	System PLL Configuration Register (SCG_SPLLCFG)	32	R/W	0000_0000h	19.3.21/479

19.3.1 Version ID Register (SCG_VERID)

Note: Writing to this register will result in a transfer error.

Address: 4006_4000h base + 0h offset = 4006_4000h



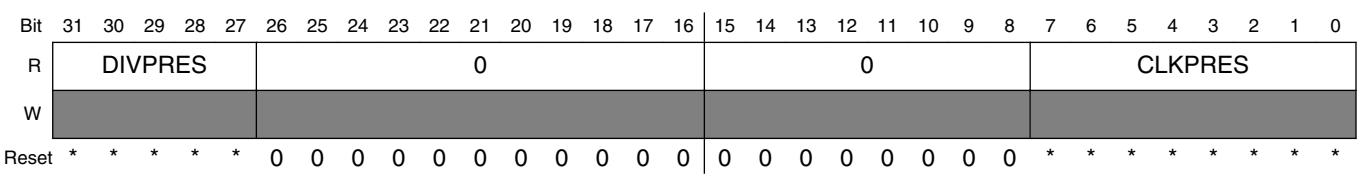
SCG_VERID field descriptions

Field	Description
VERSION	SCG Version Number

19.3.2 Parameter Register (SCG_PARAM)

Note: Writing to this register will result in a transfer error.

Address: 4006_4000h base + 4h offset = 4006_4004h



* Notes:
- DIVPRES field: The reset value is controlled by which SCG System Dividers are used by Soc.
- CLKPRES field: The reset value is controlled by which SCG Clock Sources are used by Soc. Please reference the Reference manual clocking chapter.

SCG_PARAM field descriptions

Field	Description
31–27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG.

Table continues on the next page...

SCG_PARAM field descriptions (continued)

Field	Description
	DIVPRES[27]=1 System DIVSLOW is present. DIVPRES[28]=1 System DIVBUS is present DIVPRES[31]=1 System DIVCORE is present
26–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKPRES	Clock Present Indicates which clock sources are present in this instance of SCG. Any bits not defined in this bit field are Reserved and always has the value 0 when read. CLKPRES[0] Reserved CLKPRES[1]=1System OSC (SOSC) is present CLKPRES[2]=1Slow IRC (SIRC) is present CLKPRES[3]=1Fast IRC (FIRC) is present CLKPRES[6]=1System PLL (SPLL) is present

19.3.3 Clock Status Register (SCG_CSR)

This register returns the currently configured system clock source and the system clock dividers for the core (DIVCORE) and peripheral interface clock (DIVSLOW). The SCG_CSR reflects the configuration set by one of three clock control registers SCG_RCCR, SCG_VCCR, SCG_HCCR.

Note: Writing to this register will result in a transfer error.

Address: 4006_4000h base + 10h offset = 4006_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	0				SCS				0				DIVCORE				0				0				DIVBUS				DIVSLOW			
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	*	*	*	*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

* Notes:

- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The valid reset values are div-by-1 or div-by-2 when resetting into RUN mode or div-by-4 or div-by-8 when resetting into VLPR mode.

SCG_CSR 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 SCS	System Clock Source

Table continues on the next page...

SCG_CSR field descriptions (continued)

Field	Description
	Returns the currently configured clock source generating the system clock. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio If SPLL is selected as system clock source, the maximum DIVCORE value is Divide-by-4. 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 DIVBUS	Bus Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10

Table continues on the next page...

SCG_CSR field descriptions (continued)

Field	Description
	1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
DIVSLOW	Slow Clock Divide Ratio 0000 Reserved 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Reserved 1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

19.3.4 Run Clock Control Register (SCG_RCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in Run mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in RUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in RUN, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4006_4000h base + 14h offset = 4006_4014h

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

* Notes:

- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The two valid reset values are div-by-1 and div-by-2

SCG_RCCR 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 SCS	System Clock Source Selects the clock source generating the system clock in Run mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in Run mode requires that clock source to be enabled first and be valid before system clocks are allowed to switch to that clock source. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. Software should write 0 to these bits to maintain compatibility. This field is reserved.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 DIVBUS	Bus Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5

Table continues on the next page...

SCG_RCCR field descriptions (continued)

Field	Description
	0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
DIVSLOW	Slow Clock Divide Ratio 0000 Reserved 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Reserved 1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

19.3.5 VLPR Clock Control Register (SCG_VCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in VLPR mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in VLPR requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in VLPR, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4006_4000h base + 18h offset = 4006_4018h

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

* Notes:
- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-4 and div-by-8.

SCG_VCCR 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 SCS	System Clock Source Selects the clock source generating the system clock in VLPR mode. Attempting to select a clock that is not valid will be ignored. Selects the clock source generating the system clock. Selecting a different clock source when in VLPR mode requires that clock source to be enabled first and be valid before system clocks switch to that clock source. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3

Table continues on the next page...

SCG_VCCR field descriptions (continued)

Field	Description
	0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. Software should write 0 to these bits to maintain compatibility. This field is reserved.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 DIVBUS	Bus Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
DIVSLOW	Slow Clock Divide Ratio 0000 Reserved 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Reserved

Table continues on the next page...

SCG_VCCR field descriptions (continued)

Field	Description
	1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

19.3.6 HSRUN Clock Control Register (SCG_HCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in HSRUN mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in HSRUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in HSRUN, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4006_4000h base + 1Ch offset = 4006_401Ch

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

SCG_HCCR 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 SCS	System Clock Source Selects the clock source generating the system clock in HSRUN mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in HSRUN mode will enable that clock source and switch to that clock mode when it is valid. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved

Table continues on the next page...

SCG_HCCR 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 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. Software should write 0 to these bits to maintain compatibility. This field is reserved.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 DIVBUS	Bus Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
DIVSLOW	Slow Clock Divide Ratio 0000 Reserved 0001 Divide-by-2 0010 Divide-by-3

Table continues on the next page...

SCG_HCCR field descriptions (continued)

Field	Description
	0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Reserved 1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

19.3.7 SCG CLKOUT Configuration Register (SCG_CLKOUTCNFG)

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

Address: 4006_4000h base + 20h offset = 4006_4020h

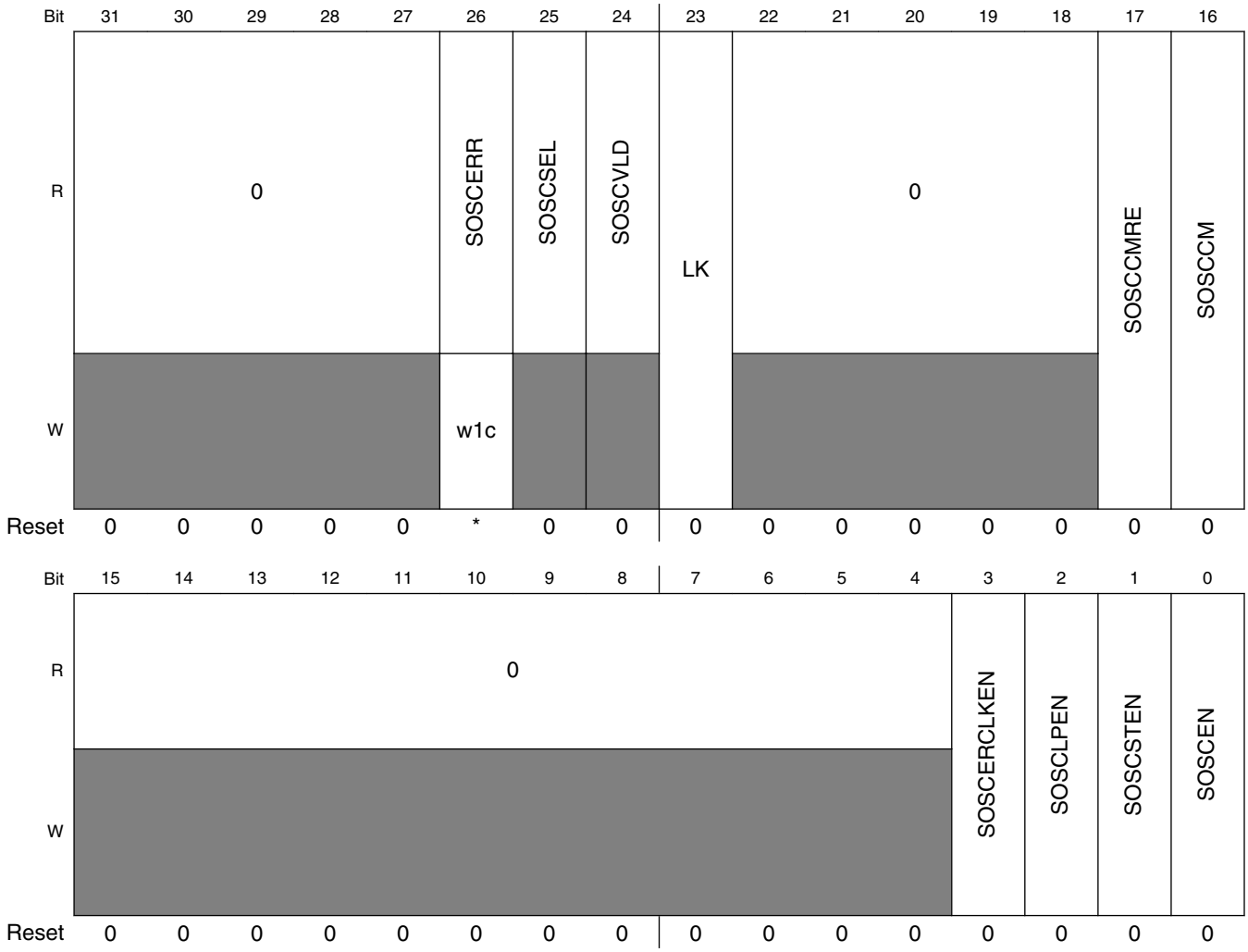
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CLKOUTSEL				0																							
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCG_CLKOUTCNFG 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 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000 SCG SLOW Clock 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved 1111 Reserved
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

19.3.8 System OSC Control Status Register (SCG_SOSCCSR)

Address: 4006_4000h base + 100h offset = 4006_4100h



* Notes:
- SOSCERR field: This flag is reset on Chip POR only

SCG_SOSCCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SOSCERR	System OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0 System OSC Clock Monitor is disabled or has not detected an error 1 System OSC Clock Monitor is enabled and detected an error

Table continues on the next page...

SCG_SOSCCSR field descriptions (continued)

Field	Description
25 SOSCSEL	System OSC Selected 0 System OSC is not the system clock source 1 System OSC is the system clock source
24 SOSCVLD	System OSC Valid The SOSC is considered valid after 4096 xtal counts. 0 System OSC is not enabled or clock is not valid 1 System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0 This Control Status Register can be written. 1 This Control Status Register cannot be written.
22–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SOSCCMRE	System OSC Clock Monitor Reset Enable 0 Clock Monitor generates interrupt when error detected 1 Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enables the clock monitor when SOSCVLD is set. If the clock source is disabled in a low power mode then the clock monitor is also disabled in the low power mode. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. 0 System OSC Clock Monitor is disabled 1 System OSC Clock Monitor is enabled
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SOSCERCLKEN	System OSC 3V ERCLK Enable SOSCERCLKEN is required for stop modes. 0 System OSC 3V ERCLK output clock is disabled. 1 System OSC 3V ERCLK output clock is enabled when SYSOSC is enabled.
2 SOSCLPEN	System OSC Low Power Enable SOSCLPEN is required for low power modes. In VLPS mode (low power stop mode), if you want the clock to remain ON, then both SOSCLPEN and SOSCSTEN bits must be enabled. 0 System OSC is disabled in VLP modes 1 System OSC is enabled in VLP modes
1 SOSCSTEN	System OSC Stop Enable 0 System OSC is disabled in Stop modes 1 System OSC is enabled in Stop modes if SOSCEN=1.
0 SOSCEN	System OSC Enable

Table continues on the next page...

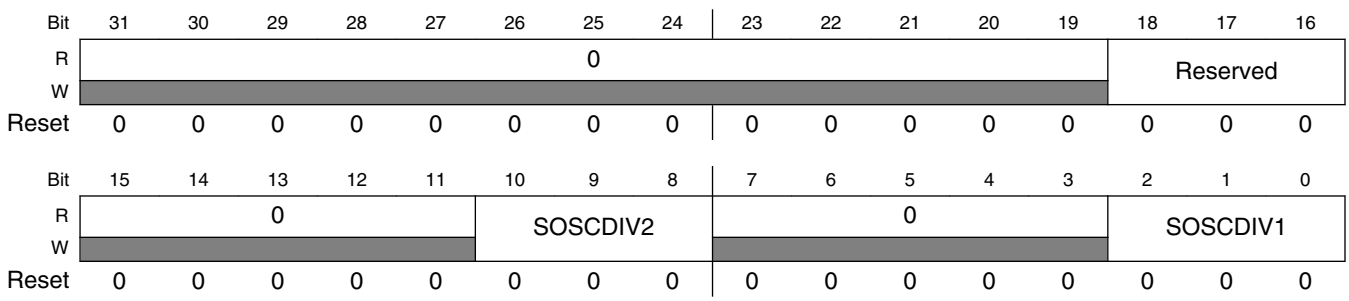
SCG_SOSCCSR field descriptions (continued)

Field	Description
	If this bit written during clock switching, it should be read back and confirmed before proceeding. 0 System OSC is disabled 1 System OSC is enabled

19.3.9 System OSC Divide Register (SCG_SOSCDIV)

The SCG_SOSCDIV register provides the control of 2 clock trees which can be used to provide optional peripheral functional clocks, or alternative module clocks. Each clock tree has optional dividers of the input SOSC clock. Changes to SOSCDIV should be done when System OSC is disabled to prevent glitches to output divided clock.

Address: 4006_4000h base + 104h offset = 4006_4104h



SCG_SOSCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved. This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SOSCDIV2	System OSC Clock Divide 2 Clock divider 2 for System OSC. Used by modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

Table continues on the next page...

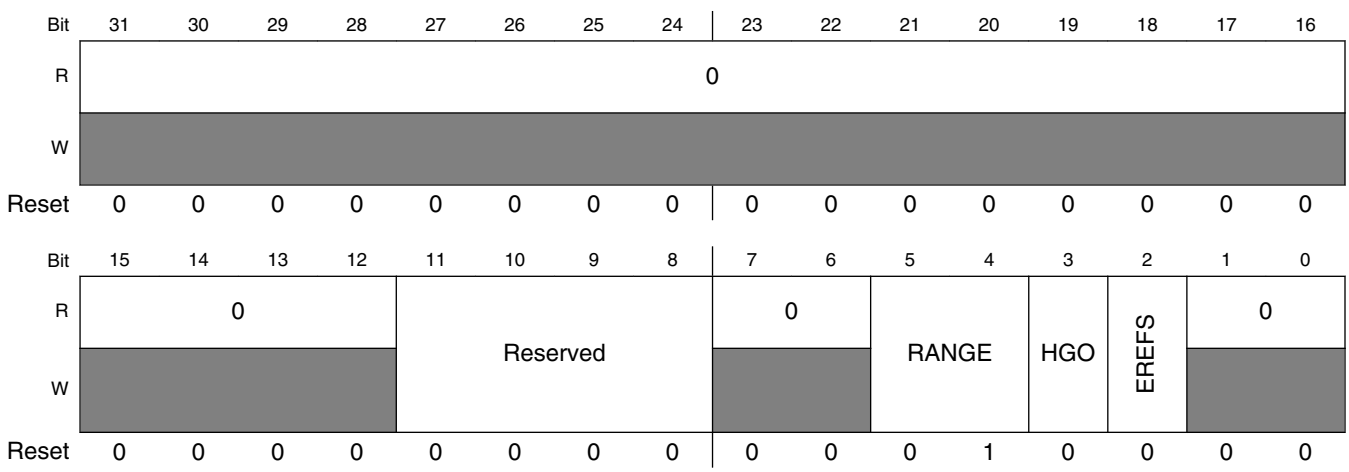
SCG_SOSCDIV field descriptions (continued)

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SOSCDIV1	System OSC Clock Divide 1 Clock divider 1 for System OSC. Used to generate the clock source for modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

19.3.10 System Oscillator Configuration Register (SCG_SOSCCFG)

The SOSCCFG register cannot be changed when the System OSC is enabled. When the System OSC is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4006_4000h base + 108h offset = 4006_4108h



SCG_SOSCCFG field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This bit is reserved. Software should write 0 to this bit field.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

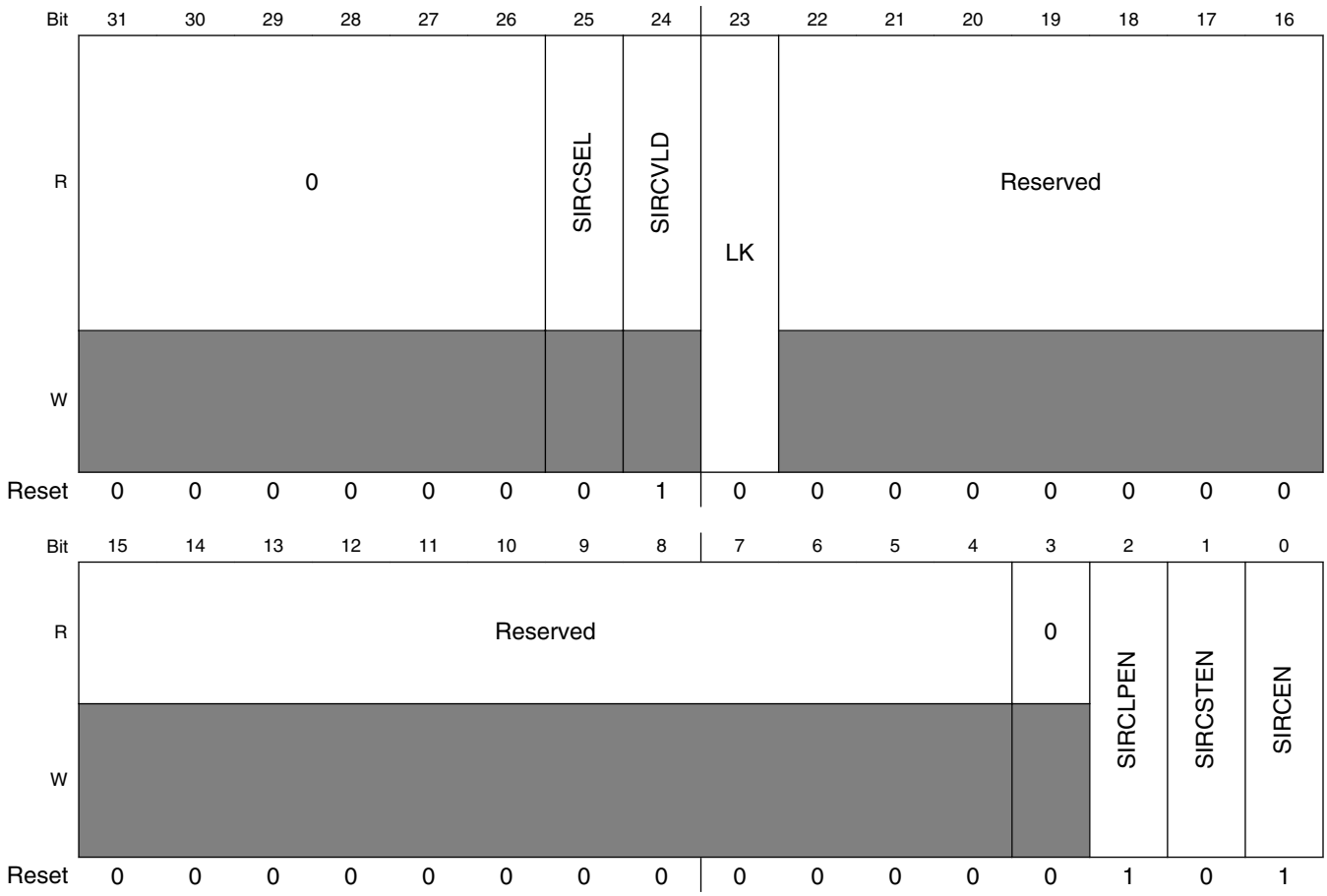
Table continues on the next page...

SCG_SOSCCFG field descriptions (continued)

Field	Description
5–4 RANGE	System OSC Range Select Selects the frequency range for the system crystal oscillator (OSC) **See chip-specific information for supported crystal oscillator ranges.** 00 Reserved 01 Low frequency range selected for the crystal oscillator 10 Medium frequency range selected for the crytstal oscillator 11 High frequency range selected for the crystal oscillator
3 HGO	High Gain Oscillator Select Controls the crystal oscillator power mode of operations. 0 Configure crystal oscillator for low-gain operation 1 Configure crystal oscillator for high-gain operation
2 EREFS	External Reference Select Selects the source for the external reference clock. This bit selects which clock is output from the System OSC (SOSC) into the SCG, thus either the crystal oscillator or from an external clock input 0 External reference clock selected 1 Internal crystal oscillator of OSC selected.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

19.3.11 Slow IRC Control Status Register (SCG_SIRCCSR)

Address: 4006_4000h base + 200h offset = 4006_4200h



SCG_SIRCCSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 SIRCSEL	Slow IRC Selected 0 Slow IRC is not the system clock source 1 Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0 Slow IRC is not enabled or clock is not valid 1 Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time.

Table continues on the next page...

SCG_SIRCCSR field descriptions (continued)

Field	Description
	0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–4 Reserved	This field is reserved and is always has the value 0 This field is reserved.
3 Reserved	This field is reserved and is always has the value 0 This field is reserved. This read-only field is reserved and always has the value 0.
2 SIRCLPEN	Slow IRC Low Power Enable 0 Slow IRC is disabled in VLP modes 1 Slow IRC is enabled in VLP modes
1 SIRCSTEN	Slow IRC Stop Enable 0 Slow IRC is disabled in supported Stop modes 1 Slow IRC is enabled in supported Stop modes
0 SIRCEN	Slow IRC Enable If this bit written during clock switching, it should be read back and confirmed before proceeding. 0 Slow IRC is disabled 1 Slow IRC is enabled

19.3.12 Slow IRC Divide Register (SCG_SIRCDIV)

To prevent glitches to the output divided clock, change SIRDIV when the Slow IRC is disabled.

Address: 4006_4000h base + 204h offset = 4006_4204h

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

SCG_SIRCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved. This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SIRCDIV2	Slow IRC Clock Divide 2

Table continues on the next page...

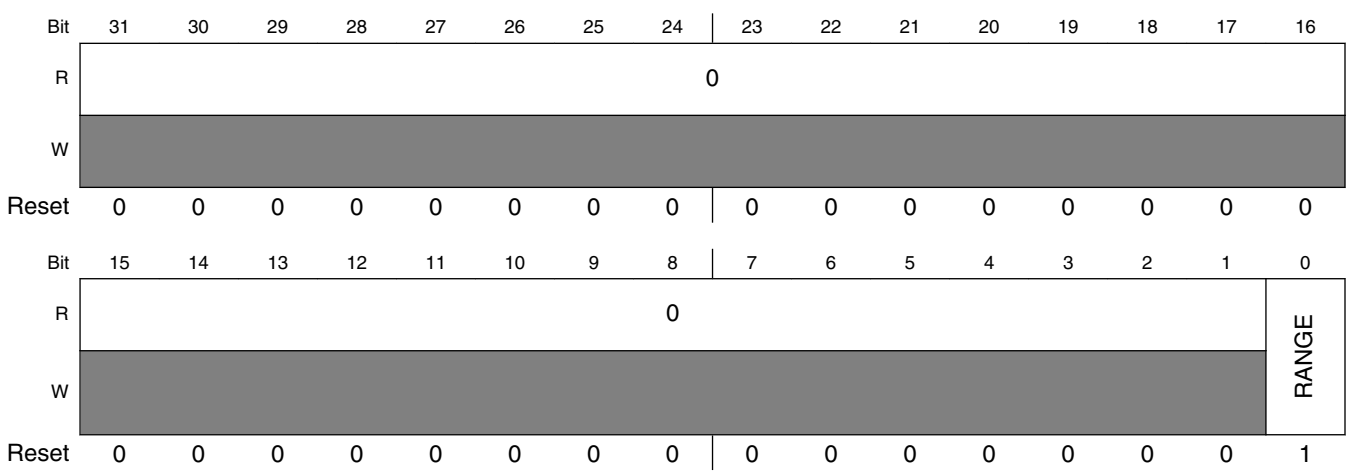
SCG_SIRCDIV field descriptions (continued)

Field	Description
	Clock divider 2 for Slow IRC. Used by modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SIRCDIV1	Slow IRC Clock Divide 1 Clock divider 1 for Slow IRC. Used to generate the clock source for modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

19.3.13 Slow IRC Configuration Register (SCG_SIRCCFG)

The SIRCCFG register cannot be changed when the slow IRC clock is enabled. When the slow IRC clock is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4006_4000h base + 208h offset = 4006_4208h

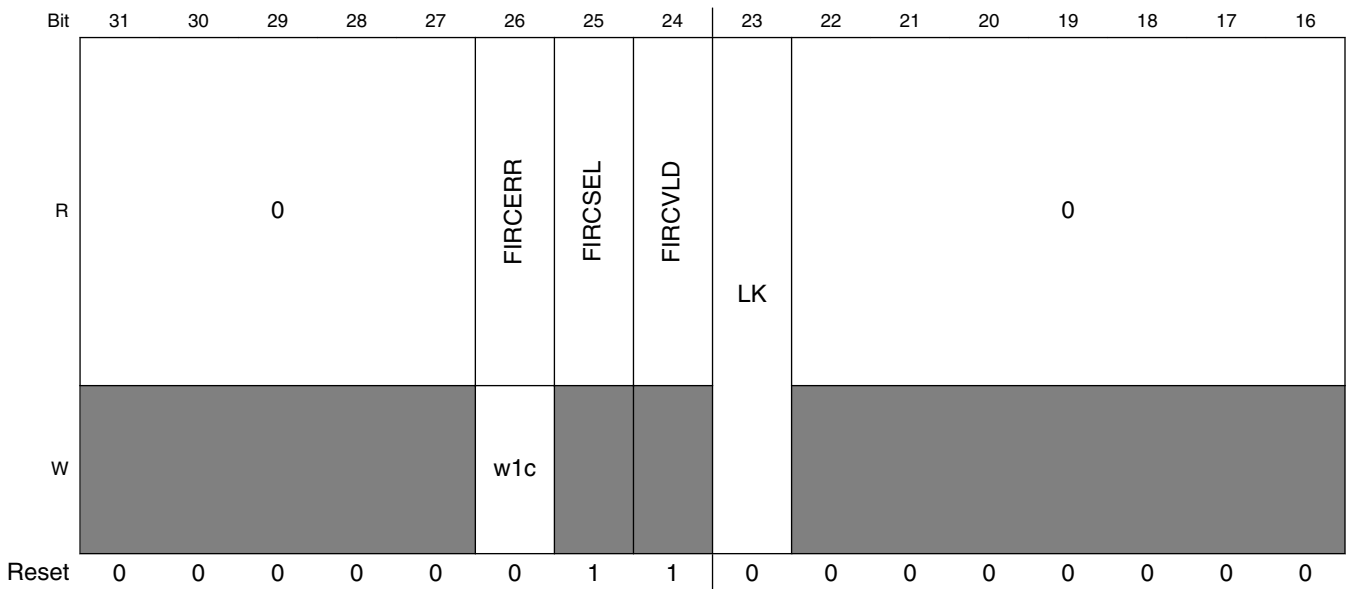


SCG_SIRCCFG field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 RANGE	Frequency Range **See chip-specific information for supported frequency ranges.** 0 Slow IRC low range clock (2 MHz) 1 Slow IRC high range clock (8 MHz)

19.3.14 Fast IRC Control Status Register (SCG_FIRCCSR)

Address: 4006_4000h base + 300h offset = 4006_4300h



SCG_FIRCCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 FIRCERR	Fast IRC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0 Error not detected with the Fast IRC trimming. 1 Error detected with the Fast IRC trimming.
25 FIRCSEL	Fast IRC Selected status 0 Fast IRC is not the system clock source 1 Fast IRC is the system clock source
24 FIRCVLD	Fast IRC Valid status 0 Fast IRC is not enabled or clock is not valid. 1 Fast IRC is enabled and output clock is valid. The clock is valid once there is an output clock from the FIRC analog.
23 LK	Lock Register This bit field can be cleared/set at any time. 0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 FIRCTRUP	Fast IRC Trim Update 0 Disable Fast IRC trimming updates 1 Enable Fast IRC trimming updates
8 FIRCTREN	Fast IRC Trim Enable 0 Disable trimming Fast IRC to an external clock source 1 Enable trimming Fast IRC to an external clock source
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FIRCREGOFF	Fast IRC Regulator Enable 0 Fast IRC Regulator is enabled. 1 Fast IRC Regulator is disabled.
2 FIRCLPEN	Fast IRC Low Power Enable 0 Fast IRC is disabled in VLP modes 1 Fast IRC is enabled in VLP modes
1 FIRCSTEN	Fast IRC Stop Enable 0 Fast IRC is disabled in Stop modes. When selected as the reference clock to the System PLL and if the System PLL is enabled in STOP mode, the Fast IRC will stay enabled even if FIRCSTEN=0. 1 Fast IRC is enabled in Stop modes
0 FIRCEN	Fast IRC Enable

Table continues on the next page...

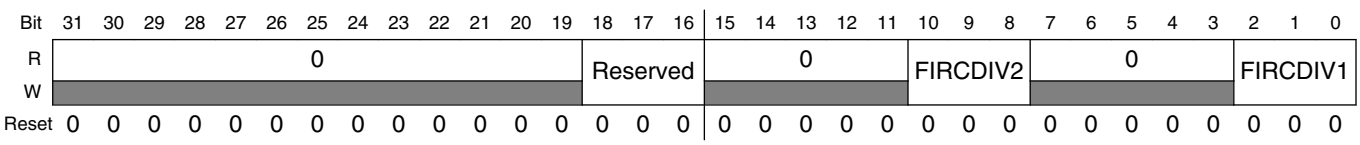
SCG_FIRCCSR field descriptions (continued)

Field	Description
	If this bit written during clock switching, it should be read back and confirmed before proceeding. FIRCEN should not toggle when FIRCTREN = 1. 0 Fast IRC is disabled 1 Fast IRC is enabled

19.3.15 Fast IRC Divide Register (SCG_FIRCDIV)

Changes to FIRCDIV should be done when FAST IRC is disabled to prevent glitches to output divided clock.

Address: 4006_4000h base + 304h offset = 4006_4304h



SCG_FIRCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved. This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FIRCDIV2	Fast IRC Clock Divide 2 Clock divider 2 for the Fast IRC. Used by modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FIRCDIV1	Fast IRC Clock Divide 1 Clock divider 1 for Fast IRC. Used to generate the clock source for modules that need an asynchronous clock source. 000 Output disabled

Table continues on the next page...

SCG_FIRCDIV field descriptions (continued)

Field	Description
	001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

19.3.16 Fast IRC Configuration Register (SCG_FIRCCFG)

The FIRCCFG register cannot be changed when the Fast IRC is enabled. When the Fast IRC is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4006_4000h base + 308h offset = 4006_4308h

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														RANGE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

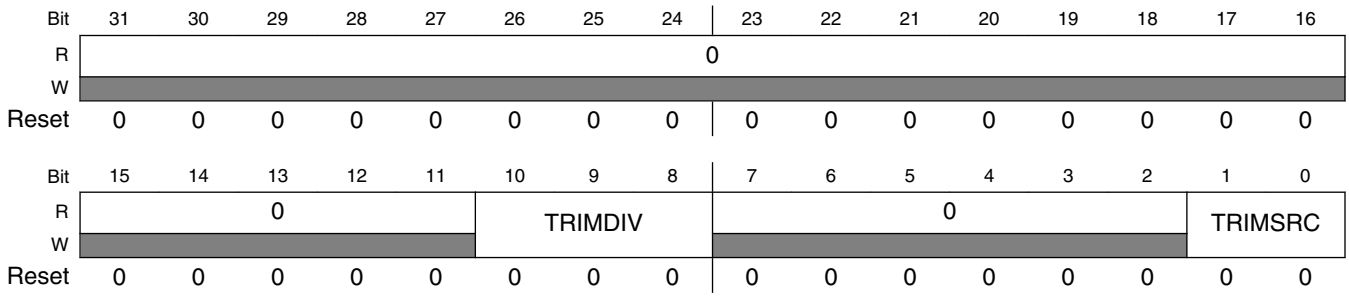
SCG_FIRCCFG field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RANGE	Frequency Range **See chip-specific information for supported frequency ranges.** 00 Fast IRC is trimmed to 48 MHz 01 Reserved 10 Reserved 11 Reserved

19.3.17 Fast IRC Trim Configuration Register (SCG_FIRCTCFG)

The FIRCTCFG register cannot be changed when Fast IRC tuning is enabled. When the Fast IRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4006_4000h base + 30Ch offset = 4006_430Ch



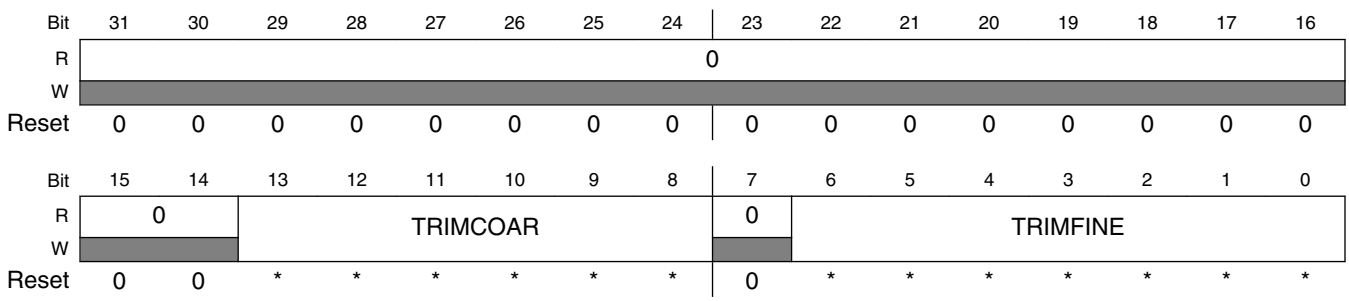
SCG_FIRCTCFG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 TRIMDIV	Fast IRC Trim Predivide Divide the System OSC down for Fast IRC trimming. 000 Divide by 1 001 Divide by 128 010 Divide by 256 011 Divide by 512 100 Divide by 1024 101 Divide by 2048 110 Reserved. Writing this value will result in Divide by 1. 111 Reserved. Writing this value will result in a Divide by 1.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRIMSRC	Trim Source Configures the external clock source to tune the Fast IRC. TRMSRC must be configured before programming FIRCSTAT register for trim update 00 Reserved 01 Reserved 10 System OSC 11 Reserved

19.3.18 Fast IRC Status Register (SCG_FIRCSTAT)

This register is loaded from IFR during reset. These register gets uploaded with the trim values generated by FIRC auto trimming which is enabled when FIRC is enabled and FIRCTREN=1 and FIRCTRUP=1. When FIRC auto trimming is enabled and FIRCTRUP is off (Note: TRIMSRC needs to be programmed to TRIMSRC=10 or TRIMSRC=11), writes to this register is allowed and values written to this register are used to trim FIRC clock.

Address: 4006_4000h base + 318h offset = 4006_4318h



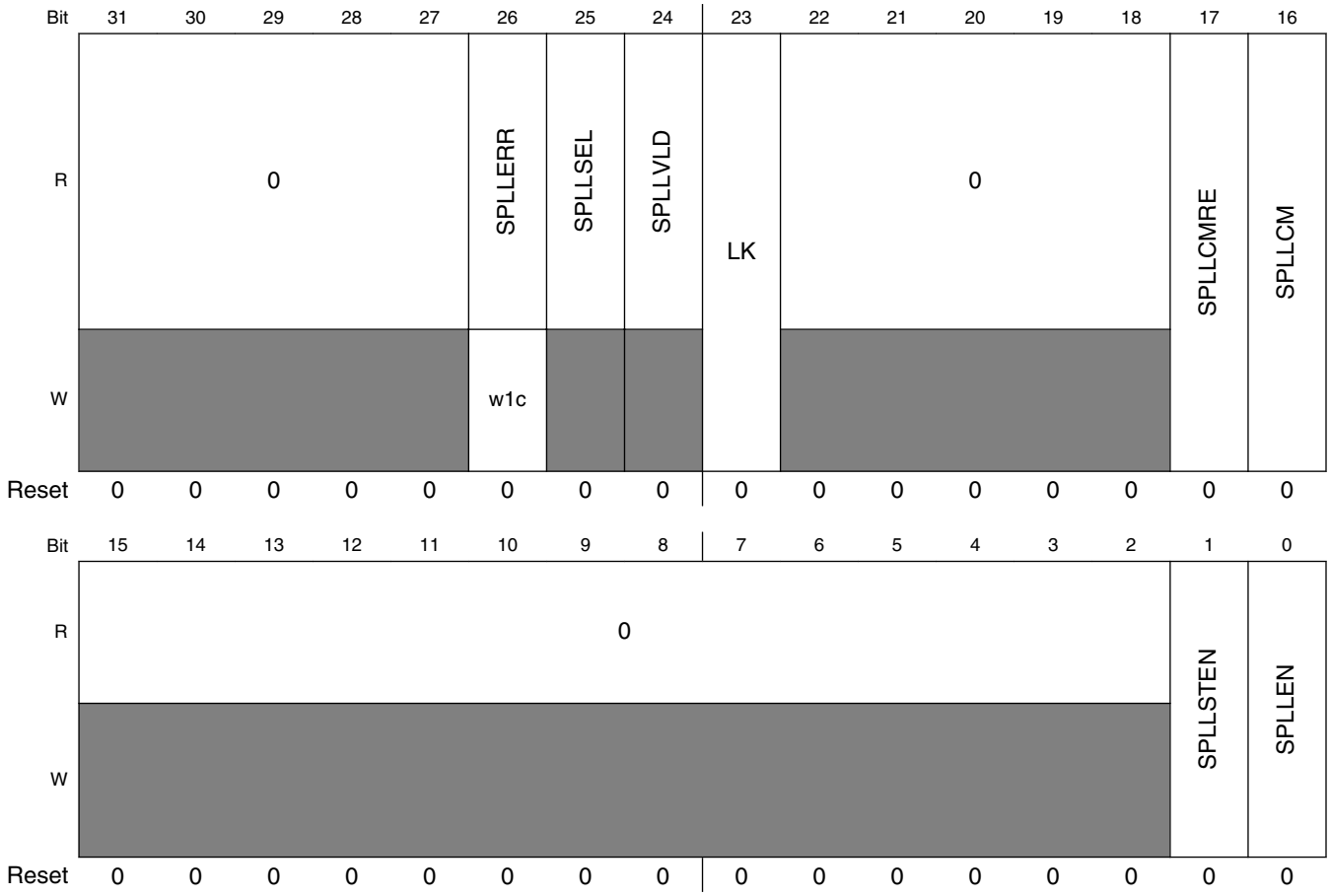
* Notes:
- TRIMCOAR field: Reset values are loaded out of IFR.
- TRIMFINE field: Reset values are loaded out of IFR.

SCG_FIRCSTAT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coursely trim the Fast IRC Clock to within approximately ±0.7% of the target frequency. When FIRC is enabled and auto trimming is enabled (FIRCTREN=1 and FIRCTRUP=1), then TRIMCOAR register gets uploaded with the trimmed coarse value. When FIRCTRUP=0, TRIMCOAR bitfield is writable, to allow user programming of coarse trim values.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRIMFINE	Trim Fine Status Once the Fast IRC Clock is trimmed to ±0.7% of the target frequency using the TRIMCOAR bits, the TRIMFINE bits can be used to trim the Fast IRC Clock to within ±0.04% of the target frequency. When FIRC is enabled and auto trimming is enabled (FIRCTREN=1 and FIRCTRUP=1), TRIMFINE register gets uploaded with the trimmed fine value. When FIRCTRUP=0, TRIMFINE bitfield is writeable, to allow user programming of fine trim values.

19.3.19 System PLL Control Status Register (SCG_SPLLCSR)

Address: 4006_4000h base + 600h offset = 4006_4600h



SCG_SPLLCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SPLLERR	System PLL Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one **NOTE:** The LOL Flag is set when the PLL reference is out of range (Dunl in datasheet) and is constantly modulated such that 3 consecutive un-locked samples of the reference clock are generated. 0 System PLL Clock Monitor is disabled or has not detected an error 1 System PLL Clock Monitor is enabled and detected an error. System PLL Clock Error flag will not set when System OSC is selected as its source and SOSCERR has set.
25 SPLLSEL	System PLL Selected 0 System PLL is not the system clock source 1 System PLL is the system clock source

Table continues on the next page...

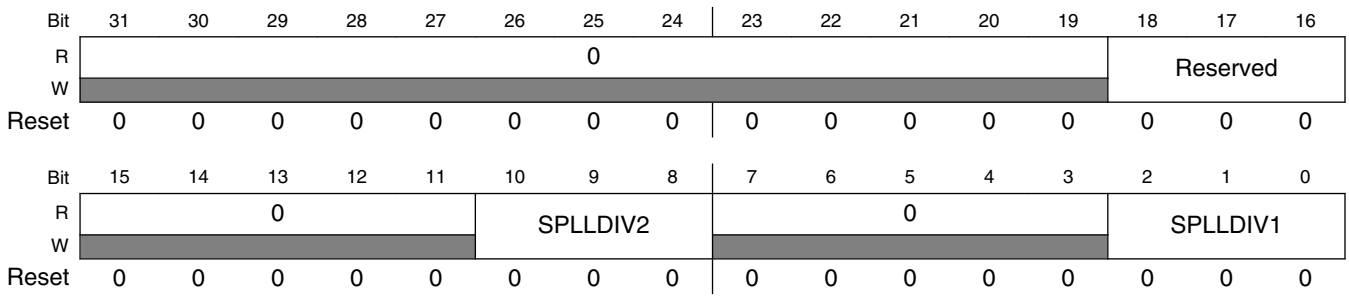
SCG_SPLLCSR field descriptions (continued)

Field	Description
24 SPLLVLD	System PLL Valid Indicates when the SPLL clock is valid. When the System PLL (SPLL) is disabled, the System PLL Valid bit (SPLLVLD) will clear without causing the System PLL Clock Error bit (SPLLERR) to get set. In a similar way, if the System PLL (SPLL) is using the System Oscillator (SOSC) as its reference clock, and a System OSC Clock Error (SOSCCSR[SOSCERR]) is detected, then the System PLL Valid bit (SPLLVLD) will clear without asserting a System PLL Clock Error (SPLLERR). Lock detect is determined by a lock detect circuit. Three samples of lock detect determines whether or not the clock is valid. **NOTE:** The System PLL Valid bit (SPLLVLD) should only be used to verify that the SPLL is locked after initialization. To monitor the SPLL clock, ensure that the System PLL Clock Monitor is enabled, using the System PLL Clock Monitor bit (SPLLCM). 0 System PLL is not enabled or clock is not valid 1 System PLL is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SPLLCMRE	System PLL Clock Monitor Reset Enable 0 Clock Monitor generates interrupt when error detected 1 Clock Monitor generates reset when error detected
16 SPLLCM	System PLL Clock Monitor Enables the clock monitor, if the clock source is disabled in a low power mode then the clock monitor is also disabled in the low power mode. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. 0 System PLL Clock Monitor is disabled 1 System PLL Clock Monitor is enabled
15–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 SPLLSTEN	System PLL Stop Enable 0 System PLL is disabled in Stop modes 1 System PLL is enabled in Stop modes
0 SPLLEN	System PLL Enable **NOTE:** If this bit written during clock switching, it should be read back and confirmed before proceeding. As the device exits reset, the SCG_RCCR register should be configured as per the supported frequency ranges of the device BEFORE enabling the SPLL (SPLLEN =1). 0 System PLL is disabled 1 System PLL is enabled

19.3.20 System PLL Divide Register (SCG_SPLLDIV)

Changes to SPLLDIV should be done when System PLL is disabled to prevent glitches to output divided clock.

Address: 4006_4000h base + 604h offset = 4006_4604h



SCG_SPLLDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved. This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SPLLDIV2	System PLL Clock Divide 2 Clock divider 2 for System PLL. Used by modules that need an asynchronous clock source. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SPLLDIV1	System PLL Clock Divide 1 Clock divider 1 for System PLL. Used to generate the clock source for modules that need an asynchronous clock source. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8

Table continues on the next page...

SCG_SPLLDIV field descriptions (continued)

Field	Description
	101 Divide by 16 110 Divide by 32 111 Divide by 64

19.3.21 System PLL Configuration Register (SCG_SPLLCFG)

The SPLLCFG register cannot be changed when the System PLL is enabled. When the System PLL is enabled, writes to this register are ignored, and there is no transfer error.

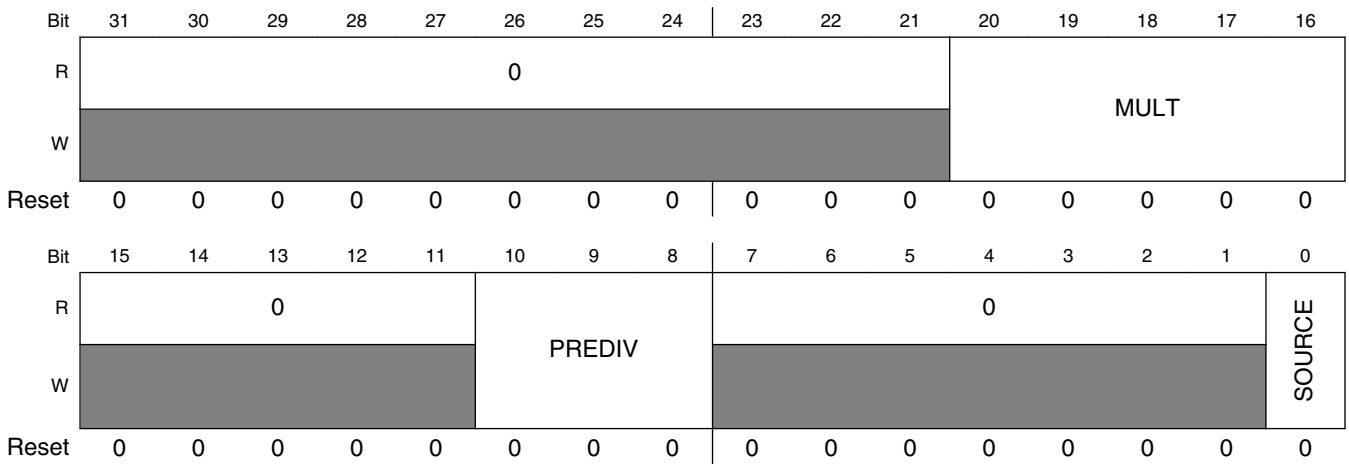
The below information applies to VCO_CLK.

The SPLL_CLK = (VCO_CLK)/2

The VCO_CLK = SPLL_SOURCE/(PREDIV + 1) X (MULT + 16)

SPLL_SOURCE is the clock source selected from the SOURCE bitfield of this register.

Address: 4006_4000h base + 608h offset = 4006_4608h



SCG_SPLLCFG field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 MULT	System PLL Multiplier

Table continues on the next page...

SCG_SPLLCFG field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>Multiplier for the System PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency.

Table 19-1. PLL VCO Multiply Factor
<table><tr><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td></tr><tr><td>00000</td><td>16</td><td></td><td>01000</td><td>24</td><td></td><td>10000</td><td>32</td><td></td><td>11000</td><td>40</td></tr><tr><td>00001</td><td>17</td><td></td><td>01001</td><td>25</td><td></td><td>10001</td><td>33</td><td></td><td>11001</td><td>41</td></tr><tr><td>00010</td><td>18</td><td></td><td>01010</td><td>26</td><td></td><td>10010</td><td>34</td><td></td><td>11010</td><td>42</td></tr><tr><td>00011</td><td>19</td><td></td><td>01011</td><td>27</td><td></td><td>10011</td><td>35</td><td></td><td>11011</td><td>43</td></tr><tr><td>00100</td><td>20</td><td></td><td>01100</td><td>28</td><td></td><td>10100</td><td>36</td><td></td><td>11100</td><td>44</td></tr><tr><td>00101</td><td>21</td><td></td><td>01101</td><td>29</td><td></td><td>10101</td><td>37</td><td></td><td>11101</td><td>45</td></tr><tr><td>00110</td><td>22</td><td></td><td>01110</td><td>30</td><td></td><td>10110</td><td>38</td><td></td><td>11110</td><td>46</td></tr><tr><td>00111</td><td>23</td><td></td><td>01111</td><td>31</td><td></td><td>10111</td><td>39</td><td></td><td>11111</td><td>47</td></tr></table></td></tr>
<tr><td>15–11
Reserved</td><td>This field is reserved.
This read-only field is reserved and always has the value 0.</td></tr>
<tr><td>10–8
PREDIV</td><td>PLL Reference Clock Divider

Selects the amount to divide down the reference clock for the System PLL. The resulting frequency must be in the range specified in the datasheet.

Table 19-2. System PLL Reference Divide Factor
<table><tr><td>PREDIV</td><td>Divide Factor</td></tr><tr><td>000</td><td>1</td></tr><tr><td>001</td><td>2</td></tr><tr><td>010</td><td>3</td></tr><tr><td>011</td><td>4</td></tr><tr><td>100</td><td>5</td></tr><tr><td>101</td><td>6</td></tr><tr><td>110</td><td>7</td></tr><tr><td>111</td><td>8</td></tr></table></td></tr>
<tr><td>7–1
Reserved</td><td>This field is reserved.
This read-only field is reserved and always has the value 0.</td></tr>
<tr><td>0
SOURCE</td><td>Clock Source

Configures the input clock source for the System PLL.

0 System OSC (SOSC)
1 Fast IRC (FIRC)</td></tr>
</table>

19.4 Functional description

19.4.1 SCG Clock Mode Transitions

The following figure shows the valid clock mode transitions supported by SCG.

Slow IRC (SIRC) boot mode is not supported on this device.

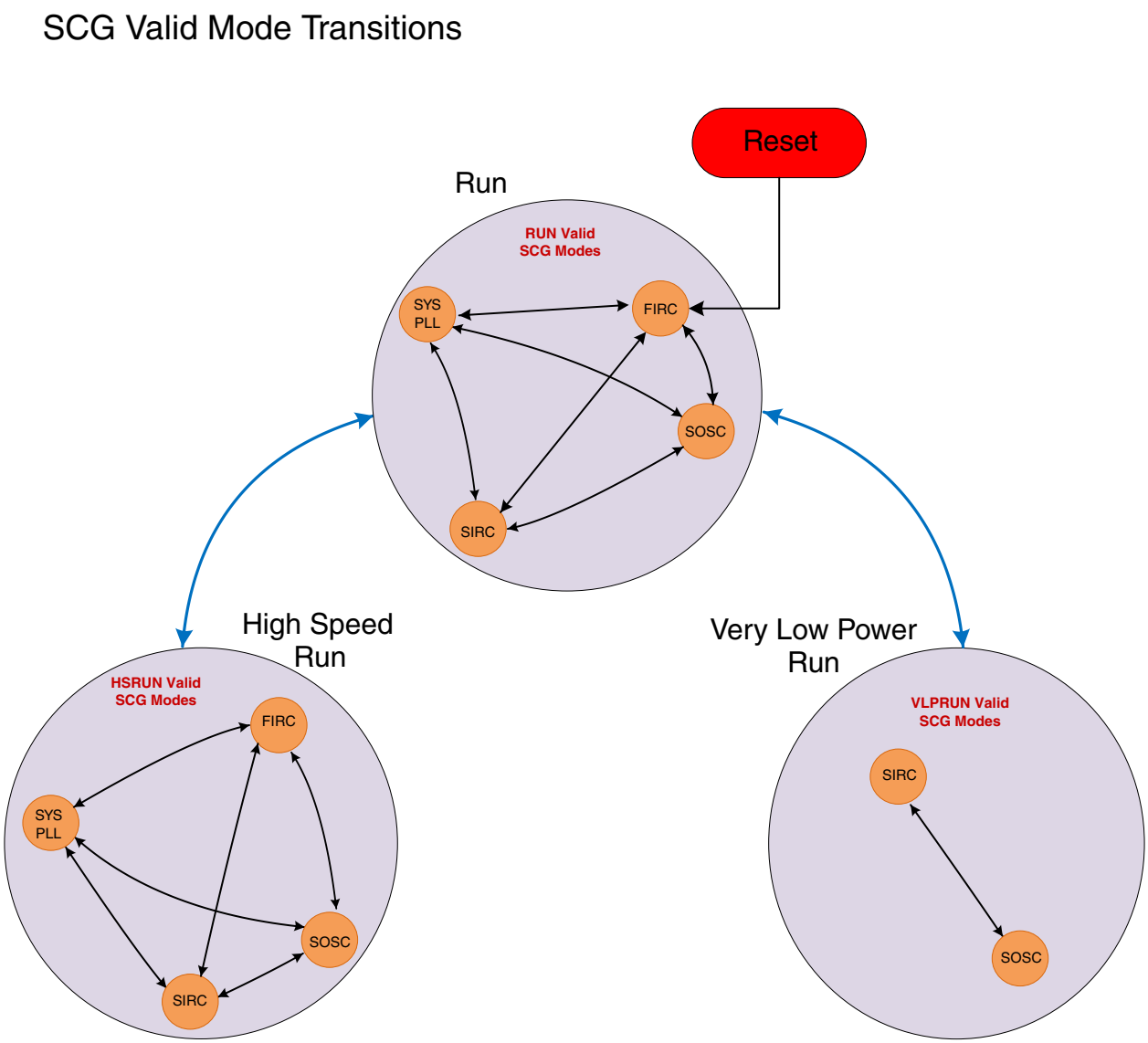


Figure 19-2. SCG Valid Mode Transition Diagram

NOTE

When a transition between run modes (RUN, HSRUN, VLRUN) is required, the SCG should complete the switch to the clock mode as defined in the SCG clock control register first. Once the switch to the clock mode is completed, the system can then initiate the request for the selected run mode.

For example, if a transition from RUN mode to VLRUN is required, first complete any required clock change. Initiate the VLRUN request **after** the clock change has completed.

The power modes are chip specific. For more details about power mode assignments, see power management and system mode control information.

The modes of operation listed in the following table are the valid modes for this implementation of the SCG.

Table 19-3. SCG modes of operation

Mode	Description
System Oscillator Clock (SOSC)	System Oscillator Clock (SOSC) mode is entered when all the following conditions occur: • RUN MODE: 0001 is written to RCCR[SCS]. VLRUN MODE: 0001 is written to VCCR[SCS]. HSRUN MODE: 0001 is written to HCCR[SCS]. • SOSCEN = 1 • SOSCVLD = 1 In SOSC mode, SCGCLKOUT and system clocks are derived from the external System Oscillator Clock (SOSC). Information regarding SOSC operation during normal and low power stop modes is found in the "Stop" row of this table.
Slow Internal Reference Clock (SIRC)	Slow Internal Reference Clock (SIRC) mode is entered when all the following conditions occur: • RUN MODE: 0010 is written to RCCR[SCS]. VLRUN MODE: 0010 is written to VCCR[SCS] and 1 is written to SIRCCSR[SIRCLPEN]. HSRUN MODE: 0010 is written to HCCR[SCS]. • SIRCEN = 1 • SIRCVLD = 1 In SIRC mode, SCGCLKOUT and system clocks are derived from the slow internal reference clock. Two frequency ranges are available for SIRC clock as described in the SIRCCFG[RANGE] register definition. Changes to SIRC range settings will be ignored when SIRC clock is enabled. Information regarding SIRC operation during normal and low power stop modes is found in the "Stop" row of this table.
Fast Internal Reference Clock (FIRC)	Fast Internal Reference Clock (FIRC) mode is the default clock mode of operation and is entered when all the following conditions occur:

Table continues on the next page...

Table 19-3. SCG modes of operation (continued)

Mode	Description
	• RUN MODE: 0011 is written to RCCR[SCS]. VLRUN MODE: Invalid mode. Programming SCG into FIRC mode will be ingored. HSRUN MODE: 0011 is written to HCCR[SCS]. • FIRCEN = 1 • FIRCVLD = 1 In FIRC mode, SCGCLKOUT and system clocks are derived from the fast internal reference clock. Four frequency range settings are available for FIRC clock as described in the FIRC[RANGE] register definition. Changes to FIRC range settings will be ignored when FIRC clock is enabled. Information regarding FIRC operation during normal and low power stop modes is found in the "Stop" row of this table.
Sys PLL (SPLL)	Sys PLL (SPLL) mode is entered when all the following conditions occur: • RUN MODE: 0110 is written to RCCR[SCS]. VLRUN MODE: Invalid mode. Programming SCG into SPLL mode will be ingored. HSRUN MODE: 0110 is written to HCCR[SCS]. • SPLLEN = 1 • SPLLVLD = 1 In SPLL mode, the SCGCLKOUT and system clocks are derived from the output of PLL which is controlled by either the System Oscillator (SOSC) clock or the Fast internal reference clock (FIRC). The selected PLL clock frequency locks to a multiplication factor, as specified by its corresponding SCG_SPLLCFG[MULT], times the selected PLL reference frequency. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. This divide value is defined by the SCG_SPLLCFG[PREDIV] bits. Information regarding SPLL operation during normal and low power stop modes is found in the "Stop" row of this table.
Stop	Entered whenever the MCU enters a Stop state. The power modes are chip specific. For power mode assignments, see the chapter that describes how modules are configured and SCG behaviour during Stop recovery. Entering Stop mode, all SCG clock signals are static except the following clocks which can continue to run and stay enabled in the following cases: SIRCCLK is available in Normal Stop and VLPS mode when all the following conditions become true: • SIRCCSR[SIRCEN] = 1 • SIRCCSR[SIRCSTEN] = 1 • SIRCCSR[SIRCLPEN] = 1 in VLPS FIRCCLK is available only in Normal Stop mode when all the following conditions become true: • FIRCCSR[FIRCEN] = 1 • FIRCCSR[FIRCSTEN] = 1 SOSCLK is available in following low power stop modes (Normal Stop, VLPS) when all the below conditions are true. • SOSCCSR[SOSCEN] = 1

Table 19-3. SCG modes of operation

Mode	Description
	• SOSCCSR[SOSCSTEN] = 1 • SOSCCSR[SOSCLPEN] = 1 (required only for Low Power Stop modes (VLPS) SPLLCLK is available in Normal Stop mode when all the following conditions are true: • SPLLCSR[SPLLEN] = 1 • SPLLCSR[SPLLSTEN] = 1 • SPLLSTEN control bit has no affect in VLPS Power mode.

Chapter 20
RTC Oscillator (OSC32)

20.1 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.

20.1.1 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
- Automatic Gain Control (AGC) to optimize power consumption

The RTC oscillator operations are described in detail in Functional Description .

20.1.2 Block Diagram

The following is the block diagram of the RTC oscillator.

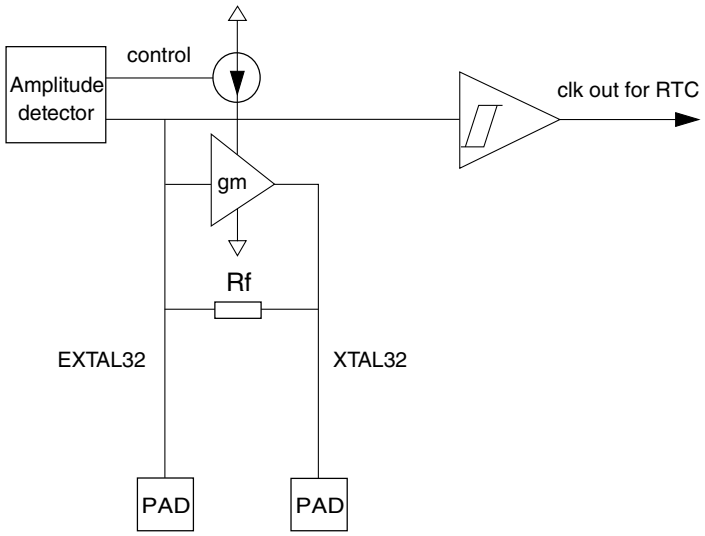


Figure 20-1. RTC Oscillator Block Diagram

20.2 RTC Signal Descriptions

The following table shows the user-accessible signals available for the RTC oscillator. See the Pinouts or Signal Multiplexing and Pin Assignment section to find out which signals are actually connected to the external pins.

Table 20-1. RTC Signal Descriptions

Signal	Description	I/O
EXTAL32	Oscillator Input	I
XTAL32	Oscillator Output	O

20.2.1 EXTAL32 — Oscillator Input

This signal is the analog input of the RTC oscillator.

20.2.2 XTAL32 — Oscillator Output

This signal is the analog output of the RTC oscillator module.

20.3 External Crystal Connections

The connections with a crystal is shown in the following figure. External load capacitors are required based on crystal parameters, but feedback resistor is not required.

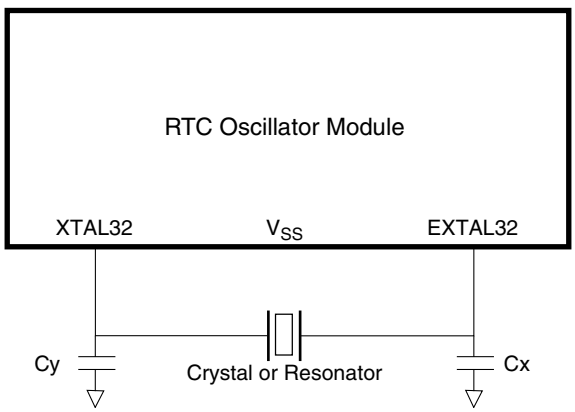


Figure 20-2. Crystal Connections

20.4 Memory Map/Register Descriptions

The following section shows the memory map and explains the register.

OSC32 memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_0000	RTC Oscillator Control Register (OSC32_CR)	8	R/W	00h	20.4.1/487

20.4.1 RTC Oscillator Control Register (OSC32_CR)

Address: 4006_0000h base + 0h offset = 4006_0000h

Bit	7	6	5	4	3	2	1	0
Read	ROSCEN	ROSCSTPEN	ROSCSTB	ROSCEREFS	0			0
Write								
Reset	0	0	0	0	0	0	0	0

OSC32_CR field descriptions

Field	Description
7 ROSCEN	RTC 32k Oscillator (OSC32) module enable This bit is used to enable the RTC 32k VLP Oscillator (OSC32) module. **NOTE: It is necessary to set this bit even when the external 32k clock mode (ROSCEREFS=0) is in use.** **NOTE:** If RTC_CR[OSCE] is set, this bit will be bypassed. OSC32 then works in crystal mode. 0 OSC32 module is disabled. 1 OSC32 module is enabled.
6 ROSCSTPEN	RTC 32k Oscillator stop mode enable This bit is used to enable the RTC 32k VLP Oscillator in stop mode together with ROSCEN bitfield. 0 Oscillator is disabled regardless the state of ROSCEN. 1 Oscillator is enabled in Stop mode when ROSCEN is set.
5 ROSCSTB	RTC 32k Oscillator stable flag This flag indicates when using the crystal mode if the oscillator has started up stably after 4096 cycles. 0 RTC 32k oscillator is unstable now and no clock will go out of the block. 1 RTC 32k oscillator is stable.
4 ROSCEREFS	RTC 32k Oscillator external reference clcok selection **NOTE:** If RTC_CR[OSCE] is set, this bit will be bypassed. OSC32 then works in crystal mode. 0 Bypass mode. RTC oscillator selects the external 32k clock. 1 Crystal mode.
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.

20.5 Functional Description

As shown in Figure 20-1, the module includes an amplifier which supplies the negative resistor for the RTC oscillator. The gain of the amplifier is controlled by the amplitude detector, which optimizes the power consumption. A schmitt trigger is used to translate the sine-wave generated by this oscillator to a pulse clock out, which is a reference clock for the RTC digital core.

The oscillator includes an internal feedback resistor of approximately 100 MΩ between EXTAL32 and XTAL32.

20.6 Reset Overview

There is no reset state associated with the RTC oscillator.

20.7 Interrupts

The RTC oscillator does not generate any interrupts.

Chapter 21
Peripheral Clock Controller (PCC)

21.1 Chip-specific information for this module

21.1.1 Information of PCC on this device

The clock connection information for this module is as follows.

Clock Source : SCG	Clock Source Descriptions	PCS Clock Names of PCC
SOSCDIV1_CLK / SOSCDIV2_CLK	SOSCDIV1 (or DIV2) of system OSC clock	OSCCLK
SIRCDIV1_CLK / SIRCDIV2_CLK	SIRCDIV1 (or DIV2) of slow IRC clock	SCGIRCLK
FIRCDIV1_CLK / FIRCDIV2_CLK	FIRCDIV1 (or DIV2) of fast IRC clock	SCGFIRCLK
SPLLDIV1_CLK / SPLLDIV2_CLK	SPLLDIV1 (or DIV2) of PLL clock	SCGPLLCLK

21.2 Introduction

The Peripheral Clock Control module (PCC) provides peripheral clock control and configuration registers.

In addition to the peripheral clock gates, clock multiplexers, and clock dividers, the PCC contains an interface between the peripherals and the system to control and acknowledge the Stop, Doze, and Debug signals.

21.2.1 Features

The PCC module enables software to configure the following clocking options for each peripheral:

- Clock gating
- Clock source selection
- Clock divide values

The following figure is a block diagram of the PCC clock source selection and clock gating. Some peripherals also have a clock divider available. See the peripheral's PCC register for more information.

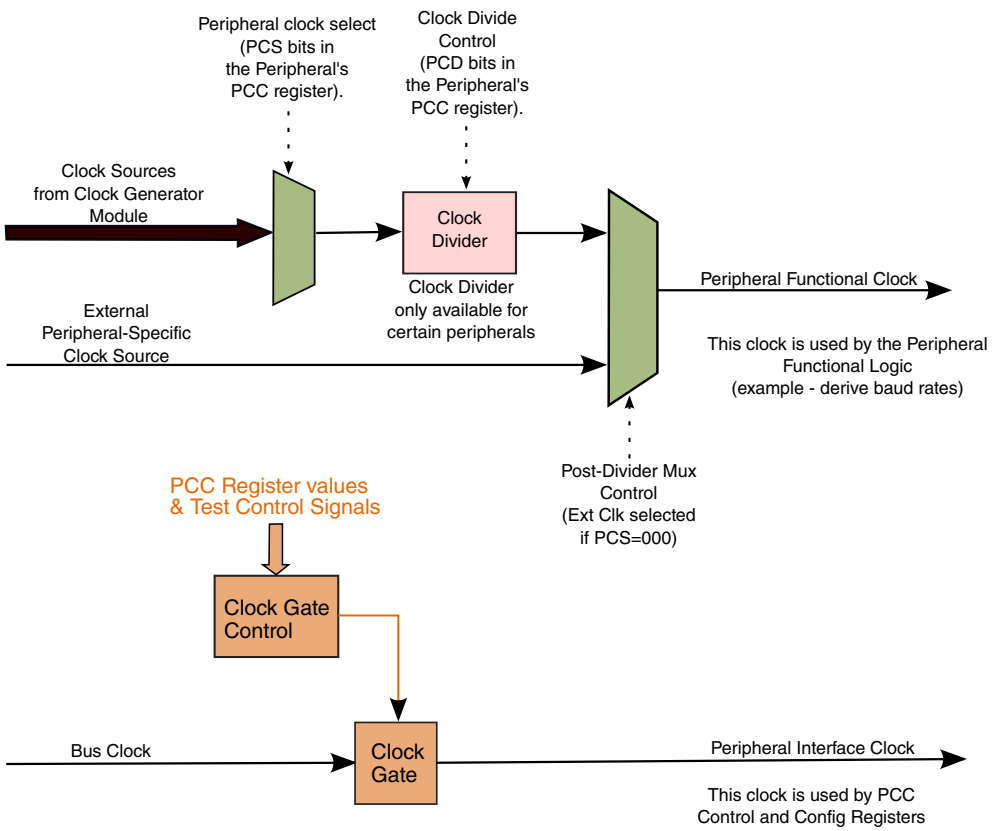


Figure 21-1. PCC Clock Source Selection and Gating

21.3 Functional description

The Peripheral Clock Control (PCC) module provides clock gating and clock source selection to each peripheral.

Each peripheral has its own unique PCCn register that provides clock gating for the peripheral's interface clock and its functional clock where applicable.

Optional peripheral functional clock sources can come from the SCG (*xxx*DIV*y*_CLK) or other modules such as LPO. Not all peripherals will make use of all available peripheral functional clocks.

See each peripheral's PCCn register for details.

21.4 Memory map and register definition

Each peripheral has its own dedicated PCC Register, which controls the clock gating, clock source and dividers for that specific peripheral.

NOTE

For each core processor or SOC, the number of PCC registers is different. Write accesses to unused PCC registers are blocked and result in a bus error. Read accesses are allowed and do not cause a bus error, but may return different results from different cores on a mulit-core SOC.

21.4.1 PCC Register Descriptions

21.4.1.1 PCC Memory Map

Absolute address	Register	Width (In bits)	Access	Reset value
40065020h	PCC DMA0 (PCC_DMA0)	32	RW	C0000000h
40065034h	PCC MPU (PCC_MPU)	32	RW	C0000000h
40065080h	PCC FLASH (PCC_FLASH)	32	RW	C0000000h
40065084h	PCC DMAMUX0 (PCC_DMAMUX0)	32	RW	80000000h
40065090h	PCC CAN0 (PCC_CAN0)	32	RW	80000000h
40065094h	PCC CAN1 (PCC_CAN1)	32	RW	80000000h
40065098h	PCC FLEXTMR3 (PCC_FLEXTMR3)	32	RW	80000000h
4006509Ch	PCC ADC1 (PCC_ADC1)	32	RW	C0000000h
400650B0h	PCC LPSPI0 (PCC_LPSPI0)	32	RW	80000000h
400650B4h	PCC LPSPI1 (PCC_LPSPI1)	32	RW	80000000h

Table continues on the next page...

Absolute address	Register	Width (In bits)	Access	Reset value
400650C4h	PCC PDB1 (PCC_PDB1)	32	RW	80000000h
400650C8h	PCC CRC (PCC_CRC)	32	RW	80000000h
400650CCh	PCC PDB2 (PCC_PDB2)	32	RW	80000000h
400650D8h	PCC PDB0 (PCC_PDB0)	32	RW	80000000h
400650DCh	PCC LPIT0 (PCC_LPIT0)	32	RW	80000000h
400650E0h	PCC FLEXTMR0 (PCC_FLEXTMR0)	32	RW	80000000h
400650E4h	PCC FLEXTMR1 (PCC_FLEXTMR1)	32	RW	80000000h
400650E8h	PCC FLEXTMR2 (PCC_FLEXTMR2)	32	RW	80000000h
400650ECh	PCC ADC0 (PCC_ADC0)	32	RW	C0000000h
400650F0h	PCC ADC2 (PCC_ADC2)	32	RW	C0000000h
400650F4h	PCC RTC (PCC_RTC)	32	RW	80000000h
400650FCh	PCC DAC0 (PCC_DAC0)	32	RW	80000000h
40065100h	PCC LPTMR0 (PCC_LPTMR0)	32	RW	80000000h
40065124h	PCC PORTA (PCC_PORTA)	32	RW	80000000h
40065128h	PCC PORTB (PCC_PORTB)	32	RW	80000000h
4006512Ch	PCC PORTC (PCC_PORTC)	32	RW	80000000h
40065130h	PCC PORTD (PCC_PORTD)	32	RW	80000000h
40065134h	PCC PORTE (PCC_PORTE)	32	RW	80000000h
40065158h	PCC PWT (PCC_PWT)	32	RW	80000000h
40065168h	PCC FLEXIO (PCC_FLEXIO)	32	RW	80000000h
40065180h	PCC OSC32 (PCC_OSC32)	32	RW	80000000h
40065184h	PCC EWM (PCC_EWM)	32	RW	80000000h
40065198h	PCC LPI2C0 (PCC_LPI2C0)	32	RW	80000000h
4006519Ch	PCC LPI2C1 (PCC_LPI2C1)	32	RW	80000000h
400651A8h	PCC LPUART0 (PCC_LPUART0)	32	RW	80000000h
400651ACh	PCC LPUART1 (PCC_LPUART1)	32	RW	80000000h
400651B0h	PCC LPUART2 (PCC_LPUART2)	32	RW	80000000h
400651CCh	PCC CMP0 (PCC_CMP0)	32	RW	80000000h
400651D0h	PCC CMP1 (PCC_CMP1)	32	RW	80000000h
400651D4h	PCC CMP2 (PCC_CMP2)	32	RW	80000000h

21.4.1.2 PCC DMA0 (PCC_DMA0)

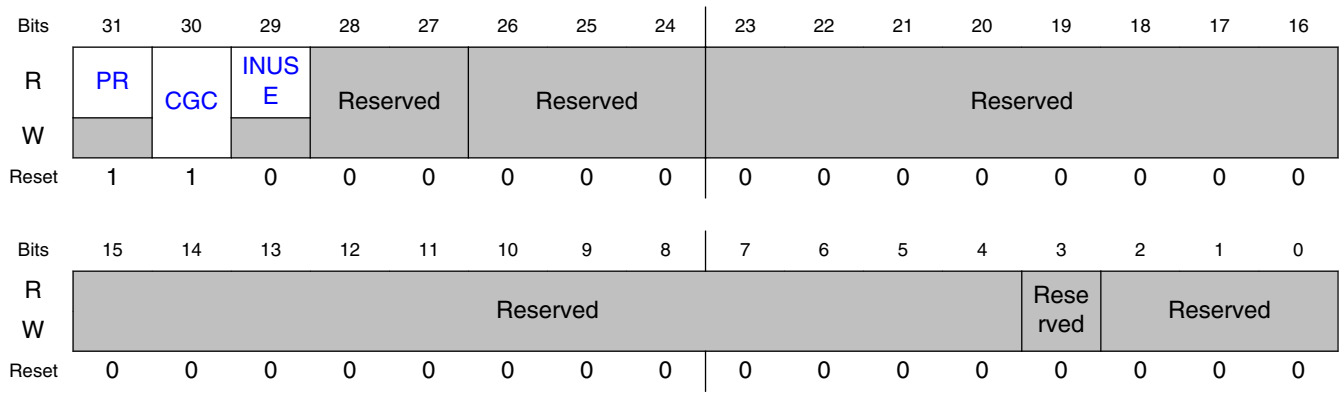
21.4.1.2.1 Address

Register	Offset
PCC_DMA0	40065020h

21.4.1.2.2 Function

PCC Register

21.4.1.2.3 Diagram



21.4.1.2.4 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.3 PCC MPU (PCC_MPU)

21.4.1.3.1 Address

Register	Offset
PCC_MPU	40065034h

PCC Register

21.4.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.3.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.4 PCC FLASH (PCC_FLASH)

21.4.1.4.1 Address

Register	Offset
PCC_FLASH	40065080h

PCC Register

21.4.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.4.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.5 PCC DMAMUX0 (PCC_DMAMUX0)

21.4.1.5.1 Address

Register	Offset
PCC_DMAMUX0	40065084h

PCC Register

21.4.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.5.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.6 PCC CAN0 (PCC_CAN0)

21.4.1.6.1 Address

Register	Offset
PCC_CAN0	40065090h

PCC Register

21.4.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.6.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.7 PCC CAN1 (PCC_CAN1)

21.4.1.7.1 Address

Register	Offset
PCC_CAN1	40065094h

PCC Register

21.4.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.7.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.8 PCC FLEXTMR3 (PCC_FLEXTMR3)

21.4.1.8.1 Address

Register	Offset
PCC_FLEXTMR3	40065098h

PCC Register

21.4.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.8.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.9 PCC ADC1 (PCC_ADC1)

21.4.1.9.1 Address

Register	Offset
PCC_ADC1	4006509Ch

PCC Register

21.4.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.9.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral.

Table continues on the next page...

Field	Function
	0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.10 PCC LPSPI0 (PCC_LPSPI0)

21.4.1.10.1 Address

Register	Offset
PCC_LPSPI0	400650B0h

PCC Register

21.4.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.10.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.11 PCC LPSPI1 (PCC_LPSPI1)

21.4.1.11.1 Address

Register	Offset
PCC_LPSPI1	400650B4h

PCC Register

21.4.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.11.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral.

Table continues on the next page...

Field	Function
	0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.12 PCC PDB1 (PCC_PDB1)

21.4.1.12.1 Address

Register	Offset
PCC_PDB1	400650C4h

PCC Register

21.4.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.12.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.13 PCC CRC (PCC_CRC)

21.4.1.13.1 Address

Register	Offset
PCC_CRC	400650C8h

PCC Register

21.4.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.13.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.14 PCC PDB2 (PCC_PDB2)

21.4.1.14.1 Address

Register	Offset
PCC_PDB2	400650CCh

PCC Register

21.4.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.14.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.15 PCC PDB0 (PCC_PDB0)

21.4.1.15.1 Address

Register	Offset
PCC_PDB0	400650D8h

PCC Register

21.4.1.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.15.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.16 PCC LPIT0 (PCC_LPIT0)

21.4.1.16.1 Address

Register	Offset
PCC_LPIT0	400650DCh

PCC Register

21.4.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.16.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.17 PCC FLEXTMR0 (PCC_FLEXTMR0)

21.4.1.17.1 Address

Register	Offset
PCC_FLEXTMR0	400650E0h

PCC Register

21.4.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.17.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

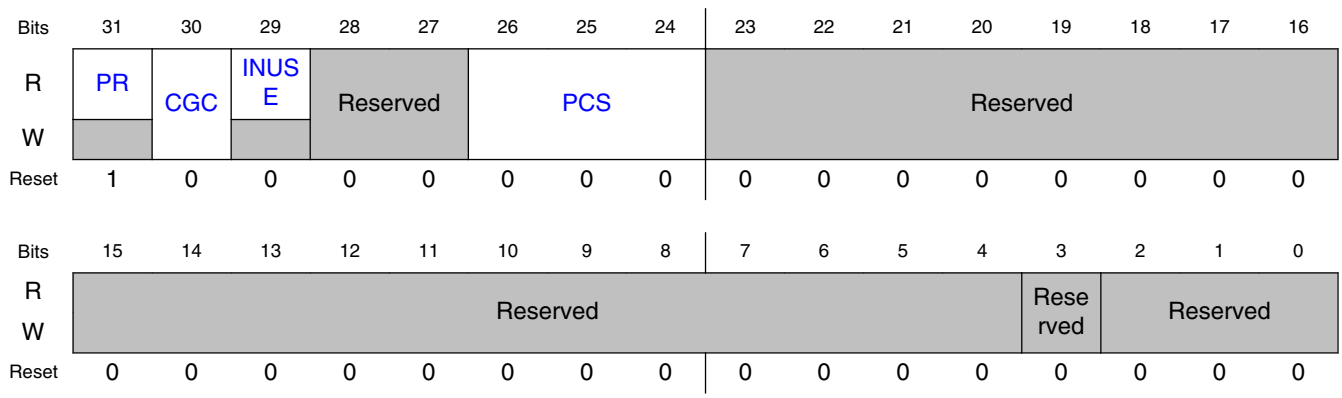
21.4.1.18 PCC FLEXTMR1 (PCC_FLEXTMR1)

21.4.1.18.1 Address

Register	Offset
PCC_FLEXTMR1	400650E4h

PCC Register

21.4.1.18.2 Diagram



21.4.1.18.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked.

Table continues on the next page...

Field	Function
	000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.19 PCC FLEXTMR2 (PCC_FLEXTMR2)

21.4.1.19.1 Address

Register	Offset
PCC_FLEXTMR2	400650E8h

PCC Register

21.4.1.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.19.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.20 PCC ADC0 (PCC_ADC0)

21.4.1.20.1 Address

Register	Offset
PCC_ADC0	400650ECh

PCC Register

21.4.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.20.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved.

Table continues on the next page...

Field	Function
	110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.21 PCC ADC2 (PCC_ADC2)

21.4.1.21.1 Address

Register	Offset
PCC_ADC2	400650F0h

PCC Register

21.4.1.21.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.21.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.22 PCC RTC (PCC_RTC)

21.4.1.22.1 Address

Register	Offset
PCC_RTC	400650F4h

PCC Register

21.4.1.22.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.22.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.23 PCC DAC0 (PCC_DAC0)

21.4.1.23.1 Address

Register	Offset
PCC_DAC0	400650FCh

PCC Register

21.4.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.23.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.24 PCC LPTMR0 (PCC_LPTMR0)

21.4.1.24.1 Address

Register	Offset
PCC_LPTMR0	40065100h

PCC Register

21.4.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.24.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.25 PCC PORTA (PCC_PORTA)

21.4.1.25.1 Address

Register	Offset
PCC_PORTA	40065124h

PCC Register

21.4.1.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.25.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.26 PCC PORTB (PCC_PORTB)

21.4.1.26.1 Address

Register	Offset
PCC_PORTB	40065128h

PCC Register

21.4.1.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.26.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.27 PCC PORTC (PCC_PORTC)

21.4.1.27.1 Address

Register	Offset
PCC_PORTC	4006512Ch

PCC Register

21.4.1.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.27.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.28 PCC PORTD (PCC_PORTD)

21.4.1.28.1 Address

Register	Offset
PCC_PORTD	40065130h

PCC Register

21.4.1.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.28.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.29 PCC PORTE (PCC_PORTE)

21.4.1.29.1 Address

Register	Offset
PCC_PORTE	40065134h

PCC Register

21.4.1.29.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.29.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.30 PCC PWT (PCC_PWT)

21.4.1.30.1 Address

Register	Offset
PCC_PWT	40065158h

PCC Register

21.4.1.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.30.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.31 PCC FLEXIO (PCC_FLEXIO)

21.4.1.31.1 Address

Register	Offset
PCC_FLEXIO	40065168h

PCC Register

21.4.1.31.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.31.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select **NOTE:** When FlexIO clock source is off (from PCC or SCG), FlexIO cannot be accessed. This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.32 PCC OSC32 (PCC_OSC32)

21.4.1.32.1 Address

Register	Offset
PCC_OSC32	40065180h

PCC Register

21.4.1.32.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.32.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30	Clock Control

Table continues on the next page...

Field	Function
CGC	This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

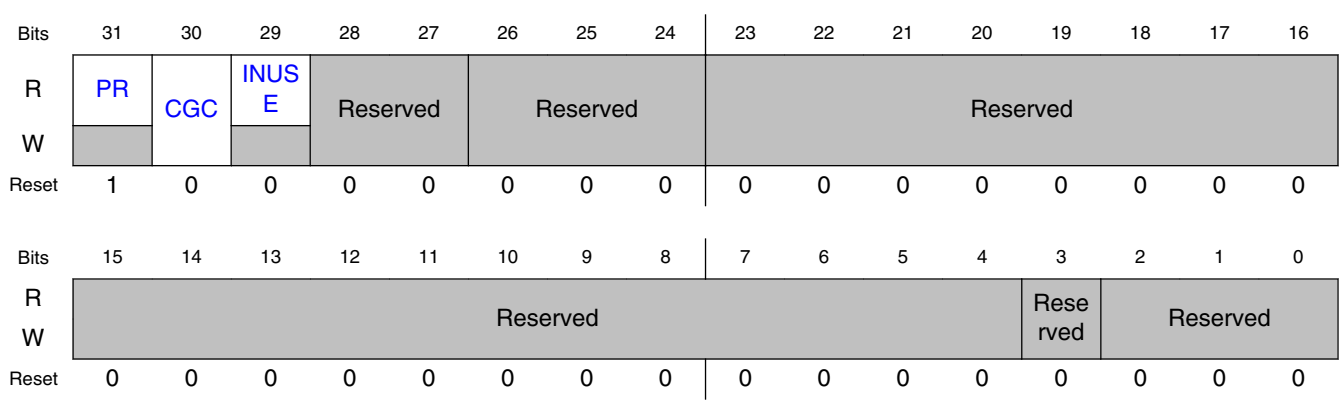
21.4.1.33 PCC EWM (PCC_EWM)

21.4.1.33.1 Address

Register	Offset
PCC_EWM	40065184h

PCC Register

21.4.1.33.2 Diagram



21.4.1.33.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

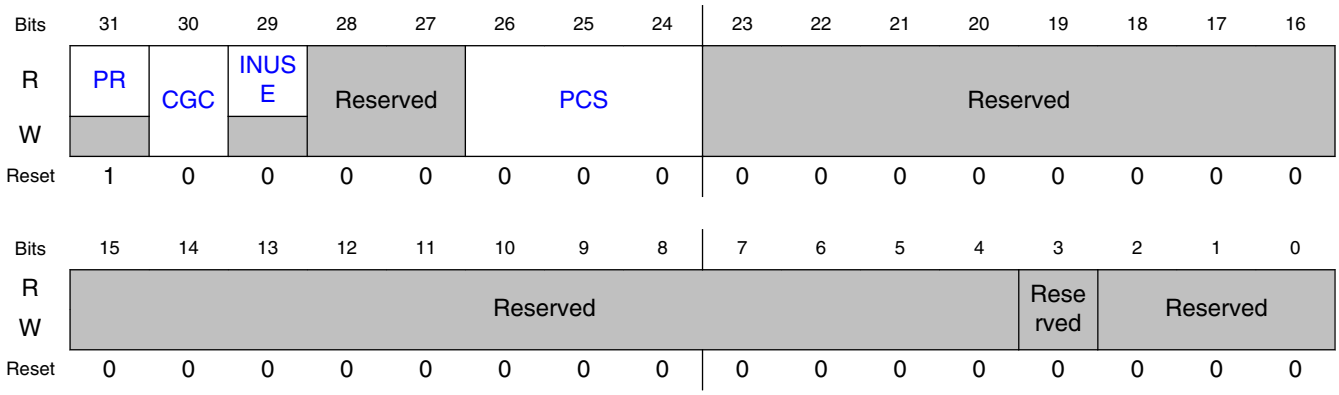
21.4.1.34 PCC LPI2C0 (PCC_LPI2C0)

21.4.1.34.1 Address

Register	Offset
PCC_LPI2C0	40065198h

PCC Register

21.4.1.34.2 Diagram



21.4.1.34.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.35 PCC LPI2C1 (PCC_LPI2C1)

21.4.1.35.1 Address

Register	Offset
PCC_LPI2C1	4006519Ch

PCC Register

21.4.1.35.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.35.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral.

Table continues on the next page...

Field	Function
	0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.36 PCC LPUART0 (PCC_LPUART0)

21.4.1.36.1 Address

Register	Offset
PCC_LPUART0	400651A8h

PCC Register

21.4.1.36.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.36.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.37 PCC LPUART1 (PCC_LPUART1)

21.4.1.37.1 Address

Register	Offset
PCC_LPUART1	400651ACh

PCC Register

21.4.1.37.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.37.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral.

Table continues on the next page...

Field	Function
	0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

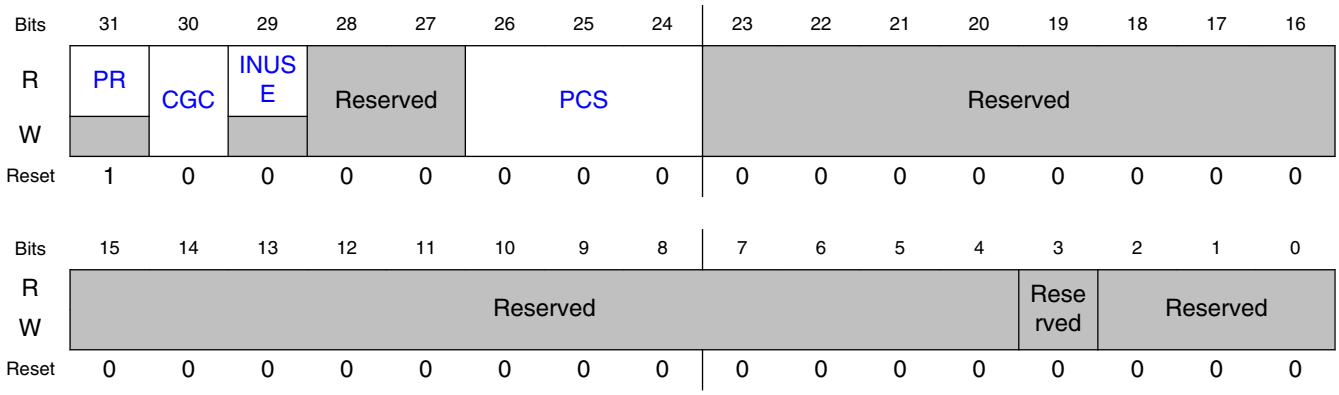
21.4.1.38 PCC LPUART2 (PCC_LPUART2)

21.4.1.38.1 Address

Register	Offset
PCC_LPUART2	400651B0h

PCC Register

21.4.1.38.2 Diagram



21.4.1.38.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000b - Clock is off . 001b - System Oscillator Bus Clock. 010b - Slow IRC Clock. 011b - Fast IRC Clock. 100b - Reserved. 101b - Reserved. 110b - System PLL clock. 111b - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.39 PCC CMP0 (PCC_CMP0)

21.4.1.39.1 Address

Register	Offset
PCC_CMP0	400651CCh

PCC Register

21.4.1.39.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.39.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral.

Table continues on the next page...

Field	Function
	0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

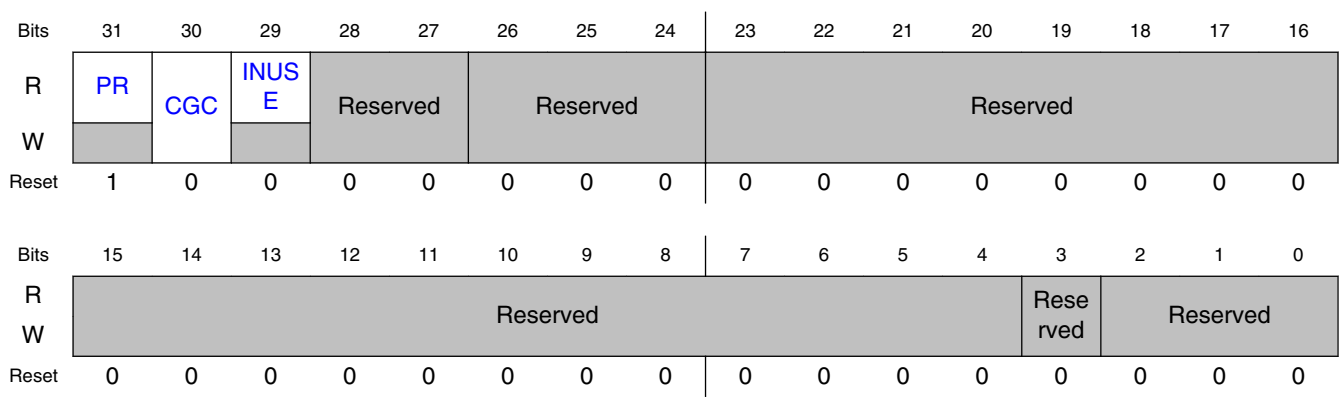
21.4.1.40 PCC CMP1 (PCC_CMP1)

21.4.1.40.1 Address

Register	Offset
PCC_CMP1	400651D0h

PCC Register

21.4.1.40.2 Diagram



21.4.1.40.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

21.4.1.41 PCC CMP2 (PCC_CMP2)

21.4.1.41.1 Address

Register	Offset
PCC_CMP2	400651D4h

PCC Register

21.4.1.41.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

21.4.1.41.3 Fields

Field	Function
31 PR	Enable This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Control This read/write bit enables the clock for the peripheral. 0b - Clock disabled 1b - Clock enabled
29 INUSE	Clock Gate Control >This read-only bit shows that this peripheral is being used . 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

Chapter 22
Reset and Boot

22.1 Introduction

The following reset sources are supported in this MCU:

Table 22-1. Reset sources

Reset sources	Description
POR reset	• Power-on reset (POR)
System resets	• External pin reset (PIN) • Low voltage detect (LVD) • Software watchdog reset (WDOG) • Clock generator loss of clock (LOC) reset • Clock generator loss of lock (LOL) reset • Stop mode acknowledge error (SACKERR) • Software reset (SW) • Lockup reset (LOCKUP) • MDM DAP system reset
Debug reset	• JTAG reset • nTRST reset

Each of the reset sources has an associated bit in the system reset status (RCM_SRS) register. Besides immediate reset, the RCM module also supports optional delays of the system resets for a period of time with an interrupt generated. This provides software an option to perform a graceful shutdown. See the Reset Control Module (RCM) chapter for register details.

The MCU exits reset in functional mode where the CPU is executing code. See Boot options for more details.

The following figure shows a block diagram of the reset sources for this device.

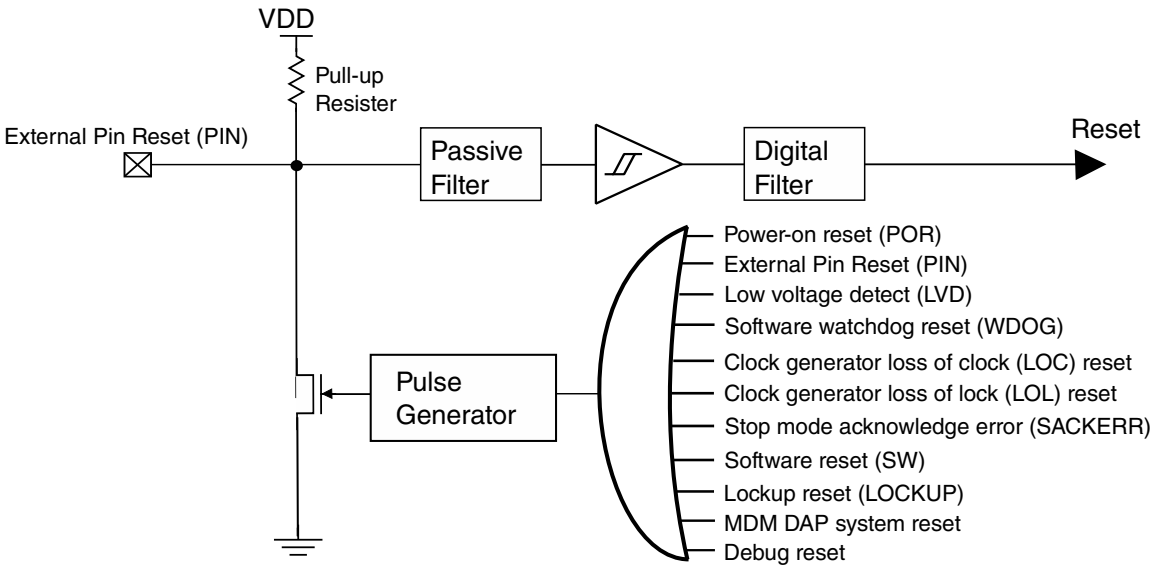


Figure 22-1. Reset Sources

22.2 Reset

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

22.2.1 Power-on reset (POR)

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

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

22.2.2 System resets

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 program counter (PC) from vector-table offset 4
- Link register (LR) is set to 0xFFFF_FFFF

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

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

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

and associated output pin configured as:

- TDO with no pull-down or pull-up

NOTE

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

- nTRST in PU

22.2.2.1 External pin reset (PIN)

On this device, asserting $\overline{\text{RESET}}$ wakes and resets the device from any mode. During a pin reset, RCM_SRS[PIN] is set.

The $\overline{\text{RESET}}$ pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. RCM_RPC[RSTFLTSS], RCM_RPC[RSTFLTSRW], and RCM_RPC[RSTFLTSEL] control this functionality; see the RCM chapter. The filters are asynchronously reset by Chip POR. The reset value for each filter assumes the $\overline{\text{RESET}}$ pin is negated.

For all stop modes where LPO clock is still active, the only filtering option is the LPO-based digital filter. The filtering logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected.

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

22.2.2.2 Low voltage detect (LVD)

The chip includes a system for managing low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system consists of a power-on reset (POR) circuit and an LVD circuit. The LVD system can always be enabled in HSRUN, normal Run, or Wait mode. The LVD system is disabled (LVR active only) when entering VLPx modes or Stop mode.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting the PMC_LVDSC1[LVDRE] bit to 1. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage has risen above the low voltage detection threshold. The RCM_SRS[LVD] bit is set following either an LVD reset or POR.

Refer to the "Low-voltage Detect (LVD) System" section in the Power Management Controller (PMC) chapter for more information. For LVR related content, see Low Voltage Reset (LVR) Operation.

22.2.2.3 Watchdog timer (WDOG)

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

22.2.2.4 Clock generator loss-of-clock (LOC)

The SCG module contains a clock monitor with reset and interrupt request capability for ROSC (OSC32) and SOSC clocks.

NOTE

To prevent unexpected loss of clock reset events, all clock monitors should be disabled before entering any low power modes, including VLPR and VLPW.

22.2.2.5 Loss-of-lock (LOL) reset

The SCG module contains a loss-of-lock detector, to indicate a reset has been caused by a loss of lock in the SCG PLL/FLL.

NOTE

This reset source does not cause a reset if the chip is in VLPR/ VLPW/VLPS mode.

22.2.2.6 Stop mode acknowledge error (SACKERR)

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

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

The RCM_SRS[SACKERR] bit is set to indicate this reset source.

22.2.2.7 Software reset (SW)

The SYSRESETREQ bit in the System Control Block's (SCB) application interrupt and reset control register can be set to force a software reset on the device. (See ARM's Cortex-M user guide for the full description of the register fields, especially the VECTKEY field requirements.) Setting SYSRESETREQ generates a software reset request. This reset forces a system reset of all major components except for the debug module. A software reset causes the RCM_SRS[SW] bit to set.

22.2.2.8 Lockup reset (LOCKUP)

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

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

22.2.2.9 MDM-AP system reset request

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

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

22.2.3 MCU Resets

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

22.2.3.1 POR Only

The POR Only reset asserts on the POR reset source only. It resets the PMC registers.

The POR Only reset also causes all other reset types to occur.

22.2.3.2 Chip POR

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

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

22.2.3.3 Early Chip Reset

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

22.2.3.4 Chip Reset

Chip Reset asserts on all reset sources and only negates after flash initialization has completed and the RESET_b pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

22.2.4 Reset Pin

For all reset sources, the RESET_b pin is driven low by the MCU for at least 128 bus clock cycles and until flash initialization has completed.

After flash initialization has completed, the RESET_b pin is released, and the internal Chip Reset negates after the RESET_b pin is pulled high. Keeping the RESET_b pin asserted externally delays the negation of the internal Chip Reset.

22.2.5 Debug resets

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

22.2.5.1 JTAG reset

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

22.2.5.2 nTRST reset

The nTRST pin causes a reset of the JTAG logic when asserted. Asserting the nTRST pin allows the debugger to gain control of the TAP controller state machine , without resetting the state of the debug modules.

The nTRST pin does not cause a system reset.

22.2.5.3 Resetting the Debug subsystem

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

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

- SWJ-DP
- AHB-AP
- TPIU
- MDM-AP (MDM control and status registers)

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

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

- AHB-AP [1]
- Private peripheral bus [1]

22.3 Boot

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

22.3.1 Boot options

The Flash Option (FOPT) register in the Flash Memory module (FTFE_FOPT) 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 default setting for all values in the FTFE_FOPT register is logic 1 since it is copied from the option byte residing in flash, which has all bits as logic 1 in the flash erased state. To configure for alternate settings, program the appropriate bits in the NVM option byte. The new settings will take effect on subsequent POR and any system reset. For more details on programming the option byte, see the flash memory chapter.

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

Table 22-2. Flash Option Register (FTFE_FOPT) definition

Bit Num	Field	Value	Definition
7	BOOTSRC_SEL	Boot Source Selection: these bits select the boot sources if boot pin option bit BOOTPIN_OPT (FOPT[1]) = 1. BOOTSRC_SEL (FOPT[7]) and BOOTPIN_OPT (FOPT[1]) value as below:	
		00	Boot from ROM with BOOTCFG0/NMI pin low, or Boot from Flash with BOOTCFG0/NMI pin high
		01	Boot from Flash
		10	Boot from ROM
		11	Boot from ROM
6	Reserved	Reserved for future expansion	
5-4	Reserved	Reserved for future expansion	
3	RESET_PIN_CFG	Enables/disables control for the RESET pin.	
		0	RESET_b pin is disabled following a POR and cannot be enabled as reset function. When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin low prior to establishing the

Table continues on the next page...

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

Table 22-2. Flash Option Register (FTFE_FOPT) definition (continued)

Bit Num	Field	Value	Definition
			setting of this option and releasing the reset function on the pin. When the RESET pin is disabled and configured as a GPIO output, it operates as a pseudo open drain output. This bit is preserved through system resets and low-power modes. When RESET_b pin function is disabled, it cannot be used as a source for low-power mode wake-up. **NOTE:** When the reset pin has been disabled and security has been enabled by means of the FSEC register, a mass erase can be performed only by setting both the Mass Erase and System Reset Request fields in the MDM-AP register.
		1	RESET_b pin is dedicated. The port is configured with pullup enabled, open drain, passive filter enabled.
2	NMI_DIS	Enables/disables control for the NMI function.	
		0	NMI interrupts are always blocked. The associated pin continues to default to NMI_b pin controls with internal pullup enabled. When NMI_b pin function is disabled, it cannot be used as a source for low-power mode wake-up. If the NMI function is not required, either for an interrupt or wake up source, it is recommended that the NMI function be disabled by clearing NMI_DIS.
		1	NMI_b pin/interrupts reset default to enabled.
1	BOOTPIN_OPT	External pin selects boot options	
		0	Force Boot from ROM with update if BOOTCFG0 asserted, where BOOTCFG0 is the boot config function which is muxed with NMI pin. The RESET pin should be enabled when this option is selected.
		1	Boot source configured by FOPT[7] (BOOTSRC_SEL) bitfield
0	LPBOOT	Controls the reset value of clock divider of IRC48M to feed the core clock. Larger divide value selections produce lower average power consumption during POR and reset sequencing and after reset exit. The recovery times are also extended .	
		0	Low-power boot: Core and system clock divider (DIVCORE) is 0x1 (divide by 2).
		1	Normal boot: Core and system clock divider (DIVCORE) is 0x0 (divide by 1).

This device supports cold booting from either internal flash or Boot ROM.

When the device boots from internal flash, the reset vectors are located at address 0x0 (initial SP_main) and 0x4 (initial PC).

When the device boots from ROM, the chip will re-map the reset vectors to ROM start address at 0x1C00_0000 where SP_main is offset 0x0 and PC is offset 0x4. When Boot ROM completes, software can clear RCM mode register (RCM_MR) to disable remapping of vector fetches. Boot source can change between reset, but is always known before core reset negation. NMI input is disabled to platform when booting from ROM. See FOPT section and Reset Control Module for more detail options.

The device also supports relocating the exception vector table to RAM. This is implemented through a programmable Vector Table Offset Register (VTOR) in SCB module.

The boot options can be overridden by using RCM_FM[2:1] and RCM_MR[2:1] which can be written by software. The boot source remains set until the next System Reset or software can write logic one to clear one or both of the mode bits.

22.3.2 Boot sequence

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

1. A system reset is held on internal logic, the RESET_b pin is driven out low, and the SCG is enabled in its default clocking mode.
2. Required clocks are enabled (Core Clock, System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control reset to disabled).
3. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Reset Control logic continues to drive the RESET_b pin out low.
4. Early in reset sequencing the NVM option byte is read and stored to the Flash Memory module's FOPT register.
5. When Flash Initialization completes, the RESET_b pin is released. If RESET_b continues to be asserted (an indication of a slow rise time on the RESET_b pin or external drive in low), the system continues to be held in reset. Once the RESET_b pin is detected high, the Core clock is enabled and the system is released from reset.
6. 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. What happens next depends on the NMI input and the FOPT[NMI_DIS] field in the Flash Memory module:
   - If the NMI input is high or the NMI function is disabled in the NMI_DIS field, the CPU begins execution at the PC location.
   - If the NMI input is low and the NMI function is enabled in the NMI_DIS field, 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.
7. If FlexNVM is enabled, the flash controller continues to restore the FlexNVM data. This data is not available immediately out of reset and the system should not access

this data until the flash controller completes this initialization step as indicated by the EEERDY flag.

Subsequent system resets follow this same reset flow.

The following figure shows the boot sequence.

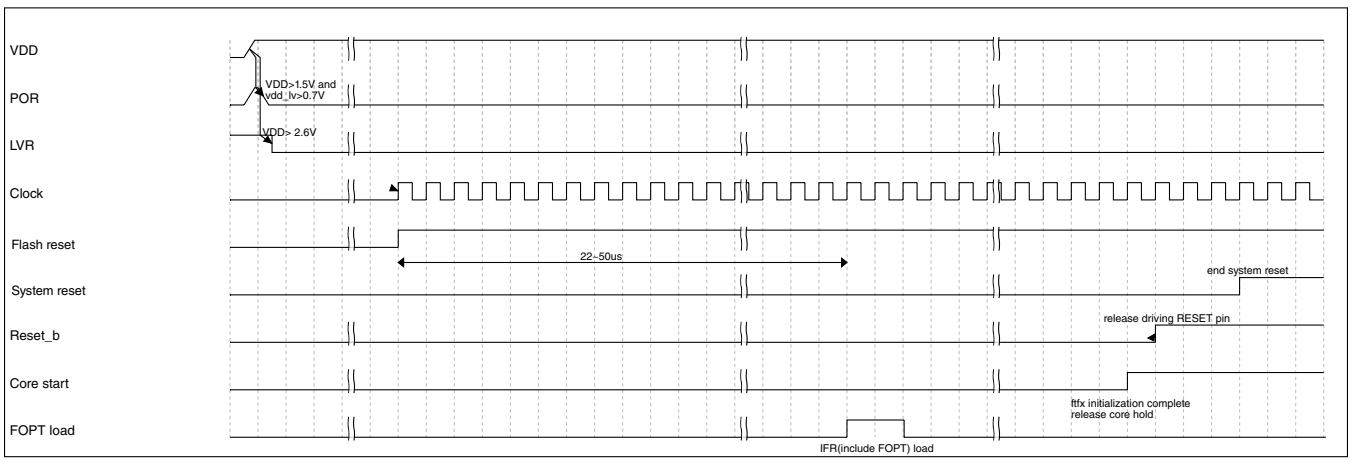


Figure 22-2. Boot Sequence

Chapter 23
Kinetis ROM Bootloader

23.1 Chip-specific information for this module

23.1.1 Boot ROM Configuration

This device contains an on-chip ROM bootloader which supports booting from LPUART, LPSPI, FlexCAN or LPI2C. The pinout table for peripherals supported by ROM is shown as follows.

Package			Peripheral	Instance	Signal	GPIO	ALT
100 QFP	80 QFP[1]	64 QFP					
53	41	33	LPUART	0	LPUART0_TX	PTB1	2
54	42	34			LPUART0_RX	PTB0	2
80	63	51		1	LPUART1_TX	PTC7	2
81	64	52			LPUART1_RX	PTC6	2
27	22	18	LPSPI	0	LPSPI0_PCS1	PTB5	3
28	23	19			LPSPI0_SOUT	PTB4	3
47	39	31			LPSPI0_SIN	PTB3	3
48	40	32			LPSPI0_SCK	PTB2	3
58	46	38		1	LPSPI1_PCS1	PTA6	3
71	58	46			LPSPI1_SOUT	PTD2	3
3	1	1			LPSPI1_SIN	PTD1	3
4	2	2			LPSPI1_SCK	PTD0	3
72	59	47	LPI2C	0	LPI2C0_SCL	PTA3	3
73	60	48			LPI2C0_SDA	PTA2	3
93	75	59		1	LPI2C1_SCL	PTE1	4

Table continues on the next page...

Package			Peripheral	Instance	Signal	GPIO	ALT
100 QFP	80 QFP[1]	64 QFP					
94	76	60			LPI2C1_SDA	PTE0	4
8	6	5	FlexCAN	0	CAN0_TX	PTE5	5
9	7	6			CAN0_RX	PTE4	5

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

ROM bootloader also supports application integrity check. It performs a CRC check over a configurable range of flash if requested via BCA option, and asserts a pin to indicate the failure. The available CRC check failure pin is shown as below:

Package			Function	GPIO	ALT
100 QFP	80 QFP[1]	64 QFP			
91	73	57	CRC check failure pin	PTA11	1

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

NOTE

The values of RAM properties returned by Boot ROM are incorrectly, as the RAM-related items in the table Table 23-69. See Chip Errata Kinetis_E_0N79P for details of this issue.

23.2 Introduction

The Kinetis bootloader is the program residing in the on-chip read-only memory (ROM) of a Kinetis microcontroller device. There is hardware logic in place at boot time that either starts execution of an embedded image available on the internal flash memory, or starts the execution of the Kinetis Bootloader from on-chip ROM.

The Kinetis Bootloader's main task is to provision the internal flash memory with an embedded firmware image during manufacturing, or at any time during the life of the Kinetis device. The Kinetis Bootloader does the provisioning by acting as a slave device, and listening to various peripheral ports where a master can start communication.

For the Kinetis device, the Kinetis Bootloader can interface with I2C, SPI, and UART peripherals in slave mode and respond to the commands sent by a master (or host) communicating on one of those ports. The host/master can be a firmware-download application running on a PC or an embedded host communicating with the Kinetis Bootloader. Regardless of the host/master (PC or embedded host), the Kinetis Bootloader always uses a command protocol to communicate with that host/master. Commands are

provided to write to memory (internal flash or RAM), erase flash, and get/set bootloader options and property values. The host application can query the set of available commands.

On start-up, the bootloader reads optional configuration parameters from a fixed area on flash called the bootloader configuration area (BCA). These parameters can be modified by the write memory command or by downloaded flash image. BCA parameters include configuration data such as enabled peripherals, peripheral-specific settings, etc.

This chapter describes Kinetis Bootloader features, functionality, command structure and which peripherals are supported.

Features supported by the Kinetis Bootloader in Kinetis ROM:

- Supports I2C, SPI, CAN and UART peripheral interfaces
- Automatic detection of the active peripheral
- Ability to disable any peripheral
- UART peripheral implements autobaud
- Common packet-based protocol for all peripherals
- Packet error detection and retransmission
- Flash-resident configuration options
- Fully supports internal flash security, including ability to mass erase or unlock security via the backdoor key
- Protection of RAM used by the bootloader while it is running
- Provides command to read properties of the device, such as flash and RAM size
- Multiple options for executing the bootloader either at system start-up or under application control at runtime
- Supports internal flash
- Supports encrypted image download

Table 23-1. Commands supported by the Kinetis Bootloader in ROM

Command	Description	When flash security is enabled, then this command is
Execute	Run user application code that never returns control to the bootloader	Not supported
FlashEraseAll	Erase the entire flash array	Not supported
FlashEraseRegion	Erase a range of sectors in flash	Not supported
WriteMemory	Write data to memory	Not supported
ReadMemory	Read data from memory	Not supported
FlashSecurityDisable	Attempt to unlock flash security using the backdoor key	Supported
GetProperty	Get the current value of a property	Supported
Reset	Reset the chip	Supported

Table continues on the next page...

Table 23-1. Commands supported by the Kinetis Bootloader in ROM (continued)

Command	Description	When flash security is enabled, then this command is
SetProperty	Attempt to modify a writable property	Supported
FlashEraseAllUnsecure	Erase the entire flash array, including protected sectors	Supported

23.3 Functional Description

The following sub-sections describe the Kinetis Bootloader in ROM functionality.

23.3.1 Memory Maps

While executing, the Kinetis Bootloader uses ROM and RAM memory.

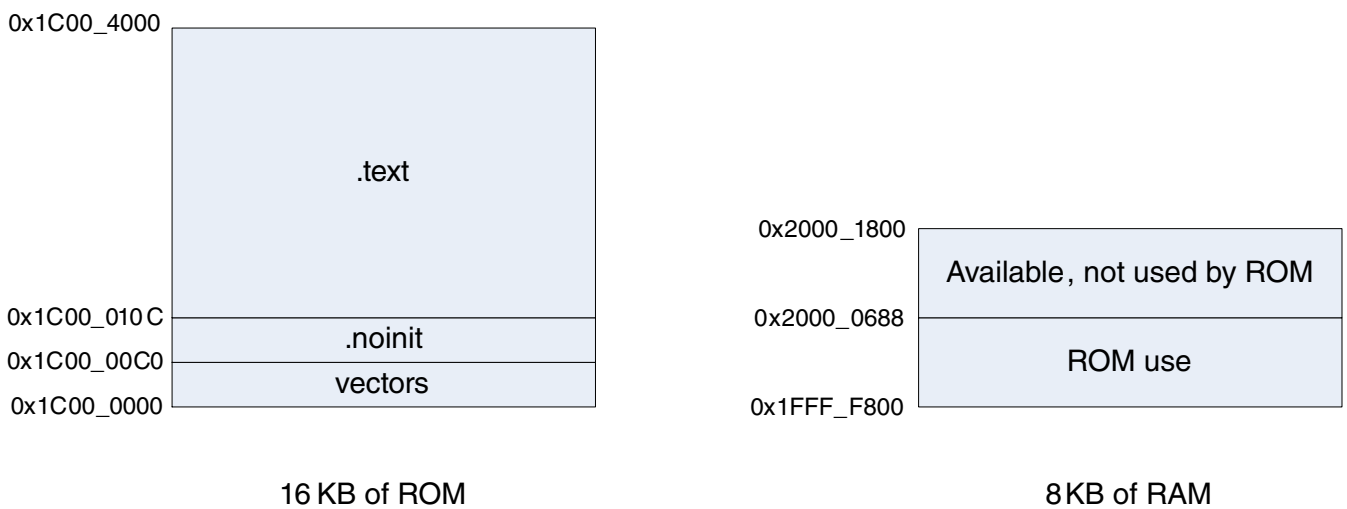


Figure 23-1. Kinetis Bootloader ROM/RAM Memory Maps

23.3.2 The Kinetis Bootloader Configuration Area (BCA)

The Kinetis Bootloader reads data from the Bootloader Configuration Area (BCA) to configure various features of the bootloader. The BCA resides in flash memory at offset 0x3C0, and provides all of the parameters needed to configure the Kinetis Bootloader

operation. For uninitialized flash, the Kinetis Bootloader uses a predefined default configuration. A host application can use the Kinetis Bootloader to program the BCA for use during subsequent initializations of the bootloader.

Table 23-2. Configuration Fields for the Kinetis Bootloader

Offset	Size (bytes)	Configuration Field	Description
0x00 - 0x03	4	tag	Magic number to verify bootloader configuration is valid. Must be set to 'kcfg'.
0x04 - 0x07	4	crcStartAddress	Start address for application image CRC check. To generate the CRC, refer to the CRC chapter. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x08 - 0x0B	4	crcByteCount	Byte count for application image CRC check. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x0C - 0x0F	4	crcExpectedValue	Expected CRC value for application CRC check. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x10	1	enabledPeripherals	Bitfield of peripherals to enable. bit 0 LPUART bit 1 LPI2C bit 2 LPSPI bit 3 CAN Kinetis bootloader will enable the peripheral if corresponding bit is set to 1.
0x11	1	i2cSlaveAddress	If not 0xFF, used as the 7-bit I2C slave address. If 0xFF, defaults to 0x10 for I2C slave address
0x12 - 0x13	2	peripheralDetectionTimeout	Timeout in milliseconds for active peripheral detection. If 0xFFFF, defaults to 5 seconds.
0x14 - 0x15	2	-	Reserved
0x16- 0x17	2	-	Reserved
0x18 - 0x1B	4	-	Reserved
0x1C	1	clockFlags	See Table 23-4, clockFlags Configuration Field
0x1D	1	clockDivider	Inverted value of the divider to use for core and bus clocks when in high speed mode
0x1F	1	pad byte	N/A
0x20 - 0x23	4	Reserved	-
0x24 - 0x27	4	Reserved	-

Table continues on the next page...

Table 23-2. Configuration Fields for the Kinetis Bootloader (continued)

Offset	Size (bytes)	Configuration Field	Description
0x28	1	Reserved	-
0x29	1	canConfig1	ClkSel[1], PropSeg[3], SpeedIndex[4]
0x2A - 0x2B	2	canConfig2	Pdiv[8], Pseg[3], Pseg2[3], rjw[2]
0x2C - 0x2D	2	canTxId	txId
0x2E - 0x2F	2	canRxId	rxId
0x30 - 0x33	4	Reserved	-
0x34	12	Reserved	-

NOTE

The flash sector containing the BCA should not be located in the execute-only region, because the Kinetis bootloader cannot read an execute-only region.

The first configuration field 'tag' is a tag value or magic number. The tag value must be set to 'kcfg' for the bootloader configuration data to be recognized as valid. If tag-field verification fails, then the Kinetis Bootloader assumes that the flash is not initialized and uses a predefined default configuration (all as 0xFF). The tag value is treated as a character string, so bytes 0-3 must be set as shown in the table.

Table 23-3. tag Configuration Field

Offset	tag Byte Value
0	'k' (0x6B)
1	'c' (0x63)
2	'f' (0x66)
3	'g' (0x67)

The flags in the clockFlags configuration field are enabled if the corresponding bit is cleared (0).

Table 23-4. clockFlags Configuration Field

Bit	Flag	Description
0	HighSpeed	Enable high speed mode (i.e., 48 MHz). Read Clock Configuration section for more information on the high speed mode.
1 - 7	Reserved	

23.3.3 Start-up Process

Any of the following conditions will force the hardware to start the Kinetis Bootloader:

- FOPT [7] is set to 1. This forces the ROM to run out of reset.
- The BOOTCFG0 pin is asserted. The pin must be configured as BOOTCFG0 by setting the BOOTPIN_OPT bit of FOPT to 0.
- A user applications running on flash or RAM calls into the Kinetis Bootloader entry point address in ROM, to start Kinetis Bootloader execution.

The FOPT[BOOTSRC_SEL] determines the boot source. The FOPT register is located in the flash configuration field at address 0x40D in the flash memory array. For a complete list of options, see the Boot options section in the Reset and Boot chapter. If FOPT [7] is set to 1, then the device will boot to ROM out of reset. Flash memory defaults to all 1s when erased, so a blank chip will automatically boot to ROM.

The BOOTCFG0 pin is shared with the NMI pin, with NMI being the default usage. Regardless of whether the NMI pin is enabled or not, the NMI functionality is disabled if the ROM is executed out of reset, for as long as the ROM is running.

When the ROM is executed out of reset, vector fetches from the CPU are redirected to the ROM's vector table in ROM memory at offset 0x1C00_0000. This ensures that any exceptions will be handled by the ROM.

After the Kinetis Bootloader has started, the following procedure starts bootloader operations:

1. The RCM_MR [BOOTROM] bits are set, so that the device will reboot back into the ROM if/when the device is reset.
2. Initializes the bootloader's .data and .bss sections.
3. Reads bootloader configuration data from flash at address 0x3C0. The configuration data is only used if the tag field is set to the expected 'kcfg' value. If the tag is incorrect, then the configuration values are set to default, as if the data was all 0xFF bytes.
4. Clocks are configured. See the Clock Configuration section.
5. Enabled peripherals are initialized.
6. The bootloader waits for communication to begin on a peripheral.
   - If detection times out, then the bootloader jumps to the user application in flash. See Bootloader Exit state section.
   - If communication is detected, then all inactive peripherals are shut down, and the command phase is entered.

NOTE

The flash sector containing the vector table should not be located in the execute-only region, because the Kinetis

bootloader cannot read the PC and SP addresses in an execute-only region.

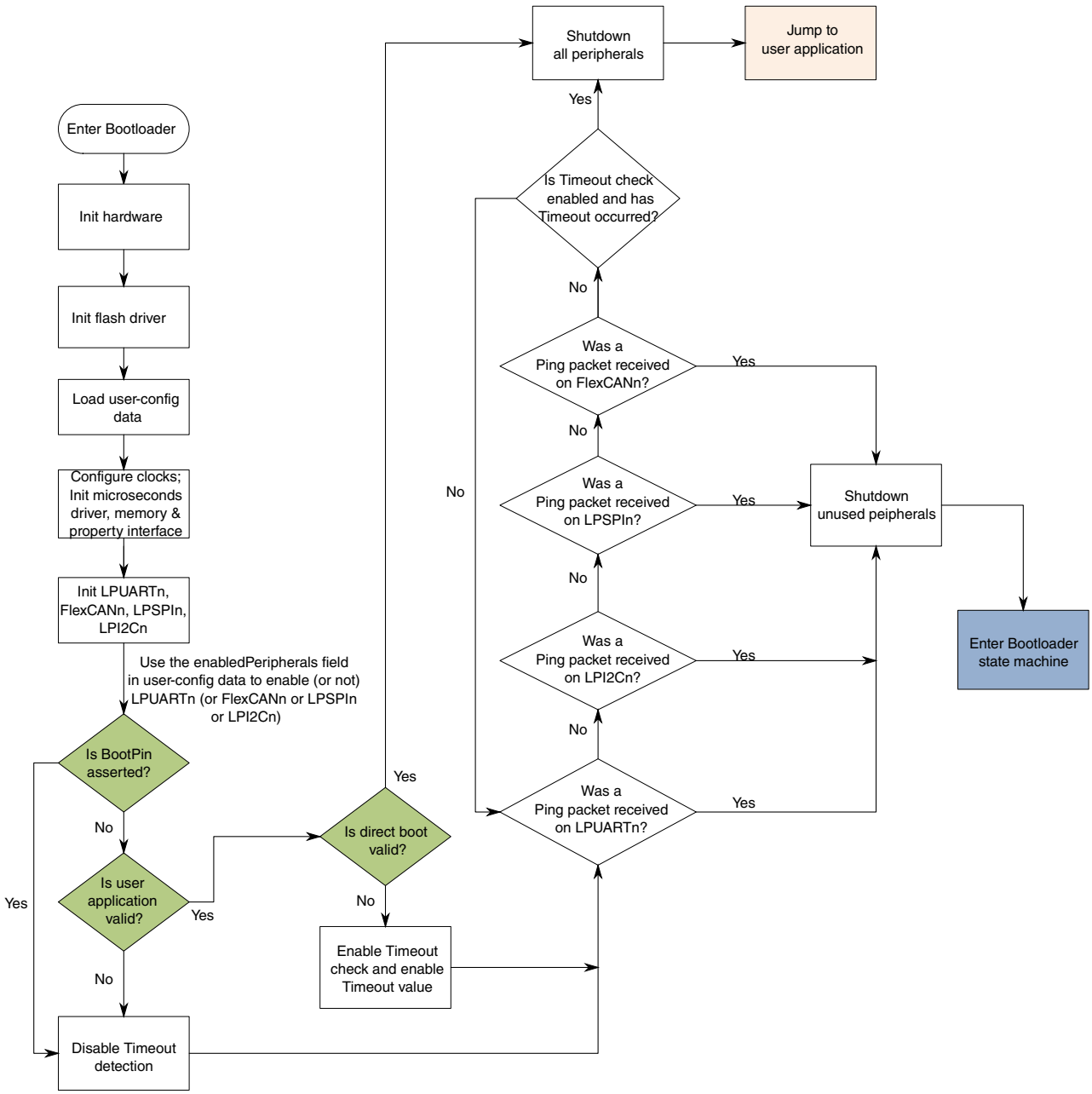


Figure 23-2. Kinetis Bootloader Start-up Flowchart

23.3.4 Clock Configuration

By default, the bootloader does not modify clocks. The Kinetis Bootloader in ROM will use the clock configuration of the chip out of reset unless the clock configuration bits in the FOPT register are cleared.

- Alternate clock configurations are supported, by setting fields in the Bootloader Configuration Area (BCA) shown in Table 23-2.
- If the HighSpeed flag of the clockFlags configuration value is cleared, the bootloader will enable the internal 48 MHz reference clock.
- In high speed mode, the core and bus clock frequencies are determined by the clockDivider configuration value.
- The core clock divider is set directly from the inverted value of the clockDivider.
- The bus clock divider is set to 1, unless the resulting bus clock frequency would be greater than the maximum supported value. In this case, the bus clock divider is increased until the bus clock frequency is at or below the maximum.
- Note that the maximum baud rate of serial peripherals is related to the core and bus clock frequencies. To achieve the desired baud rates, high speed mode should be enabled in BCA.

23.3.5 Bootloader Entry Point / API Tree

To run the Kinetis Bootloader, a user application simply calls the runBootloader function. To get the address of the entry point, the user application reads the word containing the pointer to the bootloader API tree at offset 0x1C of the bootloader's vector table. The vector table is placed at the base of the bootloader's address range, which for the ROM is 0x1C00_0000; the API tree pointer is at address 0x1C00_001C.

The bootloader API tree is a structure that contains pointers to other structures, which have the function and data addresses for the bootloader. The bootloader entry point is always the 1st word of the API tree.

```
typedef struct BootloaderTree
{
    void (*runBootloader)(void *arg);           //!< Function to start the bootloader
executing.
    standard_version_t version;                 //!< Bootloader version number.
    const char *copyright;                      //!< Copyright string.
    const bootloader_context_t *runtimeContext; //!< Pointer to the bootloader's runtime
context.
    const flash_driver_interface_t *flashDriver; //!< Flash driver API.
    const aes_driver_interface_t *aesDriver;    //!< AES driver API.

} bootloader_tree_t;
```

The prototype of the entry point is:

```
void run_bootloader(void * arg);
```

The `arg` parameter is currently unused, and is intended for future expansion (for example, passing options to the bootloader). To ensure future compatibility, a value of NULL should be passed for `arg`.

Example: code to get the entry pointer address from the ROM and start the bootloader.

NOTE

This entry must be called in supervisor (privileged) mode.

```
// Variables

uint32_t runBootloaderAddress;

void (*runBootloader)(void * arg);


// Read the function address from the ROM API tree.

runBootloaderAddress = **(uint32_t **)(0x1c00001c);

runBootloader = (void (*)(void * arg))runBootloaderAddress;


// Start the bootloader.

runBootloader(NULL);
```

23.3.6 Bootloader Protocol

This section explains the general protocol for the packet transfers between the host and the Kinetis Bootloader. The description includes the transfer of packets for different transactions, such as commands with no data phase and commands with incoming or outgoing data phase. The next section describes various packet types used in a transaction.

Each command sent from the host is replied to with a response command.

Commands may include an optional data phase:

- If the data phase is **incoming** (from host to bootloader), then the data phase is part of the **original command**.
- If the data phase is **outgoing** (from bootloader to host), then the data phase is part of the **response command**.

NOTE

In all protocols (described in the next subsections), the Ack sent in response to a Command or Data packet can arrive at any time *before, during, or after* the Command/Data packet has processed.

23.3.6.1 Command with no data phase

The protocol for a command with no data phase contains:

- Command packet (from host)
- Generic response command packet (to host)

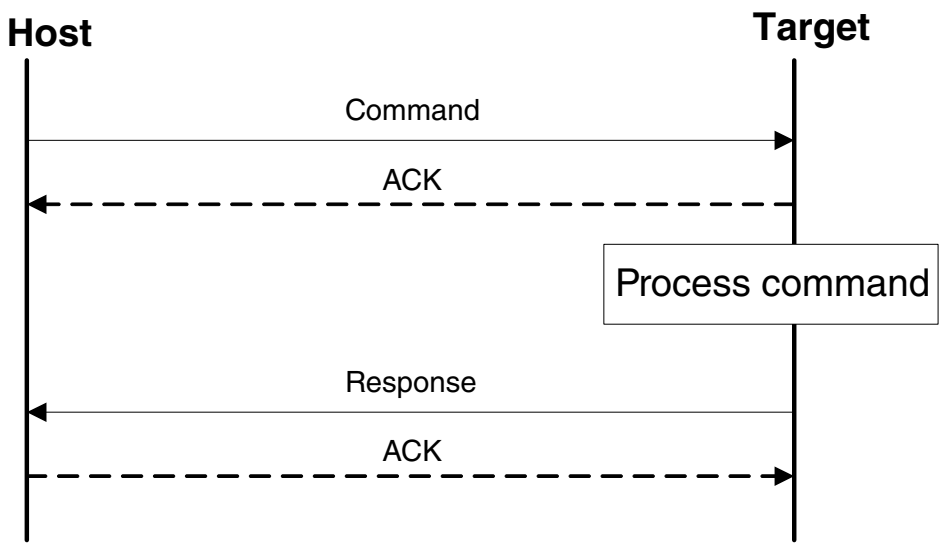


Figure 23-3. Command with No Data Phase

23.3.6.2 Command with incoming data phase

The protocol for a command with an incoming data phase contains:

- Command packet (from host)
- Generic response command packet (to host)
- Incoming data packets (from host)
- Generic response command packet (to host)

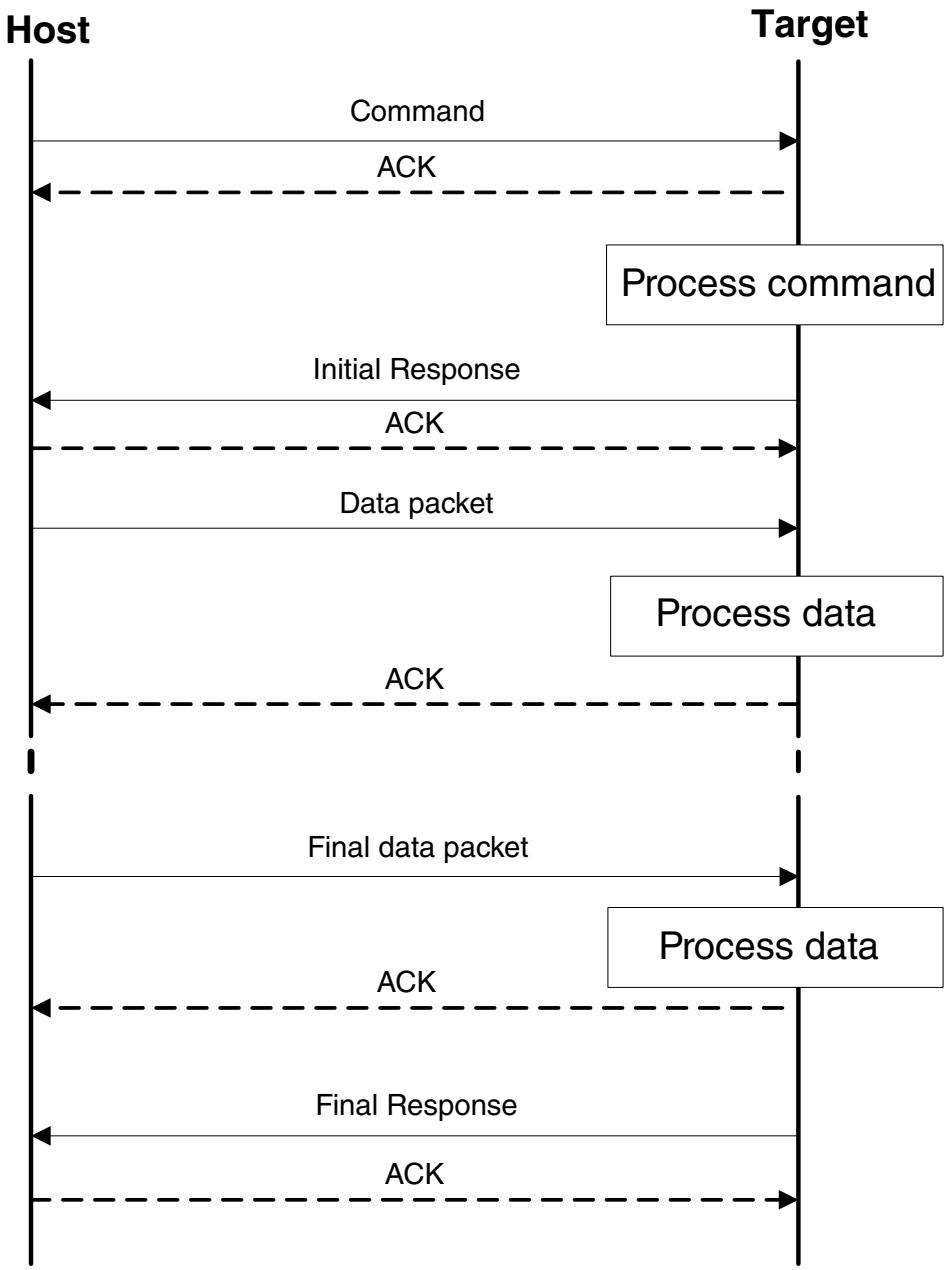


Figure 23-4. Command with incoming data phase

NOTE

- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the Generic Response packet prior to the start of the data phase does not have a status of kStatus_Success, then the data phase is aborted.
- Data phases may be aborted by the receiving side by sending the final Generic Response early with a status of

kStatus_AbortDataPhase. The host may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

23.3.6.3 Command with outgoing data phase

The protocol for a command with an outgoing data phase contains:
- Command packet (from host)
- ReadMemory Response command packet (to host) (kCommandFlag_HasDataPhase set)
- Outgoing data packets (to host)
- Generic response command packet (to host)

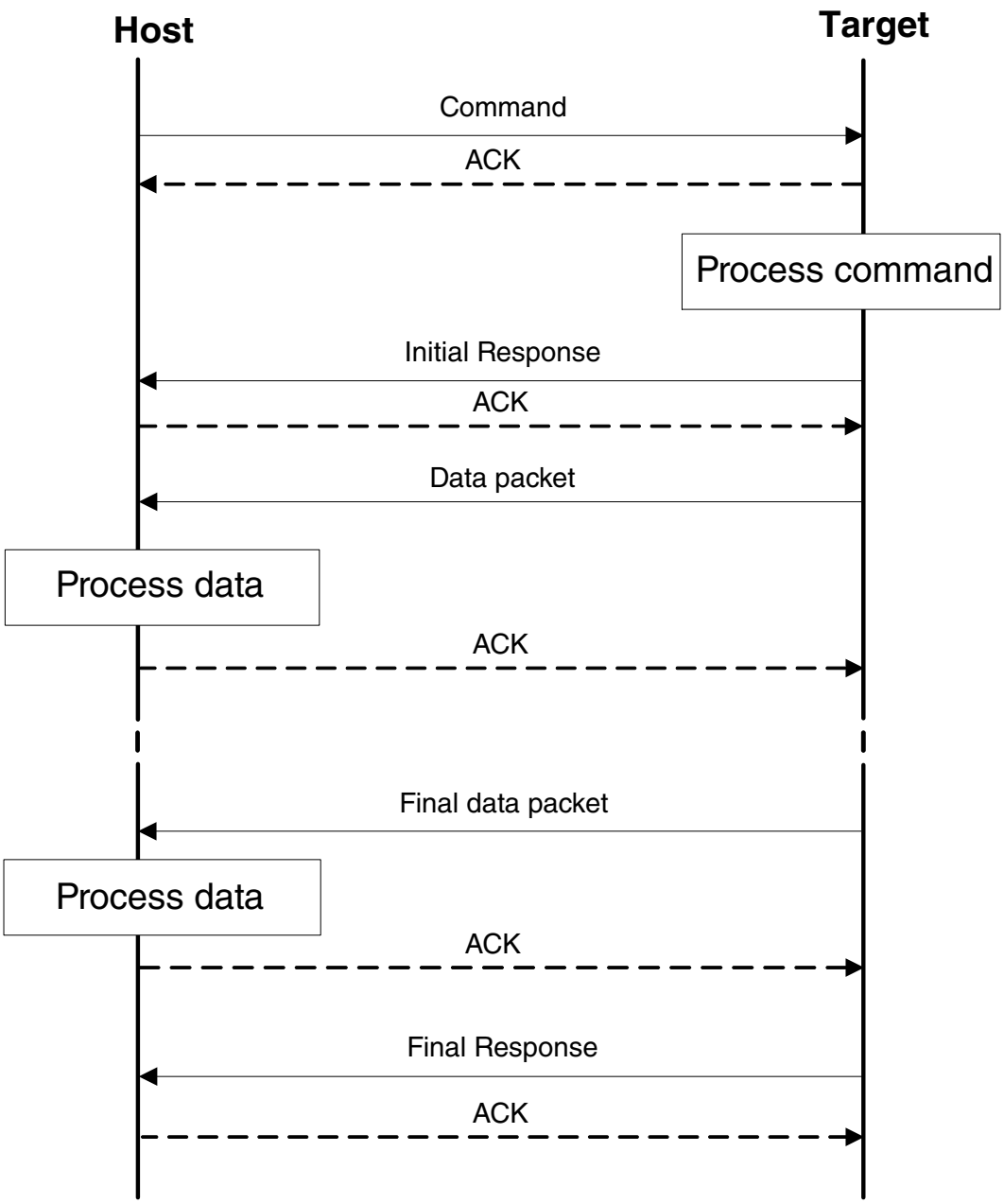


Figure 23-5. Command with outgoing data phase

NOTE

- For the outgoing data phase sequence above, the data phase is really considered part of the response command.
- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the ReadMemory Response command packet prior to the start of the data phase does not contain the kCommandFlag_HasDataPhase flag, then the data phase is aborted.

- Data phases may be aborted by the host sending the final Generic Response early with a status of kStatus_AbortDataPhase. The sending side may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

23.3.7 Bootloader Packet Types

The Kinetis Bootloader device works in slave mode. All data communication is initiated by a host, which is either a PC or an embedded host . The Kinetis Bootloader device is the target, which receives a command or data packet. All data communication between host and target is packetized.

NOTE

The term "target" refers to the "Kinetis Bootloader device."

There are 6 types of packets used in the device:

- Ping packet
- Ping Response packet
- Framing packet
- Command packet
- Data packet
- Response packet

All fields in the packets are in little-endian byte order.

23.3.7.1 Ping packet

The Ping packet is the first packet sent from a host to the target (Kinetis Bootloader), to establish a connection on a selected peripheral. For a UART peripheral, the Ping packet is used to determine the baudrate. A Ping packet must be sent before any other communications. In response to a Ping packet, the target sends a Ping Response packet.

Table 23-5. Ping Packet Format

Byte #	Value	Name
0	0x5A	start byte
1	0xA6	ping

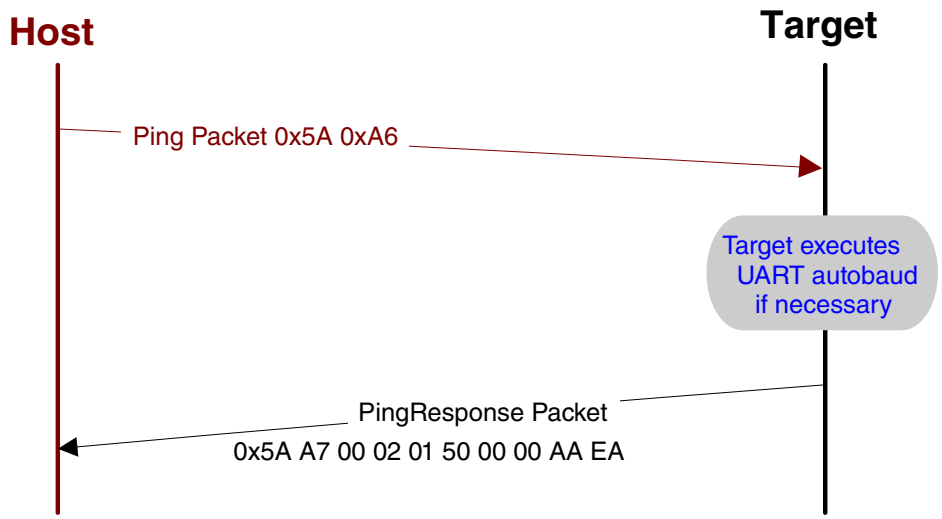


Figure 23-6. Ping Packet Protocol Sequence

23.3.7.2 Ping Response Packet

The target (Kinetis Bootloader) sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over a UART peripheral, the target uses the incoming Ping packet to determine the baud rate before replying with the Ping Response packet. Once the Ping Response packet is received by the host, the connection is established, and the host starts sending commands to the target (Kinetis Bootloader).

Table 23-6. Ping Response Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA7	Ping response code
2		Protocol bugfix
3		Protocol minor
4		Protocol major
5		Protocol name = 'P' (0x50)
6		Options low
7		Options high
8		CRC16 low
9		CRC16 high

23.3.7.3 Framing Packet

The framing packet is used for flow control and error detection, and it (the framing packet) wraps command and data packets as well.

The framing packet described in this section is used for serial peripherals including UART, I2C, SPI, and CAN.

Table 23-7. Framing Packet Format

Byte #	Value	Parameter	
0	0x5A	start byte	
1		packetType	
2		length_low	Length is a 16-bit field that specifies the entire command or data packet size in bytes.
3		length_high	
4		crc16_low	This is a 16-bit field. The CRC16 value covers entire framing packet, including the start byte and command or data packets, but does not include the CRC bytes. See the CRC16 algorithm after this table.
5		crc16_high	
6 . . .n		Command or Data packet payload	

A special framing packet that contains only a start byte and a packet type is used for synchronization between the host and target.

Table 23-8. Special Framing Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA*n*	packetType

The Packet Type field specifies the type of the packet from one of the defined types (below):

Table 23-9. packetType Field

packetType	Name	Description
0xA1	kFramingPacketType_Ack	The previous packet was received successfully; the sending of more packets is allowed.
0xA2	kFramingPacketType_Nak	The previous packet was corrupted and must be re-sent.
0xA3	kFramingPacketType_AckAbort	Data phase is being aborted.
0xA4	kFramingPacketType_Command	The framing packet contains a command packet payload.
0xA5	kFramingPacketType_Data	The framing packet contains a data packet payload.
0xA6	kFramingPacketType_Ping	Sent to verify the other side is alive. Also used for UART autobaud.

Table continues on the next page...

Table 23-9. packetType Field (continued)

packetType	Name	Description
0xA7	kFramingPacketType_PingResponse	A response to Ping; contains the framing protocol version number and options.

This device uses the Cyclic Redundancy Check module (CRC) to perform the CRC algorithm. See the CRC chapter for more details.

23.3.7.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 23-10. Command Packet Format

Command Packet Format (32 bytes)										
Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param1 (32-bit)	Param2 (32-bit)	Param3 (32-bit)	Param4 (32-bit)	Param5 (32-bit)	Param6 (32-bit)	Param7 (32-bit)
byte 0	byte 1	byte 2	byte 3							

Table 23-11. Command Header Format

Byte #	Command Header Field	
0	Command or Response tag	The command header is 4 bytes long, with these fields.
1	Flags	
2	Reserved. Should be 0x00.	
3	ParameterCount	

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. As mentioned earlier, command packets and data packets are embedded into framing packets for all of the transfers.

Table 23-12. Commands that are supported

Command	Name
0x01	FlashEraseAll
0x02	FlashEraseRegion
0x03	ReadMemory

Table continues on the next page...

Table 23-12. Commands that are supported (continued)

Command	Name
0x04	WriteMemory
0x05	Reserved
0x06	FlashSecurityDisable
0x07	GetProperty
0x08	Reserved
0x09	Execute
0x0A	Reserved
0x0B	Reset
0x0C	SetProperty
0x0D	FlashEraseAllUnsecure
0x0E	Reserved
0x0F	Reserved
0x10	Reserved
0x11	Reserved
0x12	Reserved

Table 23-13. Responses that are supported

Response	Name
0xA0	GenericResponse
0xA3	ReadMemoryResponse (used for sending responses to ReadMemory command only)
0xA7	GetPropertyResponse (used for sending responses to GetProperty command only)

Flags: Each command packet contains a Flag byte. Only bit 0 of the flag byte is used. If bit 0 of the flag byte is set to 1, then data packets will follow in the command sequence. The number of bytes that will be transferred in the data phase is determined by a command-specific parameter in the parameters array.

ParameterCount: The number of parameters included in the command packet.

Parameters: The parameters are word-length (32 bits). With the default maximum packet size of 32 bytes, a command packet can contain up to 7 parameters.

23.3.7.5 Data packet

The data packet carries just the data, either host sending data to target, or target sending data to host. The data transfer direction is determined by the last command sent from the host. The data packet is also wrapped within a framing packet, to ensure the correct packet data is received.

The contents of a data packet are simply the data itself. There are no other fields, so that the most data per packet can be transferred. Framing packets are responsible for ensuring that the correct packet data is received.

23.3.7.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse
- GetPropertyResponse
- ReadMemoryResponse

GenericResponse: After the Kinetis Bootloader has processed a command, the bootloader will send a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters (defined in the next section). The parameter count field in the header is always set to 2, for status code and command tag parameters.

Table 23-14. GenericResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The Status codes are errors encountered during the execution of a command by the target (Kinetis Bootloader). If a command succeeds, then a kStatus_Success code is returned. Table 23-73, Kinetis Bootloader Status Error Codes, lists the status codes returned to the host by the Kinetis Bootloader for ROM.
4 - 7	Command tag	The Command tag parameter identifies the response to the command sent by the host.

GetPropertyResponse: The GetPropertyResponse packet is sent by the target in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

Table 23-15. GetPropertyResponse Parameters

Byte #	Value	Parameter
0 - 3		Status code
4 - 7		Property value
. . .		. . .
		Can be up to maximum 6 property values, limited to the size of the 32-bit command packet and property type.

ReadMemoryResponse: The ReadMemoryResponse packet is sent by the target in response to the host sending a ReadMemory command. The ReadMemoryResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a ReadMemoryResponse tag value (0xA3), the flags field set to kCommandFlag_HasDataPhase (1).

The parameter count set to 2 for the status code and the data byte count parameters shown below.

Table 23-16. ReadMemoryResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The status of the associated Read Memory command.
4 - 7	Data byte count	The number of bytes sent in the data phase.

23.3.8 Bootloader Command API

All Kinetis Bootloader command APIs follow the command packet format that is wrapped by the framing packet, as explained in previous sections.

- For a list of commands supported by the Kinetis Bootloader in ROM, see Table 23-1, Commands supported.
- For a list of status codes returned by the Kinetis Bootloader in ROM, see Table 23-73, Kinetis Bootloader Status Error Codes.

NOTE

All the examples in this section depict byte traffic on serial peripherals that use framing packets.

23.3.8.1 Execute command

The execute command results in the bootloader setting the program counter to the code at the provided jump address, R0 to the provided argument, and a Stack pointer to the provided stack pointer address. Prior to the jump, the system is returned to the reset state.

The Jump address, function argument pointer, and stack pointer are the parameters required for the Execute command.

Table 23-17. Parameters for Execute Command

Byte #	Command
0 - 3	Jump address
4 - 7	Argument word
8 - 11	Stack pointer address

The Execute command has no data phase.

Response: Before executing the Execute command, the target (Kinetis Bootloader) will validate the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

23.3.8.2 Reset command

The Reset command will result in bootloader resetting the chip.

The Reset command requires no parameters.

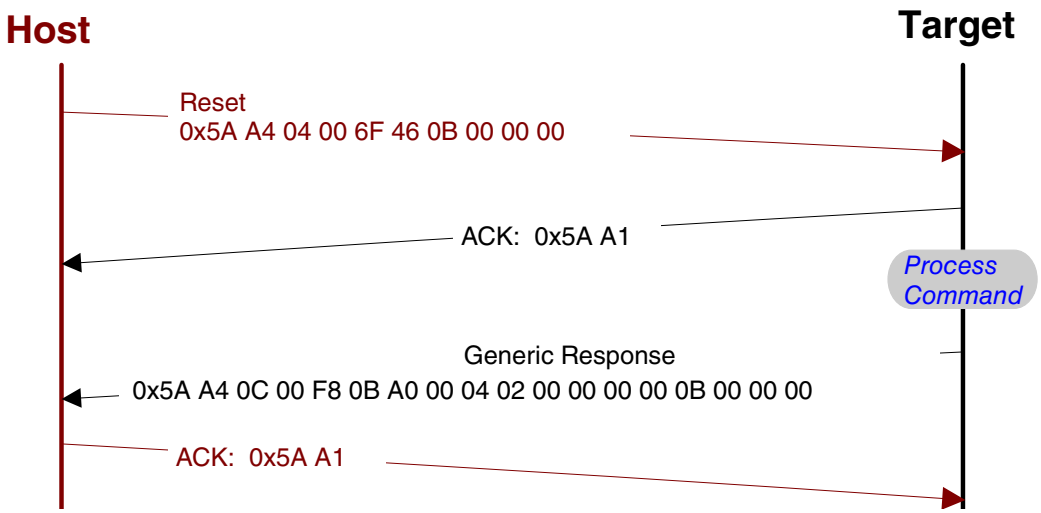


Figure 23-7. Protocol Sequence for Reset Command

Table 23-18. Reset Command Packet Format (Example)

Reset	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0x6F 0x46
Command packet	commandTag	0x0B - reset
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The Reset command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

23.3.8.3 GetProperty command

The GetProperty command is used to query the bootloader about various properties and settings. Each supported property has a unique 32-bit tag associated with it. The tag occupies the first parameter of the command packet. The target returns a GetPropertyResponse packet with the property values for the property identified with the tag in the GetProperty command.

Properties are the defined units of data that can be accessed with the GetProperty or SetProperty commands. Properties may be read-only or read-write. All read-write properties are 32-bit integers, so they can easily be carried in a command parameter.

For a list of properties and their associated 32-bit property tags supported by the Kinetis Bootloader in ROM, see Table 23-69.

The 32-bit property tag is the only parameter required for GetProperty command.

Table 23-19. Parameters for GetProperty Command

Byte #	Command
0 - 3	Property tag

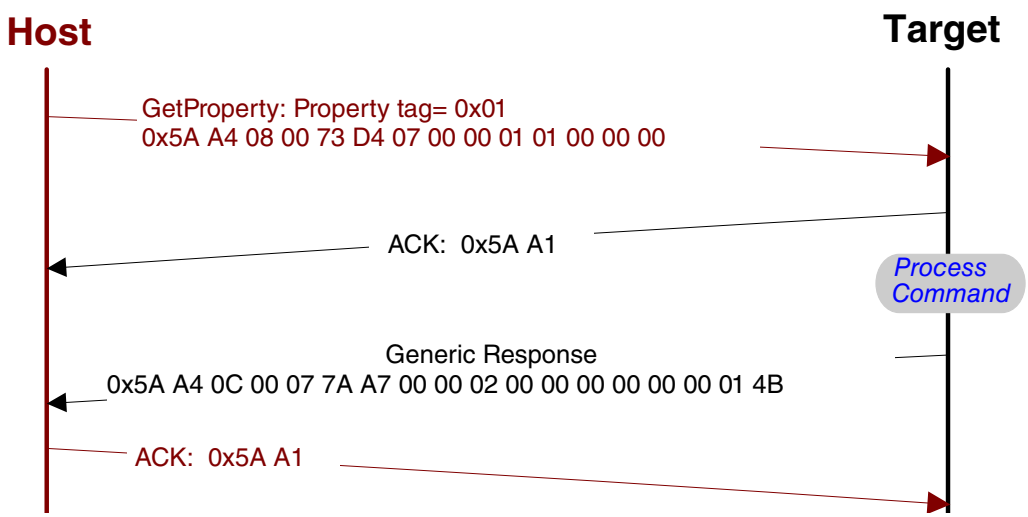


Figure 23-8. Protocol Sequence for GetProperty Command

Table 23-20. GetProperty Command Packet Format (Example)

GetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x08 0x00
	crc16	0x73 0xD4
Command packet	commandTag	0x07 – GetProperty
	flags	0x00
	reserved	0x00
	parameterCount	0x01
	propertyTag	0x00000001 - CurrentVersion

The GetProperty command has no data phase.

Response: In response to a GetProperty command, the target will send a GetPropertyResponse packet with the response tag set to 0xA7. The parameter count indicates the number of parameters sent for the property values, with the first parameter showing status code 0, followed by the property value(s). The next table shows an example of a GetPropertyResponse packet.

Table 23-21. GetProperty Response Packet Format (Example)

GetPropertyResponse	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0c 0x00 (12 bytes)
	crc16	0x07 0x7a
Command packet	responseTag	0xA7
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	status	0x00000000
	propertyValue	0x0000014b - CurrentVersion

23.3.8.4 SetProperty command

The SetProperty command is used to change or alter the values of the properties or options in the Kinetis Bootloader ROM. However, the SetProperty command can only change the value of properties that are writable—see Table 23-69, Properties used by Get/SetProperty Commands. If you try to set a value for a read-only property, then the Kinetis Bootloader will return an error.

The property tag and the new value to set are the 2 parameters required for the SetProperty command.

Table 23-22. Parameters for SetProperty Command

Byte #	Command
0 - 3	Property tag
4 - 7	Property value

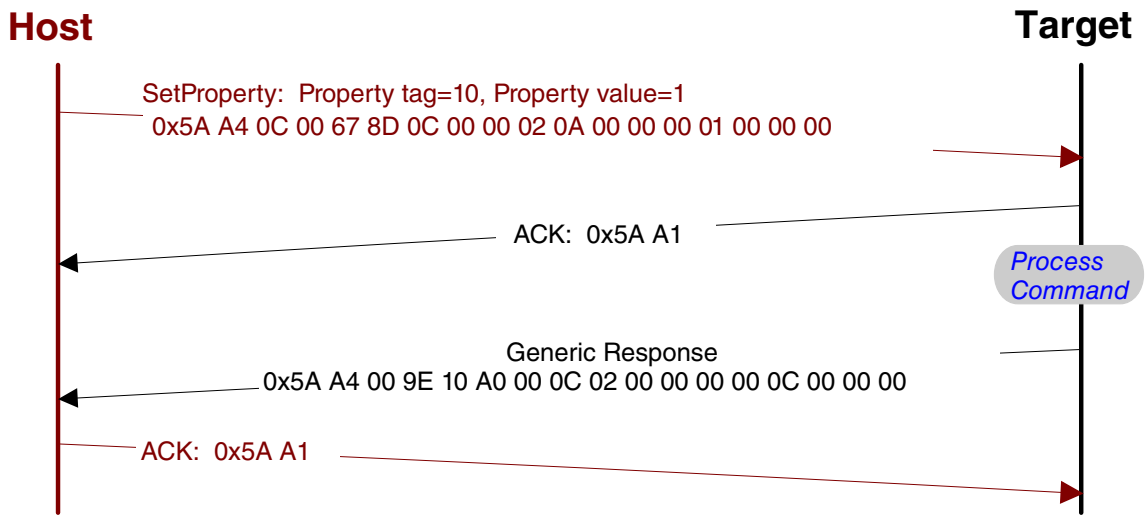


Figure 23-9. Protocol Sequence for SetProperty Command

Table 23-23. SetProperty Command Packet Format (Example)

SetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x67 0x8D
Command packet	commandTag	0x0C – SetProperty with property tag 10
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	propertyTag	0x0000000A - VerifyWrites
	propertyValue	0x00000001

The SetProperty command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following status codes:

Table 23-24. SetProperty Response Status Codes

Status Code
kStatus_Success
kStatus_ReadOnly
kStatus_UnknownProperty
kStatus_InvalidArgument

23.3.8.5 FlashEraseAll command

The FlashEraseAll command performs an erase of the entire flash memory. If any flash regions are protected, then the FlashEraseAll command will fail and return an error status code. Executing the FlashEraseAll command will release flash security if it (flash security) was enabled, by setting the FTFE_FSEC register. However, the FSEC field of the flash configuration field is erased, so unless it is reprogrammed, the flash security will be re-enabled after the next system reset. The Command tag for FlashEraseAll command is 0x01 set in the commandTag field of the command packet.

The FlashEraseAll command requires no parameters.

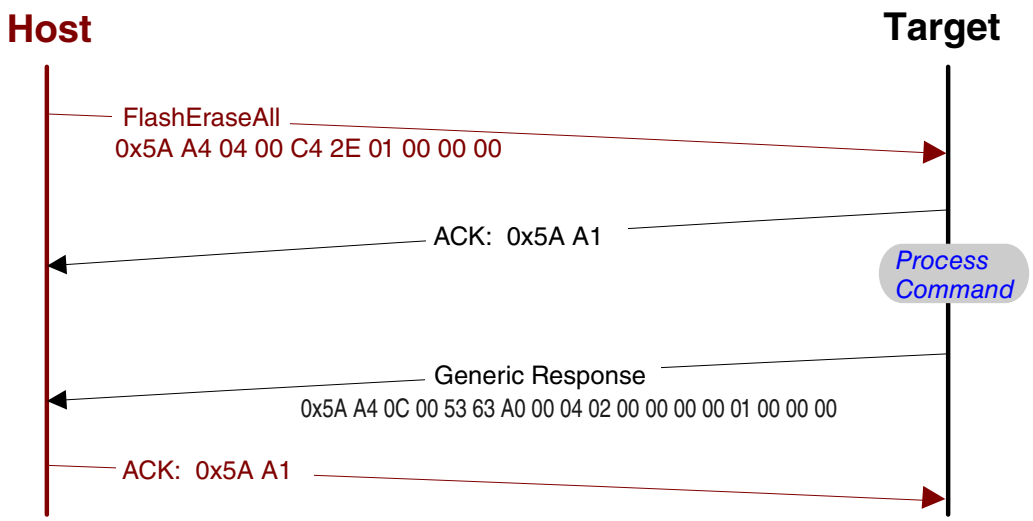


Figure 23-10. Protocol Sequence for FlashEraseAll Command

Table 23-25. FlashEraseAll Command Packet Format (Example)

FlashEraseAll	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0xC4 0x2E
Command packet	commandTag	0x01 - FlashEraseAll
	flags	0x00
	reserved	0x00
	parameterCount	0x00
	MemoryID	• If MemoryID = 0x00h, then internal flash. • If MemoryID = 0x01h, then QSPI0 memory.

The FlashEraseAll command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

23.3.8.6 FlashEraseRegion command

The FlashEraseRegion command performs an erase of one or more sectors of the flash memory or a specified range of flash within the connected SPI flash devices.

The start address and number of bytes are the 2 parameters required for the FlashEraseRegion command. The start and byte count parameters must be 4-byte aligned ([1:0] = 00), or the FlashEraseRegion command will fail and return kStatus_FlashAlignmentError (0x101). If the region specified does not fit in the flash memory space, the FlashEraseRegion command will fail and return kStatus_FlashAddressError (0x102). If any part of the region specified is protected, the FlashEraseRegion command will fail and return kStatus_MemoryRangeInvalid (0x10200).

Table 23-26. Parameters for FlashEraseRegion Command

Byte #	Parameter
0 - 3	Start address
4 - 7	Byte count

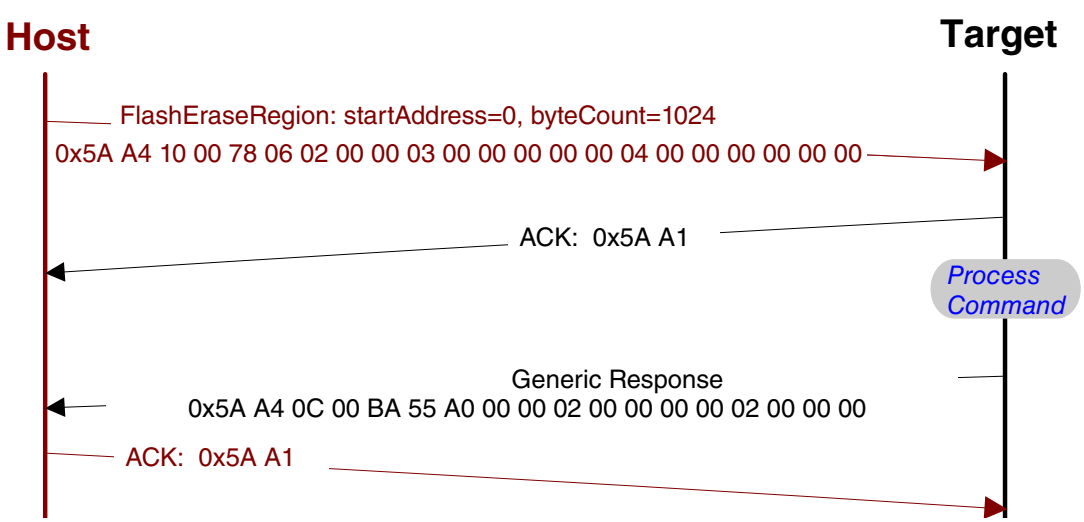


Figure 23-11. Protocol Sequence for FlashEraseRegion Command

Table 23-27. FlashEraseRegion Command Packet Format (Example)

FlashEraseRegion	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0x78 0x06
Command packet	commandTag	0x02, kCommandTag_FlashEraseRegion
	flags	0x00
	reserved	0x00
	parameterCount	0x03
	startAddress	0x00 0x00 0x00 0x00 (0x0000_0000)
	byte count	0x00 0x04 0x00 0x00 (0x400)
	memory_id	0x00 0x00 0x00 0x00 (internal flash)

The FlashEraseRegion command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following error status codes.

Table 23-28. FlashEraseRegion Response Status Codes

Status Code
kStatus_Success (0x0)
kStatus_MemoryRangeInvalid (0x10200)
kStatus_FlashAlignmentError (0x101)
kStatus_FlashAddressError (0x102)
kStatus_FlashAccessError (0x103)
kStatus_FlashProtectionViolation (0x104)
kStatus_FlashCommandFailure (0x105)

23.3.8.7 FlashEraseAllUnsecure command

The FlashEraseAllUnsecure command performs a mass erase of the flash memory, including protected sectors. Flash security is immediately disabled if it (flash security) was enabled, and the FSEC byte in the flash configuration field at address 0x40C is programmed to 0xFE. However, if the mass erase enable option in the FSEC field is disabled, then the FlashEraseAllUnsecure command will fail.

The FlashEraseAllUnsecure command requires no parameters.

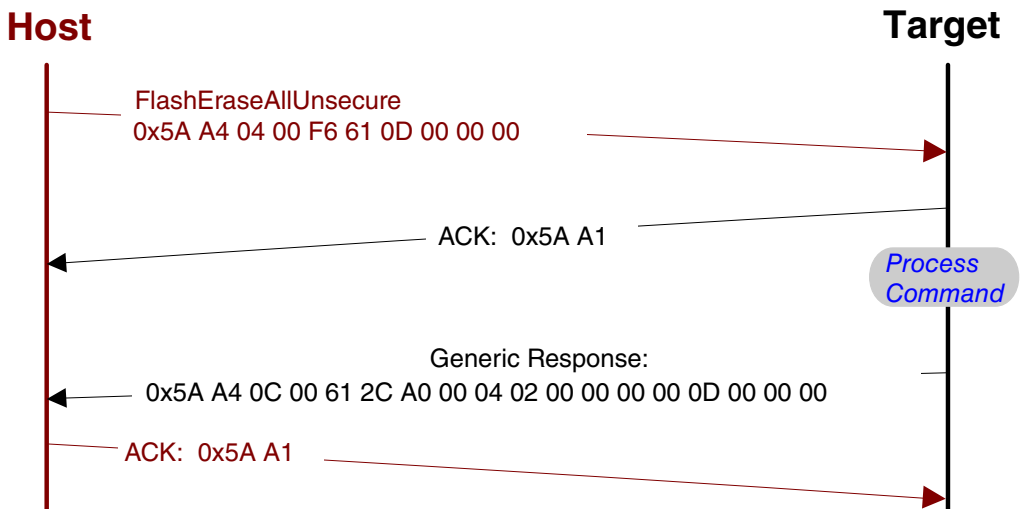


Figure 23-12. Protocol Sequence for FlashEraseAllUnsecure Command

Table 23-29. FlashEraseAllUnsecure Command Packet Format (Example)

FlashEraseAllUnsecure	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0xF6 0x61
Command packet	commandTag	0x0D - FlashEraseAllUnsecure
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The FlashEraseAllUnsecure command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

23.3.8.8 FlashSecurityDisable command

The FlashSecurityDisable command performs the flash security disable operation, by comparing the 8-byte backdoor key (provided in the command) against the backdoor key stored in the flash configuration field (at address 0x400 in the flash).

The backdoor low and high words are the only parameters required for FlashSecurityDisable command.

Table 23-30. Parameters for FlashSecurityDisable Command

Byte #	Command
0 - 3	Backdoor key low word
4 - 7	Backdoor key high word

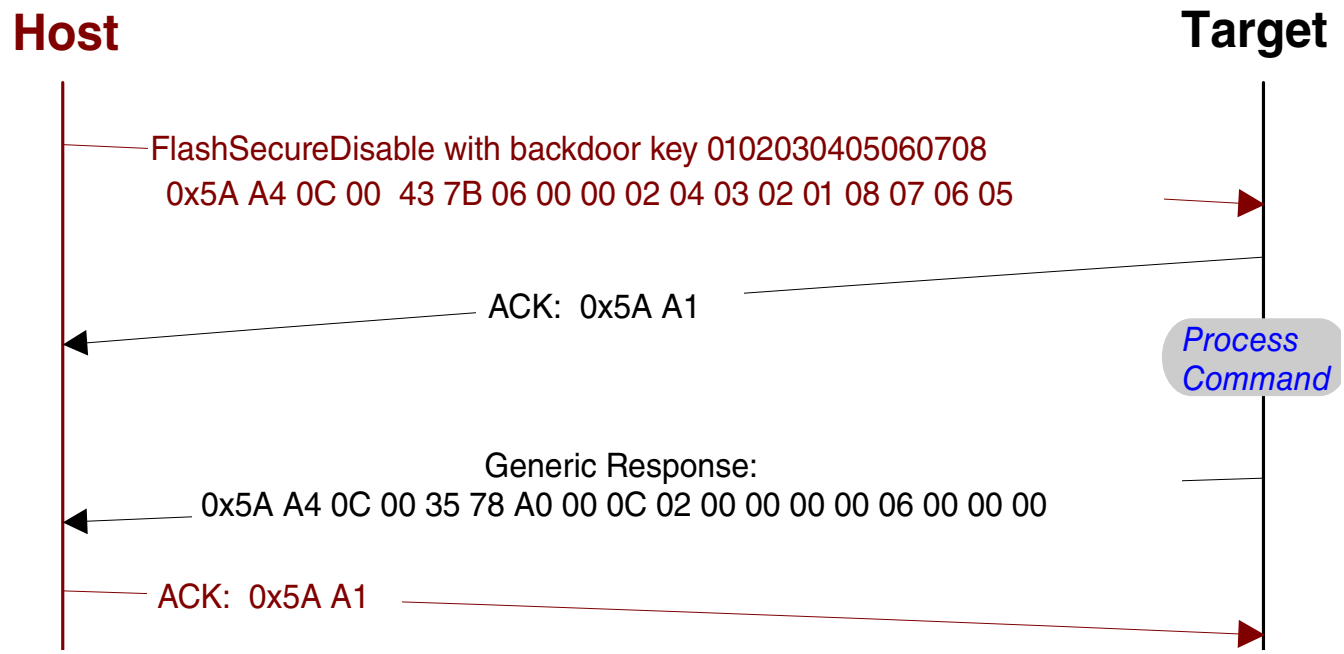


Figure 23-13. Protocol Sequence for FlashSecurityDisable Command

Table 23-31. FlashSecurityDisable Command Packet Format (Example)

FlashSecurityDisable	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x43 0x7B
Command packet	commandTag	0x06 - FlashSecurityDisable
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	Backdoorkey_low	0x04 0x03 0x02 0x01
	Backdoorkey_high	0x08 0x07 0x06 0x05

The FlashSecurityDisable command has no data phase.

Response: The target (KinetisBootloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

23.3.8.9 WriteMemory command

The WriteMemory command writes data provided in the data phase to a specified range of bytes in memory (flash or RAM). However, if flash protection is enabled, then writes to protected sectors will fail.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll, FlashEraseRegion, or FlashEraseAllUnsecure command.
- Writing to flash requires the start address to be 4-byte aligned ([1:0] = 00).
- The byte count will be rounded up to a multiple of 4, and the trailing bytes will be filled with the flash erase pattern (0xff).
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

The start address and number of bytes are the 2 parameters required for WriteMemory command.

Table 23-32. Parameters for WriteMemory Command

Byte #	Command
0 - 3	Start address
4 - 7	Byte count

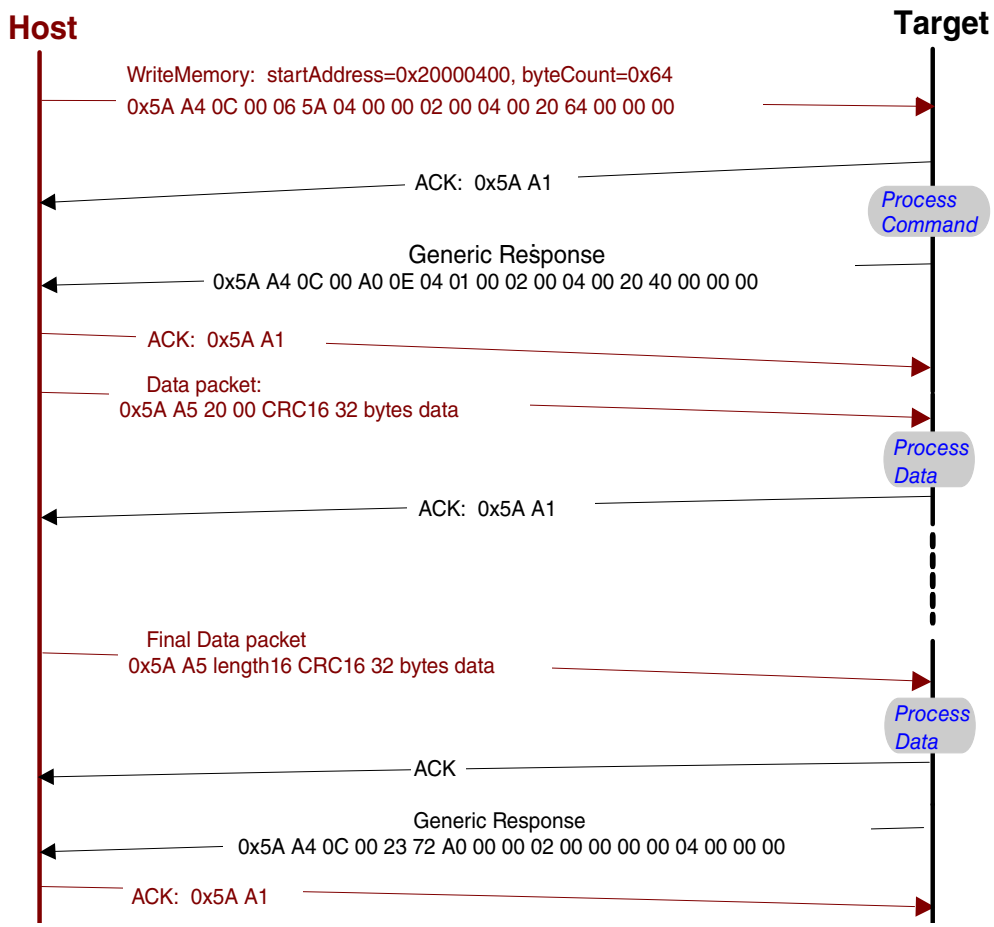


Figure 23-14. Protocol Sequence for WriteMemory Command

Table 23-33. WriteMemory Command Packet Format (Example)

WriteMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x06 0x5A
Command packet	commandTag	0x04 - writeMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The WriteMemory command has a data phase; the host will send data packets until the number of bytes of data specified in the byteCount parameter of the WriteMemory command are received by the target.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code set to kStatus_Success upon successful execution of the command, or to an appropriate error status code.

23.3.8.10 Read memory command

The ReadMemory command returns the contents of memory at the given address, for a specified number of bytes. This command can read any region of Flash memory, SRAM_L and SRAM_U memory accessible by the CPU and not protected by security.

The start address and number of bytes are the 2 parameters required for ReadMemory command.

Table 23-34. Parameters for read memory command

Byte	Parameter	Description
0-3	Start address	Start address of memory to read from
4-7	Byte count	Number of bytes to read and return to caller

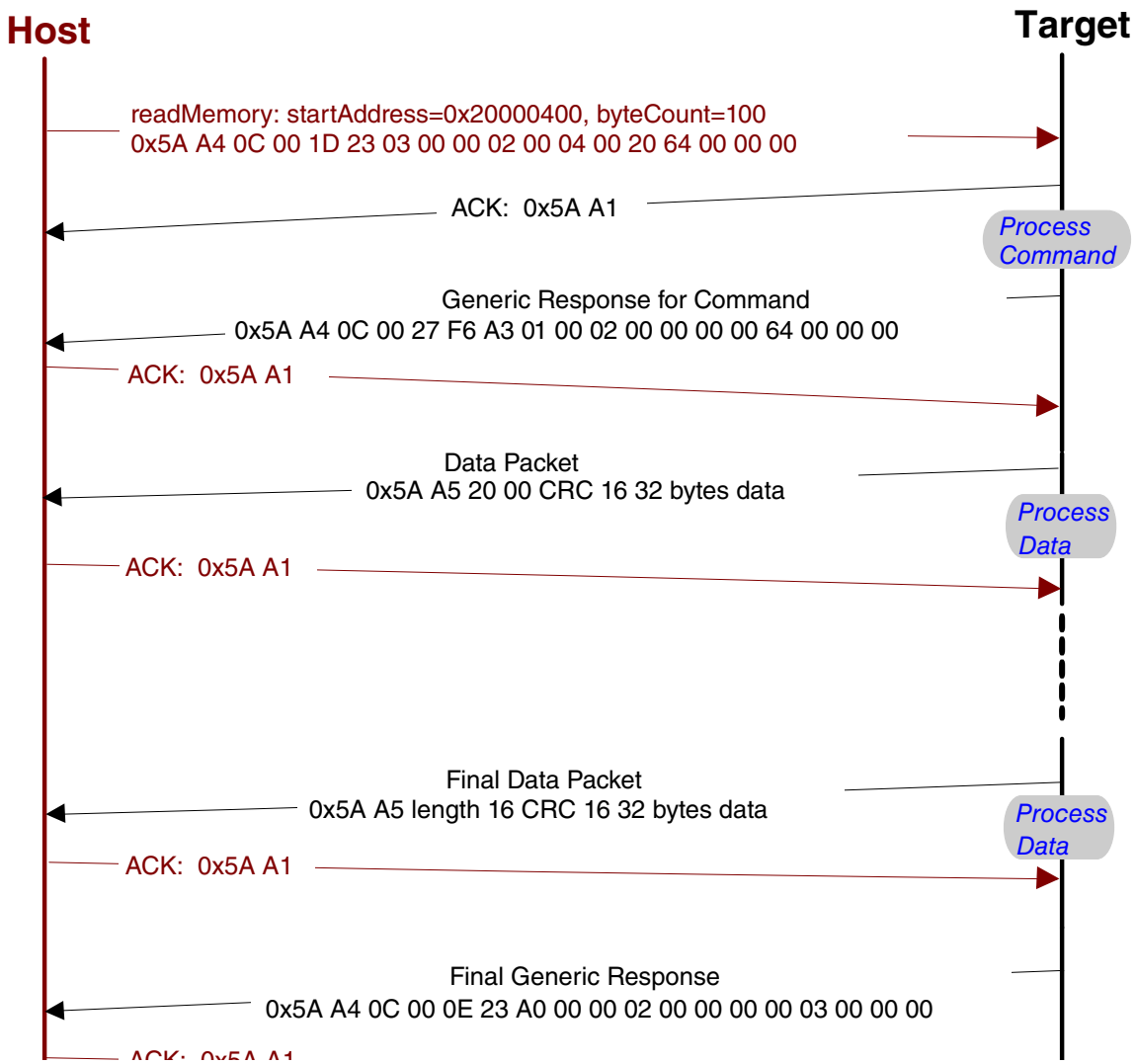


Figure 23-15. Command sequence for read memory

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A0xA4,
	packetType	kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x1D 0x23
Command packet	commandTag	0x03 - readMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The ReadMemory command has a data phase. Since the target (Kinetis Bootloader) works in slave mode, the host need pull data packets until the number of bytes of data specified in the byteCount parameter of ReadMemory command are received by host.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

23.3.9 Bootloader Exit state

The Kinetis Bootloader tries to reconfigure the system back to the reset state in the following situations:

- After completion of an Execute command, but before jumping to the specified entry point.
- After a peripheral detection timeout, but before jumping to the application entry point.

23.4 Kinetis Flash Driver API

To simplify flash code development, the Kinetis ROM Bootloader has flash driver APIs that user applications can use, and provides pointers to where these APIs are located. This section describes how to use each flash driver API provided in the Kinetis flash driver API tree.

NOTE

For more information on how to use the ROM-resident Flash Driver API from an application space, see the "Kinetis Flash Driver API" chapter of the latest "Kinetis Bootloader Reference Manual," http://www.nxp.com/KBOOT.

23.4.1 Flash Driver Entry Point

The MCU ROM bootloader provides a flash driver API tree entry (flashDriver) that a user application can use to get the entry points for the whole flash API set that is supported by the bootloader.

NOTE

The flashloader and flash-resident bootloader do not support this feature (flash driver API tree).

To get the address of the entry point, the user application reads the word containing the pointer to the bootloader API tree at offset 0x1C of the bootloader's vector table. The vector table is placed at the base of the bootloader's address range.

23.4.2 Flash driver API Tree

The bootloader API tree is a structure that contains pointers to other structures, which have the function and data address for the bootloader. The Flash driver API tree entry is always the 5th word of the API tree.

The prototype of the entry point is:

```
flash_driver_interface_t flashDriver;
```

There are several slightly different versions of the flash driver API among different targets with ROM bootloader.

Table 23-35. Different versions of the flash driver

Flash driver API version	Supported targets
V1.0	KL03Z4 KL43Z4 KL33Z4 KL27Z4 KL17Z4
V1.1	KL27Z644 KL17Z644
V1.2	KL13Z644 KL33Z644 K80F256 K81F256 K82F256 KL81Z7 KL82Z7 KL28Z7

There are minor differences in the flash driver interface among the flash driver API versions. See the definitions below.

```
typedef union BootloaderVersion
{
    struct
    {
        uint32_t bugfix : 8; //!lt; bugfix version [7:0]
        uint32_t minor : 8;  //!lt; minor version [15:8]
        uint32_t major : 8;  //!lt; major version [23:16]
        uint32_t name : 8;   //!lt; name [31:24]
    } B;
    uint32_t version; //!lt; combined version numbers
} standard_version_t;

//! @brief Interface for the flash driver.
typedef struct FlashDriverInterface
{
#if !defined(FLASH_API_TREE_1_0)
    standard_version_t version; //!lt; flash driver API version number.
#endif
```

```
    status_t (*flash_init)(flash_config_t *config);

#if defined(FLASH_API_TREE_1_0)
    status_t (*flash_erase_all)(flash_config_t *config);
    status_t (*flash_erase_all_unsecure)(flash_config_t *config);
    status_t (*flash_erase)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
#else
    status_t (*flash_erase_all)(flash_config_t *config, uint32_t key);
    status_t (*flash_erase_all_unsecure)(flash_config_t *config, uint32_t key);
    status_t (*flash_erase)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes,
uint32_t key);
#endif

    status_t (*flash_program)(flash_config_t *config, uint32_t start, uint32_t *src,
uint32_t lengthInBytes);
    status_t (*flash_get_security_state)(flash_config_t *config, flash_security_state_t
*state);
    status_t (*flash_security_bypass)(flash_config_t *config, const uint8_t *backdoorKey);
    status_t (*flash_verify_erase_all)(flash_config_t *config, flash_margin_value_t margin);
    status_t (*flash_verify_erase)(flash_config_t *config,
                                   uint32_t start,
                                   uint32_t lengthInBytes,
                                   flash_margin_value_t margin);
    status_t (*flash_verify_program)(flash_config_t *config,
                                     uint32_t start,
                                     uint32_t lengthInBytes,
                                     const uint32_t *expectedData,
                                     flash_margin_value_t margin,
                                     uint32_t *failedAddress,
                                     uint32_t *failedData);
    status_t (*flash_get_property)(flash_config_t *config, flash_property_tag_t
whichProperty, uint32_t *value);

#if (!defined(FLASH_API_TREE_1_0)) && (!defined(FLASH_API_TREE_1_1))
    status_t (*flash_register_callback)(flash_config_t *config, flash_callback_t callback);
    status_t (*flash_program_once)(flash_config_t *config, uint32_t index, uint32_t *src,
uint32_t lengthInBytes);
    status_t (*flash_read_once)(flash_config_t *config, uint32_t index, uint32_t *dst,
uint32_t lengthInBytes);
    status_t (*flash_read_resource)(flash_config_t *config,
                                    uint32_t start,
                                    uint32_t *dst,
                                    uint32_t lengthInBytes,
                                    flash_read_resource_option_t option);
#endif
} flash_driver_interface_t;
```

The NXP standard flash driver (C90TFS flash driver) is the basis for the flash driver API.

23.4.3 Quick demo using Kinetis Flash Driver API

The example code below uses the Kinetis Flash Driver API to erase a region of flash memory. For more code examples, get the latest Kinetis bootloader package at http://www.nxp.com/KBOOT/

```
bootloader_tree_t* tree;                // pointer points to bootloader tree
flash_config_t flash_config;          // variable used to keep runtime state of flash driver

// Get bootloader API tree from ROM
tree = (bootloader_tree_t*)(*(uint32_t*)0x1c00001c);
```

```
// Init flash driver.
status_t status = tree->flashDriver->flash_init(&flash_config);
if  (status == kStatus_Success)
   {
// Erase flash region from 0x800 to 0xc00.
     status = tree->flashDriver->flash_erase(&flash_config, 0x800, 1024);
   }
```

23.4.4 Flash driver data structures

23.4.4.1 flash_config_t

The `flash_config_t` data structure is a required argument for all flash driver API functions. `flash_config_t` is initialized by calling FLASH_Init. For other functions, an initialized instance of this data structure should be passed as an argument.

Table 23-36. `flash_config_t` data structure

Offset (hex)	Size	Field	Description
0	4	PFlashBlockBase	Base address of the first PFlash block
4	4	PFlashTotalSize	Size of all combined PFlash blocks
8	4	PFlashBlockCount	Number of PFlash blocks
C	4	PFlashSectorSize	Size (in bytes) of sector of PFlash
10	4	PFlashCallback	Pointer to a callback function used to do extra operations during erasure (for example, service watchdog)
14	4	PFlashAccessSegmentSize	Size of FAC access segment
18	4	PFlashAccessSegmentCount	Count of FAC access segment
1C	4	flashExecuteInRamFunctionInfo	Info struct of flash execute-in-ram function
20	4	FlexRAMBlockBase	• **FlexNVM device**: FlexRAM base address • **non-FlexNVM device**: acceleration RAM memory base address
24	4	FlexRAMTotalSize	• **FlexNVM device**: FlexRAM size • **non-FlexNVM device**: acceleration RAM memory size
28	4	DFlashBlockBase	• **FlexNVM device**: D-Flash memory (FlexNVM memory) base address • **non-FlexNVM device**: unused
2C	4	DFlashTotalSize	• **FlexNVM device**: FlexNVM memory total size • **non-FlexNVM device**: unused
30	4	EEpromTotalSize	• **FlexNVM device**: the size (in bytes) of the EEPROM area that was partitioned from FlexRAM • **non-FlexNVM device**: unused

`flash_config_t` prototype:

```
typedef struct _flash_config
{
    uint32_t PFlashBlockBase;                /*!< Base address of the first PFlash block */
    uint32_t PFlashTotalSize;                /*!< Size of all combined PFlash block. */
    uint32_t PFlashBlockCount;               /*!< Number of PFlash blocks. */
    uint32_t PFlashSectorSize;               /*!< Size in bytes of a sector of PFlash. */
    flash_callback_t PFlashCallback;         /*!< Callback function for flash API. */
    uint32_t PFlashAccessSegmentSize;        /*!< Size in bytes of a access segment of
PFlash. */
    uint32_t PFlashAccessSegmentCount;       /*!< Number of PFlash access segments. */
    uint32_t *flashExecuteInRamFunctionInfo; /*!< Info struct of flash execute-in-ram
function. */
    uint32_t FlexRAMBlockBase;               /*!< For FlexNVM device, this is the base
address of FlexRAM
                                                  For non-FlexNVM device, this is the base
address of acceleration RAM memory */
    uint32_t FlexRAMTotalSize;               /*!< For FlexNVM device, this is the size of
FlexRAM
                                                  For non-FlexNVM device, this is the size
of acceleration RAM memory */
    uint32_t DFlashBlockBase; /*!< For FlexNVM device, this is the base address of D-Flash
memory (FlexNVM memory);
                                   For non-FlexNVM device, this field is unused */
    uint32_t DFlashTotalSize; /*!< For FlexNVM device, this is total size of the FlexNVM
memory;
                                   For non-FlexNVM device, this field is unused */
    uint32_t EEpromTotalSize; /*!< For FlexNVM device, this is the size in byte of EEPROM
area which was partitioned
                                 from FlexRAM;
                                   For non-FlexNVM device, this field is unused */
} flash_config_t;
```

23.4.5 Flash driver API

This section describes each function supported in the flash driver API.

23.4.5.1 FLASH_Init

Checks and initializes the flash module for the other flash API functions.

NOTE

FLASH_Init must be always called before calling other API functions.

Prototype:

```
status_t FLASH_Init(flash_config_t *config);
```

Table 23-37. Parameters

Parameter	Description
config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.

Table 23-38. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config pointer is NULL.
100	kStatus_FLASH_SizeError	Returned flash is incorrect.
0	kStatus_Success	This function has performed successfully.

Example:

```
flash_config_t flashInstance;
status_t status = FLASH_Init(&flashInstance);
```

23.4.5.2 FLASH_EraseAll

Erases the entire flash array.

Prototype:

```
status_t FLASH_EraseAll(flash_config_t *config, uint32_t key);
```

Table 23-39. Parameters

Parameter	Description
config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
key	Key used to validate erase operation. Must be set to 0x6B65666B.

Table 23-40. Possible status response

Value	Constants	Description
4	kStatus_InvalidArgument	Config pointer is NULL.
103	kStatus_FLASH_AccessError	Command is not available under current mode/ security.
104	kStatus_FLASH_ProtectionViolation	Any region of the program flash memory is protected.
107	kStatus_FLASH_EraseKeyError	Key is incorrect.
0	kStatus_Success	This function has performed successfully.

Example:

```
status_t status = FLASH_EraseAll(&flashInstance, kFLASH_ApiEraseKey);
```

23.4.5.3 FLASH_EraseAllUnsecure

Erases the entire flash (including protected sectors) and restores flash to unsecured mode.

Prototype:

```
status_t FLASH_EraseAllUnsecure(flash_config_t *config, uint32_t key);
```

Table 23-41. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Key	Key used to validate erase operation. Must be set to 0x6B65666B.

Table 23-42. Possible Status Response

Value	Constant	Description
4	kStatus_InvalidArgument	Config pointer is NULL.
103	kStatus_FLASH_AccessError	Command is not available under current mode/ security.
107	kStatus_FLASH_EraseKeyError	Key is incorrect.
0	kStatus_Success	This function has performed successfully.

Example:

```
status_t status = FLASH_EraseAllUnsecure(&flashInstance, kFLASH_ApiEraseKey);
```

23.4.5.4 FLASH_Erase

Erases expected flash sectors specified by parameters. For Kinetis devices, the minimum erase unit is one sector.

Prototype:

```
status_t FLASH_Erase(flash_config_t *config, uint32_t start, uint32_t lengthInBytes, uint32_t
      key);
```

Table 23-43. Parameters

Parameters	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Start	The start address of the desired flash memory to be erased.

Table continues on the next page...

Table 23-43. Parameters (continued)

Parameters	Description
	The start address does not need to be sector aligned, but must be word-aligned.
lengthInBytes	The length, given in bytes (not words or long words) to be erased. Must be word-aligned.
Key	Key is used to validate erase operation. Must be set to 0x6B65666B.

Table 23-44. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config pointer is NULL.
100	kStatus_FLASH_AlignmentError	Start or lengthInBytes; is not long word-aligned.
102	kStatus_FLASH_AddressError	The range to be erased is not a valid flash range.
103	kStatus_FLASH_AccessError	Command is not available under current mode/ security.
104	kStatus_FLASH_ProtectionViolation	The selected program flash sector is protected.
107	kStatus_FLASH_EraseKeyError	Key is incorrect.
0	kStatus_Success	This function has performed successfully.

Example:

```
status_t status = FLASH_Erase (&flashInstance, 0x800, 1024, kFLASH_ApiEraseKey);
```

23.4.5.5 FLASH_Program

Programs the flash memory with data at locations that are passed in using parameters.

Prototype:

```
status_t FLASH_Program(flash_config_t *config, uint32_t start, uint32_t *src, uint32_t
      lengthInBytes);
```

Table 23-45. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Start	The start address of the desired flash memory to be erased. The start address does not need to be sector-aligned, but the start address must be word-aligned.
src	Pointer to the source buffer of data that is to be programmed into flash.
lengthInBytes	The length in bytes (not words or long words) to be erased; the length must also be word-aligned.

Table 23-46. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or src pointers are NULL.
101	kStatus_FLASH_AlignmentError	Start or lengthInBytes is not longword aligned.
102	kStatus_FLASH_AddressError	The range to be programmed is invalid.
103	kStatus_FLASH_AccessError	Command is not available under current mode/ security.
104	kStatus_FLASH_ProtectionViolation	The selected program flash address is protected.
0	kStatus_Success	This function has performed successfully.

Example:

```
uint32_t m_content[] = {0x01234567, 0x89abcdef};
status_t status = FLASH_Program (&flashInstance, 0x800, &m_content[0], sizeof(m_content));
```

NOTE

Before calling `flash_program`, make sure that the region to be programmed is empty and is not protected.

23.4.5.6 FLASH_GetSecurityState

Retrieves the current flash security status, including the security enabling state and the backdoor key enabling state.

Prototype:

```
status_t FLASH_GetSecurityState(flash_config_t *config, flash_security_state_t *state);
```

Table 23-47. Parameters

Parameters	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
State	Pointer to the value returned for the current security status code: **Table 23-48. Returned value** kFLASH_SecurityStateNotSecure — 0 — Flash is under unsecured mode. kFLASH_SecurityStateBackdoorEnabled — 1 — Flash is under secured mode and Backdoor is enabled. kFLASH_SecurityStateBackdoorDisabled — 2 — Flash is under secured mode and Backdoor is disabled.

Table 23-49. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or state pointers are NULL.
0	kStatus_Success	This function has performed successfully.

Example:

```
flash_security_state_t state;
status_t status = FLASH_GetSecurityState (&flashInstance, &state);
```

23.4.5.7 FLASH_SecurityBypass

Allows the user to bypass security with a backdoor key. If the MCU is in a secured state, then the FLASH_SecurityBypass function unsecures the MCU, by comparing the provided backdoor key with keys in the Flash Configuration Field.

Prototype:

```
status_t FLASH_SecurityBypass(flash_config_t *config, const uint8_t *backdoorKey);
```

Table 23-50. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
backdoorKey	Pointer to the user buffer containing the backdoor key.

Table 23-51. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or backdoorKey pointers are NULL.
103	kStatus_FLASH_AccessError	The following condition causes this return value: 1. An incorrect backdoor key is supplied 2. Backdoor key access has not been enabled.
0	kStatus_Success	This function has performed successfully.

Example:

Assume that the flash range from 0x400 to 0x40c contains the following content after the last reset, which means that the backdoor key is valid and the backdoor key access has been enabled.

0x11 0x22 0x33 0x44 0x55 0x66 0x77 0x88 0xff 0xff 0xff 0xbf

```
uint8_t backdoorKey[] = {0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88};
status_t status = FLASH_SecurityBypass (&flashInstance, & backdoorKey[0]);
```

23.4.5.8 FLASH_VerifyEraseAll

Checks if the entire flash has been erased to the specified read margin level.

To verify if the entire flash has been fully erased (after executing an FLASH_EraseAll), call FLASH_VerifyEraseAll.

Prototype:

```
status_t FLASH_VerifyEraseAll(flash_config_t *config, flash_margin_value_t margin);
```

Table 23-52. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Margin1	Read margin choice: • kFLASH_MarginValueNormal 0 • kFLASH_MarginValueUser 1 • kFLASH_MarginValueFactory 2

Table 23-53. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or backdoorKey pointers are NULL.
103	kStatus_FLASH_AccessError	An invalid margin choice is specified.
105	kStatus_FLASH_CommandFailure	The entire flash is not fully erased.
0	kStatus_Success	This function has performed successfully.

Example:

Assume that flash_erase_all has been successfully executed.

```
status_t status = flash_verify_erase_all (&flashInstance, kFLASH_MarginValueUser);
```

NOTE

For the choice of margin, see the FTFA chapter in the reference manual for detailed information.

23.4.5.9 FLASH_VerifyErase

Verifies the erasure of the desired flash area at a specified margin level. This function checks the appropriate number of flash sectors based on the desired start address and length, to see if the flash has been erased at the specified read margin level.

FLASH_VerifyErase is often called after successfully performing the FLASH_Erase API.

Prototype:

```
status_t FLASH_VerifyErase(flash_config_t *config, uint32_t start, uint32_t lengthInBytes,
flash_margin_value_t margin);
```

Table 23-54. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Start	The start address of the desired flash memory to be verified.
lengthInBytes	The length, given in bytes (not words or long words) to be verified. Must be word-aligned.
margin	Read margin choice as follows: kFLASH_MarginValueNormal 0 kFLASH_MarginValueUser 1 kFLASH_MarginValueFactory 2

Table 23-55. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or backdoorKey pointers are NULL.
101	kStatus_FLASH_AlignmentError	Start or lengthInBytes is not longword aligned.
102	kStatus_FLASH_AddressError	The range to be verified is not a valid flash range.
103	kStatus_FlashAccessError	The following situation causes this response: 1. Command is not available under current mode/ security 2. An invalid margin code is provided 3. The requested number of bytes is 0 4. The requested sector crosses a flash block boundary
105	kStatus_FLASH_CommandFailure	The flash range to be verified is not fully erased.
0	kStatus_Success	This function has performed successfully.

Example:

Assume that flash region from 0x800 to 0xc00 has been successfully erased.

```
status_t status = FLASH_VerifyErase(&flashInstance, 0x800, 1024, kFLASH_MarginValueUser);
```

NOTE

For the choice of margin, see the FTFA chapter in the reference manual for detailed information.

23.4.5.10 FLASH_VerifyProgram

Verifies the data programmed in the flash memory (using the Flash Program Check Command), and compares it with expected data for a given flash area (as determined by the start address and length).

FLASH_VerifyProgram is often called after successfully doing FLASH_Program().

Prototype:

```
status_t FLASH_VerifyProgram(flash_config_t *config,
                             uint32_t start,
                             uint32_t lengthInBytes,
                             const uint32_t *expectedData,
                             flash_margin_value_t margin,
                             uint32_t *failedAddress,
                             uint32_t *failedData);
```

Table 23-56. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Start	The start address of the desired flash memory to be verified.
LengthInBytes	The length, given in bytes (not words or long-words) to be verified. Must be word-aligned.
ExpectedData	Pointer to the expected data that is to be verified against.
Margin	Read margin choice as follows: kFLASH_MarginValueUser 1 kFLASH_MarginValueFactory 2
FailedAddress	Pointer to returned failing address.
FailedData	Pointer to return failing data. Some derivatives do not include failed data as part of the FCCOBx registers. In this instance, 0x00s are returned upon failure.

Table 23-57. Possible status response

Value	Contants	Description
4	kStatus_InvalidArgument	Config or expectedData pointers are NULL.
101	kStatus_FlashAlignmentError	Start or lengthInBytes is not longword-aligned.
102	kStatus_FLASH_AddressError	The range to be verified is invalid.
103	kStatus_FLASH_AccessError	The following situation causes this response: 1. Command is not available under current mode/ security. 2. An invalid margin code is supplied.

Table continues on the next page...

Table 23-57. Possible status response (continued)

Value	Contants	Description
105	kStatus_FLASH_CommandFailure	Either of the margin reads does not match the expected data.
0	kStatus_Success	This function has performed successfully.

Example:

Assume that flash region from 0x800 to 0x807 is successfully programmed with:

0x01 0x23 0x45 0x67 0x89 0xab 0xcd 0xef

```
uint8_t expectedData[] = {0x01, 0x23, 0x45, 0x67, 0x89, 0xab, 0xcd, 0xef };
status_t status = FLASH_VerifyProgram (&flashInstance, 0x800, 8,
                   &expectedData[0], kFlashMargin_User, NULL, NULL);
```

NOTE

For the choice of margin, see the FTFA chapter in the reference manual for detailed information.

23.4.5.11 FLASH_GetProperty

Returns the desired flash property, which includes base address, sector size, and other options.

Prototype:

```
status_t flash_get_property(flash_driver_t * driver, flash_property_t whichProperty, uint32_t
* value);
```

Table 23-58. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory to store driver runtime state.
whichProperty	The desired property from the list of properties.

Table 23-59. Properties

Definition	Value	Description
kFLASH_PropertyPflashSectorSize	0	Get Flash Sector size
kFLASH_PropertyPflashTotalSize	1	Get total flash size
kFLASH_PropertyPflashBlockBaseAddr	4	Get flash base address
kFLASH_PropertyPflashFacSupport	5	Get FAC support status
kFLASH_PropertyPflashAccessSegmentSize	6	Get FAC segment size

Table continues on the next page...

Table 23-58. Parameters (continued)

Parameter	Description
	Table 23-59. Properties (continued)

Definition	Value	Description
kFLASH_PropertyPflashAccessSegmentCount	7	Get FAC segment count
kFLASH_PropertyVersion	32	Get version of Flash Driver API

Value	Pointer to the value returned for the desired flash property.

Table 23-60. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or value pointers are invalid.
106	kStatus_FLASH_UnknownProperty	Invalid property is supplied.
0	kStatus_Success	This function has performed successfully.

Example:

```
uint32_t propertyValue;
status_t status = FLASH_GetProperty (&flashInstance, kFLASH_PropertyPflashSectorSize,
&propertyValue);
```

23.4.5.12 FLASH_ProgramOnce

Programs a certain Program Once Field with the expected data for a given IFR region (as determined by the index and length).

- For each Program Once Field, FLASH_ProgramOnce can only allowed to be called once; otherwise, an error code is returned.
- For targets which do not support FLASH_ProgramOnce, the value of the FLASH_ProgramOnce pointer is 0.

Prototype

```
status_t flash_program_once (flash_driver_t * driver, uint32_t index, uint32_t *src, uint32_t
      lengthInBytes);
```

Table 23-61. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Index	Index for a certain Program Once Field.

Table continues on the next page...

Table 23-61. Parameters (continued)

Parameter	Description
src	Pointer to the source buffer of data that is to be programmed into the Program Once Field.
Lengthinbytes	The length, in bytes (not words or long words) to be programmed. Must be word-aligned.

Table 23-62. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or src pointers are NULL.
101	kStatus_FLASH_AlignmentError	index or lengthInBytes is invalid.
103	kStatus_FLASH_AddressError	The following situation causes this response: 1. Command is not available under current mode/ security. 2. An invalid index is supplied. 3. The requested Program Once field has already been programmed to a non-FFFF value. 4. The requested sector crosses a flash block boundary.
115	kStatus_FLASH_CommandNotSupported	This function is not supported.
0	kStatus_Success	This function has performed successfully.

Example:

Assume the Program Once Field has not been programmed before.

```
uint32_t expectedData = 0x78563412;

status_t status = FLASH_ProgramOnce(&flashInstance, 0, &expectedData, 4);
```

NOTE

For the choice of index and length, see the FTFA chapter in RM for detailed information.

23.4.5.13 FLASH_ReadOnce

Reads a certain flash Program Once Field according to parameters passed by index and length.

For targets that do not support FLASH_ReadOnce, the value of the FLASH_ReadOnce pointer is 0.

Prototype:

```
status_t flash_read_once (flash_driver_t * driver, uint32_t index, uint32_t *dst, uint32_t
lengthInBytes);
```

Table 23-63. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Index	Index for a certain Program Once Field.
dst	Pointer to the destination buffer of data that stores data reads from the Program Once Field.
Lengthinbytes	The length, in bytes (not words or long words) to be read. Must be word-aligned.

Table 23-64. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or dst pointers are NULL.
101	kStatus_FlashAlignmentError	Index or lengthInBytes is invalid.
103	kStatus_FLASH_AddressError	The following situation causes this response: 1. Command is not available under current mode/ security. 2. An invalid index is supplied.
115	kStatus_FLASH_CommandNotSupported	This function is not supported.
0	kStatus_Success	This function has performed successfully.

Example:

```
uint32_t temp;
      status_t status = FLASH_ReadOnce(&flashInstance, 0, &temp, 4);
```

NOTE

For the choice of index and length, see the FTFA chapter in RM for detailed information.

23.4.5.14 FLASH_ReadResource

Reads certain regions of IFR determined by the start address, length, and option.

For targets that do not support FLASH_ReadResource, the value of the FLASH_ReadResource pointer is 0.

Prototype:

```
status_t FLASH_ReadResource(
    flash_config_t *config, uint32_t start, uint32_t *dst, uint32_t lengthInBytes,
flash_read_resource_option_t option);
```

Table 23-65. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Start	Index for a certain Program Once Field.
dst	Pointer to the destination buffer of data that stores data reads from IFR.
Lengthinbytes	The length, in bytes (not words or long words), to be read. Must be word-aligned.
Option	The resource option which indicates the area that needs be read back. • 0 IFR • 1 Version ID of the flash module

Table 23-66. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or dst pointers are NULL.
101	kStatus_FLASH_AlignmentError	Start, lengthInBytes, or option is invalid.
103	kStatus_FLASH_AccessError	The following situation causes this response: 1. Command is not available under current mode/ security. 2. An invalid index is supplied. 3. An invalid resource option. 4. Address is out-of-rage for the targeted resource. 5. Address is not long word aligned.
115	kStatus_FLASH_CommandNotSupported	This function is not supported.
0	kStatus_Success	This function has performed successfully.

Example:

```
uint32_t temp[256];
status_t status = FLASH_ReadResource(&flashInstance, 0, &temp[0], 256, 0);
```

NOTE

See the FTFA chapter in RM for detailed information regarding the start, length, and option choices.

23.4.5.15 FLASH_SetCallback

Registers (like to write into a list) expected callback functions into the flash driver, for example, like a function that services a watchdog.

Prototype:

```
status_t FLASH_SetCallback(flash_config_t *config, flash_callback_t callback);
```

Table 23-67. Parameters

Parameter	Description
Config	Pointer to `flash_config_t` data structure in memory, to store driver runtime state.
Callback	A pointer points to a function that is called during erasure. A use for this function is to service the watchdog during an erase operation.

Table 23-68. Possible status response

Value	Constant	Description
4	kStatus_InvalidArgument	Config or dst pointers are NULL.
115	kStatus_FLASH_CommandNotSupported	This function is not supported.
0	kStatus_Success	This function has performed successfully.

Example:

Assume that there is a function.

```
void led_toggle(void).
status_t status = FLASH_SetCallback(&flashInstance, led_toggle);
```

23.5 Peripherals Supported

This section describes the peripherals supported by the Kinetis ROM Bootloader. To use an interface for bootloader communications, the peripheral must be enabled in the BCA, as shown in Table 23-2. If the BCA is invalid (such as all 0xFF bytes), then all peripherals will be enabled by default.

23.5.1 I2C Peripheral

The Kinetis Bootloader in ROM supports loading data into flash via the I2C peripheral, where the I2C peripheral serves as the I2C slave. A 7-bit slave address is used during the transfer.

Customizing an I2C slave address is also supported. This feature is enabled if the Bootloader Configuration Area (BCA) (shown in Table 23-2) is enabled (tag field is filled with ‘kcfg’) and the i2cSlaveAddress field is filled with a value other than 0xFF. 0x10 is used as the default I2C slave address.

The maximum supported I2C baud rate depends on corresponding clock configuration field in the BCA. Typical supported baud rate is 400 kbps with factory settings. Actual supported baud rate may be lower or higher than 400 kbps, depending on the actual value of the clockFlags and the clockDivider fields.

Because the I2C peripheral serves as an I2C slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

- An incoming packet is sent by the host with a selected I2C slave address and the direction bit is set as write.
- An outgoing packet is read by the host with a selected I2C slave address and the direction bit is set as read.
- 0x00 will be sent as the response to host if the target is busy with processing or preparing data.

The following flow charts demonstrate the communication flow of how the host reads ping packet, ACK and response from the target.

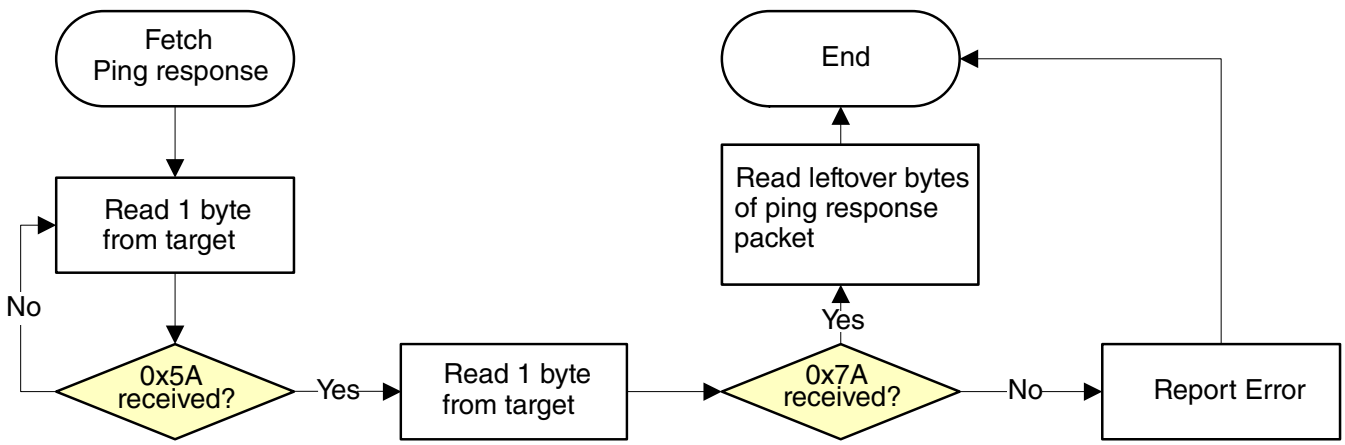


Figure 23-16. Host reads ping response from target via I2C

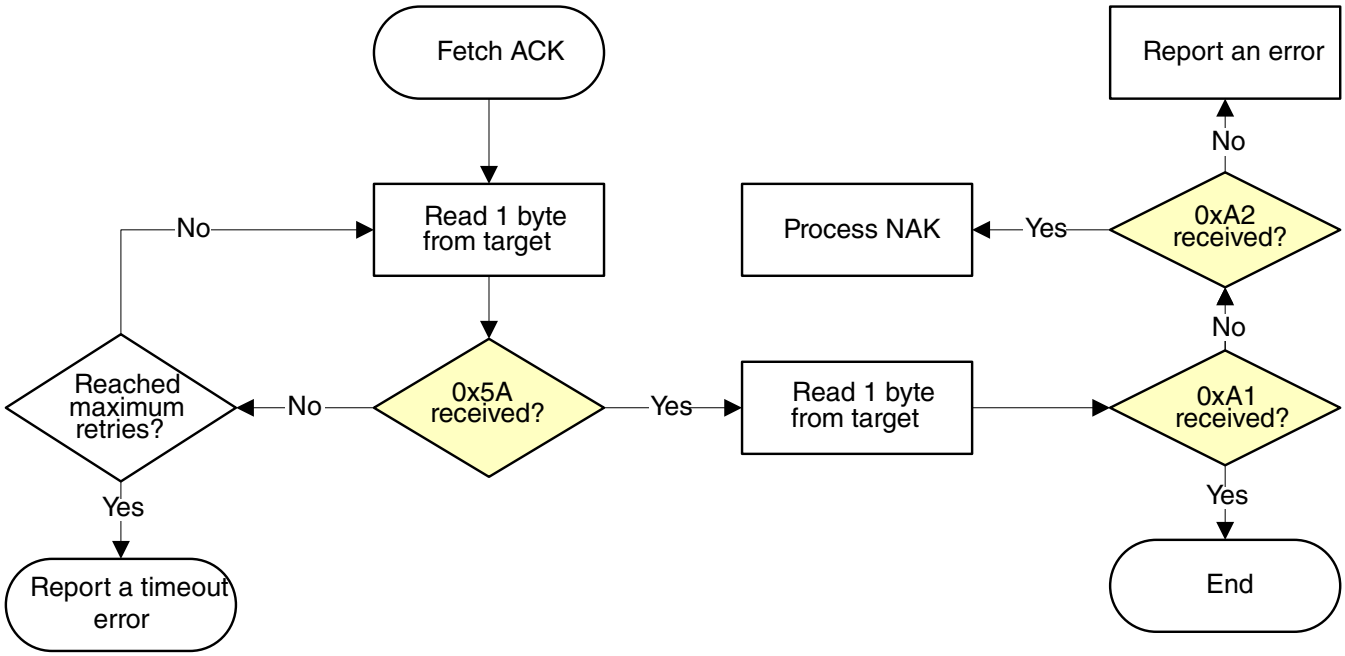


Figure 23-17. Host reads ACK packet from target via I2C

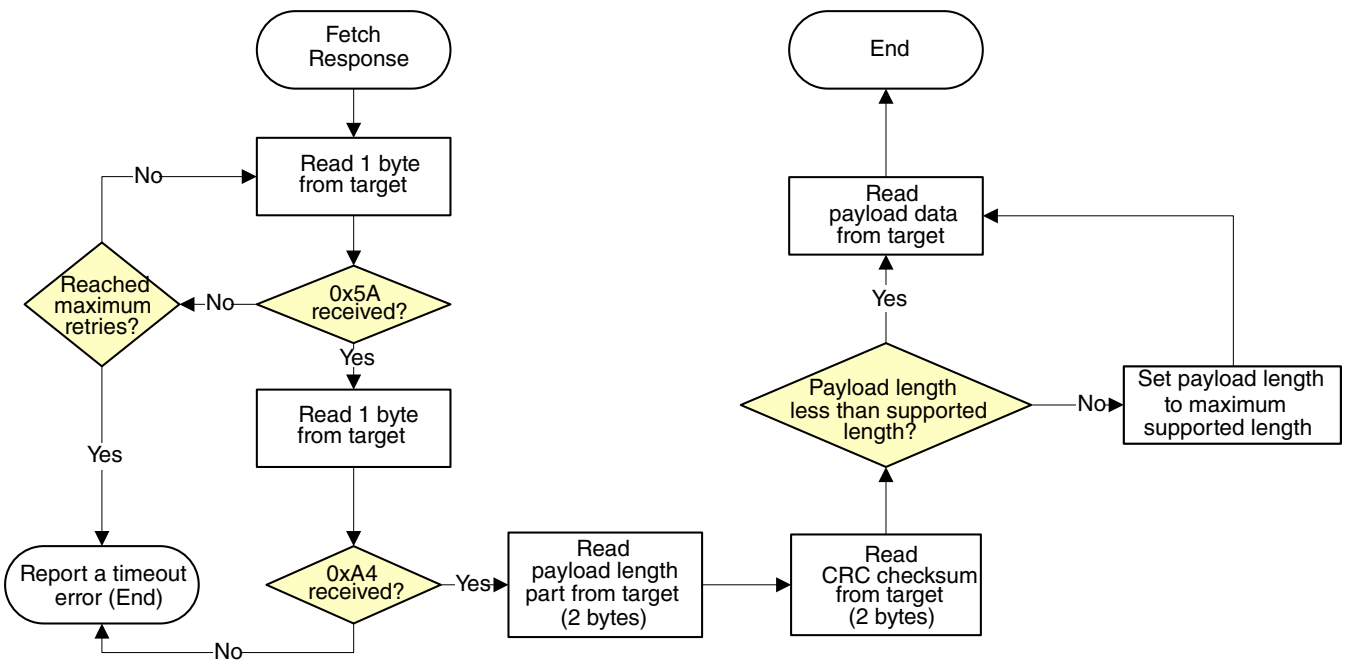


Figure 23-18. Host reads response from target via I2C

23.5.2 SPI Peripheral

The Kinetis Bootloader in ROM supports loading data into flash via the SPI peripheral, where the SPI peripheral serves as a SPI slave.

Maximum supported baud rate of SPI depends on the clock configuration fields in the Bootloader Configuration Area (BCA) shown in Table 23-2. The typical supported baud rate is 400 kbps with the factory settings. The actual baud rate is lower or higher than 400 kbps, depending on the actual value of the clockFlags and clockDivider fields in the BCA.

The SPI peripheral uses the following bus attributes:

- Clock Phase = 1 (Second Edge)
- Clock Polarity = 1 (Active Low)

Because the SPI peripheral in ROM serves as a SPI slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

The transfer on SPI is slightly different from I2C:

- Host will receive 1 byte after it sends out any byte.
- Received bytes should be ignored when host is sending out bytes to target
- Host starts reading bytes by sending 0x00s to target
- The byte 0x00 will be sent as response to host if target is under the following conditions:
  - Processing incoming packet
  - Preparing outgoing data
  - Received invalid data

The SPI bus configuration is:

- Phase = 1; data is sampled on rising edges
- Polarity = 1; idle is high
- MSB is transmitted first

For any transfer where the target does not have actual data to send, the target (slave) is responsible for ensuring that 0x00 bytes will be returned to the host (master). The host uses framing packets to identify real data and not "dummy" 0x00 bytes (which do not have framing packets).

The following flowcharts demonstrate how the host reads a ping response, an ACK and a command response from target via SPI.

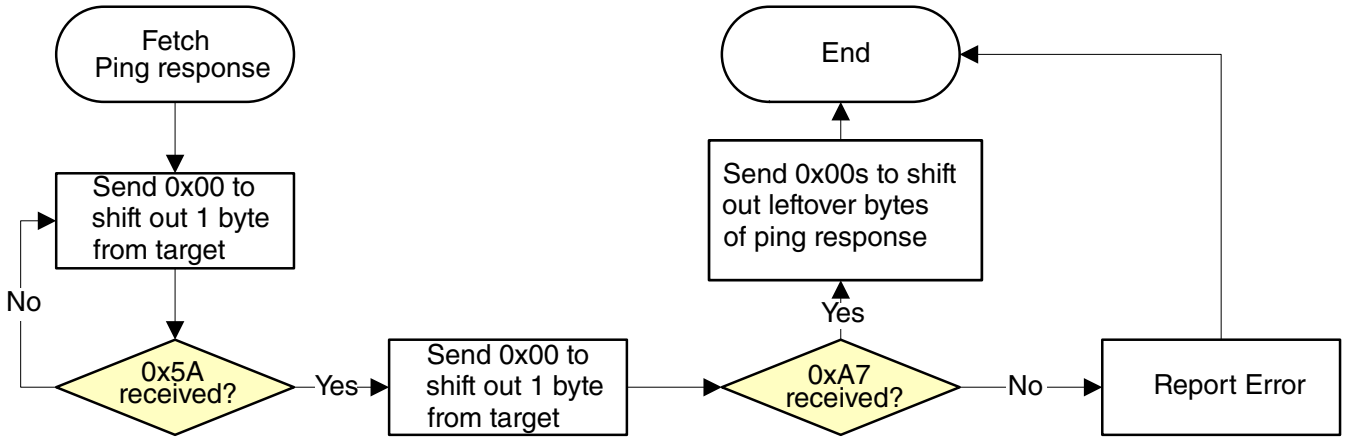


Figure 23-19. Host reads ping packet from target via SPI

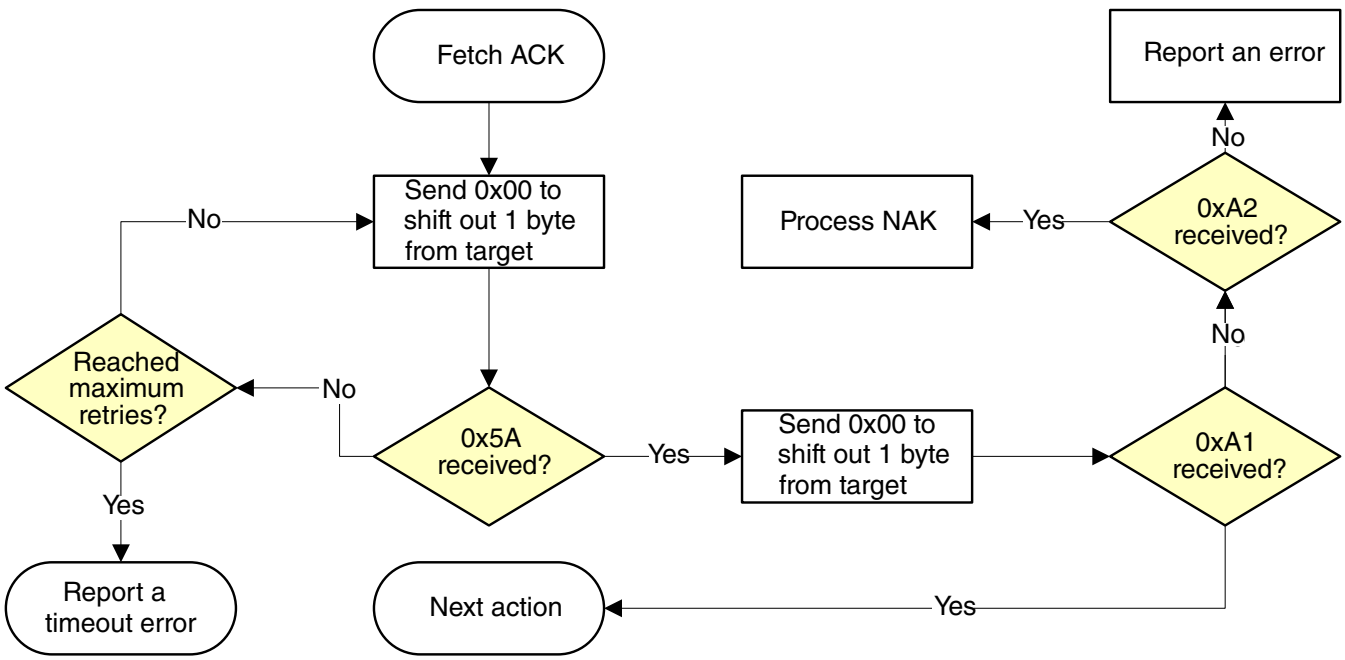


Figure 23-20. Host reads ACK from target via SPI

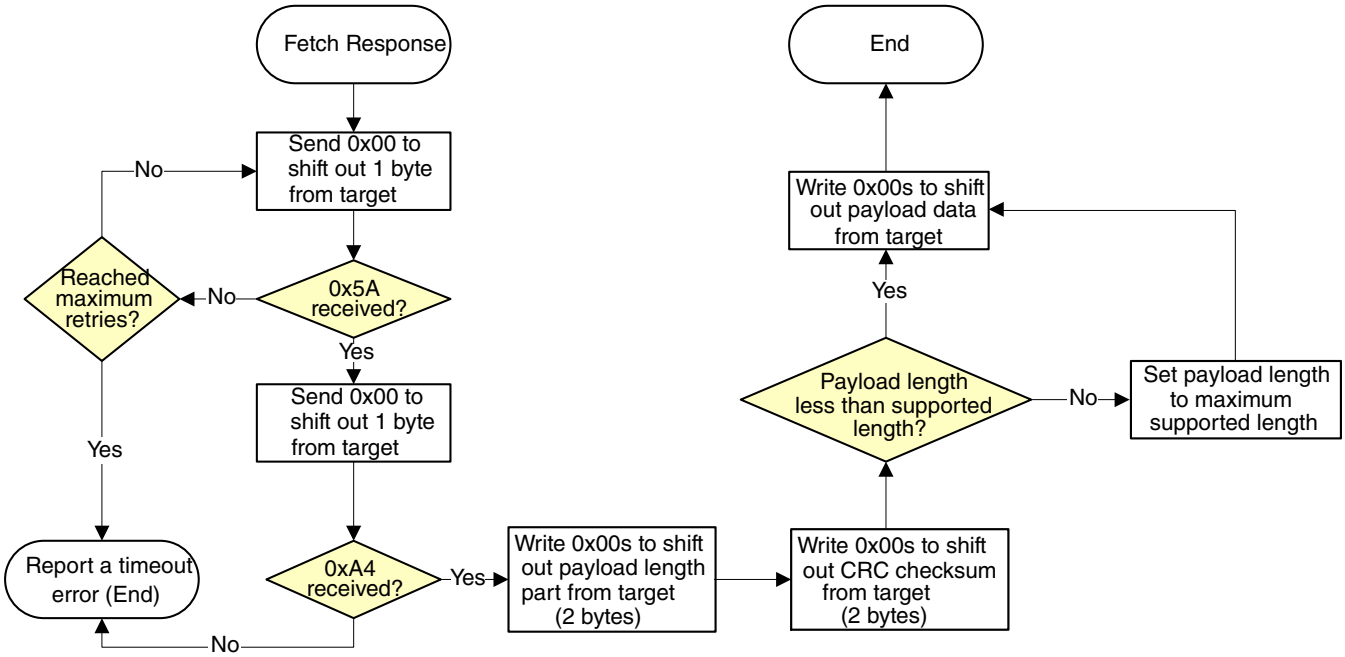


Figure 23-21. Host reads response from target via SPI

23.5.3 UART Peripheral

The Kinetis Bootloader integrates an autobaud detection algorithm for the UART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If UART*n* is used to connect to the bootloader, then the UART*n*_RX pin must be kept high and not left floating during the detection phase in order to comply with the autobaud detection algorithm. After the bootloader detects the ping packet (0x5A 0xA6) on UART*n*_RX, the bootloader firmware executes the autobaud sequence. If the baudrate is successfully detected, then the bootloader will send a ping packet response [(0x5A 0xA7), protocol version (4 bytes), protocol version options (2 bytes) and crc16 (2 bytes)] at the detected baudrate. The Kinetis Bootloader then enters a loop, waiting for bootloader commands via the UART peripheral.

NOTE

- The autobaud feature requires a ping packet with a higher accuracy (+/-3%), or the ping packet will be ignored as noise.
- The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed UART transmission mode (8-bit data, no parity bit and 1 stop bit). If the bytes of the ping packet are sent one-by-one with more than 80 ms delay between them, then the autobaud

detection algorithm may calculate an incorrect baud rate. In this case, the autobaud detection state machine should be reset.

Supported baud rates: The baud rate is closely related to the MCU core and system clock frequencies. Typical baud rates supported are 9600, 19200, 38400, 57600, and 115200. Of course, to influence the performance of autobaud detection, the clock configuration in BCA can be changed.

Packet transfer: After autobaud detection succeeds, bootloader communications can take place over the UART peripheral. The following flow charts show:

- How the host detects an ACK from the target
- How the host detects a ping response from the target
- How the host detects a command response from the target

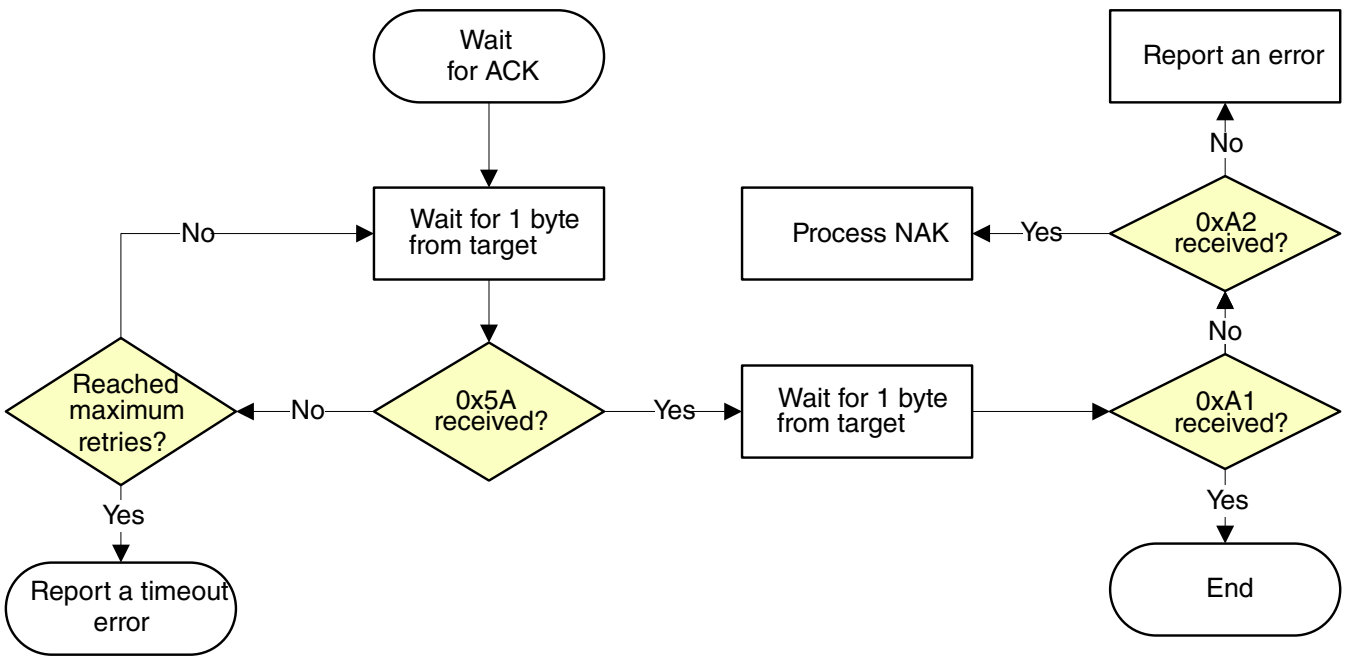


Figure 23-22. Host reads an ACK from target via UART

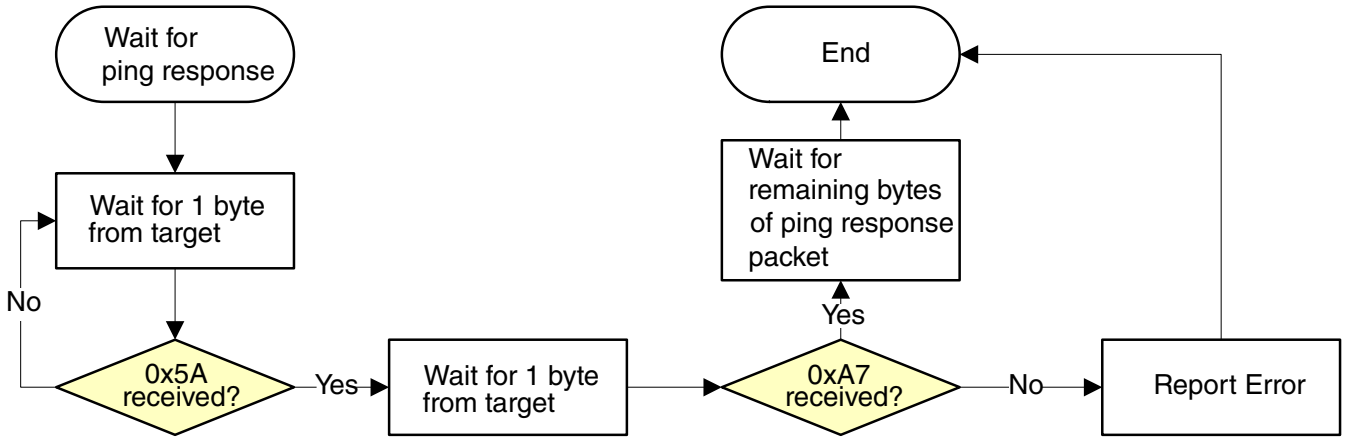


Figure 23-23. Host reads a ping response from target via UART

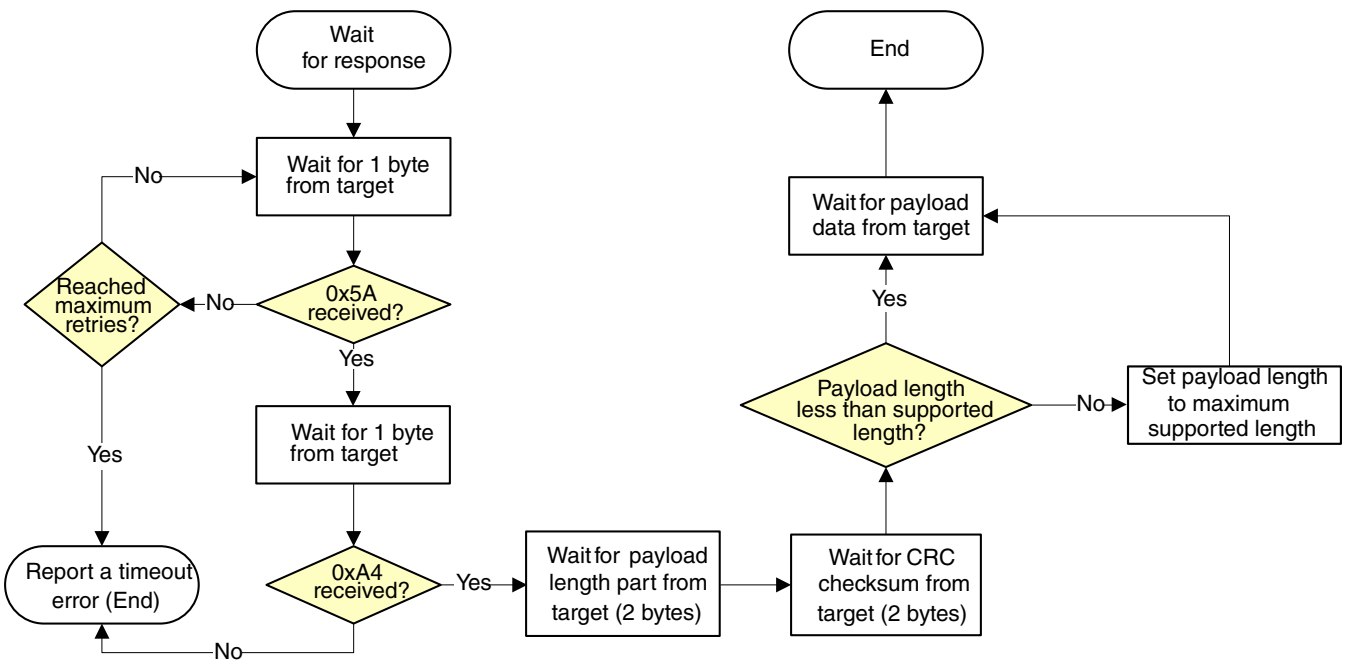


Figure 23-24. Host reads a command response from target via UART

23.5.4 FlexCAN Peripheral

The Kinetis Bootloader in this device's ROM supports loading data into flash via the FlexCAN peripheral.

It supports 5 predefined speeds on FlexCAN transferring:

- 125 kHz
- 250 kHz
- 500 kHz

- 750 kHz
- 1 MHz

The current FlexCAN module can support up to 1 MHz speed. So the default speed is set to 1 MHz.

In host applications, user can specify the speed for FlexCAN by giving the speed index 0 through 4 that represents those 5 speeds.

In Bootloader, it supports auto speed detection feature within supported speeds. The Bootloader will enter the listen mode in the beginning with the initial speed (default speed 1 MHz). Once the host starts sending a ping to a specific node, it will generate traffic on FlexCAN bus. Since the Bootloader is in a listen mode, it will be able to check if the local node speed is correct by detecting errors. If there is an error, that means we see some traffic but may not be on the right speed to see the real data, it will change the speed setting and check again. If there is no error, which means the speed should be correct, now it changes the settings back to normal receiving mode to see if there is a package for this node. (It then will stay in this speed until another host is using another speed and try to communicate with any node. It will repeat the process to detect a right speed before sending host timeout and abort the request.)

The host side should also have reasonable time tolerance during the auto speed detect period. If the sending is timeout, which means there is no response from the specific node or there is a real error, it will need to report the error to the application.

The following flow charts demonstrate the communication flow of how the host reads ping packet, ACK and response from the target.

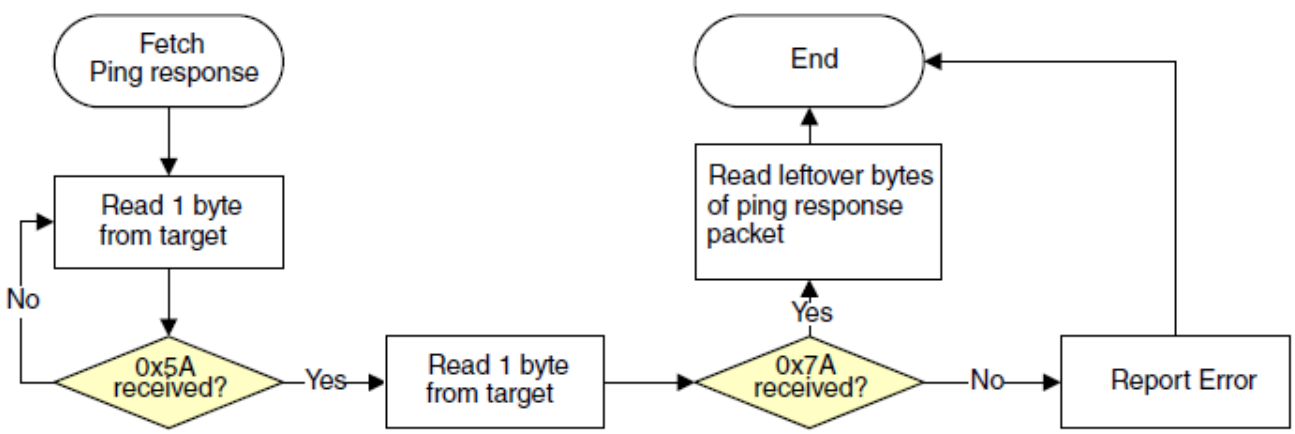


Figure 23-25. Host reads ping response from target via FlexCAN

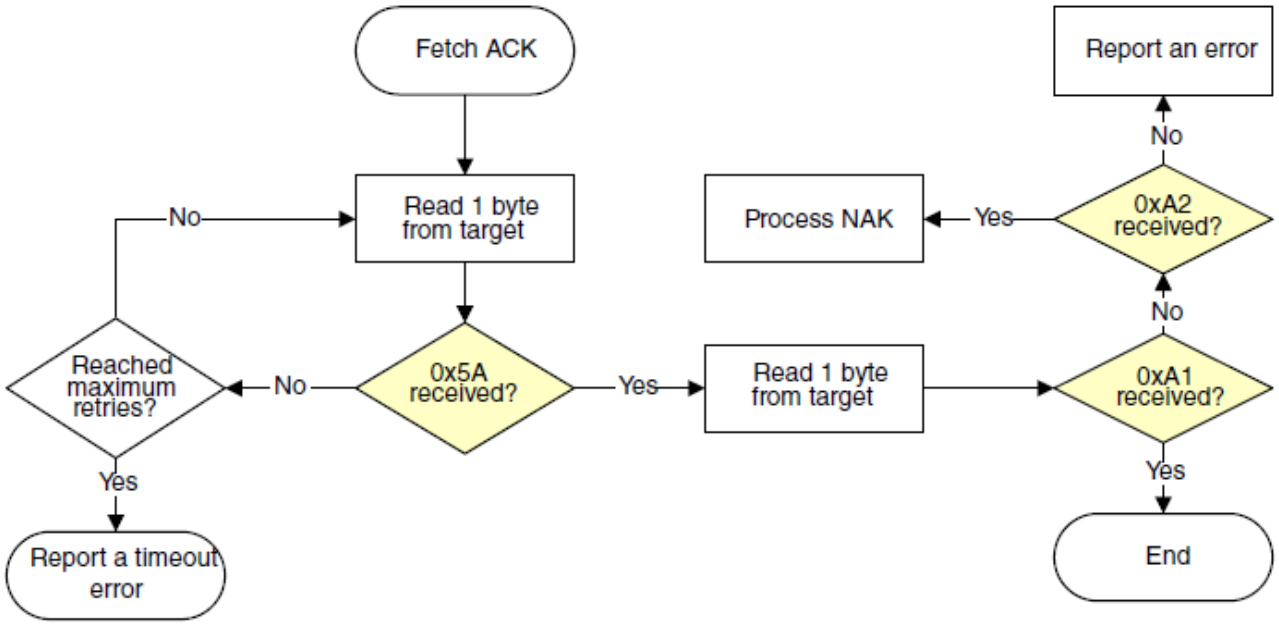


Figure 23-26. Host reads ACK packet from target via FlexCAN

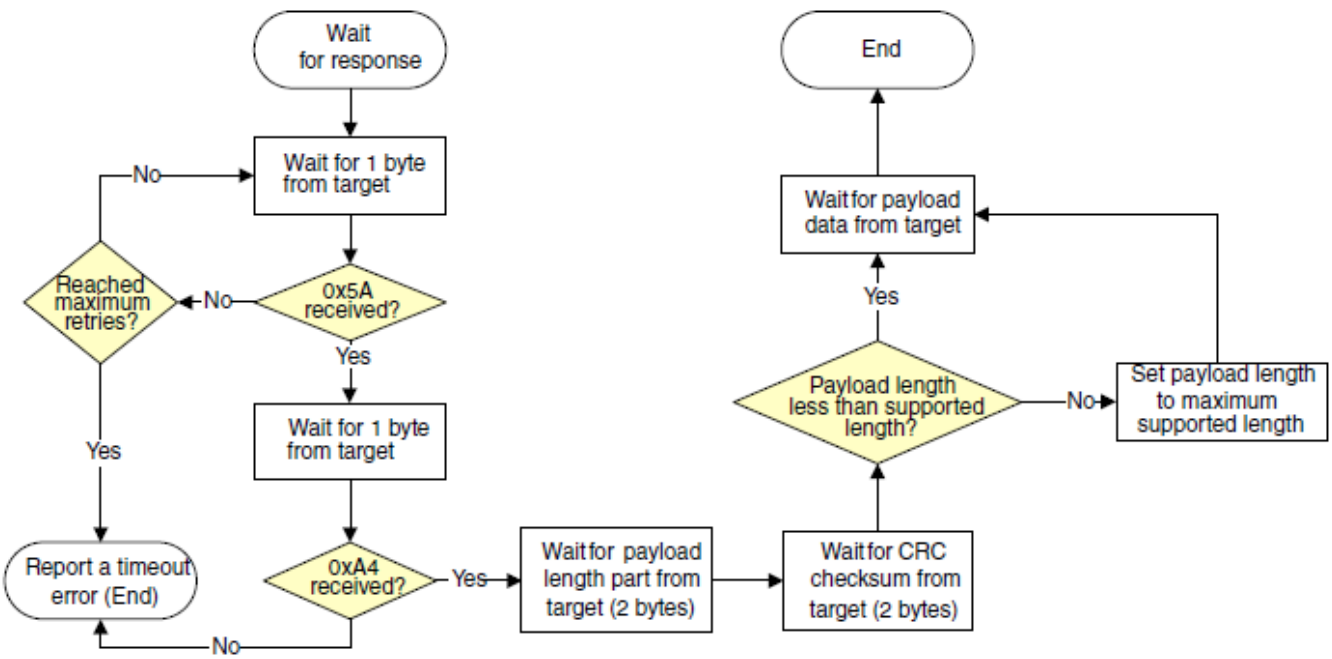


Figure 23-27. Host reads a command response from target via FlexCAN

23.6 Get/SetProperty Command Properties

This section lists the properties of the GetProperty and SetProperty commands.

Table 23-69. Properties used by Get/SetProperty Commands, sorted by Value

Property	Writable	Tag Value	Size	Description
CurrentVersion	No	01h	4	Current bootloader version.
AvailablePeripherals	No	02h	4	The set of peripherals supported on this chip.
FlashStartAddress	No	03h	4	Start address of program flash.
FlashSizeInBytes	No	04h	4	Size in bytes of program flash.
FlashSectorSize	No	05h	4	The size in bytes of one sector of program flash. This is the minimum erase size.
FlashBlockCount	No	06h	4	Number of blocks in the flash array.
AvailableCommands	No	07h	4	The set of commands supported by the bootloader.
VerifyWrites	Yes	0Ah	4	Controls whether the bootloader will verify writes to flash. VerifyWrites feature is enabled by default. 0 - No verification is done. 1 - Enable verification.
MaxPacketSize	No	0Bh	4	Maximum supported packet size for the currently active peripheral interface.
ReservedRegions	No	0Ch	16	List of memory regions reserved by the bootloader. Returned as value pairs (<start-address-of-region>, <end-address-of-region>). • If HasDataPhase flag is not set, then the Response packet parameter count indicates the number of pairs. • If HasDataPhase flag is set, then the second parameter is the number of bytes in the data phase.
ValidateRegions	Yes	0Dh	4	Controls whether the bootloader will validate attempts to write to memory regions (i.e., check if they are reserved before attempting to write). ValidateRegions feature is enabled by default. 0 - No validation is done 1 - Enable validation
RAMStartAddress	No	0Eh	4	Start address of RAM segment. The first parameter to GetProperty command identifies the segment. See the device specific memory map for number of RAM segments the device contains.
RAMSizeInBytes	No	0Fh	4	Size in bytes of RAM segment. The first parameter to GetProperty command identifies the segment. See the device specific memory map for number of RAM segments the device contains.
SystemDeviceId	No	10h	4	Value of the Kinetis System Device Identification register.
FlashSecurityState	No	11h	4	Indicates whether Flash security is enabled 0 - Flash security is disabled 1 - Flash security is enabled
UniqueDeviceId	No	12h	16	Unique device identification, value of Kinetis Unique Identification registers (16 for K series devices, 12 for KL series devices)

Table continues on the next page...

Table 23-69. Properties used by Get/SetProperty Commands, sorted by Value (continued)

Property	Writable	Tag Value	Size	Description
FacSupport	No	13h	4	FAC (Flash Access Control) support flag 0 - FAC not supported 1 - FAC supported
FlashAcessSegmentSize	No	14h	4	The size in bytes of 1 segment of flash
FlashAcessSegmentCount	No	15h	4	FAC segment count (The count of flash access segments within the flash model.)

23.6.1 Property Definitions

Get/Set property definitions are provided in this section.

23.6.1.1 CurrentVersion Property

The value of this property is a 4-byte structure containing the current version of the bootloader.

Table 23-70. Fields of CurrentVersion property:

Bits	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'K' (0x4B)	Major version	Minor version	Bugfix version

23.6.1.2 AvailablePeripherals Property

The value of this property is a bitfield that lists the peripherals supported by the bootloader and the hardware on which it is running.

Table 23-71. Peripheral bits:

Bit	[31:7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Peripheral	Reserved	Reserved	Reserved	Reserved	CAN Slave	SPI Slave	I2C Slave	UART

If the peripheral is available, then the corresponding bit will be set in the property value. All reserved bits must be set to 0.

23.6.1.3 AvailableCommands Property

This property value is a bitfield with set bits indicating the commands enabled in the bootloader. Only commands that can be sent from the host to the target are listed in the bitfield. Response commands such as GenericResponse are excluded.

The bit number that identifies whether a command is present is the command's tag value minus 1. 1 is subtracted from the command tag because the lowest command tag value is 0x01. To get the bit mask for a given command, use this expression:

```
mask = 1 << (tag - 1)
```

Table 23-72. Command bits:

Bit	[31:18]	[17]	[16]	[15]	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Command	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FlashEraseAllUnsecure	SetProperty	Reset	Reserved	Execute	Reserved	GetProperty	FlashSecurityDisable	Reserved	WriteMemory	ReadMemory	FlashEraseRegion	FlashEraseAll

23.7 Verifying the application in flash using CRC-32

Using CRC-32 and a given address range, the ROM bootloader supports performing an application integrity check.

Before application integrity testing: the following fields in the bootloader configuration area must be set:

- Set crcStartAddress to the start address that should be used for the CRC check.
- Set crcByteCount to the number of bytes to run the CRC check on, from the start address.
- Set crcExpectedValue to the value that the CRC calculation should result in.

Application integrity testing:

- If all of the above fields are unset (all 0xFF bytes for crcStartAddress, crcByteCount, and crcExpectedValue), then the ROM bootloader returns kStatus_AppCrcCheckInvalid.
- If any one of the above fields are set (crcStartAddress, crcByteCount, and crcExpectedValue), then the ROM bootloader checks if the given address range of the application is valid, and if the application resides in internal flash, by running a CRC check on the image in flash:
  - If the the given address range of the application is not valid (false), then the bootloader returns kStatus_AppCrcCheckOutOfRange.
  - If the given address range of the application is valid (true), then the CRC check is performed.
    - If the CRC check fails, then the bootloader returns kStatus_AppCrcCheckFailed;
    - If the CRC check succeeds, then it returns kStatus_AppCrcCheckPassed.

After application integrity testing:

- If the bootloader returns kStatus_AppCrcCheckOutOfRange or kStatus_AppCrcCheckFailed, then an external pin (PTA6)(PTA11) will also be asserted, to indicate CRC check failure.
- The bootloader will only jump to the application if kStatus_AppCrcCheckPassed is returned; if kStatus_AppCrcCheckPassed is not returned, then the bootloader will stay active, and wait for further commands.

23.8 Kinetis Bootloader Status Error Codes

This section describes the status error codes that the Kinetis Bootloader returns to the host.

Table 23-73. Kinetis Bootloader Status Error Codes, sorted by Value

Error Code	Value	Description
kStatus_Success	0	Operation succeeded without error.
kStatus_Fail	1	Operation failed with a generic error.
kStatus_ReadOnly	2	Requested value cannot be changed because it is read-only.
kStatus_OutOfRange	3	Requested value is out of range.
kStatus_InvalidArgument	4	The requested command's argument is undefined.
kStatus_Timeout	5	A timeout occurred.
kStatus_FlashSizeError	100	Not used.
kStatus_FlashAlignmentError	101	Address or length does not meet required alignment.
kStatus_FlashAddressError	102	Address or length is outside addressable memory.

Table continues on the next page...

Table 23-73. Kinetis Bootloader Status Error Codes, sorted by Value (continued)

Error Code	Value	Description
kStatus_FlashAccessError	103	The FTFE_FSTAT[ACCERR] bit is set.
kStatus_FlashProtectionViolation	104	The FTFE_FSTAT[FPVIOL] bit is set.
kStatus_FlashCommandFailure	105	The FTFE_FSTAT[MGSTAT0] bit is set.
kStatus_FlashUnknownProperty	106	Unknown Flash property.
kStatus_FlashEraseKeyError	107	The key provided does not match the programmed flash key.
kStatus_FlashRegionExecuteOnly	108	The area of flash is protected as execute only.
kStatus_I2C_SlaveTxUnderrun	200	I2C Slave TX Underrun error.
kStatus_I2C_SlaveRxOverrun	201	I2C Slave RX Overrun error.
kStatus_I2C_AribtrationLost	202	I2C Arbitration Lost error.
kStatus_SPI_SlaveTxUnderrun	300	SPI Slave TX Underrun error.
kStatus_SPI_SlaveRxOverrun	301	SPI Slave RX Overrun error.
kStatus_SPI_Timeout	302	SPI tranfser timed out.
kStatus_SPI_Busy	303	SPI instance is already busy performing a transfer.
kStatus_SPI_NoTransferInProgress	304	Attempt to abort a transfer when no transfer was in progress.
kStatus_UnknownCommand	10000	The requested command value is undefined.
kStatus_SecurityViolation	10001	Command is disallowed because flash security is enabled.
kStatus_AbortDataPhase	10002	Abort the data phase early.
kStatus_Ping	10003	Internal: received ping during command phase.
kStatusMemoryRangeInvalid	10200	Memory range conflicts with a protected region.
kStatus_UnknownProperty	10300	The requested property value is undefined.
kStatus_ReadOnlyProperty	10301	The requested property value cannot be written.
kStatus_InvalidPropertyValue	10302	The specified property value is invalid.
kStatus_AppCrcCheckPassed	10400	CRC check is valid and passed.
kStatus_AppCrcCheckFailed	10401	CRC check is valid but failed.
kStatus_AppCrcCheckInactive	10402	CRC check is inactive.
kStatus_AppCrcCheckInvalid	10403	CRC check is invalid, because the BCA is invalid or the CRC parameters are unset (all 0xFF bytes).
kStatus_AppCrcCheckOutOfRange	10404	CRC check is valid but addresses are out of range.

Chapter 24
Reset Control Module (RCM)

24.1 Chip-specific information for this module

24.1.1 Instantiation Information

24.1.1.1 Information of RCM on this device

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. A bus error will generate a hard fault interrupt on this device.

NOTE

High-Voltage Detect (HVD) is not supported on this device. Therefore, HVD related descriptions are not applicable in RCM_SRS[LVD].

NOTE

In the RCM_SRIE register, bit 8 JTAG and bit 11 MDM_AP do not exist (should be *Reserved*) on this device.

24.2 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 and ColdFire+ MCUs for further details on using the RCM.

24.3 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_F000	Version ID Register (RCM_VERID)	32	R	0300_0003h	24.3.1/632
4007_F004	Parameter Register (RCM_PARAM)	32	R	See section	24.3.2/634
4007_F008	System Reset Status Register (RCM_SRS)	32	R	0000_0082h	24.3.3/636
4007_F00C	Reset Pin Control register (RCM_RPC)	32	R/W	0000_0000h	24.3.4/639
4007_F010	Mode Register (RCM_MR)	32	R/W	See section	24.3.5/640
4007_F014	Force Mode Register (RCM_FM)	32	R/W	0000_0000h	24.3.6/641
4007_F018	Sticky System Reset Status Register (RCM_SSRS)	32	R/W	0000_0082h	24.3.7/642
4007_F01C	System Reset Interrupt Enable Register (RCM_SRIE)	32	R/W	0000_0000h	24.3.8/644

24.3.1 Version ID Register (RCM_VERID)

Address: 4007_F000h base + 0h offset = 4007_F000h

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

RCM_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number

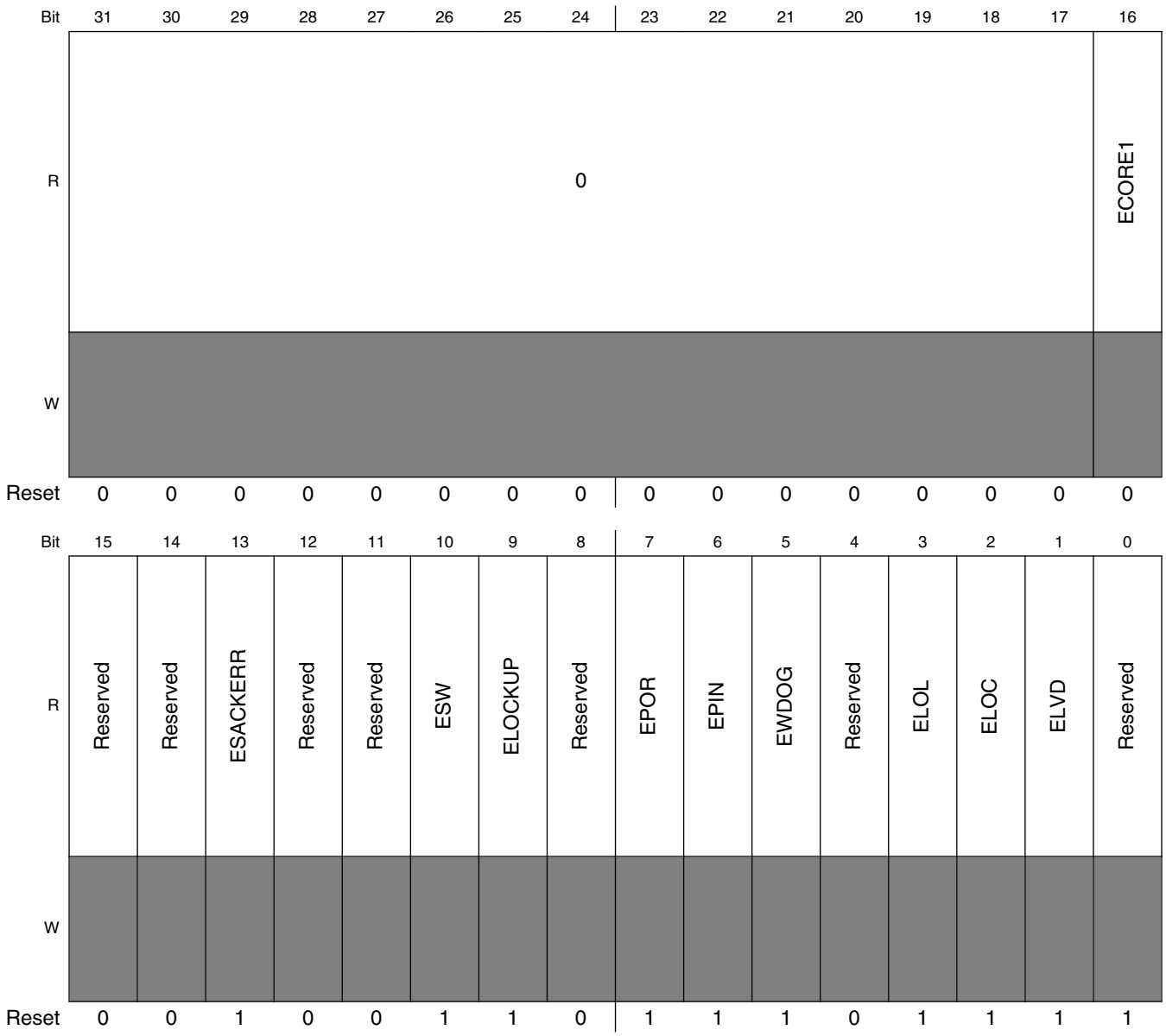
Table continues on the next page...

RCM_VERID field descriptions (continued)

Field	Description
	This read only field returns the major version number for the specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0003 Standard feature set.

24.3.2 Parameter Register (RCM_PARAM)

Address: 4007_F000h base + 4h offset = 4007_F004h



RCM_PARAM field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ECORE1	Existence of SRS[CORE1] status indication feature

Table continues on the next page...

RCM_PARAM field descriptions (continued)

Field	Description
	This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
15 Reserved	This field is reserved.
14 Reserved	This field is reserved.
13 ESACKERR	Existence of SRS[SACKERR] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
12 Reserved	This field is reserved.
11 Reserved	This field is reserved.
10 ESW	Existence of SRS[SW] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
9 ELOCKUP	Existence of SRS[LOCKUP] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
8 Reserved	This field is reserved.
7 EPOR	Existence of SRS[POR] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
6 EPIN	Existence of SRS[PIN] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
5 EWDOG	Existence of SRS[WDOG] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

Table continues on the next page...

RCM_PARAM field descriptions (continued)

Field	Description
4 Reserved	This field is reserved.
3 ELOL	Existence of SRS[LOL] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
2 ELOC	Existence of SRS[LOC] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
1 ELVD	Existence of SRS[LVD] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
0 Reserved	This field is reserved.

24.3.3 System Reset Status Register (RCM_SRS)

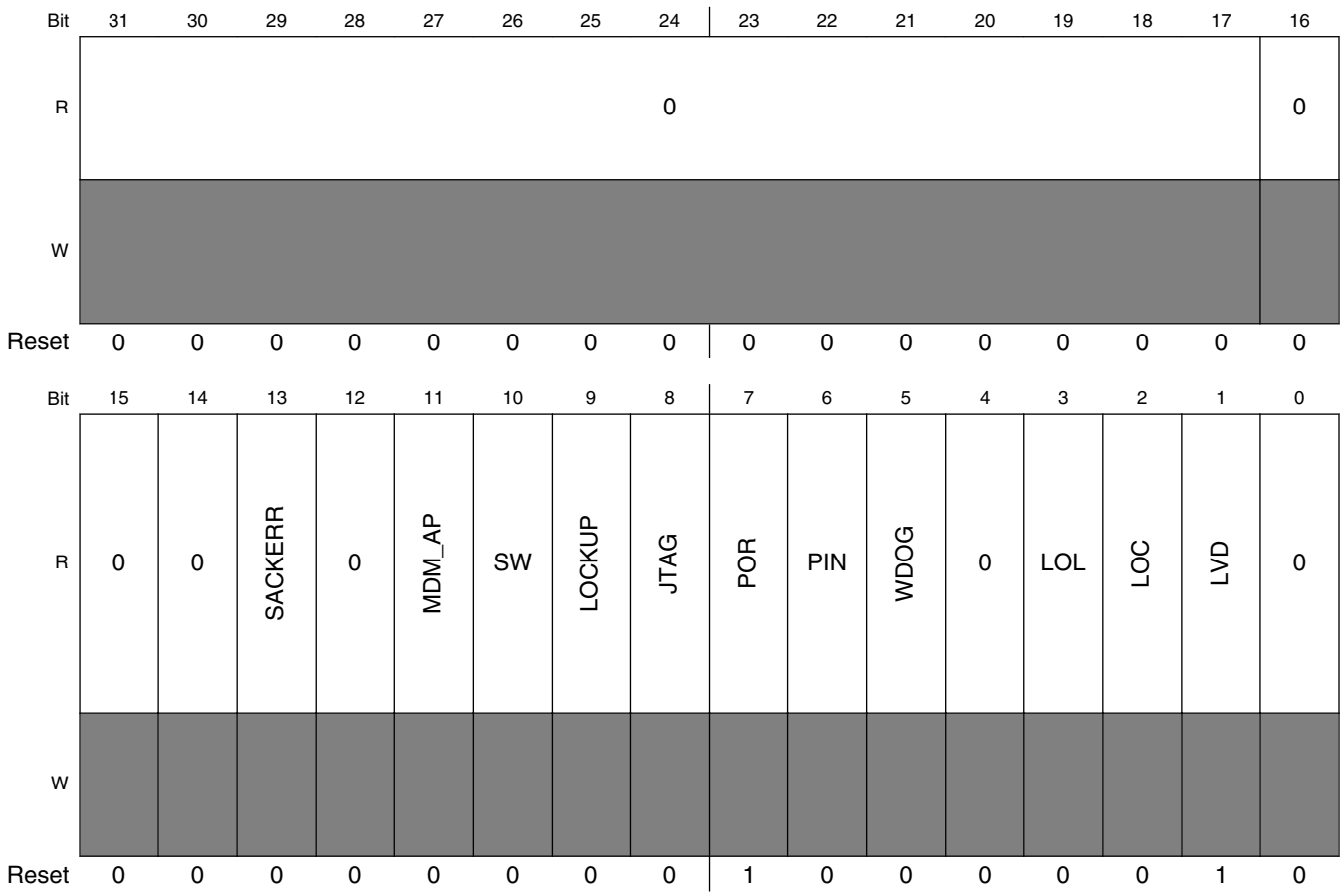
This register includes read-only status flags to indicate the source of the most recent reset. Note that multiple flags can be set if multiple reset events occur at the same time. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x82
- LVD (without POR) — 0x02
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_F000h base + 8h offset = 4007_F008h



RCM_SRS field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SACKERR	Stop Acknowledge Error 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
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 MDM_AP	MDM-AP System Reset Request

Table continues on the next page...

RCM_SRS field descriptions (continued)

Field	Description
	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 was not caused by host debugger system setting of the System Reset Request bit 1 Reset was caused by host debugger system setting of the System Reset Request bit
10 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
9 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
8 JTAG	JTAG generated reset Indicates a reset has been caused by the JTAG selection of certain IR codes: EXTEST, HIGHZ, and CLAMP. 0 Reset not caused by JTAG 1 Reset caused by JTAG
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 $\overline{\text{RESET}}$ (RESET_b) pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog 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 SCG PLL/FLL.

Table continues on the next page...

RCM_SRS field descriptions (continued)

Field	Description
	0 Reset not caused by a loss of lock in the PLL/FLL 1 Reset caused by a loss of lock in the PLL/FLL
2 LOC	Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The SCG SOSC clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed SCG 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 or High-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. If PMC_HVDSC1[HVDRE] is set and the supply rises above the HVD trip voltage, an HVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip, HVD trip or POR 1 Reset caused by LVD trip, HVD trip or POR
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

24.3.4 Reset Pin Control register (RCM_RPC)

NOTE

This register is reset on Chip POR only, it is 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_F000h base + Ch offset = 4007_F00Ch

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			RSTFLTSEL					0					RSTFLTSS	RSTFLTSRW	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RCM_RPC field descriptions

Field	Description
31–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 RSTFLTSEL	Reset Pin Filter Bus Clock Select Selects the reset pin bus clock filter width: • Transition lengths less than RSTFLTSEL cycles are filtered. • Transition lengths between RSTFLTSEL and (RSTFLTSEL+1) cycles (inclusive) may be filtered. • Transition lengths greater than (RSTFLTSEL+1) cycles are not filtered.
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 any stop mode. 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

24.3.5 Mode Register (RCM_MR)

This register includes status flags to indicate the state of the mode pins during the last Chip Reset.

Address: 4007_F000h base + 10h offset = 4007_F010h

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													BOOTROM		0
W														w1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*	0

* Notes:
- BOOTROM field: The reset state of this register depends on the boot mode.

RCM_MR field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 BOOTROM	Boot ROM Configuration Indicates the boot source, the boot source remains set until the next System Reset or software can write logic one to clear the corresponding mode bit. While either bit is set, the NMI input is disabled and the vector table is relocated to the ROM base address at $1C00_0000. These bits should be cleared by writing logic one before executing any code from either Flash or SRAM. 00 Boot from Flash 01 Boot from ROM due to BOOTCFG0 pin assertion / Reserved if no Boot pin 10 Boot form ROM due to FOPT[7] configuration 11 Boot from ROM due to both BOOTCFG0 pin assertion and FOPT[7] configuration
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

24.3.6 Force Mode Register (RCM_FM)

NOTE

The reset values of the bits in the FORCEROM field are for Chip POR only. They are unaffected by other reset types.

Address: 4007_F000h base + 14h offset = 4007_F014h

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													FORCEROM		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RCM_FM field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 FORCEROM	Force ROM Boot When either bit is set, will force boot from ROM during all subsequent system resets. 00 No effect 01 Force boot from ROM with RCM_MR[1] set.

Table continues on the next page...

RCM_FM field descriptions (continued)

Field	Description
	10 Force boot from ROM with RCM_MR[2] set. 11 Force boot from ROM with RCM_MR[2:1] set.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

24.3.7 Sticky System Reset Status Register (RCM_SSRS)

This register includes status flags to indicate all reset sources since the last POR or LVD that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 18h offset = 4007_F018h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SSACKERR	0	SMDM_AP	SSW	SLOCKUP	SJTAG	SPOR	SPIN	SWDOG	0	SLOL	SLOC	SLVD	0
W			w1c		w1c	w1c	w1c	w1c	w1c	w1c	w1c		w1c	w1c	w1c	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

RCM_SSRS field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SSACKERR	Sticky Stop Acknowledge Error 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
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 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 was not caused by host debugger system setting of the System Reset Request bit 1 Reset was caused by host debugger system setting of the System Reset Request bit
10 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
9 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
8 SJTAG	Sticky JTAG generated reset Indicates a reset has been caused by the JTAG selection of certain IR codes: EXTEST, HIGHZ, and CLAMP. 0 Reset not caused by JTAG 1 Reset caused by JTAG
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.

Table continues on the next page...

RCM_SSRS field descriptions (continued)

Field	Description
	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 $\overline{\text{RESET}}$ (RESET_b) 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. 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 SCG PLL/FLL. See the SCG description for information on the loss-of-lock event. 0 Reset not caused by a loss of lock in the PLL/FLL 1 Reset caused by a loss of lock in the PLL/FLL
2 SLOC	Sticky Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The SCG SOSC clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed SCG 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 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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

24.3.8 System Reset Interrupt Enable Register (RCM_SRIE)

This register provides the option to delay the assertion of a system reset for a period of time (DELAY field) while an interrupt is generated. When an interrupt for a reset source is enabled, software has time to perform a graceful shutdown. A Chip POR source cannot be delayed by this feature. The SRS updates only after the system reset occurs.

NOTE

This register is reset on Chip POR only, it is unaffected by other reset types.

Address: 4007_F000h base + 1Ch offset = 4007_F01Ch

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SACKERR	0	MDM_AP	SW	LOCKUP	JTAG	GIE	PIN	WDOG	0	LOL	LOC	DELAY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RCM_SRIE field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SACKERR	Stop Acknowledge Error Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 MDM_AP	MDM-AP System Reset Request 0 Interrupt disabled. 1 Interrupt enabled.
10 SW	Software Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
9 LOCKUP	Core Lockup Interrupt **NOTE:** The LOCKUP bit is useful only in devices with more than one core processor.

Table continues on the next page...

RCM_SRIE field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled.
8 JTAG	JTAG generated reset 0 Interrupt disabled. 1 Interrupt enabled.
7 GIE	Global Interrupt Enable 0 All interrupt sources disabled. 1 All interrupt sources enabled. Note that the individual interrupt-enable bits still need to be set to generate interrupts.
6 PIN	External Reset Pin Interrupt 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LOL	Loss-of-Lock Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
2 LOC	Loss-of-Clock Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
DELAY	Reset Delay Time Configures the maximum reset delay time from when the interrupt is asserted and the system reset occurs. 00 10 LPO cycles 01 34 LPO cycles 10 130 LPO cycles 11 514 LPO cycles

Chapter 25
Power Management

25.1 Introduction

This chapter describes the various chip power modes and functionality of the individual modules in these modes. Following stated are general power modes, which are supported additionally by certain clocking mode options. Clock gating technique is used for general power modes and for the additional clocking mode options.

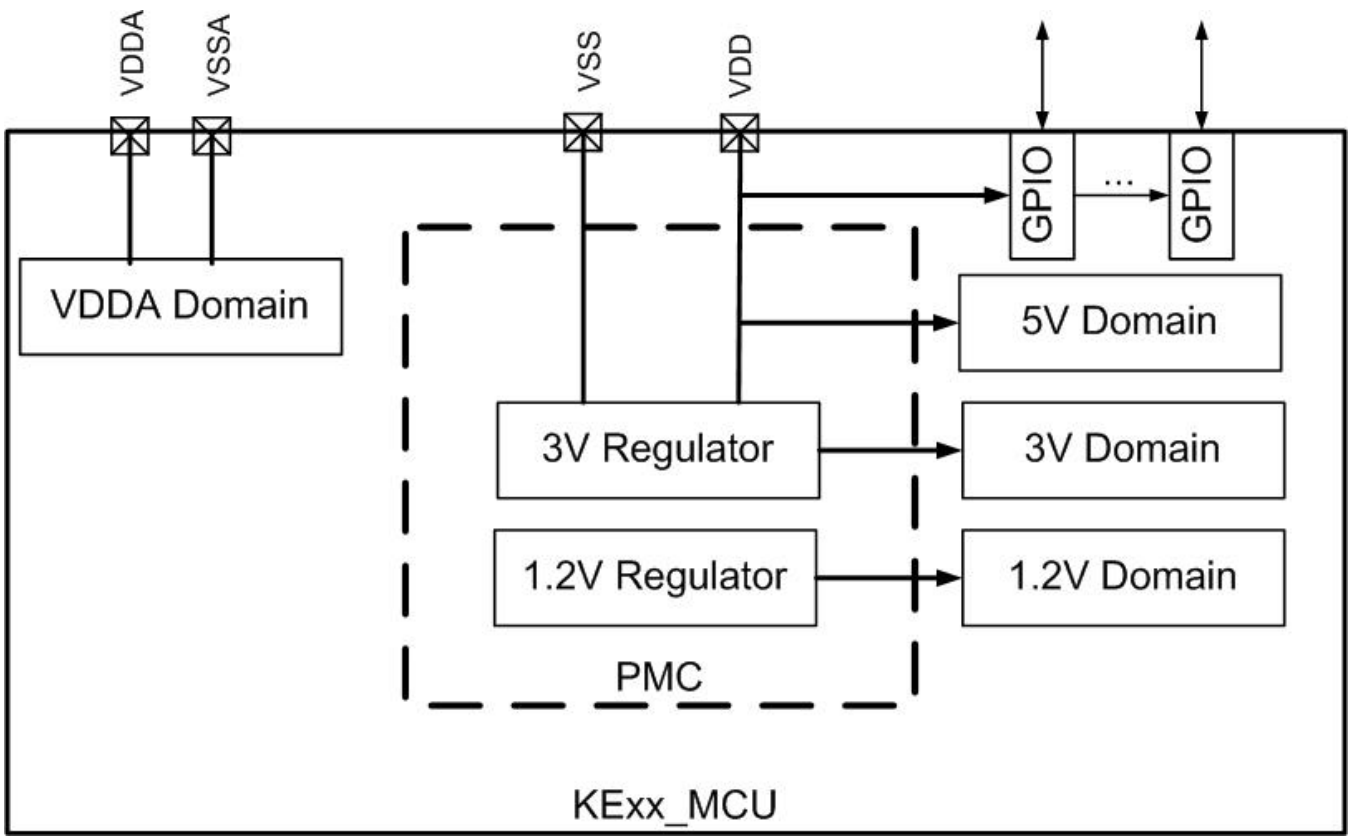


Figure 25-1. Power Infrastructure

25.2 Power Modes Description

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

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

For Run and VLPR 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.

Stop mode entry is not supported directly from HSRUN and requires transition to Run prior to an attempt to enter a stop mode.

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

Table 25-1. Chip power modes

Chip mode	Description	Core mode	Normal recovery method
Normal Run	Default mode out of reset; on-chip voltage regulator is on.	Run	-
High Speed Run	Allows maximum performance of chip. In this state, the MCU is able to operate at a faster frequency compared to normal run mode.	Run	-
Normal Wait - via WFI	Allows peripherals to function while the core is in sleep mode, reducing power. NVIC remains sensitive to interrupts; peripherals continue to be clocked.	Sleep	Interrupt
Normal Stop - via WFI	Places chip in static state. On-chip voltage regulator is in a low power mode. LVD is off while maintaining LVR and POR protection. NVIC is disabled; AWIC is used to wake up from interrupt; Peripheral clocks are stopped. All SRAM is operating (content retained and I/O state held). ADC and CMP are optional on.	Sleep Deep	Interrupt
VLPR (Very Low Power Run)	On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; internal oscillator provides a low power 4 MHz source for the core, the bus and the peripheral clocks.	Run	-
VLPW (Very Low Power Wait) -via WFI	Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.	Sleep	Interrupt

Table continues on the next page...

Table 25-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
VLPS (Very Low Power Stop)-via WFI	Same as Stop mode, but PMC_REGSC register provides options to gate off unused modules and further reduce power in low power mode.	Sleep Deep	Interrupt

25.2.1 Run mode

Run mode is the default mode after reset, and refers to any mode in which CPU execution is possible. Depending on the on-chip regulator settings, Run mode has the following configurations:

- HSRUN mode — The on-chip regulator voltage output is slightly elevated. The 1.2V domain is powered by 1.4V instead. This allows the MCU digital modules to operate at a faster frequency.
- Normal RUN mode — The on-chip regulator voltage output is normal. The 1.2V domain is powered by 1.2V. This allows the MCU digital modules to operate at a normal frequency.
- Very Low Power RUN mode — The on-chip regulator voltage is in Low Power mode. The MCU digital modules should operate at a limited frequency but with much lower power.

Run mode configurations can be selected by configuring SMC_PMCTRL.

The following sections describe optimizing power in Run modes.

25.2.1.1 Clock Gating

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

25.2.1.2 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 Run mode, HSRUN mode, or VLPR 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 VLPR mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. SCG, 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 bus (PPB) remains accessible during Compute Operation, including the MCM, System Control Space (SCS) (for NVIC), and SysTick. Although access to the GPIO registers 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, which is only accessible to the CPU. Setting or clearing the CPOREQ bit in the MCM 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 bit indicates when entry has completed. When exiting Compute Operation in Run mode, the CPOACK status bit negates immediately. When exiting Compute Operation in VLPR mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means the CPOACK bit is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wakeup is also supported during Compute Operation and causes the CPOACK status bit to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wakeup, the device transitions back into Compute Operation.

25.2.2 Wait mode

Wait mode refers to a power modes in which the CPU execution is halted. The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate.

Depending on the on-chip regulator settings, Wait mode has the following configurations:

- Normal Wait mode — The on-chip regulator voltage output is normal. The 1.2V domain is powered by 1.2V. This allows the MCU digital modules to operate at a normal frequency.
- Very Low Power Wait mode — The on-chip regulator voltage is in Low Power mode. The MCU digital modules must operate at a limited frequency but with much lower power.

After the CPU executes the WFI/WFE instruction, VLPW mode is entered when MCU is in VLPR mode and Normal Wait mode is entered when MCU is in Normal Run mode. Run mode configurations can be selected by configuring SMC_PMCTRL.

Clock gating can be used to optimize the power in Wait mode. Any interrupt can be used as a wake up source from the Wait mode. See the "Interrupt vector assignments" table in Interrupts chapter for all the available interrupt sources.

25.2.3 Stop mode

Stop mode refers to power modes in which the CPU and most peripherals are static. The SRAM and all registers are retained. The core clock, system clock, and the bus clock are gated off. NVIC is disabled; AWIC is used to wake up from interrupt. In the Stop mode, some peripherals can remain operational with asynchronous clock and can wake up the MCU as needed.

Stop mode configurations can be selected by configuring SMC_PMCTRL.

In Stop mode, the bus clock is gated as core clock and system clock. This device supports a partial Stop mode that permits peripherals to run with the bus clock.

25.2.3.1 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 SCG 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.

Any AWIC interrupt can be used as a wake up source from Stop (normal Stop and VLPS) mode. See Table 25-5 for all the available wake up source. Besides waking up the CPU from Stop mode, the DMA can perform data transfer while retaining the CPU in Low Power mode.

25.2.3.2 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 SCG, 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, SCG 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.

25.2.4 Power domains

The following table describe the power domain of this device.

Table 25-2. Power domain summary

Domain name	Description
5V	5V domain is powered by VDD/VSS directly. It contains GPIO and PMC.
VDDA	Analog domain is powered by VDDA/VSSA. It contains analog modules such as ADC and CMP.

Table continues on the next page...

Table 25-2. Power domain summary (continued)

Domain name	Description
3V	3V domain is powered by the PMC 3V regulator. It contains OSC, and Flash memory.
1.2V	1.2V domain is powered by the PMC 1.2V regulator. It contains all digital logics and SRAM.

Table 25-3. Module power domain summary

VDD (5V)	
PMC	GPIOx (all ports)
VDDA	
ADCx	CMPx
DACx	
3V CORE	
Flash Memory	OSC
1.2V	
Cortex-M4 Core	TRGMUX*x*
SRAM	EWM
SCG	WDOG
PCC	CRC
AXBS-Lite	FlexIO
Boot ROM	LPI2Cx
SIM	LPSPIx
RCM	LPUARTx
MCM	LPITx
AIPS-Lite	FTMx
AWIC	PDBx
eDMA	LPTMRx
DMAMUX	RTC
FlexCAN	PORTx

25.2.5 Entering and exiting power modes

The WFI instruction invokes wait and stop modes for the chip. The processor exits the low-power mode via an interrupt. The "Interrupt vector assignments" table in the Interrupts chapter describes interrupt operation and what peripherals can cause interrupts.

NOTE

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

25.3 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 offer a lower power operating mode than normal modes. VLPR and VLPW are limited in frequency.

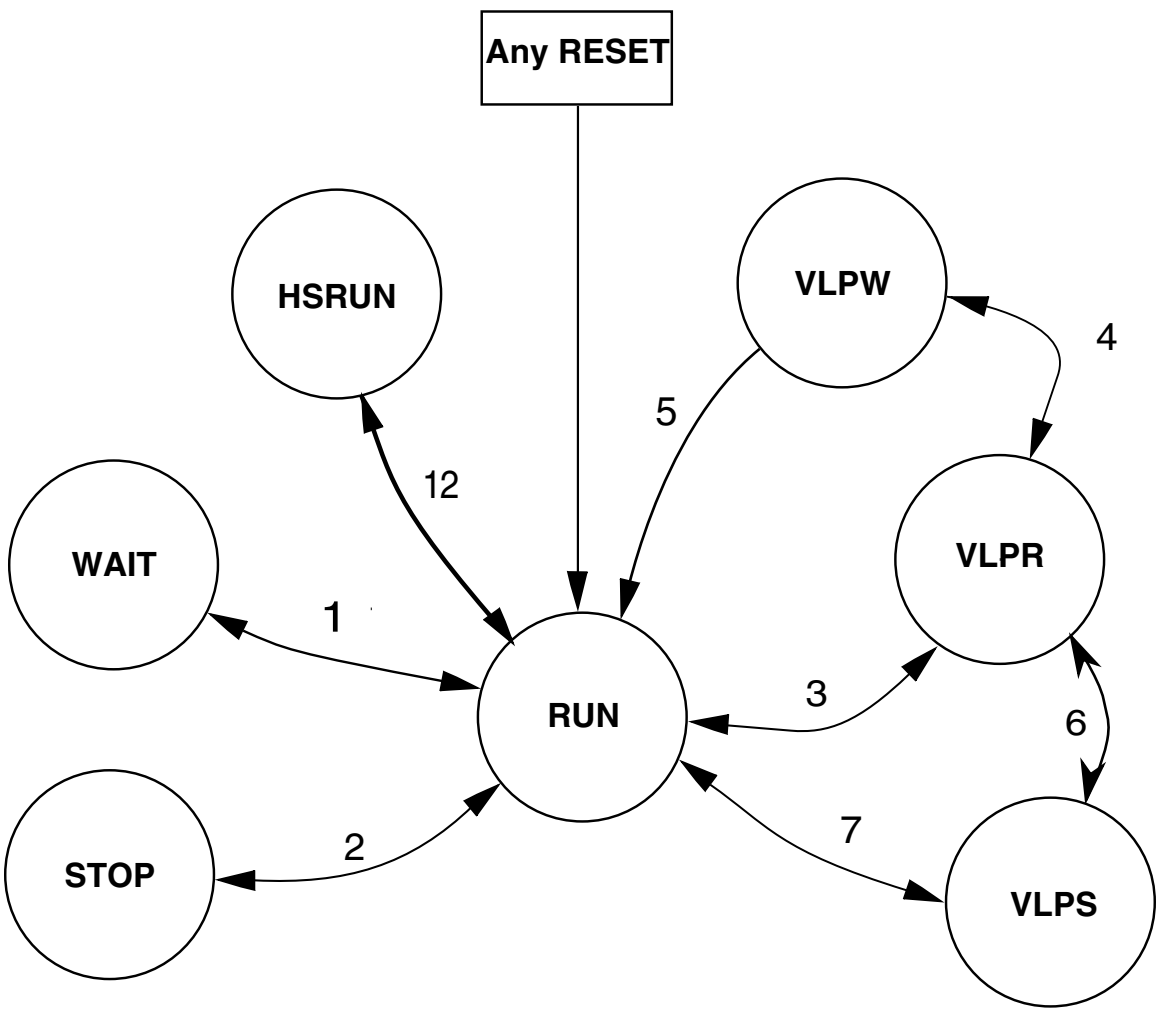


Figure 25-2. Power mode state transition diagram

NOTE

See Table 26-2 in the SMC chapter for more detailed mode transition conditions.

25.4 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-M core immediately.
- Polls stop acknowledge indications from the non-core crossbar masters (DMA), supporting peripherals (SPI, PIT) and the Flash Controller for indications that System Clocks, Bus Clock and/or Flash Clock need to be left enabled to complete a previously initiated operation, effectively stalling entry to the targeted low power mode. When all acknowledges are detected, System Clock, Bus Clock and Flash Clock are turned off at the same time.
- SCG 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-M core is shut off. Some modules support stop-in-wait functionality and have their clocks disabled under these configurations.

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

25.5 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. The standard behavior is shown with some exceptions for Compute Operation (CPO) and Partial Stop2 (PSTOP2).

Debug modules are discussed separately, see "Debug in low power modes" in the Debug chapter. Number ratings (such as 2 MHz and 1 Mbit/s) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- Async operation = Fully functional with alternate clock source, provided the selected clock source remains enabled
- static = Module register states and associated memories are retained.
- powered = Memory is powered to retain contents.
- low power = Memory is powered to retain contents in a lower power state
- OFF = Modules are powered off; module is in reset state upon wakeup. For clocks, OFF means disabled.
- wakeup = Modules can serve as a wakeup source for the chip.

Table 25-4. Module operation in low power modes

Modules	VLPR	VLPW	Stop	VLPS
Core modules				
NVIC	FF	FF	static	static
System modules				
System Mode Controller	FF	FF	FF	FF
Regulator	low power	low power	low power	low power
LVD/LVR	disabled (LVR active only)	disabled (LVR active only)	disabled (LVR active only)	disabled (LVR active only)
POR (Brown-out Detection)	FF	FF	FF	FF
DMA	FF Async operation in CPO	FF	Async operation	Async operation
Watchdog	FF	FF	FF	FF
EWM	FF static in CPO	static	static FF in PSTOP2	static
Clocks				
128 kHz LPO	FF	FF	FF	FF
System oscillator (SOSC)	SOSC_CLK optional ON	SOSC_CLK optional ON	SOSC_CLK optional ON	SOSC_CLK optional ON
32 kHz oscillator (OSC32)	Optional ON	Optional ON	Optional ON	Optional ON
SCG	SOSC, SIRC, FIRC optional ON	SOSC, SIRC, FIRC optional ON	SOSC, SIRC, FIRC, SPLL optional ON	SOSC, SIRC, FIRC optional ON
Core clock	4 MHz max	OFF	OFF	OFF
System clock	4 MHz max OFF in CPO	4 MHz max	OFF	OFF
Bus clock	4 MHz max OFF in CPO	4 MHz max	OFF FF in PSTOP2	OFF
Memory and memory interfaces				

Table continues on the next page...

Table 25-4. Module operation in low power modes (continued)

Modules	VLPR	VLPW	Stop	VLPS
Flash	1 MHz max access - no program/erase No register access in CPO	low power	low power	low power
System RAM (SRAM_U and SRAM_L)	low power [1]	low power	low power	low power
Cache	low power	low power	low power	low power
FlexMemory	low power [2]	low power	low power	low power
Communication interfaces				
LPUART	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
LPSPI	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
LPI^{2}C	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
FlexIO	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
CAN	FF wakeup in CPO	FF	wakeup FF in PSTOP2	wakeup
Security				
CRC	FF static in CPO	FF	static FF in PSTOP2	static
Timers				
FTM	FF static in CPO	FF	static	static
LPIT	FF static in CPO	FF	Async operation FF in PSTOP2	Async operation
PWT	FF static in CPO	FF	static FF in PSTOP2	static
PDB	FF static in CPO	FF	static	static
LPTMR	FF	FF	Async operation FF in PSTOP2	Async operation
RTC	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
Analog				
12-bit ADC	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only
CMP [3]	LS compare only	LS compare only	LS compare	LS compare only

Table continues on the next page...

Table 25-4. Module operation in low power modes (continued)

Modules	VLPR	VLPW	Stop	VLPS
			FF in PSTOP2	
12-bit DAC	FF static in CPO	FF	static FF in PSTOP2	static
Human-machine interfaces				
GPIO	FF static in CPO	FF	static output, wakeup input FF in PSTOP2	static output, wakeup input

1. SRAM is writable and readable in VLPR mode.
2. FlexRAM enabled as EEPROM is not writable in VLPR and writes are ignored. Read accesses to FlexRAM as EEPROM while in VLPR are allowed. There are no access restrictions for FlexRAM configured as traditional RAM.
3. CMP in stop or VLPS supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled and filtered modes of operation are not available while in stop or VLPS modes.

NOTE

- The load current should only change slowly (by peripheral modules being turned on/off, and clock speed being changed) in low power modes.
- Before entering low power modes, the peripheral clock frequencies should be set to desired values, for the modules working in 1.2 V power domain (see Table 25-2, e.g. Boot ROM, FlexIO, LPIT, LPTMR and communication modules).

25.5.1 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.

25.6 Low-power wake-up sources

Table 25-5. AWIC Stop and VLPS Wake-up Sources

Wake-up source	Description
Available system resets	RESET pin, WDOG, JTAG , loss of clock(LOC) reset and loss of lock (LOL) reset
Pin interrupts	Port Control Module - Any enabled pin interrupt is capable of waking the system
ADC	ADC is optional functional with clock source from SIRC or OSC
CMP	Functional in Stop/VLPS modes with clock source from SIRC or OSC
DAC	Functional in VLPR/VLPW modes
LPI2C	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPUART	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPSPI	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPIT	Functional in Stop/VLPS modes with clock source from SIRC or OSC
FlexIO	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPTMR	Functional in Stop/VLPS modes
RTC	Functional in Stop/VLPS modes
SCG	Functional in Stop mode
RCM	Reset wakeup
CAN	CAN stop wakeup
NMI	Non-maskable interrupt

25.7 Power supply supervisor

This device integrates the following power supervisor circuits:

- Power-on reset (POR)
- Low voltage detection (LVD)

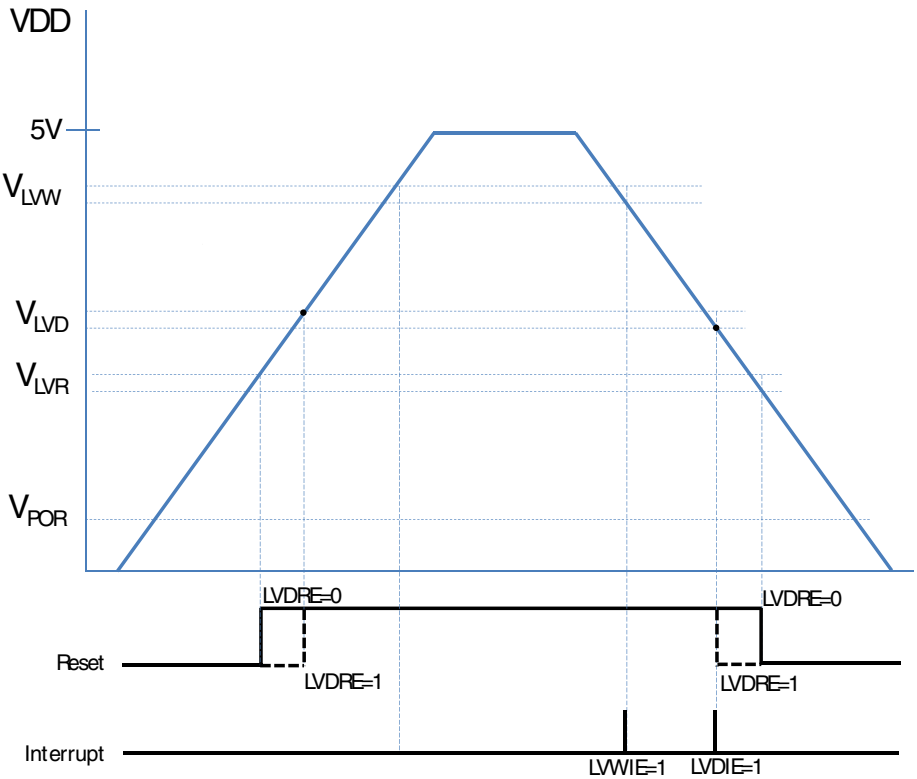


Figure 25-3. Power Supply Supervisor

NOTE

When VDD ramps up above V_{LVR} , the RESET pin is released (indicating MCU quits POR reset state), and it starts to run code immediately.

If LVDRE bit is not operated in code, the LVDRE bit is 0 after POR reset by default, then LVD does not take effect.

If LVDRE bit is configured as 1 in user's code, and the current VDD still below V_{LVD}, then LVD takes effect, and holds MCU in system reset state, keeps the RESET pin low until VDD increases above V_{LVD}.

During power-on, the POR keeps the device under reset until the supply voltage V_{DD} reaches the specified threshold. When V_{DD} is above the threshold, the device reset is released and the system can start.

The LVD circuit can be used to monitor the power supply voltage by comparing it to a configurable threshold. User can choose to generate LVD reset or LVW interrupt when power supply voltage drops below the threshold. See PMC chapters for details.

For more details on the POR/LVD reset and the LVW interrupt thresholds, see the electrical characteristics (LVR, LVD and POR) section in the Data Sheet.

Chapter 26
System Mode Controller (SMC)

26.1 Chip-specific information for this module

26.1.1 Instantiation Information

26.1.1.1 Information of SMC on this device

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. A bus error will generate a hard fault interrupt on this device.

26.2 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.

26.3 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 Kinetis microcontrollers.

The following table shows the translation between the Arm CPU modes and the Kinetis 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 Kinetis MCU documentation normally uses wait and stop.

In addition, Kinetis 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 26-1. 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.
HSRUN	The MCU can be run at a faster frequency compared with RUN mode and the internal supply is fully regulated. See the Power Management chapter for details about the maximum allowable frequencies.
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.

26.4 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	SMC Version ID Register (SMC_VERID)	32	R	0100_0000h	26.4.1/666
4007_E004	SMC Parameter Register (SMC_PARAM)	32	R	See section	26.4.2/667
4007_E008	Power Mode Protection register (SMC_PMPROT)	32	R/W	0000_0000h	26.4.3/668
4007_E00C	Power Mode Control register (SMC_PMCTRL)	32	R/W	0000_0000h	26.4.4/669
4007_E010	Stop Control Register (SMC_STOPCTRL)	32	R/W	0000_0003h	26.4.5/671
4007_E014	Power Mode Status register (SMC_PMSTAT)	32	R	0000_0001h	26.4.6/673

26.4.1 SMC Version ID Register (SMC_VERID)

Address: 4007_E000h base + 0h offset = 4007_E000h

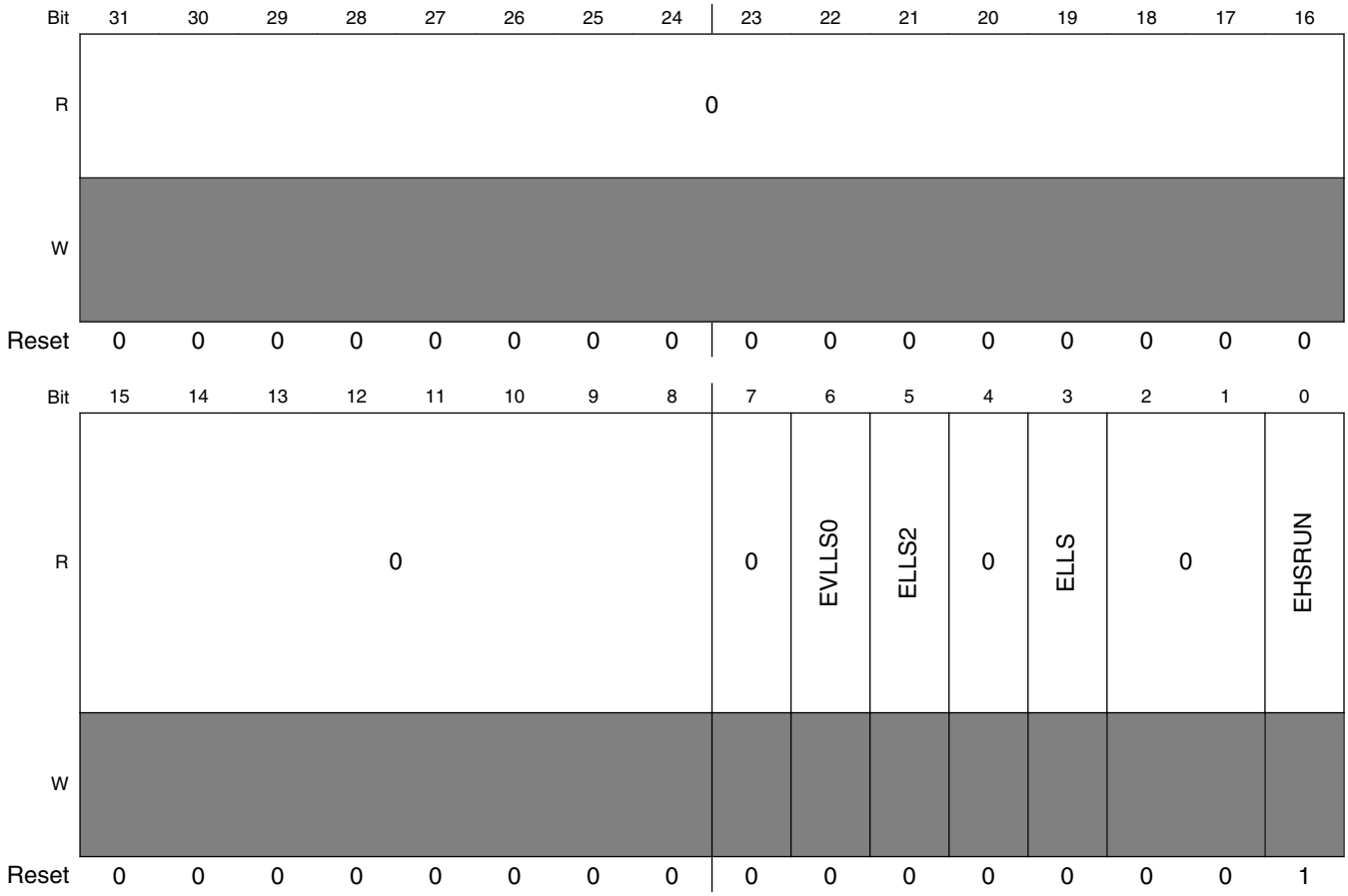
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SMC_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0000 Standard features implemented

26.4.2 SMC Parameter Register (SMC_PARAM)

Address: 4007_E000h base + 4h offset = 4007_E004h



SMC_PARAM field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 EVLLS0	Existence of VLLS0 feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
5 ELLS2	Existence of LLS2 feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

Table continues on the next page...

SMC_PARAM field descriptions (continued)

Field	Description
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ELLS	Existence of LLS feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
2–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 EHSRUN	Existence of HSRUN feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

26.4.3 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.

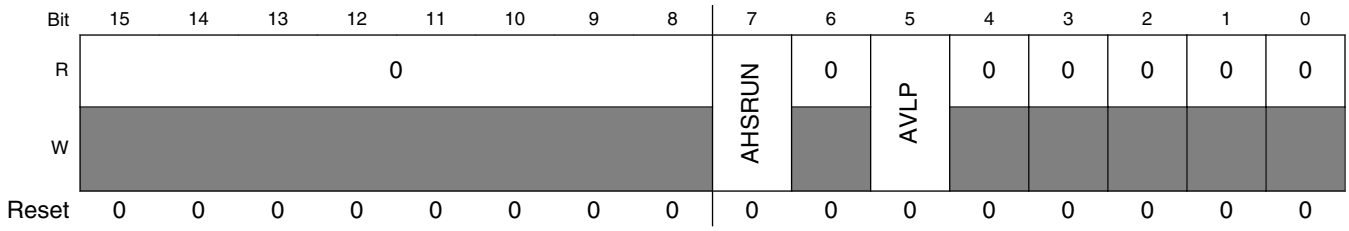
If the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the MCU is still in Normal Run mode.

NOTE

This register is reset on Chip Reset and by reset types that trigger Chip Reset. It is unaffected by reset types that do not trigger Chip Reset. See the Reset section details for more information.

Address: 4007_E000h base + 8h offset = 4007_E008h

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



SMC_PMPROT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 AHSRUN	Allow High Speed Run mode Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter High Speed Run mode (HSRUN). 0 HSRUN is not allowed 1 HSRUN is allowed
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 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 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.

26.4.4 Power Mode Control register (SMC_PMCTRL)

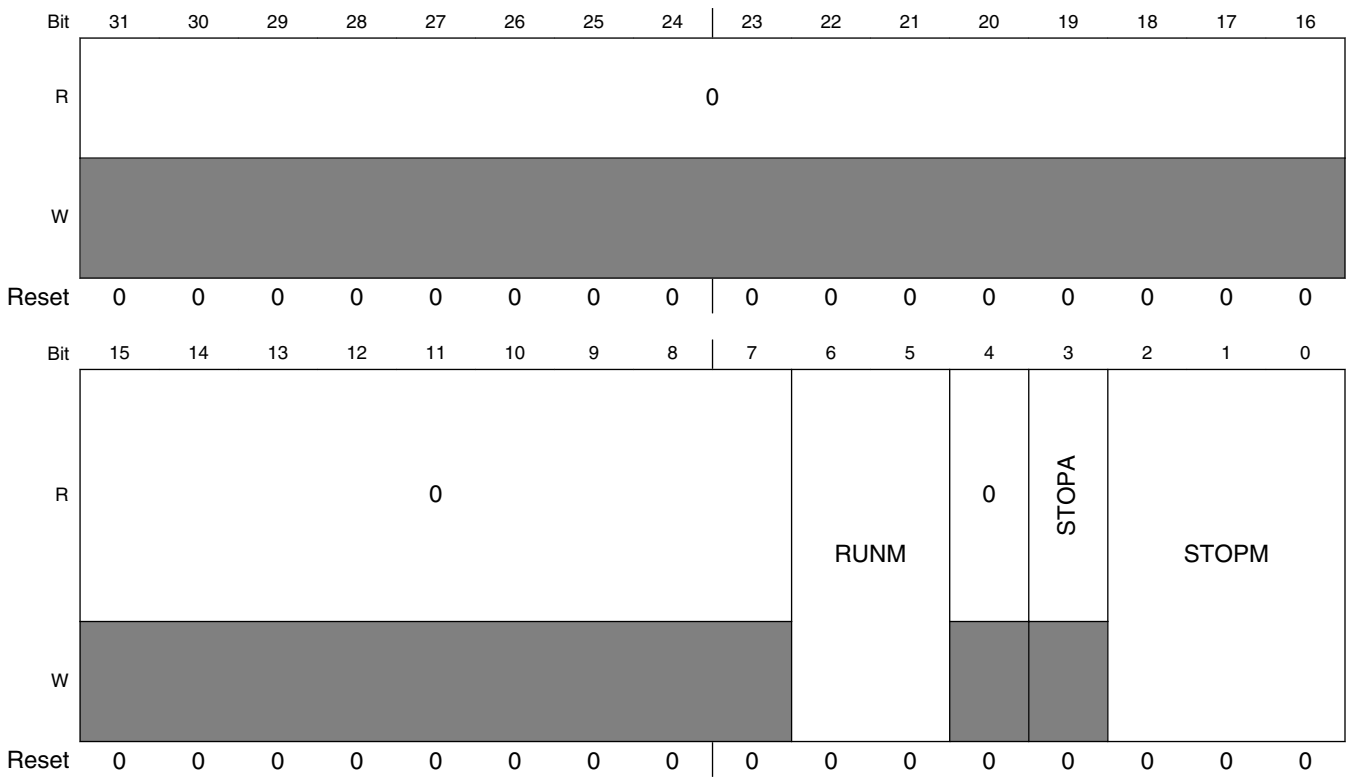
The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR and by reset types that trigger Chip POR. It is unaffected by reset types that do not

trigger Chip POR. See the Reset section details for more information.

Address: 4007_E000h base + Ch offset = 4007_E00Ch



SMC_PMCTRL field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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. **NOTE:** RUNM may be set to HSRUN only when PMSTAT=RUN. After being programmed to HSRUN, RUNM should not be programmed back to RUN until PMSTAT=HSRUN. Also, stop mode entry should not be attempted while RUNM=HSRUN or PMSTAT=HSRUN. When in HSRUN mode, any reset clears RUNM and causes the system to exit to normal RUN mode after the MCU exits its reset flow. 00 Normal Run mode (RUN) 01 Reserved 10 Very-Low-Power Run mode (VLPR) 11 High Speed Run mode (HSRUN)

Table continues on the next page...

SMC_PMCTRL field descriptions (continued)

Field	Description
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 occured during the previous stop mode entry sequence, preventing the system from entering that mode. This field is cleared by reset or 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 successful. 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 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 101 Reserved 110 Reseved 111 Reserved

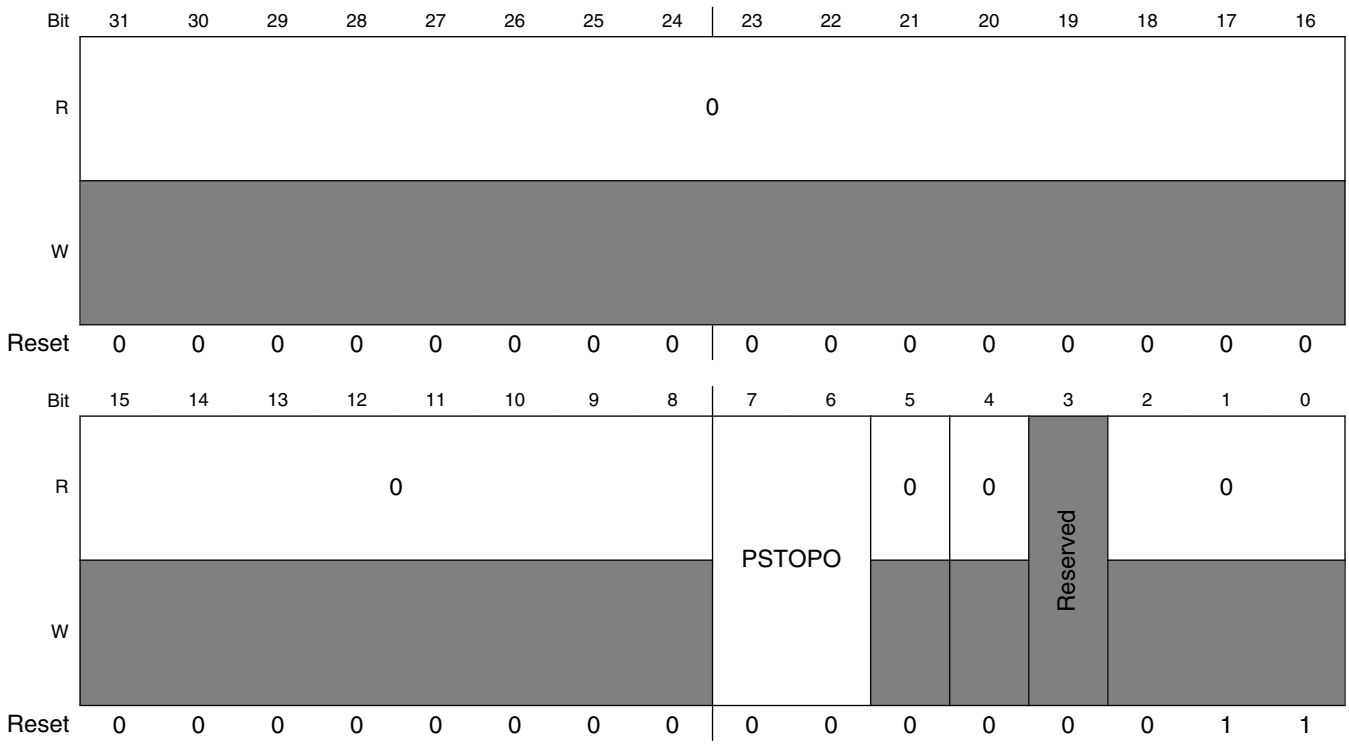
26.4.5 Stop Control Register (SMC_STOPCTRL)

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 and by reset types that trigger Chip POR. It is unaffected by reset types that do not trigger Chip POR. See the Reset section details for more information.

Address: 4007_E000h base + 10h offset = 4007_E010h



SMC_STOPCTRL field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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 mode, the PMC, SCG 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 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 bit is reserved for future expansion. Software should write 0 to this bit to maintain compatibility.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

26.4.6 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 and by reset types that trigger Chip POR. It is unaffected by reset types that do not trigger Chip POR. See the Reset section details for more information.

When in HSRUN mode, any reset causes the system to exit to normal RUN mode after the MCU exits its reset flow. The PMSTAT field is then automatically updated to show RUN as the current power mode.

Address: 4007_E000h base + 14h offset = 4007_E014h

Bit	31	30	29	28	27	26	25	24	23	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																								PMSTAT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SMC_PMSTAT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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 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 Reserved 1000_0000 Current power mode is HSRUN

26.5 Functional description

26.5.1 Power mode transitions

The following figure shows the power mode state transitions available on the chip. Any reset always brings the MCU back to the normal RUN state.

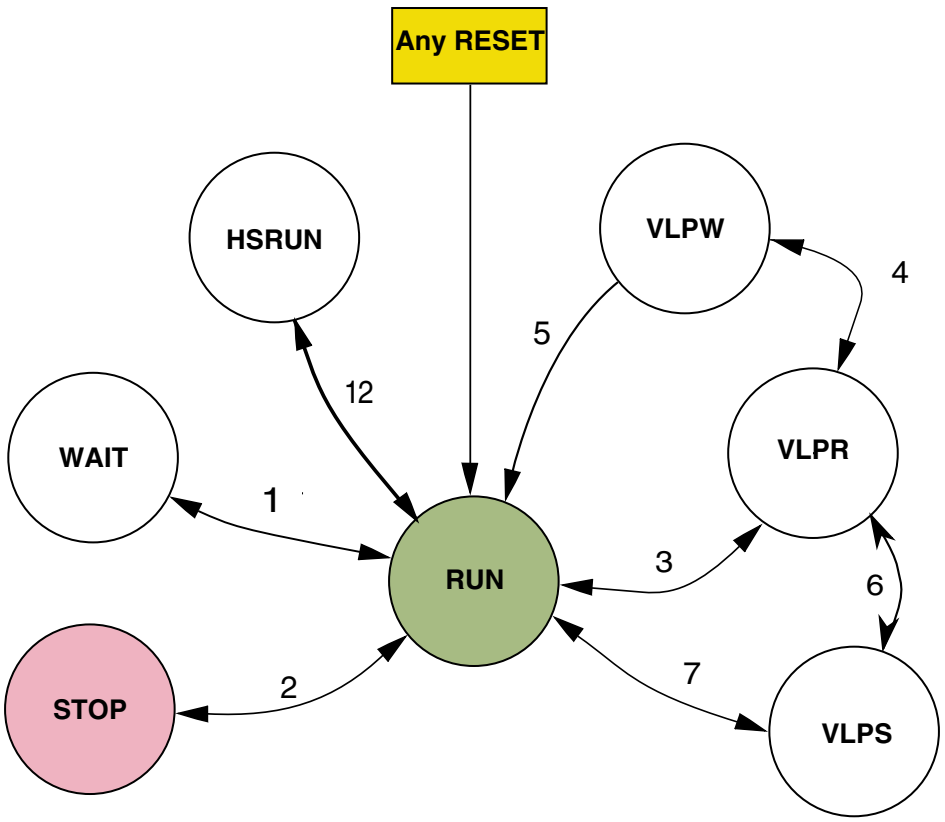


Figure 26-1. Power mode state diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 26-2. 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]

Table continues on the next page...

Table 26-2. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
	STOP	RUN	Interrupt or Reset
3	RUN	VLPR	The core, system, bus and flash clock frequencies and SCG clocking mode are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies and SCG modes supported. Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10.
	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
12	RUN	HSRUN	Set PMPROT[AHSRUN]=1, PMCTRL[RUNM]=11.
	HSRUN	RUN	Set PMCTRL[RUNM]=00 or Reset

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

26.5.2 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.

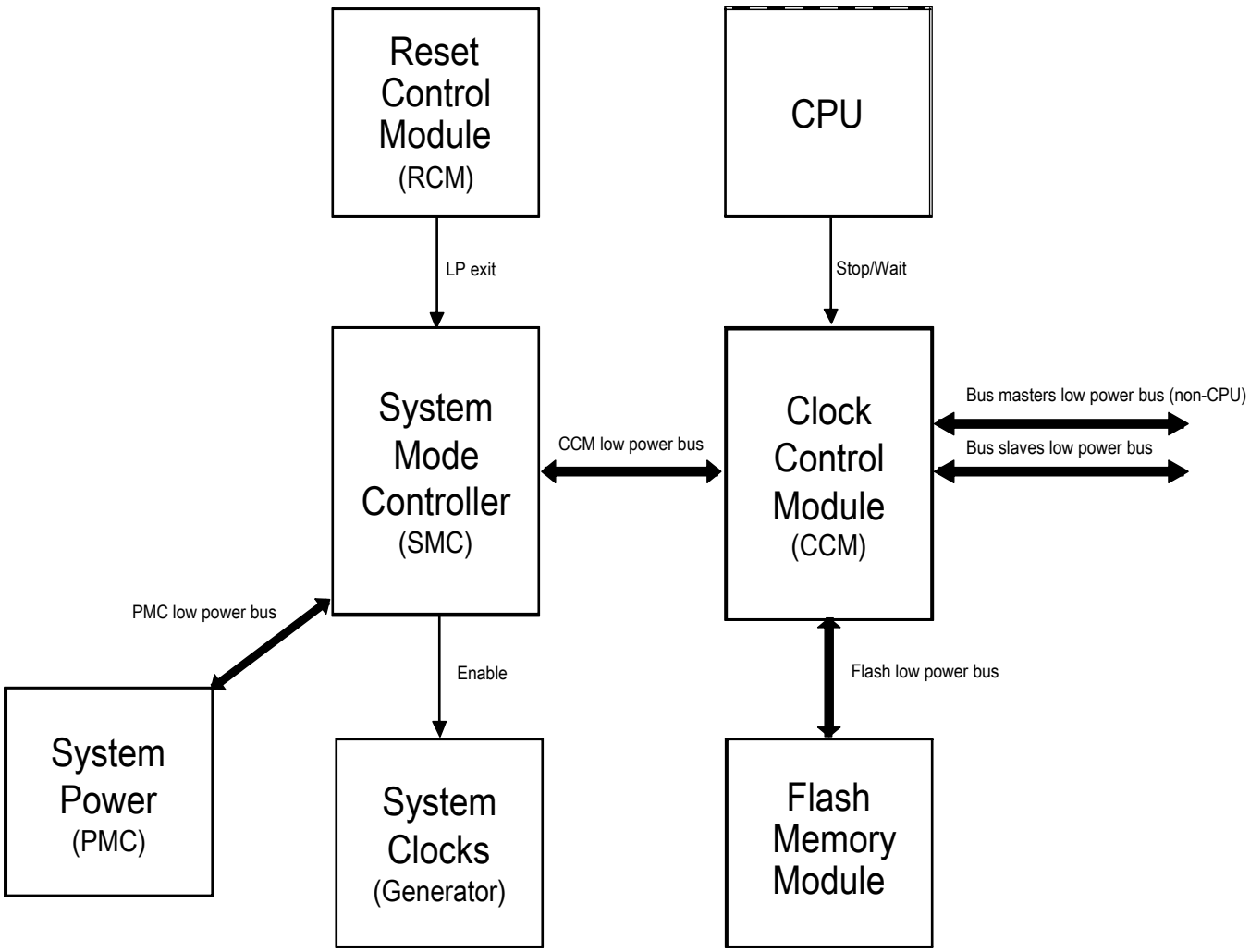


Figure 26-2. Low-power system components and connections

26.5.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS) 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 SCG unless configured to be enabled in Stop mode. See the SCG module information for the programming options.
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.

26.5.2.2 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 SCG.
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.

26.5.2.3 Aborted stop mode entry

If an interrupt 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 interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt 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.

26.5.2.4 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.

26.5.2.5 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.

26.5.3 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)
- High Speed Run (HSRUN)

26.5.3.1 RUN mode

This is the normal operating mode for the device.

This mode is selected after any reset. When the Arm processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF_FFFF.

To reduce power in this mode, disable the clocks to unused modules.

26.5.3.2 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 PCC's registers.

Before entering this mode, the following conditions must be met:

- All clock monitors in the SCG 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 SCG module or any clock divider registers. Module clock enables in the PCC can be set, but not cleared.

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 always causes an exit from VLPR and returns the device to RUN mode after the MCU exits its reset flow.

26.5.3.3 High Speed Run (HSRUN) mode

In HSRUN mode, the on-chip voltage regulator remains in a run regulation state, but with a slightly elevated voltage output. In this state, the MCU is able to operate at a faster frequency compared to normal RUN mode. For the maximum allowable frequencies, see the Power Management chapter.

While in this mode, the following restrictions must be adhered to:

- The maximum allowable change in frequency of the system, bus, flash or core clocks is restricted to 2x (double the frequency).
- Stop mode entry is not supported from HSRUN.
- Modifications to clock gating control bits are prohibited.
- Flash programming/erasing is not allowed.

To enter HSRUN mode, set PMPROT[AHSRUN] to allow HSRUN and then set PMCTRL[RUNM]=HSRUN. Before increasing clock frequencies, the PMSTAT register should be polled to determine when the system has completed entry into HSRUN mode. To reenter normal RUN mode, clear PMCTRL[RUNM]. Any reset also clears PMCTRL[RUNM] and causes the system to exit to normal RUN mode after the MCU exits its reset flow.

26.5.4 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

26.5.4.1 WAIT mode

WAIT mode is entered when the Arm core enters the Sleep-Now or Sleep-On-Exit modes while SLEEPDEEP 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.

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 causes an exit from WAIT mode, returning the device to normal RUN mode.

26.5.4.2 Very-Low-Power Wait (VLPW) mode

VLPW mode is entered by entering the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the device 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 device at a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules.

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 causes an exit from VLPW mode, returning the device to normal RUN mode.

26.5.5 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.

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)

26.5.5.1 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 SCG module can be configured to leave the reference clocks running.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, returning the device to normal RUN mode via an MCU reset.

26.5.5.2 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 also cause a VLPS exit, returning the device to normal RUN mode.

26.5.6 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 SCG-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.

Chapter 27
Power Management Controller (PMC)

27.1 Chip-specific Information for this Module

NOTE

If needed in some case, PMC_REGSC[CLKBIASDIS] should be set manually before entering STOP or VLPS mode. See CLKBIASDIS for more information. In the bitfield description, "RPM" is an alias of Low Power Mode (LPM).

27.2 Introduction

The PMC contains the internal voltage regulator, power on reset (POR), the low voltage reset (LVR) and the low voltage detect (LVD) systems.

27.3 Features

The PMC features include:

- Internal voltage regulator offering a variety of power modes
- Active POR providing brown-out detect
- Low voltage reset (LVR)
- Low voltage detect supporting two low voltage trip points (V_{LVD} and V_{LVW}) and interrupt
- Low power oscillator (LPO) with a typical frequency of 128 kHz

27.4 Modes of Operation

27.4.1 Full Performance Mode (FPM)

For the following Chip Power Modes, the internal voltage regulator is in full performance mode: HSRUN, RUN, WAIT.

27.4.2 Low Power Mode (LPM)

For the following Chip Power Modes, the internal voltage regulator is in low power mode: STOP, VLPR, VLPW, VLPS.

27.5 Low Voltage Detect (LVD) System

NOTE

The low voltage detect system (Low voltage detect flag, Low voltage warning flag and Low voltage detect reset generation) is disabled in low power mode.

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, a low voltage reset (LVR) circuit and a low voltage detect (LVD) circuit with two trip points (V_{LVD} and V_{LVW}). The LVD is disabled upon entering low power mode.

Two flags are available to indicate the status of the low voltage detect system:

- The low voltage detect flag (LVDF) operates in a level sensitive manner. The LVDF bit is set when the supply voltage falls below the trip point (V_{LVD}). The LVDF bit is cleared by writing one to the LVDACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVDF bit remains set.
- The low voltage warning flag (LVWF) operates in a level sensitive manner. The LVWF bit is set when the supply voltage falls below the selected monitor trip point (V_{LVW}). The LVWF bit is cleared by writing one to the LVWACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVWF bit remains set.

NOTE

This flag (LVDF/LVWF) gets cleared on reset. The flag is only valid after the device has come out of the reset, at which point the flag will be set accordingly to the voltage level. If supply level is higher than LVD/LVW threshold then this flag stay cleared, else this flag gets set.

27.5.1 Low Voltage Reset (LVR) Operation

If the supply voltage falls below the reset trip point (V_{LVR}), a system reset will be generated.

If PMC_LVDSC1[LVDRE] is set and the supply voltage falls below V_{LVD}, a system reset will be generated.

PMC_LVDSC1[LVDF] will be cleared by system reset, so after recovery PMC_LVDSC1[LVDF] will read zero. Usage of PMC_LVDSC1[LVDF] is intended for LVD interrupt opteration only (for example, PMC_LVDSC1[LVDIE] = 1 and PMC_LVDSC1[LVDRE] = 0).

27.5.2 LVD Interrupt Operation

By configuring the LVD circuit for interrupt operation (LVDIE set), PMC_LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. The LVDF bit is cleared by writing one to the PMC_LVDSC1[LVDACK] bit, when the supply returns to above the trip point.

27.5.3 Low-voltage warning (LVW) interrupt operation

The LVD system contains a low-voltage warning flag (LVWF) to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by setting the PMC_LVDSC2[LVWIE] bit. If enabled, an LVW interrupt request occurs when the LVWF is set. LVWF is cleared by writing one to the PMC_LVDSC2[LVWACK] bit, when the supply returns to above the trip point.

27.6 Memory Map and Register Definition

This sections provides the detailed information of all registers for the PMC module.

NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details.

NOTE

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	See section	27.6.1/686
4007_D001	Low Voltage Detect Status and Control 2 Register (PMC_LVDSC2)	8	R/W	00h	27.6.2/687
4007_D002	Regulator Status and Control Register (PMC_REGSC)	8	R/W	See section	27.6.3/688
4007_D004	Low Power Oscillator Trim Register (PMC_LPOTRIM)	8	R/W	See section	27.6.4/689

27.6.1 Low Voltage Detect Status and Control 1 Register (PMC_LVDSC1)

This register contains status and control bits to support the low voltage detect function.

NOTE

When the internal voltage regulator is in low power mode, the LVD system is disabled, regardless of the PMC_LVDSC1 settings.

Address: 4007_D000h base + 0h offset = 4007_D000h

Bit	7	6	5	4	3	2	1	0
Read	LVDF		LVDIE	LVDRE	0			
Write		LVDACK						
Reset	0	0	0	u*	0	0	0	0
POR	0	0	0	0	0	0	0	0

* Notes:
- u = Unaffected by reset.

PMC_LVDSC1 field descriptions

Field	Description
7 LVDF	Low Voltage Detect Flag This bit's read-only status bit indicates a low-voltage detect event. The threshold voltage is V_{LVD}. 0 Low-voltage event not detected 1 Low-voltage event detected
6 LVDACK	Low Voltage Detect Acknowledge This write-only bit is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Read always return 0.
5 LVDIE	Low Voltage Detect Interrupt Enable This bit enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1
4 LVDRE	Low Voltage Detect Reset Enable This bit enables the low voltage detect events to generate a system reset. 0 No system resets on low voltage detect events. 1 If the supply voltage falls below V_{LVD}, a system reset will be generated.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

27.6.2 Low Voltage Detect Status and Control 2 Register (PMC_LVDSC2)

This register contains status and control bits to support the low voltage warning (LVW) function.

NOTE

When the internal voltage regulator is in low power mode, the LVD system is disabled regardless of the PMC_LVDSC2 settings.

Address: 4007_D000h base + 1h offset = 4007_D001h

Bit	7	6	5	4	3	2	1	0
Read	LVWF		LVWIE	0				
Write		LVWACK						
Reset	0	0	0	0	0	0	0	0

PMC_LVDSC2 field descriptions

Field	Description
7 LVWF	Low-Voltage Warning Flag This bit read-only status bit indicates a low-voltage detect event. The threshold voltage is V_{LVW}. 0 Low-voltage warning event not detected 1 Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge This write-only bit is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.
5 LVWIE	Low-Voltage Warning Interrupt Enable This bit enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF=1
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

27.6.3 Regulator Status and Control Register (PMC_REGSC)

This register contains general control and status bits for the regulator and the LPO.

Address: 4007_D000h base + 2h offset = 4007_D002h

Bit	7	6	5	4	3	2	1	0
Read	LPODIS	LPOSTAT	0			REGFPM	CLKBIASDIS	BIASEN
Write								
Reset	u*	*	0	0	0	1	0	0
POR	0	*	0	0	0	1	0	0

* Notes:
- u = Unaffected by reset.
- LPOSTAT field: Reset value is undefined.

PMC_REGSC field descriptions

Field	Description
7 LPODIS	LPO Disable Bit This bit enables or disable the low power oscillator. **NOTE:** After disabling the LPO a time of 2 LPO clock cycles is required before it is allowed to enable it again. Violating this waiting time of 2 cycles can result in malfunction of the LPO. 0 Low power oscillator enabled 1 Low power oscillator disabled

Table continues on the next page...

PMC_REGSC field descriptions (continued)

Field	Description
6 LPOSTAT	LPO Status Bit This bit shows the status of the LPO clock to be either in high phase (logic 1) or low phase (logic 0) of the clock period. Software can poll this status bit to measure actual LPO clock frequency and eventually use the LPOTRIM[4:0] register to change the LPO frequency. 0 Low power oscillator in low phase 1 Low power oscillator in high phase
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 REGFPM	Regulator in Full Performance Mode Status Bit This read-only bit provides the current status of the internal voltage regulator. 0 Regulator is in low power mode or transition to/from 1 Regulator is in full performance mode
1 CLKBIASDIS	Clock Bias Disable Bit This bit disables the bias currents and reference voltages for some clock modules in order to further reduce power consumption in STOP or VLPS mode if all clocks are disabled. The bias currents and reference voltages for LPFLL (if available on device) are always disabled in LPM. **Note: Using this bit it must be ensured that respective clock modules are disabled in STOP or VLPS mode. Else severe malfunction of clock modules will happen.** 0 No effect 1 In STOP or VLPS mode the bias currents and reference voltages for the following clock modules are disabled: SIRC, FIRC, PLL. (if available on device)
0 BIASEN	Bias Enable Bit This bit enables source and well biasing for the core logic in low power mode. In full performance mode this bit has no effect. This is useful to further reduce MCU power consumption in low power mode. 0 Biasing disabled, core logic can run in full performance 1 Biasing enabled, core logic is slower and there are restrictions in allowed system clock speed (see *Data Sheet* for details)

27.6.4 Low Power Oscillator Trim Register (PMC_LPOTRIM)

This register contains the period trimming bits for the low power oscillator.

Table 27-1. Trimming effect of LPOTRIM[4:0]

LPOTRIM[4:0]	Decimal	Period of LPO clock
10000	–16	lowest
10001	–15	increasing
...	...	
11110	–2	
11111	–1	

Table continues on the next page...

Table 27-1. Trimming effect of LPOTRIM[4:0] (continued)

LPOTRIM[4:0]	Decimal	Period of LPO clock
00000	0	typical 128 kHz
00001	+1	increasing
...	...	
01110	+14	
01111	+15	highest

NOTE

The LPO trimming bits represent signed values. Starting from -16 the period of the LPO clock will increase monotonically (for example, frequency decreases monotonically).

Address: 4007_D000h base + 4h offset = 4007_D004h

Bit	7	6	5	4	3	2	1	0
Read	0			LPOTRIM				
Write								
Reset	0	0	0	*	*	*	*	*
POR	0	0	0	0*	0*	0*	0*	0*

* Notes:
- LPOTRIM field: After POR reset, automatically loaded from Flash Memory IFR after Reset (normal system reset).

PMC_LPOTRIM field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LPOTRIM	LPO trimming bits These bits are used for trimming the frequency of the low power oscillator. See the table above for trimming effect.

Chapter 28
Security

28.1 Introduction

This chapter summarizes all security related features of this device.

28.2 Flash security feature summary

The flash security features supported by this MCU are summarized here.

28.2.1 Flash security byte

Security state can be enabled via programming Flash security byte (FSEC at 0x0000 040E) in the flash configuration field (a 16 Byte region start from 0x0000 0400). User can program the FSEC byte using FTFE flash program phrase commands. The FSEC byte will be loaded into FTFE_FSEC register during boot sequence after chip reset. The FTFE_FSEC register is read-only.

The SEC bit of FSEC byte controls the chip security status. After enabling device security, the debug port (SWD) cannot access the memory resources of the MCU, and ROM boot loader also limited to access flash and not allows reading out flash information via ROM boot loader command.

The flash security byte (FSEC) also allow user to enable the flash backdoor key access feature by configuring the KEYEN bits. When backdoor Key is enabled, the software can unsecure the MCU after presenting the correct backdoor key with Verify Backdoor Access Key command.

The MEEN bit of FSEC byte can be used to disable the mass erase capability from debug port and the FlashEraseAllUnsecure command from ROM bootloader.

The FSLACC bit of FSEC byte can be used to disable the NXP failure analysis. The FSLACC bit permits the user to disable all special or test mode which is only accessible by NXP. This feature help user to achieve a highest level to control the access of MCU on chip data.

Please refer to FSEC sections of the FTFE chapter for more details.

From debug port point of view, user can only disable the secure mode by the external mass erase bit from SWD. But if Mass Erase is disabled, the debug port can no longer unsecure the MCU. Please refer to the "Debug and security" section in the Debug chapter for more details.

From ROM bootloader point of view, user can only disable the secure mode by FlashEraseAllUnsecure command or FlashSecurityDisable command. When Mass Erase is disabled, FlashEraseAllUnsecure command can no longer unsecure the MCU. When backdoor key access is disabled, FlashSecurityDisable command cannot be used. Please refer to the ROM chapter for more details.

28.2.2 Flash access protection (FAC)

Flash access controls (FAC) are a configurable memory protection scheme designed to allow end users to utilize software libraries while offering programmable restrictions to these libraries. This allows NXP or third-party vendors to pre-program software libraries into a chip and distribute parts to end customers who can use the pre-programmed software libraries.

Please refer to the Application Note AN5112: Using the Kinetis Flash Execute-Only Access Control Feature, and Flash Access Protection section in the FTFE chapter for more details.

28.3 Security hardware accelerators

28.3.1 CRC

This device contain one cyclic redundancy check (CRC) module which can generates 16/32-bit CRC code for error detection.

28.4 General security features

28.4.1 Unique ID

This device features 128-bit unique identification number, which programmed in factory and load to SIM register after power on reset. This unique ID permits the software to build a trusted device. This Unique ID generated based on the wafer lot and die series number of factory. The ID is unique for each device and it is accessible from SIM_UIDH, SIM_UIDMH, SIM_UIDML and SIM_UIDL registers. Please refer to the SIM chapter for more details.

28.4.2 Program Once Field

This device also contains 96 bytes Program Once Field in the program flash 0 IFR. User can program specific data into this field by FTFE Program Once command with index 0x00 ~ 0x07. The data can no longer be erased nor modified after programming. The Program Once Field can be read through Read Once commands. Please refer to Program Once field section in the FTFE chapter for more details.

28.5 On-chip resource access control mechanism

The Memory Protection Unit (MPU) provides hardware access control for all memory references generated in the device. 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. See the MPU chapter for more details.

Chapter 29
External Watchdog Monitor (EWM)

29.1 Introduction

For safety, a redundant watchdog system, External Watchdog Monitor (EWM), is designed to monitor external circuits, as well as the MCU software flow. This provides a back-up mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The watchdog is generally used to monitor the flow and execution of embedded software within an MCU. The watchdog consists of a counter that if allowed to overflow, forces an internal reset (asynchronous) to all on-chip peripherals and optionally assert the $\overline{\text{RESET}}$ 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.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM provides an independent $\overline{\text{EWM\_out}}$ signal that when asserted resets or places an external circuit into a safe mode. The $\overline{\text{EWM\_out}}$ signal is asserted upon the EWM counter time-out. An optional external input EWM_in is provided to allow additional control of the assertion of $\overline{\text{EWM\_out}}$ signal.

29.1.1 Features

Features of EWM module include:

- Independent LPO_CLK clock source
- Programmable time-out period specified in terms of number of EWM LPO_CLK clock cycles.
- Windowed refresh option
    - Provides robust check that program flow is faster than expected.

- Programmable window.
- Refresh outside window leads to assertion of $\overline{\text{EWM\_out}}$.

- Robust refresh mechanism
  - Write values of 0xB4 and 0x2C to EWM Refresh Register within 15 (*EWM_refresh_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 assertion of the $\overline{\text{EWM\_out}}$ signal.

29.1.2 Modes of Operation

This section describes the module's operating modes.

29.1.2.1 Stop Mode

When the EWM is in stop mode, the CPU refreshes to the EWM cannot occur. On entry to stop mode, the EWM's counter freezes.

There are two possible ways to exit from Stop mode:

- On exit from stop mode through a reset, the EWM remains disabled.
- On exit from stop mode by an interrupt, the EWM is re-enabled, and the counter continues to be clocked from the same value prior to entry to stop mode.

Note the following if the EWM enters the stop mode during CPU refresh mechanism: At the exit from stop mode by an interrupt, refresh mechanism state machine starts from the previous state which means, if first refresh command is written correctly and EWM enters the stop mode immediately, the next command has to be written within the next 15 (*EWM_refresh_time*) peripheral bus clocks after exiting from stop mode. User must mask all interrupts prior to executing EWM refresh instructions.

29.1.2.2 Wait Mode

The EWM module treats the stop and wait modes as the same. EWM functionality remains the same in both of these modes.

29.1.2.3 Debug Mode

Entry to debug mode has no effect on the EWM.

- If the EWM is enabled prior to entry of debug mode, it remains enabled.
- If the EWM is disabled prior to entry of debug mode, it remains disabled.

29.1.3 Block Diagram

This figure shows the EWM block diagram.

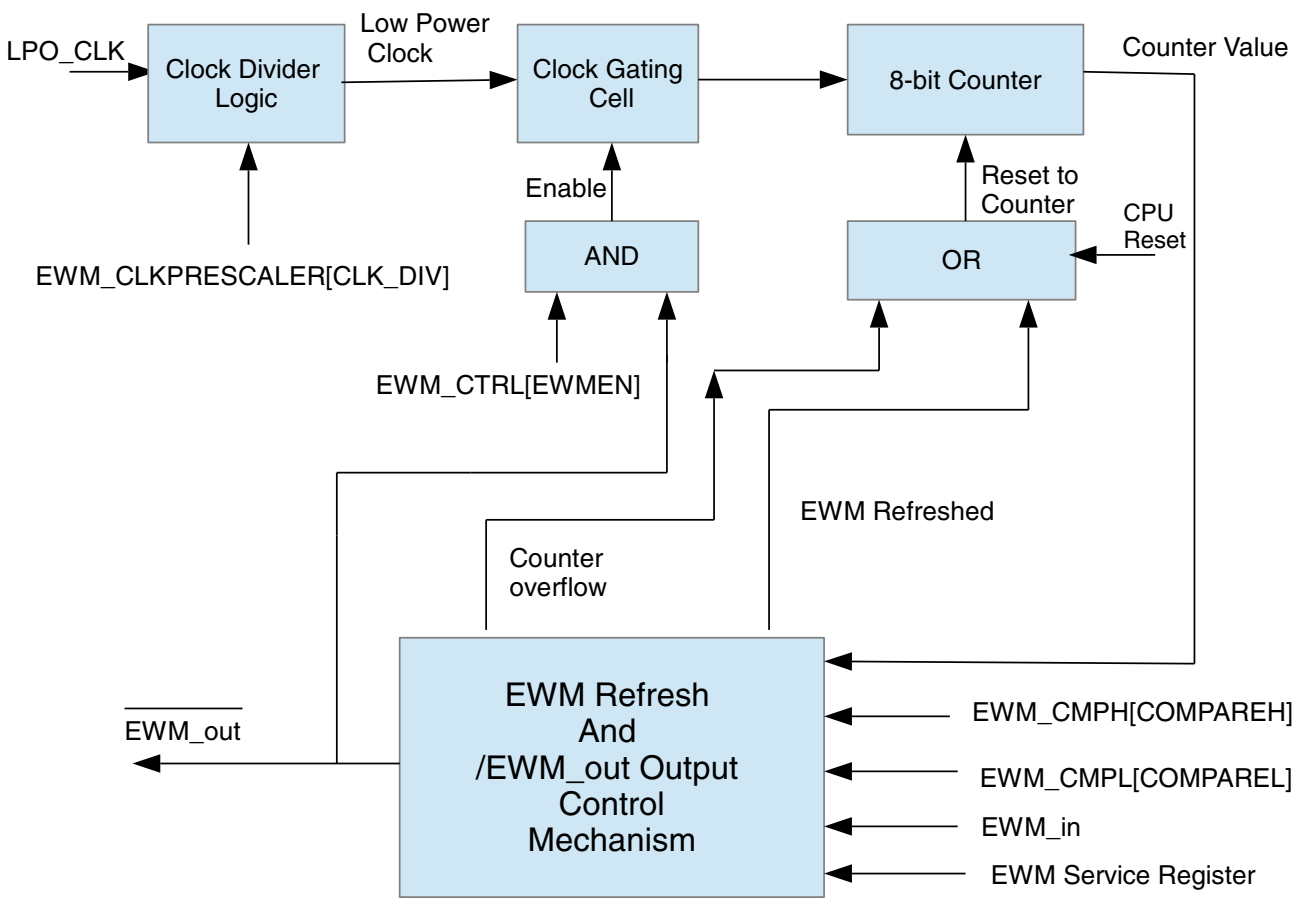


Figure 29-1. EWM Block Diagram

29.2 EWM Signal Descriptions

The EWM has two external signals, as shown in the following table.

Table 29-1. EWM Signal Descriptions

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

29.3 Memory Map/Register Definition

This section contains the module memory map and registers.

EWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_1000	Control Register (EWM_CTRL)	8	R/W	00h	29.3.1/698
4006_1001	Service Register (EWM_SERV)	8	W (always reads 0)	00h	29.3.2/699
4006_1002	Compare Low Register (EWM_CMPL)	8	R/W	00h	29.3.3/699
4006_1003	Compare High Register (EWM_CMPH)	8	R/W	FFh	29.3.4/700
4006_1005	Clock Prescaler Register (EWM_CLKPRESCALER)	8	R/W	00h	29.3.5/701

29.3.1 Control Register (EWM_CTRL)

The CTRL register is cleared by any reset.

NOTE

INEN, ASSIN and EWMEN bits can be written once after a CPU reset. Modifying these bits more than once, generates a bus transfer error.

Address: 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.
1 ASSIN	EWM_in's Assertion State Select. Default assert state of the EWM_in signal is logic zero. Setting the ASSIN bit inverts the assert state of EWM_in signal to a logic one.
0 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. This bit when unset, keeps the EWM module disabled. It cannot be re-enabled until a next reset, due to the write-once nature of this bit.

29.3.2 Service Register (EWM_SERV)

The SERV register provides the interface from the CPU to the EWM module. It is write-only and reads of this register return zero.

Address: 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 refresh mechanism requires the CPU to write two values to the SERV register: a first data byte of 0xB4, followed by a second data byte of 0x2C. The EWM refresh is invalid if either of the following conditions is true. • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte. This fixed number of cycles is called *EWM_refresh_time*.

29.3.3 Compare Low Register (EWM_CMPL)

The CMPL register is reset to zero after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

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 refresh time is required.

29.3.4 Compare High Register (EWM_CMPH)

The CMPH register is reset to 0xFF after a CPU reset. This provides a maximum of 256 clocks time, for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

The valid values for CMPH are up to 0xFE because the EWM counter never expires when CMPH = 0xFF. The expiration happens only if EWM counter is greater than CMPH.

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 refresh time is required.

29.3.5 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-specific information for 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)

29.4 Functional Description

The following sections describe functional details of the EWM module.

NOTE

When the BUS_CLK is lost, then EWM module doesn't generate the $\overline{\text{EWM\_out}}$ signal and no refresh operation is possible

29.4.1 The $\overline{\text{EWM\_out}}$ Signal

The $\overline{\text{EWM\_out}}$ is a digital output signal used to gate an external circuit (application specific) that controls critical safety functions. For example, the $\overline{\text{EWM\_out}}$ could be connected to the high voltage transistors circuits that control an AC motor in a large appliance.

The $\overline{\text{EWM\_out}}$ signal remains deasserted when the EWM is being regularly refreshed by the CPU within the programmable refresh window, indicating that the application code is executed as expected.

The $\overline{\text{EWM\_out}}$ signal is asserted in any of the following conditions:

- The EWM refresh occurs when the counter value is less than CMPL value.
- The EWM counter value reaches the CMPH value, and no EWM refresh has occurred.
- If functionality of EWM_in pin is enabled and EWM_in pin is asserted while refreshing the EWM.
- After any reset (by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ pin)

The $\overline{\text{EWM\_out}}$ is asserted after any reset by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ signal. Then, to deassert the $\overline{\text{EWM\_out}}$ signal, 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. The pad state is controlled by the $\overline{\text{EWM\_out}}$ signal only after the EWM is enabled by the EWMEN bit in the CTRL register.

Note

$\overline{\text{EWM\_out}}$ pad must be in pull down state when EWM functionality is used and when EWM is under Reset.

29.4.2 The EWM_in Signal

The EWM_in is a digital input signal for safety status of external safety circuits, that allows an external circuit to control the assertion of the $\overline{\text{EWM\_out}}$ signal. For example, in the application, an external circuit monitors a critical safety function, and if there is fault with safety function, the external circuit can then actively initiate the $\overline{\text{EWM\_out}}$ 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 start refreshing the EWM. This ensures that the $\overline{\text{EWM\_out}}$ stays in the deasserted state; otherwise, the $\overline{\text{EWM\_out}}$ output signal is asserted.

Note

The user must update the CMPH and CMPL registers prior to enabling the EWM. After enabling the EWM, the counter resets to zero, therefore the user shall provide a reasonable time after a power-on reset for the external monitoring circuit to stabilize. The user shall also ensure that the EWM_in pin is deasserted.

29.4.3 EWM Counter

It is an 8-bit ripple counter fed from a clock source that is independent of the peripheral bus clock source. As the preferred time-out is between 1 ms and 100 ms the actual clock source should be in the kHz range.

The counter is reset to zero after the CPU reset, or when EWM refresh action completes, or at counter overflow. The counter value is not accessible to the CPU.

29.4.4 EWM Compare Registers

The compare registers CMPL and CMPH are write-once after a CPU reset and cannot be modified until another CPU reset occurs.

The EWM compare registers are used to create a refresh window to refresh the EWM module.

It is illegal to program CMPL and CMPH with same value. In this case, as soon as counter reaches (CMPL + 1), $\overline{\text{EWM\_out}}$ is asserted.

29.4.5 EWM Refresh Mechanism

Other than the initial configuration of the EWM, the CPU can only access the EWM by the EWM Service Register. The CPU must access the EWM service register with correct write of unique data within the windowed time frame as determined by the CMPL and CMPH registers for correct EWM refresh operation. Therefore, three possible conditions can occur:

Table 29-2. EWM Refresh Mechanisms

Condition	Mechanism
An EWM refresh action completes when: CMPL < Counter < CMPH.	The software behaves as expected and the EWM counter is reset to zero. The EWM_out output signal remains in the deasserted state if, during the EWM refresh action, the EWM_in input has been in deasserted state..
An EWM refresh action completes when Counter < CMPL	The software refreshes the EWM before the windowed time frame, the counter is reset to zero and the EWM_out output signal is asserted irrespective of the input EWM_in.
Counter value reaches CMPH prior to completion of EWM refresh action.	Software has not refreshed the EWM. The EWM counter is reset to zero and the EWM_out output signal is asserted irrespective of the input EWM_in.

29.4.6 EWM Interrupt

When 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 EWM_out. The EWM_out signal can be deasserted only by forcing a system reset.

29.4.7 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.

29.5 Usage Guide

29.5.1 EWM low-power modes

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

Table 29-3. EWM low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS

29.5.2 $\overline{\text{EWM\_out}}$ pin state in low power modes

During Wait, Stop, and Power Down modes the $\overline{\text{EWM\_out}}$ pin preserve its state before entering Wait or Stop mode. When the CPU enters a Run mode from Wait or Stop recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode from Power Down, the pin returns to its reset state.

29.5.3 Example code

29.5.3.1 Initializing the EWM

The following code segment shows the initialize sequence of the EWM module. It enables EWM_in pin input with assert state logic zero, enables interrupt when EWM_out is assert. The compare value is also set into CMPL/H register before enabling EWM.

```
// Initialize the EWM module
EWM_CMPL = compareValue & 0xFF;
EWM_CMPH = (compareValue >> 8) 0xFF;
EWM_CTRL = EWM_CTRL_INEN(1) | EWM_CTRL_ASSIN(0) |
           EWM_CTRL_INTEN(1) | EWM_CTRL_EWMEN(1);
```

29.5.3.2 Refreshing the EWM

The following code segment shows the refresh write sequence of the EWM module.

```
// Refresh EWM
DisableInterrupts; // disable global interrupt
EWM_SERV= 0xB4; // write the 1st refresh words
EWM_SERV= 0x2C; // write the 2nd refresh words
EnableInterrupts; // enable global interrupt
```

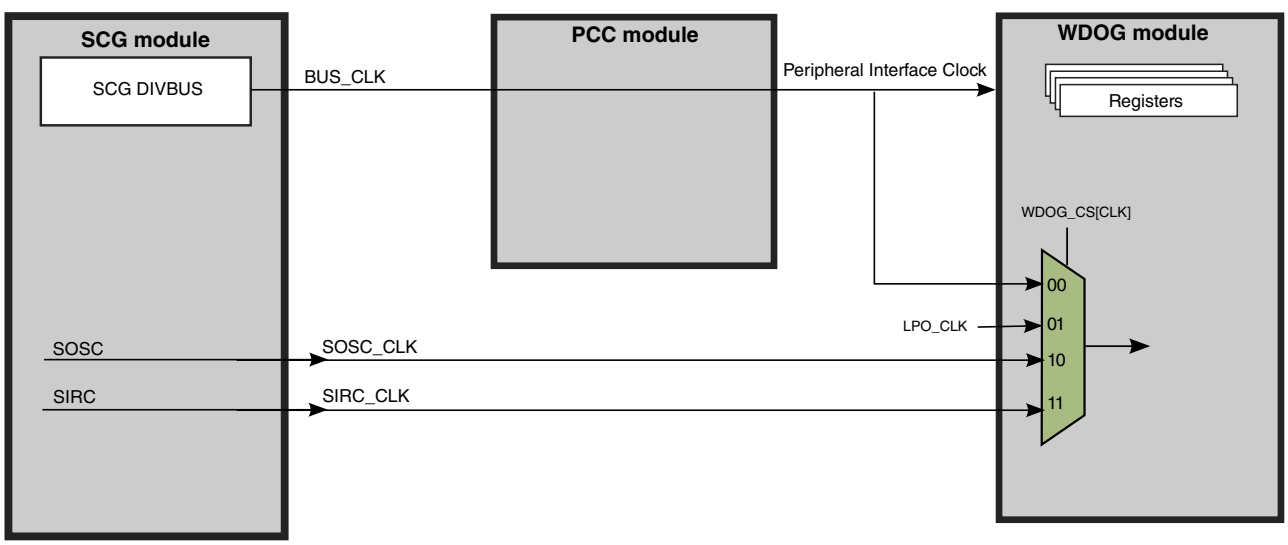
Chapter 30
Watchdog timer (WDOG)

30.1 Chip-specific information for this module

30.1.1 WDOG Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - WDOG



30.1.2 WDOG low-power modes

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

Table 30-1. WDOG low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS

30.2 Introduction

The Watchdog Timer (WDOG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the MCU.

30.2.1 Features

Features of the WDOG module include:

- Configurable clock source inputs independent from the bus clock
  - Bus clock (slow clock)
  - LPO clock (from PMC)
  - SIRC (8 MHz IRC from SCG)
  - ERCLK (external reference clock from SCG)
- Programmable timeout period
  - Programmable 16-bit timeout value
  - Optional fixed 256 clock prescaler when longer timeout periods are needed
- Robust write sequence for counter refresh
  - Refresh sequence of writing 0xA602 and then 0xB480
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value
  - Provides robust check that program flow is faster than expected
  - Early refresh attempts trigger a reset.
- Optional timeout interrupt to allow post-processing diagnostics

- Interrupt request to CPU with interrupt vector for an interrupt service routine (ISR)
- Forced reset occurs 128 bus clocks after the interrupt vector fetch.

- Configuration bits are write-once-after-reset to ensure watchdog configuration cannot be mistakenly altered.
- Robust write sequence for unlocking write-once configuration bits
  - Unlock sequence of writing 0xC520 and then 0xD928 for allowing updates to write-once configuration bits
  - Software must make updates within 128 bus clocks after unlocking and before WDOG closing unlock window.

30.2.2 Block diagram

The following figure shows a block diagram of the WDOG module.

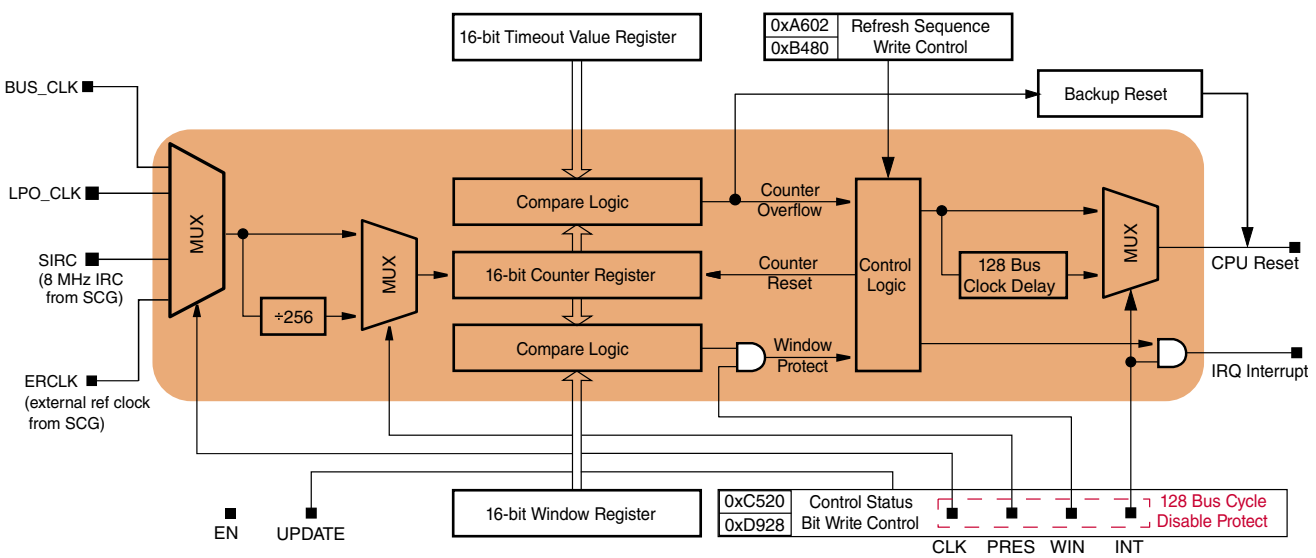


Figure 30-1. WDOG block diagram

30.3 Memory map and register definition

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_2000	Watchdog Control and Status Register (WDOG_CS)	32	R/W	See section	30.3.1/710
4005_2004	Watchdog Counter Register (WDOG_CNT)	32	R/W	0000_0000h	30.3.2/713
4005_2008	Watchdog Timeout Value Register (WDOG_TOVAL)	32	R/W	0000_0400h	30.3.3/713
4005_200C	Watchdog Window Register (WDOG_WIN)	32	R/W	0000_0000h	30.3.4/714

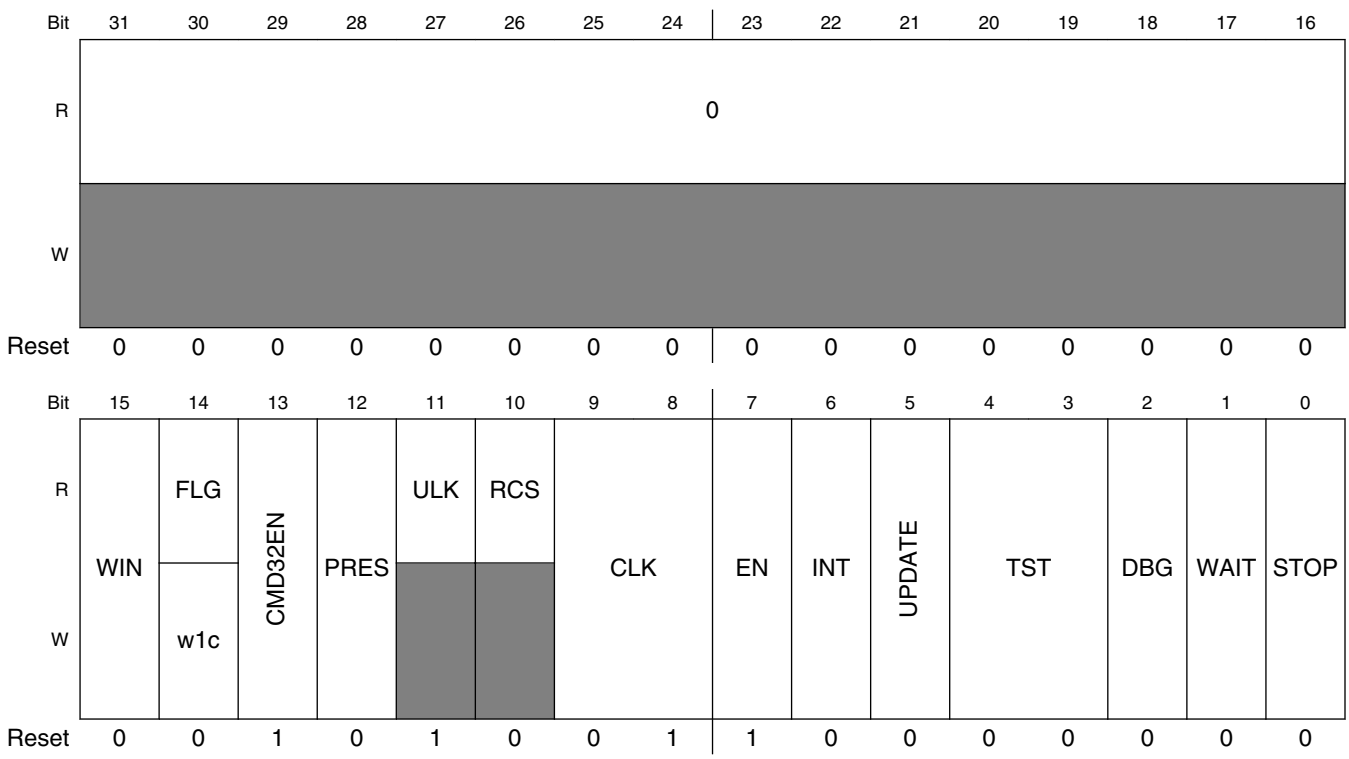
30.3.1 Watchdog Control and Status Register (WDOG_CS)

This section describes the function of Watchdog Control and Status Register.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

Address: 4005_2000h base + 0h offset = 4005_2000h



WDOG_CS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

WDOG_CS field descriptions (continued)

Field	Description
15 WIN	Watchdog Window This write-once bit enables window mode. See the Window mode section. 0 Window mode disabled. 1 Window mode enabled.
14 FLG	Watchdog Interrupt Flag This bit is an interrupt indicator when INT is set in control and status register 1. Write 1 to clear it. 0 No interrupt occurred. 1 An interrupt occurred.
13 CMD32EN	Enables or disables WDOG support for 32-bit (otherwise 16-bit or 8-bit) refresh/unlock command write words This is write-once field, and the user needs to unlock WDOG after writing this field for reconfiguration. 0 Disables support for 32-bit refresh/unlock command write words. Only 16-bit or 8-bit is supported. 1 Enables support for 32-bit refresh/unlock command write words. 16-bit or 8-bit is NOT supported.
12 PRES	Watchdog prescaler This write-once bit enables a fixed 256 pre-scaling of watchdog counter reference clock. (The block diagram shows this clock divider option.) 0 256 prescaler disabled. 1 256 prescaler enabled.
11 ULK	Unlock status This read-only bit indicates whether WDOG is unlocked or not. Default reset value is 1. 0 WDOG is locked. 1 WDOG is unlocked.
10 RCS	Reconfiguration Success This read-only bit indicates whether the reconfiguration is successful or not. Default reset value is 0.This bit is set when new configuration takes effect, and is cleared by successful unlock command. 0 Reconfiguring WDOG. 1 Reconfiguration is successful.
9–8 CLK	Watchdog Clock This write-once field indicates the clock source that feeds the watchdog counter. See the Clock source section. 00 Bus clock 01 LPO clock 10 System oscillator clock (SOSC, from SCG) 11 Slow internal reference clock (SIRC, from SCG)
7 EN	Watchdog Enable This write-once bit enables the watchdog counter to start counting.

Table continues on the next page...

WDOG_CS field descriptions (continued)

Field	Description
	0 Watchdog disabled. 1 Watchdog enabled.
6 INT	Watchdog Interrupt This write-once bit configures the watchdog to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), before forcing a reset. After the interrupt vector fetch (which comes after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. 0 Watchdog interrupts are disabled. Watchdog resets are not delayed. 1 Watchdog interrupts are enabled. Watchdog resets are delayed by 128 bus clocks from the interrupt vector fetch.
5 UPDATE	Allow updates This write-once bit allows software to reconfigure the watchdog without a reset. 0 Updates not allowed. After the initial configuration, the watchdog cannot be later modified without forcing a reset. 1 Updates allowed. Software can modify the watchdog configuration registers within 128 bus clocks after performing the unlock write sequence.
4–3 TST	Watchdog Test Enables the fast test mode. The test mode allows software to exercise all bits of the counter to demonstrate that the watchdog is functioning properly. See the Fast testing of the watchdog section. This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00 Watchdog test mode disabled. 01 Watchdog user mode enabled. (Watchdog test mode disabled.) After testing the watchdog, software should use this setting to indicate that the watchdog is functioning normally in user mode. 10 Watchdog test mode enabled, only the low byte is used. CNT[CNTLOW] is compared with TOVAL[TOVALLOW]. 11 Watchdog test mode enabled, only the high byte is used. CNT[CNTHIGH] is compared with TOVAL[TOVALHIGH].
2 DBG	Debug Enable This write-once bit enables the watchdog to operate when the chip is in debug mode. 0 Watchdog disabled in chip debug mode. 1 Watchdog enabled in chip debug mode.
1 WAIT	Wait Enable This write-once bit enables the watchdog to operate when the chip is in wait mode. 0 Watchdog disabled in chip wait mode. 1 Watchdog enabled in chip wait mode.
0 STOP	Stop Enable This write-once bit enables the watchdog to operate when the chip is in stop mode. 0 Watchdog disabled in chip stop mode. 1 Watchdog enabled in chip stop mode.

30.3.2 Watchdog Counter Register (WDOG_CNT)

This section describes the watchdog counter register.

The watchdog counter register provides access to the value of the free-running watchdog counter. Software can read the counter register at any time.

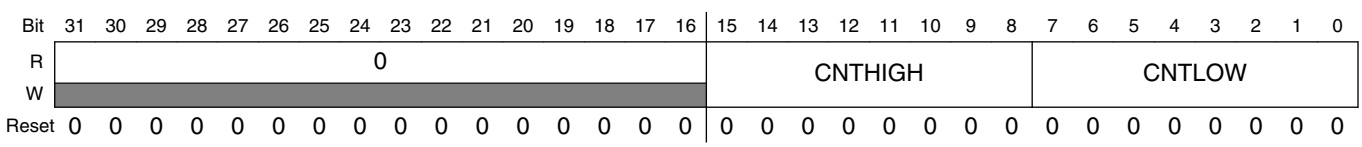
Software cannot write directly to the watchdog counter; however, two write sequences to these registers have special functions:

1. The *refresh sequence* resets the watchdog counter to 0x0000. See the "Refreshing the Watchdog" section.
2. The *unlock sequence* allows the watchdog to be reconfigured without forcing a reset (when CS[UPDATE] = 1). See the "Configure for reconfigurable" section.

NOTE

All other writes to this register are illegal and force a reset.

Address: 4005_2000h base + 4h offset = 4005_2004h



WDOG_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 CNTHIGH	High byte of the Watchdog Counter
CNTLOW	Low byte of the Watchdog Counter

30.3.3 Watchdog Timeout Value Register (WDOG_TOVAL)

This section describes the watchdog timeout value register. TOVAL contains the 16-bit value used to set the timeout period of the watchdog.

The watchdog counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, the watchdog forces a reset triggering event.

NOTE

Do not write 0 to the Watchdog Timeout Value Register; otherwise, the watchdog always generates a reset.

Address: 4005_2000h base + 8h offset = 4005_2008h

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

WDOG_TOVAL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 TOVALHIGH	High byte of the timeout value
TOVALLOW	Low byte of the timeout value

30.3.4 Watchdog Window Register (WDOG_WIN)

This section describes the watchdog window register. When window mode is enabled (CS[WIN] is set), The WIN register determines the earliest time that a refresh sequence is considered valid. See the Watchdog refresh mechanism section.

The WIN register value must be less than the TOVAL register value.

Address: 4005_2000h base + Ch offset = 4005_200Ch

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

WDOG_WIN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 WINHIGH	High byte of Watchdog Window
WINLOW	Low byte of Watchdog Window

30.4 Functional description

The WDOG module provides a fail safe mechanism to ensure the system can be reset to a known state of operation in case of system failure, such as the CPU clock stopping or there being a run away condition in the software code. The watchdog counter runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not, it generates a reset triggering event.

The timeout period, window mode, and clock source are all programmable but must be configured within 128 bus clocks after a reset.

30.4.1 Clock source

The watchdog counter has the following clock source options selected by programming CS[CLK]:

- bus clock
- internal Low-Power Oscillator clock (LPO_CLK) (This is the default source.)
- internal 8 MHz clock (SIRC)
- external clock (SOSC)

The options allow software to select a clock source independent of the bus clock for applications that need to meet more robust safety requirements. Using a clock source other than the bus clock ensures that the watchdog counter continues to run if the bus clock is somehow halted; see Backup reset.

An optional fixed prescaler for all clock sources allows for longer timeout periods. When CS[PRES] is set, the clock source is prescaled by 256 before clocking the watchdog counter.

The following table summarizes the different watchdog timeout periods available.

Table 30-2. Watchdog timeout availability

Reference clock	Prescaler	Watchdog time-out availability
Internal LPO_CLK	Pass through	~1 ms–65.5 s (if LPO_CLK = 1 kHz); (~1 ms–65.5 s)/128 (if LPO_CLK = 128 kHz). [1]
	÷256	~256 ms–16,777.2 s (if LPO_CLK = 1 kHz); ~2 ms–131.1 s (if LPO_CLK = 128 kHz).
Internal 8 MHz (SIRC)	Pass through	125 ns–8.1925 ms
	÷256	32 µs–2.09728 s
1 MHz (from bus or external)	Pass through	1 µs–65.54 ms

Table continues on the next page...

Table 30-2. Watchdog timeout availability (continued)

Reference clock	Prescaler	Watchdog time-out availability
	÷256	256 µs–16.777 s
20 MHz (from bus or external)	Pass through	50 ns–3.277 ms
	÷256	12.8 µs–838.8 ms

1. The default timeout value after reset is approximately 1 s (if LPO_CLK = 1 kHz), or 1/128 s (if LPO_CLK = 128 kHz).

NOTE

When the programmer switches clock sources during reconfiguration, the watchdog hardware holds the counter at zero for 2.5 periods of the previous clock source and 2.5 periods of the new clock source after the configuration time period (128 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

30.4.2 Watchdog refresh mechanism

The watchdog resets the MCU if the watchdog counter is not refreshed. A robust refresh mechanism makes it very unlikely that the watchdog can be refreshed by runaway code.

To refresh the watchdog counter, software must execute a refresh write sequence before the timeout period expires. In addition, if window mode is used, software must not start the refresh sequence until after the time value set in the WIN register. See the following figure.

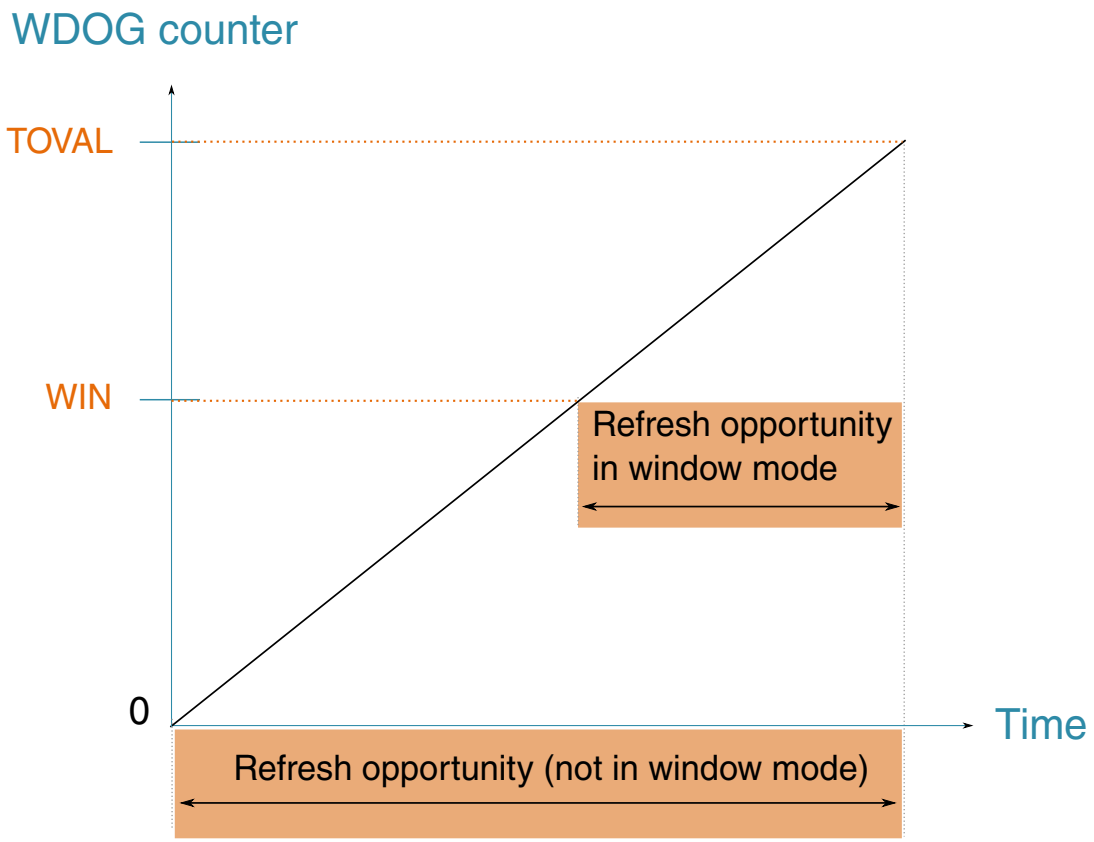


Figure 30-2. Refresh opportunity for the Watchdog counter

30.4.2.1 Window mode

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, the WDOG can be programmed to force a reset when refresh attempts are early.

When Window mode is enabled, the watchdog must be refreshed after the counter has reached a minimum expected time value; otherwise, the watchdog resets the MCU. The minimum expected time value is specified in the WIN register. Setting CS[WIN] enables Window mode.

30.4.2.2 Refreshing the Watchdog

The refresh write sequence can be

- either two 16-bit writes (0xA602, 0xB480) or four 8-bit writes (0xA6, 0x02, 0xB4, 0x80) if WDOG_CS[CMD32EN] is 0;
- one 32-bit write (0xB480_A602) if WDOG_CS[CMD32EN] is 1.

to the CNT register. Both methods must occur before the WDG timeout; otherwise, the watchdog resets the MCU.

Note

Before starting the refresh sequence, disable the global interrupts. Otherwise, an interrupt could effectively invalidate the refresh sequence, if the interrupt occurs before the refresh writes finish. After the sequence finishes, restore the global interrupt control state.

The example codes can be found in the "Application Information" section of this chapter.

30.4.3 Configuring the Watchdog

30.4.3.1 Configuring the Watchdog Once

All watchdog control bits, timeout value, and window value are write-once after reset *within 128 bus clocks*. This means that after a write has occurred they cannot be changed unless a reset occurs. This is guaranteed by the user configuring the window and timeout value first, followed by the other control bits, and ensuring that CS[UPDATE] is also set to 0.

This provides a robust mechanism to configure the watchdog and ensure that a runaway condition cannot mistakenly disable or modify the watchdog configuration after configured.

The new configuration takes effect only after all registers except CNT are written after reset. Otherwise, the WDOG uses the reset values by default. If window mode is not used (CS[WIN] is 0), writing to WIN is not required to make the new configuration take effect.

30.4.3.2 Reconfiguring the Watchdog

In some cases (like when supporting a bootloader function), you may want to reconfigure or disable the watchdog, *without forcing a reset first*.

- By setting CS[UPDATE] to 1 on the initial configuration of the watchdog after a reset, you can reconfigure the watchdog at any time by executing an unlock sequence.
- Conversely, if CS[UPDATE] remains 0, the only way to reconfigure the watchdog is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

30.4.3.2.1 Unlocking the Watchdog

The unlock sequence is a write to the CNT register of 0xC520 followed by 0xD928 within 16 bus clocks at any time after the watchdog has been configured. On completing the unlock sequence, the user must reconfigure the watchdog within 128 bus clocks; otherwise, the watchdog closes the unlock window.

NOTE

Due to the 128 bus clocks requirement for reconfiguring the watchdog, some delays must be inserted before executing STOP or WAIT instructions after reconfiguring the watchdog. This ensures that the watchdog's new configuration takes effect before the MCU enters low power mode. Otherwise, the MCU may not be waken up from low power mode.

The example codes can be found at end of this chapter.

30.4.4 Using interrupts to delay resets

- **When interrupts are enabled (CS[INT] = 1):** After a reset-triggering event (like a counter timeout or invalid refresh attempt), the watchdog first generates an interrupt request. Next, the watchdog delays 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event) before forcing a reset, to allow the interrupt service routine (ISR) to perform tasks (like analyzing the stack to debug code).
- **When interrupts are disabled (CS[INT] = 0):** the watchdog does not delay the forcing a reset.

30.4.5 Backup reset

NOTE

A clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the backup reset function is not available.

The backup reset function is a safeguard feature that independently generates a reset in case the main WDOG logic loses its clock (the bus clock) and can no longer monitor the counter. If the watchdog counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

30.4.6 Functionality in debug and low-power modes

By default, the watchdog is not functional in Debug mode, Wait mode, or Stop mode. However, the watchdog can remain functional in these modes as follows:

- **For Debug mode,** set CS[DBG]. (This way the watchdog is functional in Debug mode even when the CPU is held by the Debug module.)
- **For Wait mode,** set CS[WAIT].
- **For Stop mode,** set CS[STOP], CS[WAIT], and ensure the clock source is active in STOP mode.

NOTE

For Debug mode and Stop mode, in addition to the above configurations, a clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the watchdog cannot function.

30.4.7 Fast testing of the watchdog

Before executing application code in safety critical applications, users are required to test that the watchdog works as expected and resets the MCU. Testing every bit of a 16-bit counter by letting it run to the overflow value takes a relatively long time (64 kHz clocks).

To help minimize the startup delay for application code after reset, the watchdog has a feature to test the watchdog more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for timeout against the corresponding byte of the timeout value register. (For complete coverage when testing the high byte of the counter, the test feature feeds the input clock via the 8th bit of the low byte, thus ensuring that the overflow connection from the low byte to the high byte is tested.)

Using this test feature reduces the test time to 512 clocks (not including overhead, such as user configuration and reset vector fetches). To further speed testing, use a faster clock (such as the bus clock) for the counter reference.

On a power-on reset, the POR bit in the system reset register is set, indicating the user should perform the WDOG fast test.

30.4.7.1 Testing each byte of the counter

The test procedure follows these steps:

1. Program the preferred watchdog timeout value in the TOVAL register during the watchdog configuration time period.
2. Select a byte of the counter to test using the CS[TST] = 10b for the low byte; CS[TST] = 11b for the high byte.
3. Wait for the watchdog to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because the RAM is not affected by a watchdog reset, the timeout period of the watchdog counter can be compared with the software counter to verify the timeout period has occurred as expected.
4. The watchdog counter times out and forces a reset.
5. Confirm the WDOG flag in the system reset register is set, indicating that the watchdog caused the reset. (The POR flag remains clear.)
6. Confirm that CS[TST] shows a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, selecting the other byte in step 2.

NOTE

CS[TST] is cleared by a POR only and not affected by other resets.

30.4.7.2 Entering user mode

After successfully testing the low and high bytes of the watchdog counter, the user can configure CS[TST] to 01b to indicate the watchdog is ready for use in application user mode. Thus if a reset occurs again, software can recognize the reset trigger as a real watchdog reset caused by runaway or faulty application code.

As an ongoing test when using the default LPO clock source, software can periodically read the CNT register to ensure the counter is being incremented.

30.5 Application Information

The watchdog is enabled by default after reset. To disable or reconfigure the watchdog, it is better to be done before the first watchdog timeout. It is suggested to disable or reconfigure the watchdog at the very beginning of the software code, e.g. beginning of the startup or main function.

NOTE

When the watchdog is configured by user, it needs at least 2.5 periods of watchdog clock to take effect. This means interval between two configures by user must be larger than 2.5 clocks.

NOTE

When Chip startup from BOOT ROM then jump to flash, the watchdog would be enabled in the beginning of bootloader, and disabled when bootloader exits. If there is any code in the flash program want to reconfigure the watchdog, it must be run 2.5 watchdog clocks later after the bootloader exits.

To disable or reconfigure the watchdog without forcing a reset, the unlock sequence must be done.

30.5.1 Disable Watchdog

To disable the watchdog, first do unlock sequence, then unset the WDOG_CS[EN] bit.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; // unlock watchdog
WDOG_CS &= ~WDOG_CS_EN_MASK; // disable watchdog
EnableInterrupts; // enable global interrupt
```

30.5.2 Configure Watchdog

The watchdog can be configured once by set the WDOG_CS[UPDATE]=0. After that, the watchdog cannot be reconfigured until a reset.If set WDOG_CS[UPDATE]=1 when configuring the watchdog, the watchdog can be reconfigured without forcing a reset.The following example code shows how to configure the watchdog without window mode, clock source as LPO, interrupt enabled and timeout value to 256 clocks.

Configure once

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(0);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

Configure for reconfigurable

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
```

```
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(1);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

30.5.3 Refreshing the Watchdog

To refresh the watchdog and reset the watchdog counter to zero, a refresh sequence is required:

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xB480A602; // refresh watchdog
EnableInterrupts; // enable global interrupt
```

Chapter 31
Cyclic Redundancy Check (CRC)

31.1 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.

31.1.1 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

31.1.2 Block diagram

The following is a block diagram of the CRC.

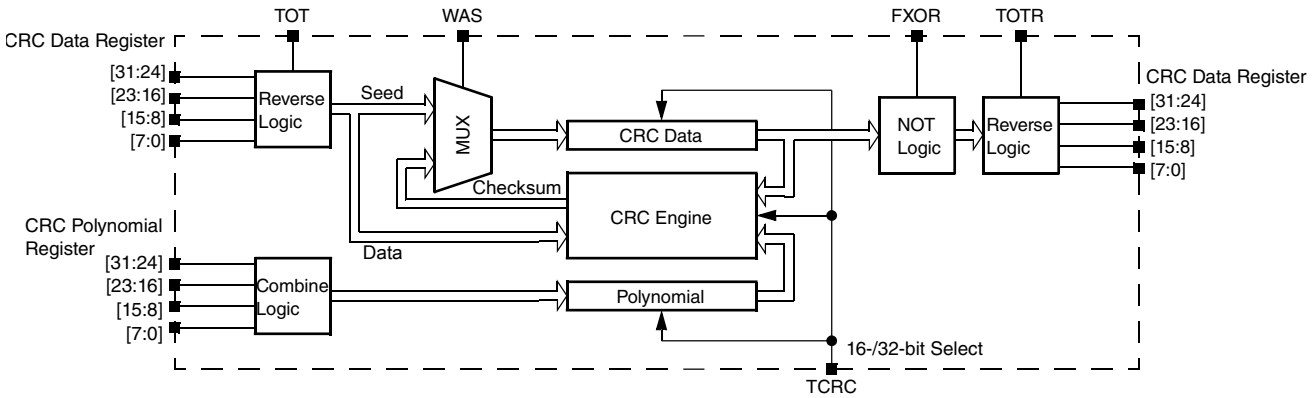


Figure 31-1. Programmable cyclic redundancy check (CRC) block diagram

31.1.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

31.1.3.1 Run mode

This is the basic mode of operation.

31.1.3.2 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.

31.2 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_2000	CRC Data register (CRC_DATA)	32	R/W	FFFF_FFFFh	31.2.1/727
4003_2004	CRC Polynomial register (CRC_GPOLY)	32	R/W	0000_1021h	31.2.2/728
4003_2008	CRC Control register (CRC_CTRL)	32	R/W	0000_0000h	31.2.3/728

31.2.1 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_2000h base + 0h offset = 4003_2000h

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

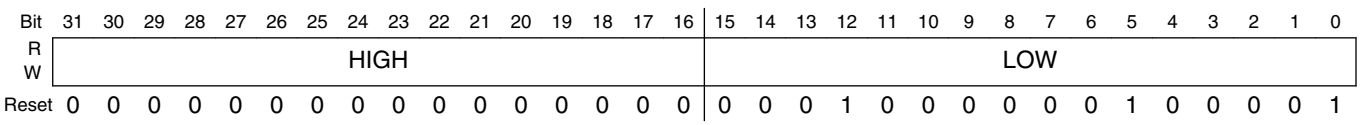
CRC_DATA field descriptions

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.

31.2.2 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_2000h base + 4h offset = 4003_2004h



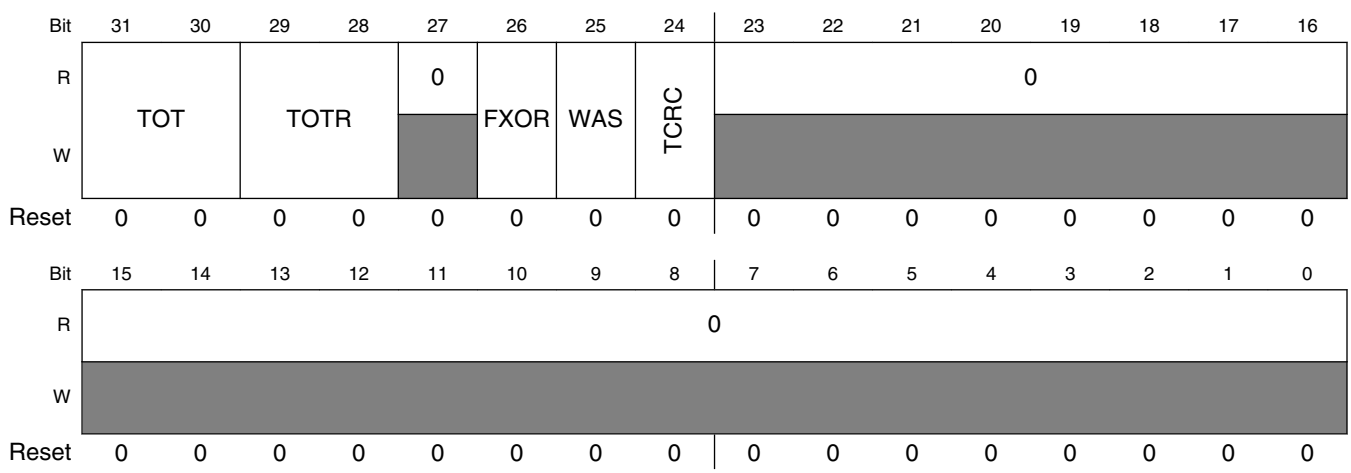
CRC_GPOLY field descriptions

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.

31.2.3 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_2000h base + 8h offset = 4003_2008h



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.

31.3 Functional description

31.3.1 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.

31.3.2 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.

31.3.2.1 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.

31.3.2.2 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.

31.3.3 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.

31.3.3.1 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]}

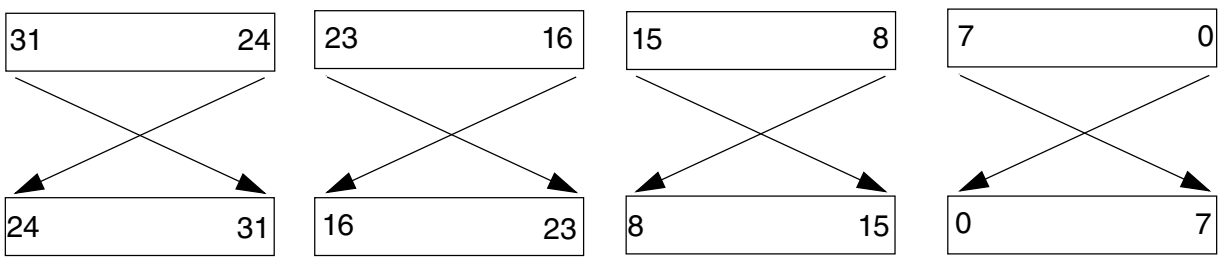


Figure 31-2. Transpose type 01

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]}

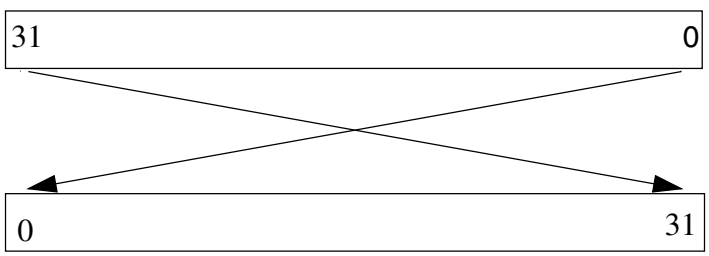


Figure 31-3. Transpose type 10

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]}

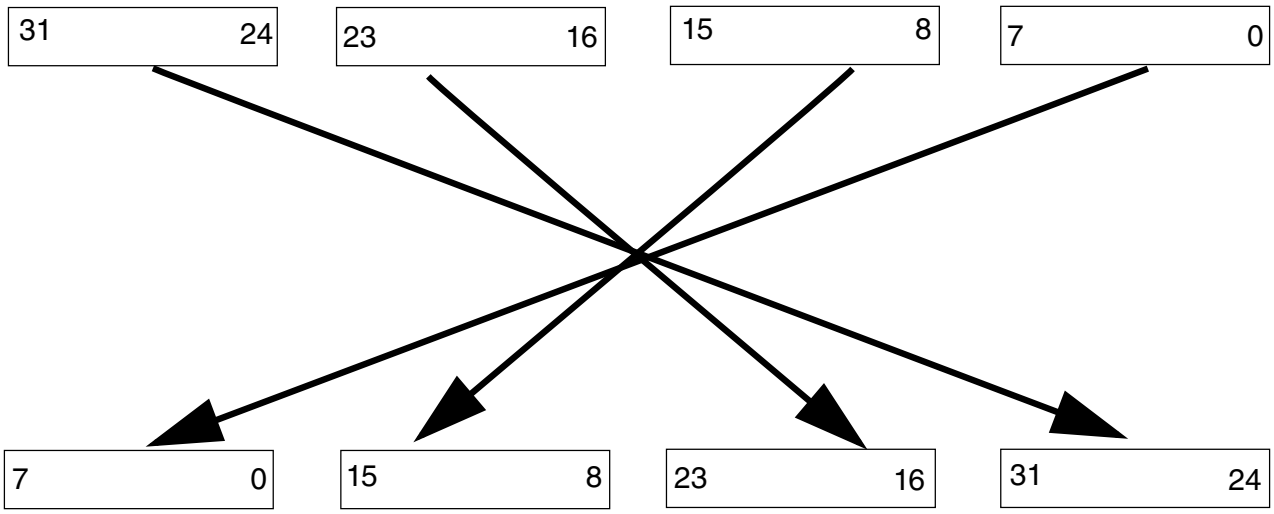


Figure 31-4. Transpose type 11

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.

31.3.4 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.

31.4 Usage Guide

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. The DATA register is written with MSB of data value first, thus the application with little-endian configured, the data write bytes transpose should be enabled when writing a 32bit value from variable to DATA register.

After all data values are written, the CRC result can be read from this data register. For a 16-bit CRC result, if transpose options 10 and 11 is used, the resulting value after transposition resides in the CRC[HU:HL] fields.

This section shows two examples of using CRC module to implement typical CRC algorithms, including both 32-bit and 16-bit algorithms.

31.4.1 32-bit POSIX CRC

CRC-32/POSIX: width=32 poly=0x04c11db7 init=0x00000000 refin=false refout=false xorout=0xffffffff check=0x765e7680

```
uint32_t checksum32, dataSize;
uint8_t data[] = "123456789";
uint32_t *data32;

// Transport Bytes for data write, as the CRC_DATA requires MSB write first
// No transport for checksum read, enable complement read as xorout not zero
CRC_CTRL = CRC_CTRL_TOT(3) | CRC_CTRL_TOTR(0) | CRC_CTRL_FXOR(1) |
           CRC_CTRL_TCRC(1) | CRC_CTRL_WAS(0);
// write polynomial register
CRC_GPOLY = 0x04c11bd7;
// write pre-computed control register value along with WAS to start checksum computation
CRC_CTRL |= CRC_CTRL_WAS(1);
// write seed (initial checksum)
CRC_DATA = 0;
// deassert WAS by writing pre-computed CRC control register value
CRC_CTRL &= ~CRC_CTRL_WAS(1);

// write data
dataSize = sizeof(data);
// 8-bit reads and writes till source address is aligned 4 bytes */
while ((data) && ((uint32_t)data & 3U))
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// use 32-bit reads and writes as long as possible
data32 = (uint32_t *)data;
while (dataSize >= sizeof(uint32_t))
{
    CRC_DATA = *data32;
    data32++;
    dataSize -= sizeof(uint32_t);
}

data = (uint8_t *)data32;

// 8-bit reads and writes till end of data buffer
while (dataSize)
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// read 32bit checksum result
checksum32 = CRC_DATA;
```

31.4.2 16-bit KERMIT CRC

CRC-16/KERMIT: width=16 poly=0x1021 init=0x0000 refin=true refout=true xorout=0x0000 check=0x2189

```
uint32_t checksum16, dataSize;
uint8_t data[] = "123456789";
uint32_t *data32;

// Transport Bytes and Bits for both data write and read
// Bytes transport is because of the CRC_DATA requires MSB write first
// Bits transport is because of the KERMIT algorithm requirement
// No complement for checksum result
CRC_CTRL = CRC_CTRL_TOT(2) | CRC_CTRL_TOTR(2) | CRC_CTRL_FXOR(0) |
           CRC_CTRL_TCRC(0) | CRC_CTRL_WAS(0);
// write polynomial register
CRC_GPOLY = 0x1021;
// write pre-computed control register value along with WAS to start checksum computation
CRC_CTRL |= CRC_CTRL_WAS(1);
// write seed (initial checksum)
CRC_DATA = 0;
// deassert WAS by writing pre-computed CRC control register value
CRC_CTRL &= ~CRC_CTRL_WAS(1);

// write data
dataSize = sizeof(data);
// 8-bit reads and writes till source address is aligned 4 bytes */
while ((data) && ((uint32_t)data & 3U))
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// use 32-bit reads and writes as long as possible
data32 = (uint32_t *)data;
while (dataSize >= sizeof(uint32_t))
{
    CRC_DATA = *data32;
    data32++;
    dataSize -= sizeof(uint32_t);
}

data = (uint8_t *)data32;

// 8-bit reads and writes till end of data buffer
while (dataSize)
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// due to the transport option TOTR >= 2
// read 16bit checksum result from CRC_DATA[HU:HL]
// otherwise, read checksum from CRC_DATA[LU:LL]
checksum16 = (CRC_DATA & 0xFFFF0000) >> 16;
```

Chapter 32
Debug

32.1 Introduction

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

Several debug interfaces are supported:

- IEEE 1149.1 JTAG
- Serial Wire Debug (SWD)
- ARM Real-Time Trace Interface(1-pin asynchronous mode only)

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

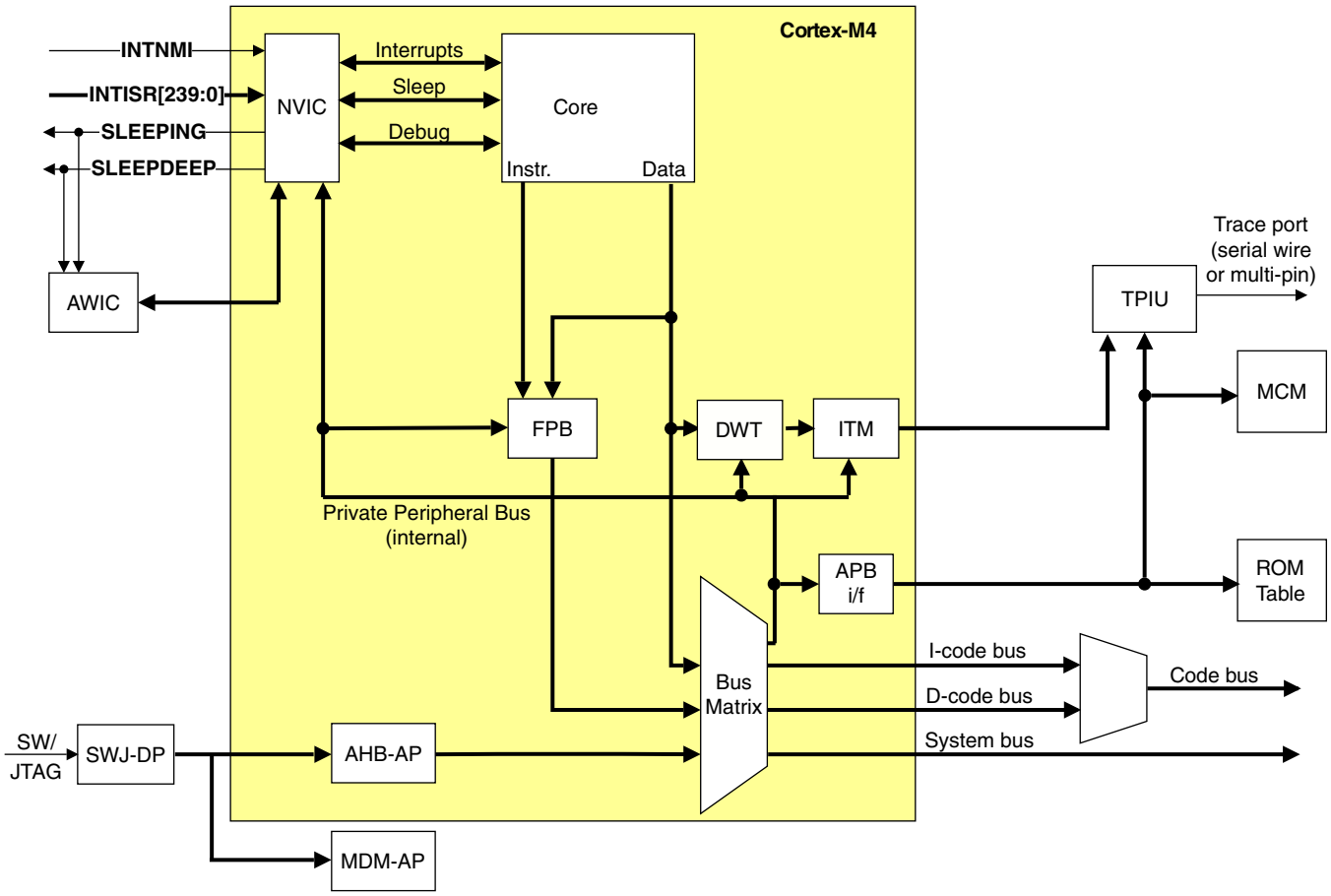


Figure 32-1. Cortex-M4 Debug Topology

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

Table 32-1. Debug Components Description

Module	Description
SWJ-DP	Modified Debug Port with support for SWD, JTAG
AHB-AP	AHB Master Interface from JTAG to debug module and SOC system memory maps
MDM-AP	Provides centralized control and status registers for an external debugger to control the device.
ROM Table	Identifies which debug IP is available.
Core Debug	Singlestep, Register Access, Run, Core Status
ITM	S/W Instrumentation Messaging + Simple Data Trace Messaging + Watchpoint Messaging
DWT (Data and Address Watchpoints)	4 data and address watchpoints
FPB (Flash Patch and Breakpoints)	The FPB implements hardware breakpoints and patches code and data from code space to system space. The FPB unit contains two literal comparators for matching against literal loads from Code space, and remapping to a corresponding area in System space.

Table continues on the next page...

Table 32-1. Debug Components Description (continued)

Module	Description
	The FPB also contains six instruction comparators for matching against instruction fetches from Code space, and remapping to a corresponding area in System space. Alternatively, the six instruction comparators can individually configure the comparators to return a Breakpoint Instruction (BKPT) to the processor core on a match, so providing hardware breakpoint capability.
TPIU (Trace Port Inteface Unit)	Asynchronous Mode (1-pin) = TRACE_SWO (available on JTAG_TDO)

32.1.1 References

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

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

32.2 CM4 ROM table

The ROM table is used to hold the information about the debug components.

The CM4 ROM table resides on the CM4 AHB AP and has entries for CM4 debug components.

Table 32-2. Bit assignments in the ROM table

Bits	Name	Description
[31:12]	Address offset	Base address of the component, relative to the ROM address. Negative values are permitted using two's complement. ComponentAddress = ROMAddress + (AddressOffset SHL 12).
[11:2]	-	Reserved SBZ.
[1]	Format	1 = 32-bit format. In the DAP Debug ROM this is set to 1. 0 = 8-bit format.
[0]	Entry present	Set HIGH to indicate an entry is present.

Table 32-3. CM4 ROM table

Component	Address	Value	Notes
NVIC	0xE00FF000	0xFFF0F003	Base address = Base address of ROM table + 0xFFF0F000

Table continues on the next page...

Table 32-3. CM4 ROM table (continued)

Component	Address	Value	Notes
DWT	0xE00FF004	0xFFF02003	Base address = Base address of ROM table + 0xFFF02000
FPB	0xE00FF008	0xFFF03003	Base address = Base address of ROM table + 0xFFF03000
ITM	0xE00FF00C	0xFFF01003	Base address = Base address of ROM table + 0xFFF01000
TPIU	0xE00FF010	0xFFF41003	Base address = Base address of ROM table + 0xFFF41000

32.3 The Debug Port

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

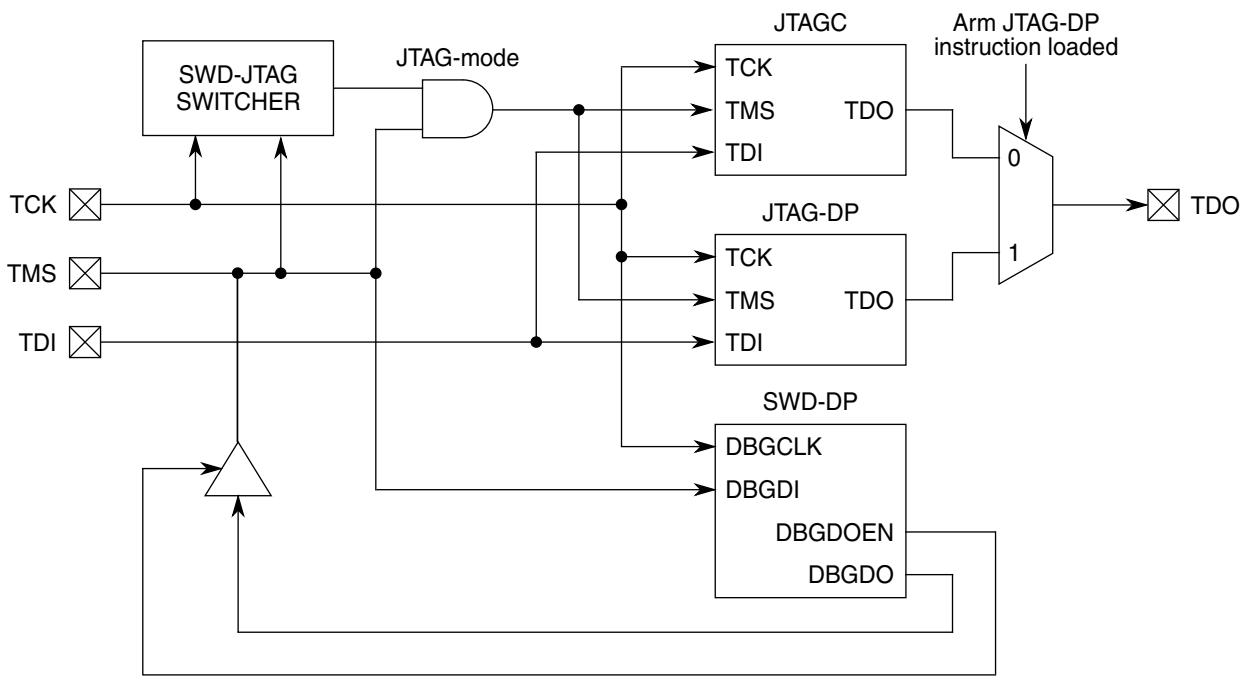


Figure 32-2. Modified Debug Port

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

32.3.1 JTAG-to-SWD change sequence

1. Send more than 50 TCK cycles with TMS (SWDIO) =1
2. Send the 16-bit sequence on TMS (SWDIO) = 0111_1001_1110_0111 (MSB transmitted first)

3. Send more than 50 TCK cycles with TMS (SWDIO) =1

NOTE

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

32.4 Debug Port Pin Descriptions

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

Table 32-4. Debug port pins

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

32.5 System TAP connection

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

32.5.1 IR Codes

Table 32-5. JTAG Instructions

Instruction	Code[3:0]	Instruction Summary
IDCODE	0000	Selects device identification register for shift
SAMPLE/PRELOAD	0010	Selects boundary scan register for shifting, sampling, and preloading without disturbing functional operation
SAMPLE	0011	Selects boundary scan register for shifting and sampling without disturbing functional operation
EXTEST	0100	Selects boundary scan register while applying preloaded values to output pins and asserting functional reset
HIGHZ	1001	Selects bypass register while three-stating all output pins and asserting functional reset
CLAMP	1100	Selects bypass register while applying preloaded values to output pins and asserting functional reset
ARM_IDCODE	1110	ARM JTAG-DP Instruction
BYPASS	1111	Selects bypass register for data operations
Factory debug reserved	0101, 0110, 0111, 1101	Intended for factory debug only
ARM JTAG-DP Reserved	1000, 1010, 1011, 1110	These instructions will go the ARM JTAG-DP controller. Please look at ARM JTAG-DP documentation for more information on these instructions.
Reserved [1]	All other opcodes	Decoded to select bypass register

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

32.6 JTAG status and control registers

Through the ARM Debug Access Port (DAP), the debugger has access to the status and control elements, implemented as registers on the DAP bus as shown in Figure 32-3. These registers provide additional control and status for low power mode recovery and typical run-control scenarios. The status register bits also provide a means for the debugger to get updated status of the core without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

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

Table 32-6. MDM-AP Register Summary

Address	Register	Description
0x0100_0000	Status	See MDM-AP Status Register

Table continues on the next page...

Table 32-6. MDM-AP Register Summary (continued)

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

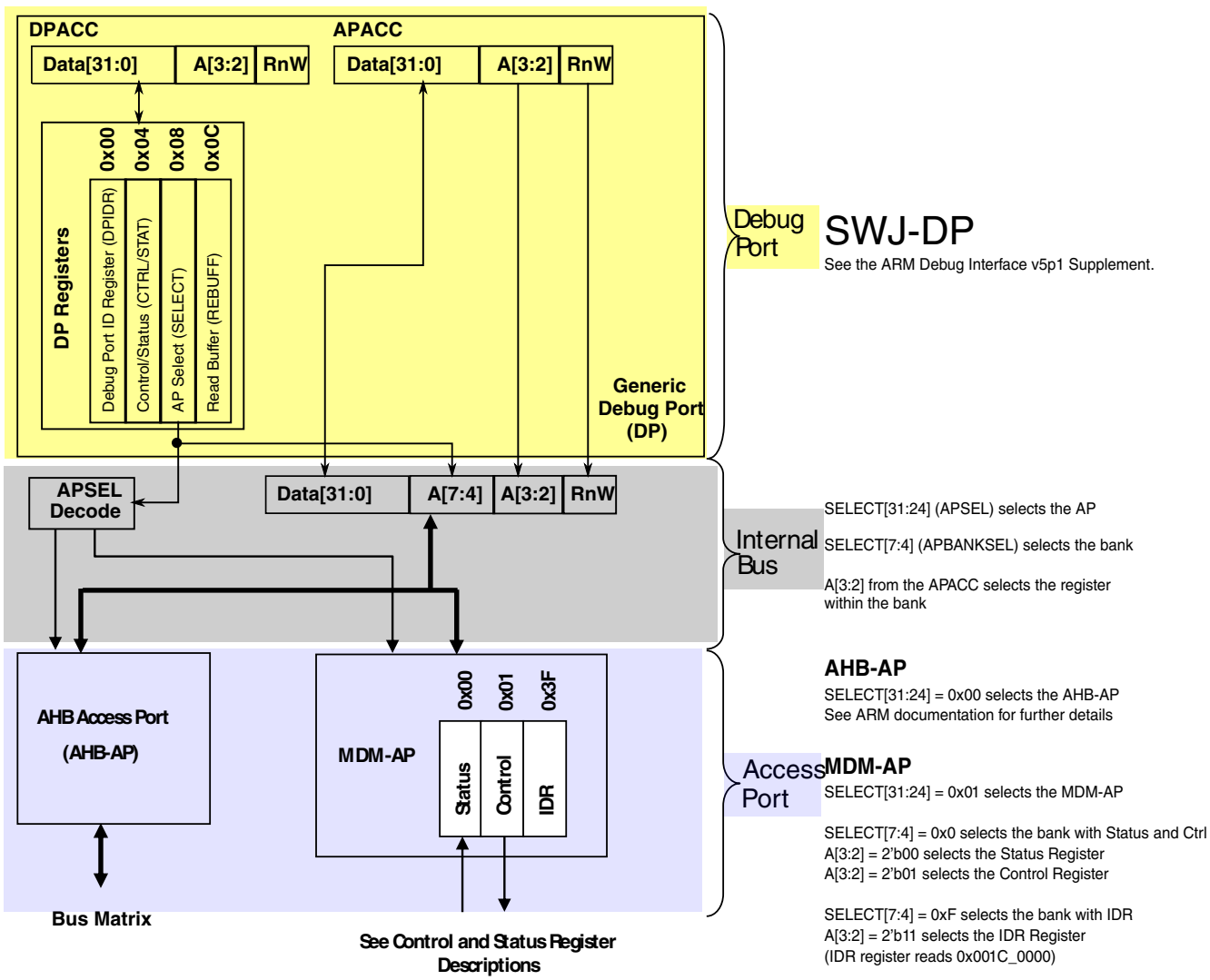


Figure 32-3. MDM AP Addressing

32.6.1 MDM-AP Control Register

Table 32-7. 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.

Table continues on the next page...

Table 32-7. MDM-AP Control register assignments (continued)

Bit	Name	Secure[1]	Description
			When mass erase is disabled (via MEEN and SEC settings), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset.
1	Debug Disable	N	Set to disable debug. Clear to allow debug operation. When set it overrides the C_DEBUGEN bit within the DHCSR and force disables Debug logic.
2	Debug Request	N	Set to force the Core to halt. If the Core is in a stop or wait mode, this bit can be used to wakeup the core and transition to a halted state.
3	System Reset Request	N	Set to force a system reset. The system remains held in reset until this bit is cleared.
4	Core Hold Reset	N	Configuration bit to control Core operation at the end of system reset sequencing. 0 Normal operation - release the Core from reset along with the rest of the system at the end of system reset sequencing. 1 Suspend operation - hold the Core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the Core from reset and CPU operation begins.
5 – 7	Reserved	N	
8	Timestamp Disable	N	Set this bit to disable the 48-bit global trace timestamp counter during debug halt mode when the core is halted. 0 The timestamp counter continues to count assuming trace is enabled. (default) 1 The timestamp counter freezes when the core has halted (debug halt mode).
9 – 31	Reserved for future use	N	

1. Command available in secure mode

32.6.2 MDM-AP Status Register

Table 32-8. MDM-AP Status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge bit is cleared after any system reset. The bit is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress bit in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation. When mass erase is disabled (via MEEN and SEC settings), an erase request due to seting of Flash Mass Erase in Progress bit is not acknowledged.
1	Flash Ready	Indicate Flash has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger.

Table continues on the next page...

Table 32-8. MDM-AP Status register assignments (continued)

Bit	Name	Description
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
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 is the selected power mode the next time the ARM Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled 1 Low Power Stop Mode is enabled Usage intended for debug operation in which Run to VLPS is attempted. Per debug definition, the system actually enters the Stop state. A debugger should interpret deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjuntion with this bit asserted as the debugger-VLPS status indication.
8	Very Low Power Mode	Indicates current power mode is VLPx. This bit is not ‘sticky’ and should always represent whether VLPx is enabled or not. This bit is used to throttle JTAG TCK frequency up/down.
9 – 10	Reserved	Always read 0.
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 indicates wait or VLPW mode. SLEEPDEEP==1 indicates stop or VLPS mode.
19 – 31	Reserved for future use	Always read 0.

32.7 Debug Resets

The debug system receives the following sources of reset:

- JTAG_TRST_b from an external signal. This signal is optional and may not be available in all packages.

- Debug reset (CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register) in the TCLK domain that allows the debugger to reset the debug logic.
- 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.

32.8 AHB-AP

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

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

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

32.9 ITM

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

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

32.10 Core Trace Connectivity

The ITM can route its data to the TPIU. (See the MCM (Miscellaneous Control Module) for controlling the routing to the TPIU.) This configuration enables the use of trace with low cost tools while maintaining the compatibility with trace probes.

32.11 TPIU

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

32.12 DWT

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

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

NOTE

An event is emitted each time a counter overflows.

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

32.13 Debug in Low Power Modes

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

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

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

32.13.1 Debug Module State in Low Power Modes

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

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

Table 32-9. Debug Module State in Low Power Modes

Module	STOP	VLPR	VLPW	VLPS
Debug Port	FF	FF	FF	OFF
AHB-AP	FF	FF	FF	OFF
ITM	FF	FF	FF	OFF
TPIU	FF	FF	FF	OFF
DWT	FF	FF	FF	OFF

32.14 Debug and Security

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

Chapter 33
JTAG Controller (JTAGC)

33.1 Introduction

The JTAGC block provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format.

33.1.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. Refer to Register description for more information about the JTAGC registers.

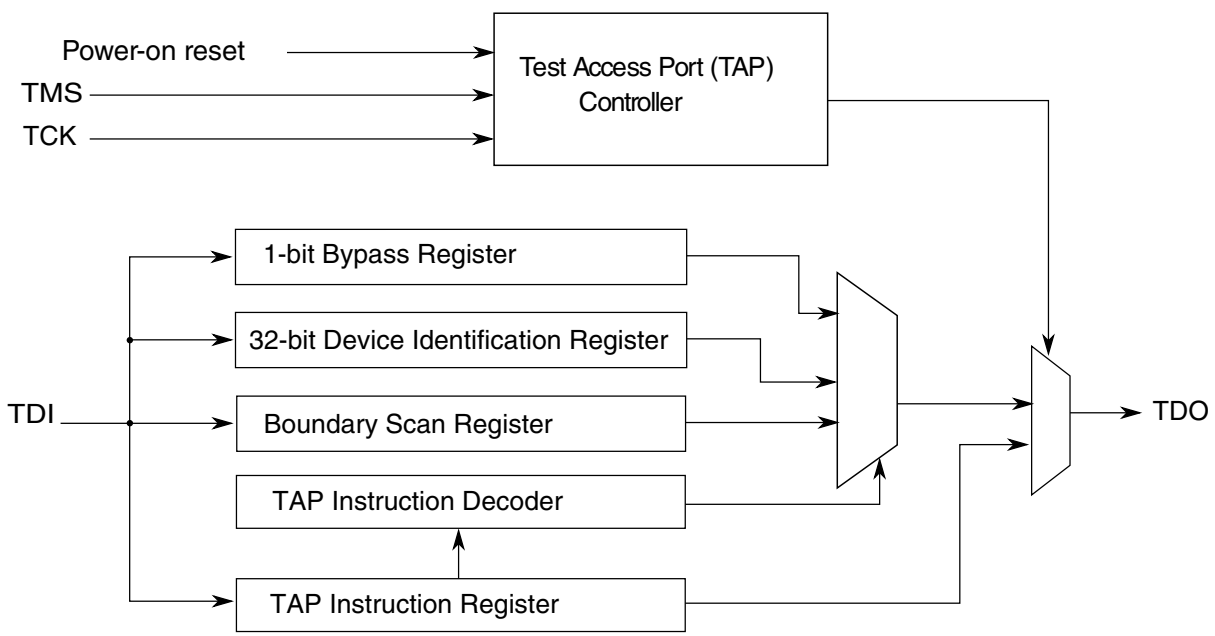


Figure 33-1. JTAG (IEEE 1149.1) block diagram

33.1.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 Test Access Port (TAP) interface
  - 4 pins (TDI, TMS, TCK, and TDO)
- Instruction register that supports several IEEE 1149.1-2001 defined instructions as well as several public and private device-specific instructions. Refer to Table 33-3 for a list of supported instructions.
- Bypass register, boundary scan register, and device identification register.
- TAP controller state machine that controls the operation of the data registers, instruction register and associated circuitry.

33.1.3 Modes of operation

The JTAGC block uses a power-on reset indication as its primary reset signals. Several IEEE 1149.1-2001 defined test modes are supported, as well as a bypass mode.

33.1.3.1 Reset

The JTAGC block is placed in reset when either power-on reset is asserted, or the TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state. Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset results in asynchronous entry into the reset state. While in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered
- The instruction register is loaded with the IDCODE instruction

33.1.3.2 IEEE 1149.1-2001 defined test modes

The JTAGC block supports several IEEE 1149.1-2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register while the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, SAMPLE and SAMPLE/PRELOAD. Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic while the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, SAMPLE or SAMPLE/PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the BYPASS, HIGHZ, CLAMP or reserved instructions are active. The functionality of each test mode is explained in more detail in JTAGC block instructions.

33.1.3.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. While in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

33.2 External signal description

The JTAGC consists of a set of signals that connect to off chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 33-1. JTAG signal properties

Name	I/O	Function	Reset State
TCK	Input	Test Clock	Weak pulldown
TDI	Input	Test Data In	Weak pullup
TDO	Output	Test Data Out	High Z[1]
TMS	Input	Test Mode Select	Weak pullup

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

33.2.1 TCK—Test clock input

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

33.2.2 TDI—Test data input

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

33.2.3 TDO—Test data output

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is three-stateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

33.2.4 TMS—Test mode select

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

33.3 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

33.3.1 Instruction register

The JTAGC block uses a 4-bit instruction register as shown in the following figure. The instruction register allows instructions to be loaded into the block to select the test to be performed or the test data register to be accessed or both. Instructions are shifted in through TDI while the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states. Synchronous entry into the

Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the Capture-IR TAP controller state, the instruction shift register is loaded with the value 0001b , making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

	3	2	1	0
R	0	0	0	1
W	Instruction Code			
Reset:	0	0	0	1

Figure 33-2. Instruction register

33.3.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the BYPASS, CLAMP, HIGHZ or reserve instructions are active. After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

33.3.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP. The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state while the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Part Revision Number				Design Center						Part Identification Number					
W																
Reset	PRN				DC						PIN					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Part Identification Number				Manufacturer Identity Code											1
W																
Reset	PIN (contd.)				MIC											1

The following table describes the device identification register functions.

Table 33-2. Device identification register field descriptions

Field	Description
PRN	Part Revision Number. Contains the revision number of the part. Value is On this device, the PIN mirrors bits 9-0 of the SIM_SDID[REVID] field. Please see the SIM_SDID register description for more detail.
DC	Design Center. Indicates the design center. Value is 0x2C.
PIN	Part Identification Number. Contains the part number of the device. On this device, the PIN mirrors bits 9-0 of the SIM_SDID register. Please see the SIM_SDID register description for more detail.
MIC	Manufacturer Identity Code. Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID. Value is 0x00E.
IDCODE ID	IDCODE Register ID. Identifies this register as the device identification register and not the bypass register. Always set to 1.

33.3.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, SAMPLE or SAMPLE/PRELOAD instructions are active. It is used to capture input pin data, force fixed values on output pins, and select a logic value and direction for bidirectional pins. Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

33.4 Functional description

This section explains the JTAGC functional description.

33.4.1 JTAGC reset configuration

While in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

33.4.2 IEEE 1149.1-2001 (JTAG) Test Access Port

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction.

Data is shifted between TDI and TDO though the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

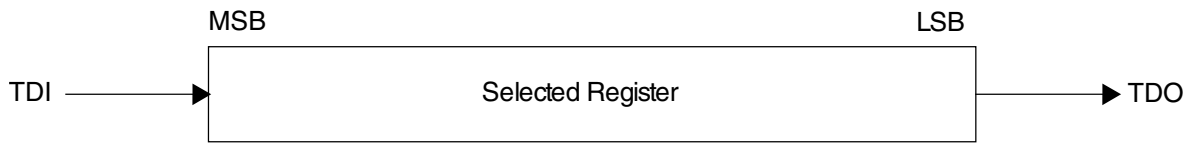
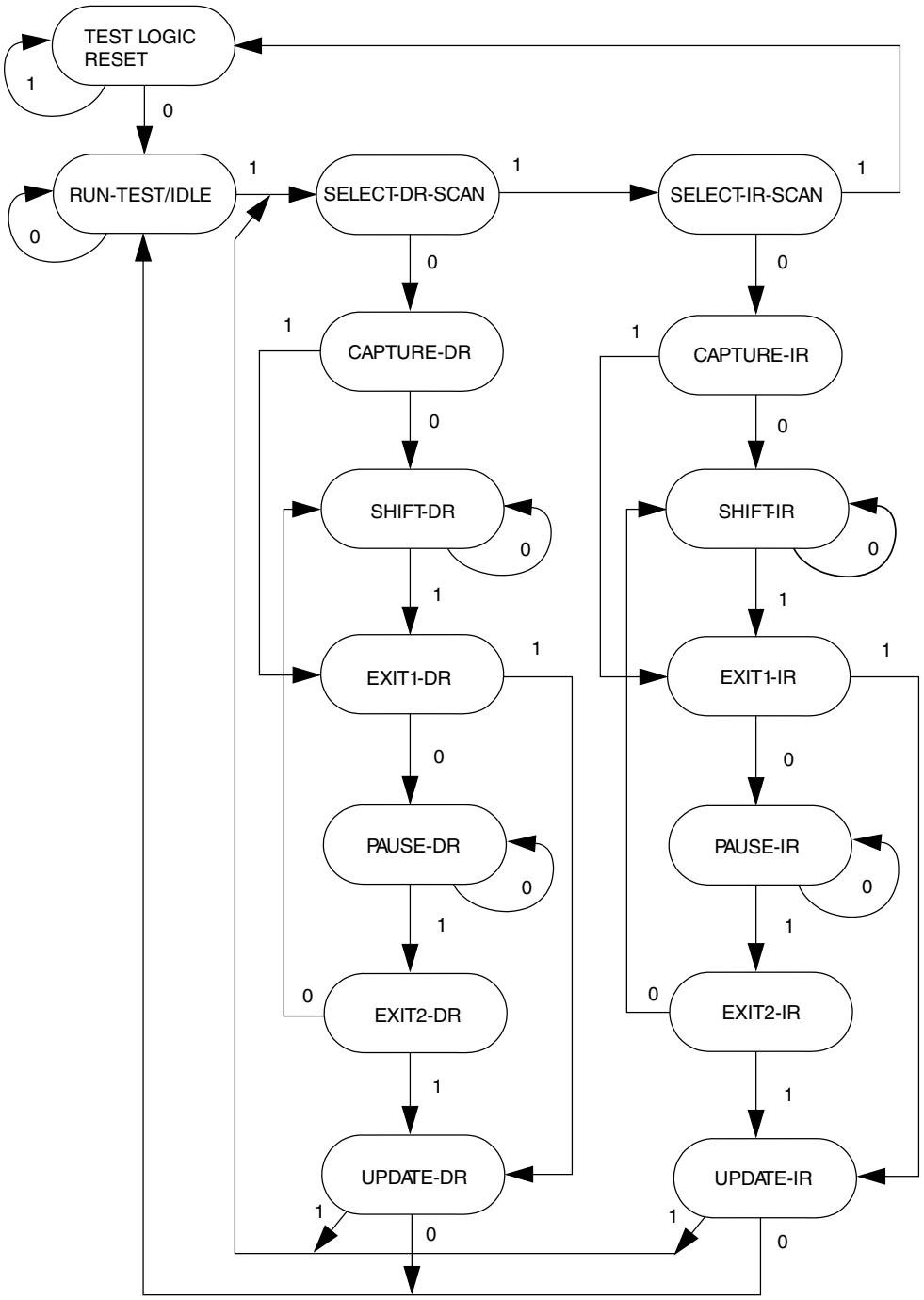


Figure 33-3. Shifting data through a register

33.4.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin. The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the following figure shows, holding TMS at logic 1 while clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 33-4. IEEE 1149.1-2001 TAP controller finite state machine

33.4.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting the JTAGC enable to a logic 1 value.

33.4.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions while the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated once the required number of bits have been acquired.

33.4.4 JTAGC block instructions

The JTAGC block implements the IEEE 1149.1-2001 defined instructions listed in the following table. This section gives an overview of each instruction; refer to the IEEE 1149.1-2001 standard for more details. All undefined opcodes are reserved.

Table 33-3. 4-bit JTAG instructions

Instruction	Code[3:0]	Instruction summary
IDCODE	0000	Selects device identification register for shift
SAMPLE/PRELOAD	0010	Selects boundary scan register for shifting, sampling, and preloading without disturbing functional operation
SAMPLE	0011	Selects boundary scan register for shifting and sampling without disturbing functional operation
EXTEST	0100	Selects boundary scan register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
Factory debug reserved	0101	Intended for factory debug only
Factory debug reserved	0110	Intended for factory debug only
Factory debug reserved	0111	Intended for factory debug only
Arm JTAG-DP Reserved	1000	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
HIGHZ	1001	Selects bypass register and three-states all output pins. **NOTE:** Execution of this instruction asserts functional reset.
Arm JTAG-DP Reserved	1010	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.

Table continues on the next page...

Table 33-3. 4-bit JTAG instructions (continued)

Instruction	Code[3:0]	Instruction summary
Arm JTAG-DP Reserved	1011	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
CLAMP	1100	Selects bypass register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
EZPORT	1101	Enables the EZPORT function for the SoC
Arm JTAG-DP Reserved	1110	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
BYPASS	1111	Selects bypass register for data operations

33.4.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

33.4.4.2 SAMPLE/PRELOAD instruction

The SAMPLE/PRELOAD instruction has two functions:

- The SAMPLE portion of the instruction obtains a sample of the system data and control signals present at the MCU input pins and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE/PRELOAD instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. Both the data capture and the shift operation are transparent to system operation.
- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

33.4.4.3 SAMPLE instruction

The SAMPLE instruction obtains a sample of the system data and control signals present at the MCU input pins and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. There is no defined action in the Update-DR state. Both the data capture and the shift operation are transparent to system operation.

33.4.4.4 EXTEST External test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the SAMPLE/PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state while performing external boundary scan operations.

33.4.4.5 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. While HIGHZ is active all output drivers are placed in an inactive drive state (e.g., high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

33.4.4.6 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register while the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) while conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

33.4.4.7 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. While the BYPASS instruction is active the system logic operates normally.

33.4.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

33.5 Initialization/Application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Place the JTAGC in reset through TAP controller state machine transitions controlled by TMS
2. Load the appropriate instruction for the test or action to be performed

Chapter 34
Signal Multiplexing and Pin Assignment

34.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. This chapter illustrates which of the device's signals are multiplexed on which external pin.

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

34.2 Pinouts

34.2.1 KE1xF Signal Multiplexing and Pin Assignments

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

NOTE

On this device, there are several special ADC channels which support hardware interleave between multiple ADCs. Taking ADC0_SE4 and ADC1_SE14 channels as an example, these two channels can work independently, but they can also be hardware interleaved. In the hardware interleaved mode, a signal on the pin PTB0 can be sampled by both ADC0 and ADC1. The interleaved mode is enabled by SIM_CHIPCTL[ADC_INTERLEAVE_EN] bits. For more information, see "ADC Hardware Interleaved Channels" in the ADC chapter of Reference Manual.

100 LQFP	64 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
—	10	VREFL/ VSS	VREFL/ VSS	VREFL/ VSS							
1	—	PTE16	DISABLED		PTE16			FTM2_CH7		FXIO_D3	TRGMUX_ OUT7
2	—	PTE15	DISABLED		PTE15			FTM2_CH6		FXIO_D2	TRGMUX_ OUT6
3	1	PTD1	ADC2_SE1	ADC2_SE1	PTD1	FTM0_CH3	LPSPI1_SIN	FTM2_CH1		FXIO_D1	TRGMUX_ OUT2
4	2	PTD0	ADC2_SE0	ADC2_SE0	PTD0	FTM0_CH2	LPSPI1_SCK	FTM2_CH0		FXIO_D0	TRGMUX_ OUT1
5	3	PTE11	ADC2_SE13	ADC2_SE13	PTE11	PWT_IN1	LPTMR0_ALT1	FTM2_CH5		FXIO_D5	TRGMUX_ OUT5
6	4	PTE10	ADC2_SE12	ADC2_SE12	PTE10	CLKOUT		FTM2_CH4		FXIO_D4	TRGMUX_ OUT4
7	—	PTE13	DISABLED		PTE13			FTM2_FLT0			
8	5	PTE5	DISABLED		PTE5	TCLK2	FTM2_QD_ PHA	FTM2_CH3	CAN0_TX	FXIO_D7	EWM_IN
9	6	PTE4	DISABLED		PTE4	BUSOUT	FTM2_QD_ PHB	FTM2_CH2	CAN0_RX	FXIO_D6	EWM_OUT_b
10	7	VDD	VDD	VDD							
11	8	VDDA	VDDA	VDDA							
12	9	VREFH	VREFH	VREFH							
13	—	VREFL	VREFL	VREFL							
14	—	VSS	VSS	VSS							
15	11	PTB7	EXTAL	EXTAL	PTB7	LPI2C0_SCL					
16	12	PTB6	XTAL	XTAL	PTB6	LPI2C0_SDA					
17	—	PTE14	ACMP2_IN3	ACMP2_IN3	PTE14	FTM0_FLT1		FTM2_FLT1			
18	13	PTE3	DISABLED		PTE3	FTM0_FLT0	LPUART2_ RTS	FTM2_FLT0		TRGMUX_IN6	ACMP2_OUT
19	—	PTE12	DISABLED		PTE12	FTM0_FLT3	LPUART2_TX				
20	—	PTD17	DISABLED		PTD17	FTM0_FLT2	LPUART2_RX				
21	14	PTD16	ACMP2_IN0	ACMP2_IN0	PTD16	FTM0_CH1					
22	15	PTD15	ACMP2_IN1	ACMP2_IN1	PTD15	FTM0_CH0					
23	16	PTE9	ACMP2_IN2/ DAC0_OUT	ACMP2_IN2/ DAC0_OUT	PTE9	FTM0_CH7	LPUART2_ CTS				
24	—	PTD14	DISABLED		PTD14	FTM2_CH5					CLKOUT
25	—	PTD13	DISABLED		PTD13	FTM2_CH4					RTC_CLKOUT
26	17	PTE8	ACMP0_IN3	ACMP0_IN3	PTE8	FTM0_CH6					
27	18	PTB5	DISABLED		PTB5	FTM0_CH5	LPSPI0_PCS1			TRGMUX_IN0	ACMP1_OUT
28	19	PTB4	ACMP1_IN2	ACMP1_IN2	PTB4	FTM0_CH4	LPSPI0_SOUT			TRGMUX_IN1	
29	20	PTC3	ADC0_SE11/ ACMP0_IN4/ EXTAL32	ADC0_SE11/ ACMP0_IN4/ EXTAL32	PTC3	FTM0_CH3	CAN0_TX				

100 LQFP	64 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
30	21	PTC2	ADC0_SE10/ ACMP0_IN5/ XTAL32	ADC0_SE10/ ACMP0_IN5/ XTAL32	PTC2	FTM0_CH2	CAN0_RX				
31	22	PTD7	DISABLED		PTD7	LPUART2_TX		FTM2_FLT3			
32	23	PTD6	DISABLED		PTD6	LPUART2_RX		FTM2_FLT2			
33	24	PTD5	DISABLED		PTD5	FTM2_CH3	LPTMR0_ALT2	FTM2_FLT1	PWT_IN2	TRGMUX_IN7	
34	—	PTD12	DISABLED		PTD12	FTM2_CH2	LPI2C1_HREQ			LPUART2_ RTS	
35	—	PTD11	DISABLED		PTD11	FTM2_CH1	FTM2_QD_ PHA			LPUART2_ CTS	
36	—	PTD10	DISABLED		PTD10	FTM2_CH0	FTM2_QD_ PHB				
37	—	VSS	VSS	VSS							
38	—	VDD	VDD	VDD							
39	25	PTC1	ADC0_SE9/ ACMP1_IN3	ADC0_SE9/ ACMP1_IN3	PTC1	FTM0_CH1				FTM1_CH7	
40	26	PTC0	ADC0_SE8/ ACMP1_IN4	ADC0_SE8/ ACMP1_IN4	PTC0	FTM0_CH0				FTM1_CH6	
41	—	PTD9	ACMP1_IN5	ACMP1_IN5	PTD9	LPI2C1_SCL		FTM2_FLT3		FTM1_CH5	
42	—	PTD8	DISABLED		PTD8	LPI2C1_SDA		FTM2_FLT2		FTM1_CH4	
43	27	PTC17	ADC0_SE15	ADC0_SE15	PTC17	FTM1_FLT3		LPI2C1_SCLS			
44	28	PTC16	ADC0_SE14	ADC0_SE14	PTC16	FTM1_FLT2		LPI2C1_SDAS			
45	29	PTC15	ADC0_SE13/ ACMP2_IN4	ADC0_SE13/ ACMP2_IN4	PTC15	FTM1_CH3					
46	30	PTC14	ADC0_SE12/ ACMP2_IN5	ADC0_SE12/ ACMP2_IN5	PTC14	FTM1_CH2					
47	31	PTB3	ADC0_SE7	ADC0_SE7	PTB3	FTM1_CH1	LPSPI0_SIN	FTM1_QD_ PHA		TRGMUX_IN2	
48	32	PTB2	ADC0_SE6	ADC0_SE6	PTB2	FTM1_CH0	LPSPI0_SCK	FTM1_QD_ PHB		TRGMUX_IN3	
49	—	PTC13	DISABLED		PTC13	FTM3_CH7	FTM2_CH7				
50	—	PTC12	DISABLED		PTC12	FTM3_CH6	FTM2_CH6				
51	—	PTC11	DISABLED		PTC11	FTM3_CH5					
52	—	PTC10	DISABLED		PTC10	FTM3_CH4					
53	33	PTB1	ADC0_SE5	ADC0_SE5	PTB1	LPUART0_TX	LPSPI0_SOUT	TCLK0			
54	34	PTB0	ADC0_SE4	ADC0_SE4	PTB0	LPUART0_RX	LPSPI0_PCS0	LPTMR0_ALT3	PWT_IN3		
55	35	PTC9	ADC2_SE15	ADC2_SE15	PTC9	LPUART1_TX	FTM1_FLT1			LPUART0_ RTS	
56	36	PTC8	ADC2_SE14	ADC2_SE14	PTC8	LPUART1_RX	FTM1_FLT0			LPUART0_ CTS	
57	37	PTA7	ADC0_SE3/ ACMP1_IN1	ADC0_SE3/ ACMP1_IN1	PTA7	FTM0_FLT2		RTC_CLKIN		LPUART1_ RTS	
58	38	PTA6	ADC0_SE2/ ACMP1_IN0	ADC0_SE2/ ACMP1_IN0	PTA6	FTM0_FLT1	LPSPI1_PCS1			LPUART1_ CTS	
59	39	PTE7	ADC2_SE2/ ACMP2_IN6	ADC2_SE2/ ACMP2_IN6	PTE7	FTM0_CH7	FTM3_FLT0				

100 LQFP	64 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
60	40	VSS	VSS	VSS							
61	41	VDD	VDD	VDD							
62	—	PTA17	DISABLED		PTA17	FTM0_CH6	FTM3_FLT0	EWM_OUT_b			
63	—	PTB17	ADC2_SE3	ADC2_SE3	PTB17	FTM0_CH5	LPSPI1_PCS3				
64	—	PTB16	ADC1_SE15	ADC1_SE15	PTB16	FTM0_CH4	LPSPI1_SOUT				
65	—	PTB15	ADC1_SE14	ADC1_SE14	PTB15	FTM0_CH3	LPSPI1_SIN				
66	—	PTB14	ADC1_SE9	ADC1_SE9	PTB14	FTM0_CH2	LPSPI1_SCK				
67	42	PTB13	ADC1_SE8	ADC1_SE8	PTB13	FTM0_CH1	FTM3_FLT1				
68	43	PTB12	ADC1_SE7	ADC1_SE7	PTB12	FTM0_CH0	FTM3_FLT2				
69	44	PTD4	ADC1_SE6/ ACMP1_IN6	ADC1_SE6/ ACMP1_IN6	PTD4	FTM0_FLT3	FTM3_FLT3				
70	45	PTD3	NMI_b	ADC1_SE3	PTD3	FTM3_CH5	LPSPI1_PCS0	FXIO_D5		TRGMUX_IN4	NMI_b
71	46	PTD2	ADC1_SE2	ADC1_SE2	PTD2	FTM3_CH4	LPSPI1_SOUT	FXIO_D4		TRGMUX_IN5	
72	47	PTA3	ADC1_SE1	ADC1_SE1	PTA3	FTM3_CH1	LPI2C0_SCL	EWM_IN		LPUART0_TX	
73	48	PTA2	ADC1_SE0	ADC1_SE0	PTA2	FTM3_CH0	LPI2C0_SDA	EWM_OUT_b		LPUART0_RX	
74	—	PTB11	ADC2_SE8	ADC2_SE8	PTB11	FTM3_CH3	LPI2C0_HREQ				
75	—	PTB10	ADC2_SE9	ADC2_SE9	PTB10	FTM3_CH2	LPI2C0_SDAS				
76	—	PTB9	ADC2_SE10	ADC2_SE10	PTB9	FTM3_CH1	LPI2C0_SCLS				
77	—	PTB8	ADC2_SE11	ADC2_SE11	PTB8	FTM3_CH0					
78	49	PTA1	ADC0_SE1/ ACMP0_IN1	ADC0_SE1/ ACMP0_IN1	PTA1	FTM1_CH1	LPI2C0_SDAS	FXIO_D3	FTM1_QD_ PHA	LPUART0_ RTS	TRGMUX_ OUT0
79	50	PTA0	ADC0_SE0/ ACMP0_IN0	ADC0_SE0/ ACMP0_IN0	PTA0	FTM2_CH1	LPI2C0_SCLS	FXIO_D2	FTM2_QD_ PHA	LPUART0_ CTS	TRGMUX_ OUT3
80	51	PTC7	ADC1_SE5	ADC1_SE5	PTC7	LPUART1_TX	CAN1_TX	FTM3_CH3			
81	52	PTC6	ADC1_SE4	ADC1_SE4	PTC6	LPUART1_RX	CAN1_RX	FTM3_CH2			
82	—	PTA16	ADC1_SE13	ADC1_SE13	PTA16	FTM1_CH3	LPSPI1_PCS2				
83	—	PTA15	ADC1_SE12	ADC1_SE12	PTA15	FTM1_CH2	LPSPI0_PCS3				
84	53	PTE6	ADC1_SE11/ ACMP0_IN6	ADC1_SE11/ ACMP0_IN6	PTE6	LPSPI0_PCS2		FTM3_CH7		LPUART1_ RTS	
85	54	PTE2	ADC1_SE10	ADC1_SE10	PTE2	LPSPI0_SOUT	LPTMR0_ALT3	FTM3_CH6	PWT_IN3	LPUART1_ CTS	
86	—	VSS	VSS	VSS							
87	—	VDD	VDD	VDD							
88	—	PTA14	DISABLED		PTA14	FTM0_FLT0	FTM3_FLT1	EWM_IN		FTM1_FLT0	BUSOUT
89	55	PTA13	ADC2_SE4	ADC2_SE4	PTA13	FTM1_CH7	CAN1_TX	LPI2C1_SCLS			
90	56	PTA12	ADC2_SE5	ADC2_SE5	PTA12	FTM1_CH6	CAN1_RX	LPI2C1_SDAS			
91	57	PTA11	DISABLED		PTA11	FTM1_CH5	LPUART0_RX	FXIO_D1			
92	58	PTA10	JTAG_TDO/ noetm_Trace_ SWO		PTA10	FTM1_CH4	LPUART0_TX	FXIO_D0			JTAG_TDO/ noetm_Trace_ SWO
93	59	PTE1	ADC2_SE6	ADC2_SE6	PTE1	LPSPI0_SIN	LPI2C0_HREQ	LPI2C1_SCL		FTM1_FLT1	
94	60	PTE0	ADC2_SE7	ADC2_SE7	PTE0	LPSPI0_SCK	TCLK1	LPI2C1_SDA		FTM1_FLT2	

100 LQFP	64 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
95	61	PTC5	JTAG_TDI		PTC5	FTM2_CH0	RTC_CLKOUT	LPI2C1_HREQ		FTM2_QD_PHB	JTAG_TDI
96	62	PTC4	JTAG_TCLK/SWD_CLK	ACMP0_IN2	PTC4	FTM1_CH0	RTC_CLKOUT		EWM_IN	FTM1_QD_PHB	JTAG_TCLK/SWD_CLK
97	63	PTA5	RESET_b		PTA5		TCLK1			JTAG_TRST_b	RESET_b
98	64	PTA4	JTAG_TMS/SWD_DIO		PTA4			ACMP0_OUT	EWM_OUT_b		JTAG_TMS/SWD_DIO
99	—	PTA9	DISABLED		PTA9			FXIO_D7	FTM3_FLT2	FTM1_FLT3	
100	—	PTA8	DISABLED		PTA8			FXIO_D6	FTM3_FLT3		

34.2.2 Pin properties

NOTE

For some pins, for example UART RXD, I2C SDA and SCL, they can be configure as pesudo opendrain pins via its module itself or the SIM module. Please see the respective module chapter and Port control and interrupt module features for details.

100 LQFP	64 LQFP	Pin Name	Driver strength	Default status after POR	Pullup/ pulldown setting after POR	Slew rate after POR	Passive pin filter after POR	True Open Drain	Pin interrupt
1		PTE16	ND	Hi-Z	—	—	—	—	Y
2		PTE15	ND	Hi-Z	—	—	—	—	Y
3	1	PTD1	HD	Hi-Z	—	—	—	—	Y
4	2	PTD0	HD	Hi-Z	—	—	—	—	Y
5	3	PTE11	ND	Hi-Z	—	—	—	—	Y
6	4	PTE10	ND	Hi-Z	—	—	—	—	Y
7		PTE13	ND	Hi-Z	—	—	—	—	Y
8	5	PTE5	ND	Hi-Z	—	—	—	—	Y
9	6	PTE4	ND	Hi-Z	—	—	—	—	Y
10	7	VDD	—	—	—	—	—	—	—
11	8	VDDA	—	—	—	—	—	—	—
12	9	VREFH	—	—	—	—	—	—	—
13	10	VREFL	—	—	—	—	—	—	—
14		VSS	—	—	—	—	—	—	—
15	11	PTB7	ND	Hi-Z	—	—	—	—	Y
16	12	PTB6	ND	Hi-Z	—	—	—	—	Y
17		PTE14	ND	Hi-Z	—	—	—	—	Y
18	13	PTE3	ND	Hi-Z	—	—	—	—	Y

Table continues on the next page...

100 LQFP	64 LQFP	Pin Name	Driver strength	Default status after POR	Pullup/ pulldown setting after POR	Slew rate after POR	Passive pin filter after POR	True Open Drain	Pin interrupt
19		PTE12	ND	Hi-Z	—	—	—	—	Y
20		PTD17	ND	Hi-Z	—	—	—	—	Y
21	14	PTD16	HD	Hi-Z	—	—	—	—	Y
22	15	PTD15	HD	Hi-Z	—	—	—	—	Y
23	16	PTE9	ND	Hi-Z	—	—	—	—	Y
24		PTD14	ND	Hi-Z	—	—	—	—	Y
25		PTD13	ND	Hi-Z	—	—	—	—	Y
26	17	PTE8	ND	Hi-Z	—	—	—	—	Y
27	18	PTB5	HD	Hi-Z	—	—	—	—	Y
28	19	PTB4	HD	Hi-Z	—	—	—	—	Y
29	20	PTC3	ND	Hi-Z	—	—	—	—	Y
30	21	PTC2	ND	Hi-Z	—	—	—	—	Y
31	22	PTD7	ND	Hi-Z	—	—	—	—	Y
32	23	PTD6	ND	Hi-Z	—	—	—	—	Y
33	24	PTD5	ND	Hi-Z	—	—	—	—	Y
34		PTD12	ND	Hi-Z	—	—	—	—	Y
35		PTD11	ND	Hi-Z	—	—	—	—	Y
36		PTD10	ND	Hi-Z	—	—	—	—	Y
37		VSS	—	—	—	—	—	—	—
38		VDD	—	—	—	—	—	—	—
39	25	PTC1	ND	Hi-Z	—	—	—	—	Y
40	26	PTC0	ND	Hi-Z	—	—	—	—	Y
41		PTD9	ND	Hi-Z	—	—	—	—	Y
42		PTD8	ND	Hi-Z	—	—	—	—	Y
43	27	PTC17	ND	Hi-Z	—	—	—	—	Y
44	28	PTC16	ND	Hi-Z	—	—	—	—	Y
45	29	PTC15	ND	Hi-Z	—	—	—	—	Y
46	30	PTC14	ND	Hi-Z	—	—	—	—	Y
47	31	PTB3	ND	Hi-Z	—	—	—	—	Y
48	32	PTB2	ND	Hi-Z	—	—	—	—	Y
49		PTC13	ND	Hi-Z	—	—	—	—	Y
50		PTC12	ND	Hi-Z	—	—	—	—	Y
51		PTC11	ND	Hi-Z	—	—	—	—	Y
52		PTC10	ND	Hi-Z	—	—	—	—	Y
53	33	PTB1	ND	Hi-Z	—	—	—	—	Y
54	34	PTB0	ND	Hi-Z	—	—	—	—	Y
55	35	PTC9	ND	Hi-Z	—	—	—	—	Y
56	36	PTC8	ND	Hi-Z	—	—	—	—	Y

Table continues on the next page...

100 LQFP	64 LQFP	Pin Name	Driver strength	Default status after POR	Pullup/ pulldown setting after POR	Slew rate after POR	Passive pin filter after POR	True Open Drain	Pin interrupt
57	37	PTA7	ND	Hi-Z	—	—	—	—	Y
58	38	PTA6	ND	Hi-Z	—	—	—	—	Y
59	39	PTE7	ND	Hi-Z	—	—	—	—	Y
60	40	VSS	—	—	—	—	—	—	—
61	41	VDD	—	—	—	—	—	—	—
62		PTA17	ND	Hi-Z	—	—	—	—	Y
63		PTB17	ND	Hi-Z	—	—	—	—	Y
64		PTB16	ND	Hi-Z	—	—	—	—	Y
65		PTB15	ND	Hi-Z	—	—	—	—	Y
66		PTB14	ND	Hi-Z	—	—	—	—	Y
67	42	PTB13	ND	Hi-Z	—	—	—	—	Y
68	43	PTB12	ND	Hi-Z	—	—	—	—	Y
69	44	PTD4	ND	Hi-Z	—	—	—	—	Y
70	45	PTD3	ND	H	PU	—	N	—	Y
71	46	PTD2	ND	Hi-Z	—	—	—	—	Y
72	47	PTA3	ND	Hi-Z	—	—	—	—	Y
73	48	PTA2	ND	Hi-Z	—	—	—	—	Y
74		PTB11	ND	Hi-Z	—	—	—	—	Y
75		PTB10	ND	Hi-Z	—	—	—	—	Y
76		PTB9	ND	Hi-Z	—	—	—	—	Y
77		PTB8	ND	Hi-Z	—	—	—	—	Y
78	49	PTA1	ND	Hi-Z	—	—	—	—	Y
79	50	PTA0	ND	Hi-Z	—	—	—	—	Y
80	51	PTC7	ND	Hi-Z	—	—	—	—	Y
81	52	PTC6	ND	Hi-Z	—	—	—	—	Y
82		PTA16	ND	Hi-Z	—	—	—	—	Y
83		PTA15	ND	Hi-Z	—	—	—	—	Y
84	53	PTE6	ND	Hi-Z	—	—	—	—	Y
85	54	PTE2	ND	Hi-Z	—	—	—	—	Y
86		VSS	—	—	—	—	—	—	—
87		VDD	—	—	—	—	—	—	—
88		PTA14	ND	Hi-Z	—	—	—	—	Y
89	55	PTA13	ND	Hi-Z	—	—	—	—	Y
90	56	PTA12	ND	Hi-Z	—	—	—	—	Y
91	57	PTA11	ND	Hi-Z	—	—	—	—	Y
92	58	PTA10	ND	Hi-Z	—	—	—	—	Y
93	59	PTE1	HD	Hi-Z	—	—	—	—	Y
94	60	PTE0	HD	Hi-Z	—	—	—	—	Y

Table continues on the next page...

100 LQFP	64 LQFP	Pin Name	Driver strength	Default status after POR	Pullup/ pulldown setting after POR	Slew rate after POR	Passive pin filter after POR	True Open Drain	Pin interrupt
95	61	PTC5	ND	H	PU	—	—	—	Y
96	62	PTC4	ND	L	PD	—	—	—	Y
97	63	PTA5	ND	H	PU	—	Y	—	Y
98	64	PTA4	ND	H	PU	—	—	—	Y
99		PTA9	ND	Hi-Z	—	—	—	—	Y
100		PTA8	ND	Hi-Z	—	—	—	—	Y

Properties	Abbreviation	Descriptions
Driver strength	ND	Normal drive
	HD	High drive
Default status after POR	Hi-Z	High impendence
	H	High level
	L	Low level
Pullup/ pulldown setting after POR	PU	Pullup
	PD	Pulldown
Slew rate after POR	FS	Fast slew rate
	SS	Slow slew rate
Passive Pin Filter after POR	N	Disabled
	Y	Enabled
Open drain	N	Disabled
	Y	Enabled
Pin interrupt	Y	Yes

34.2.3 Pinout diagram

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

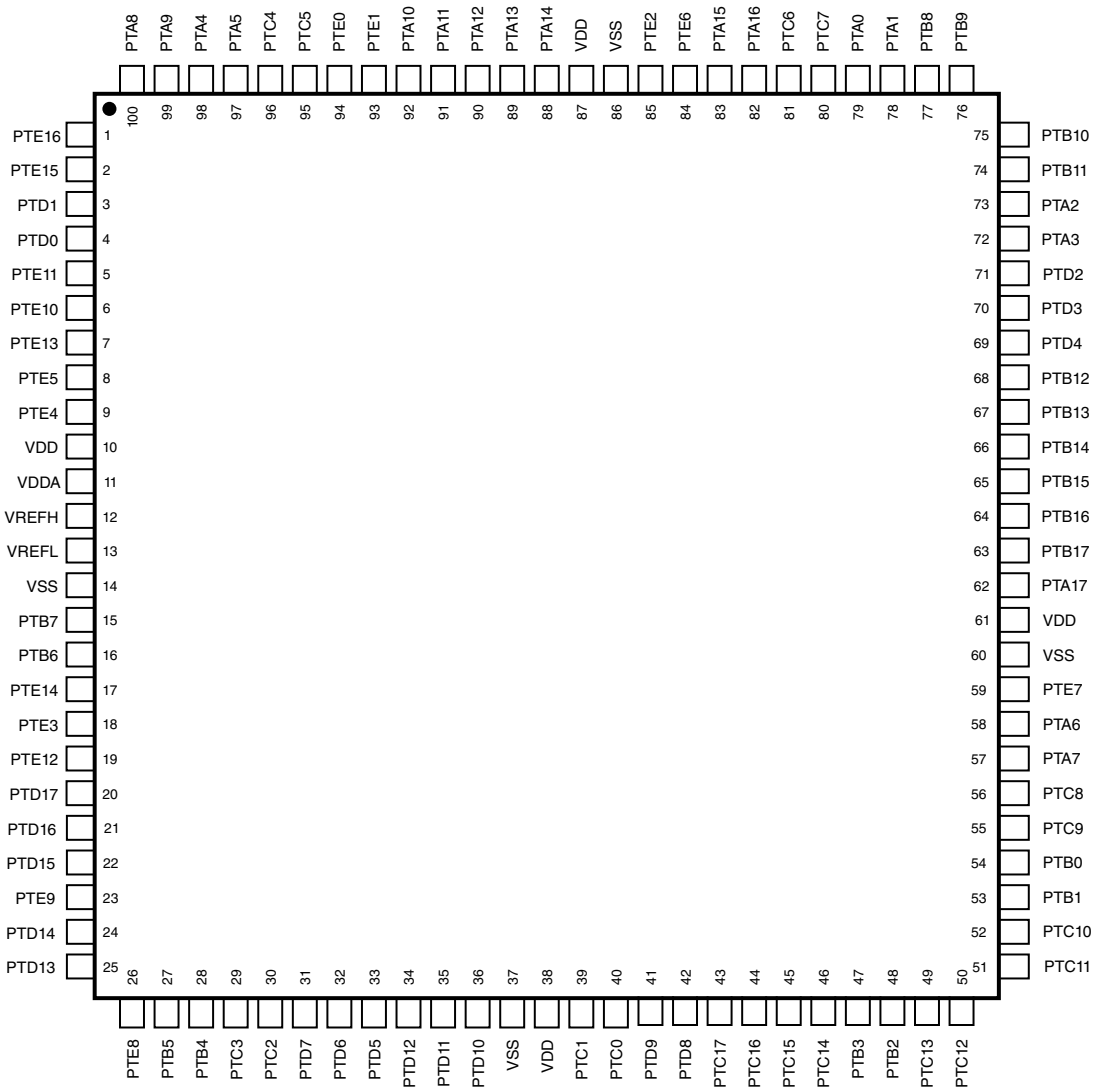


Figure 34-1. 100 LQFP Pinout Diagram

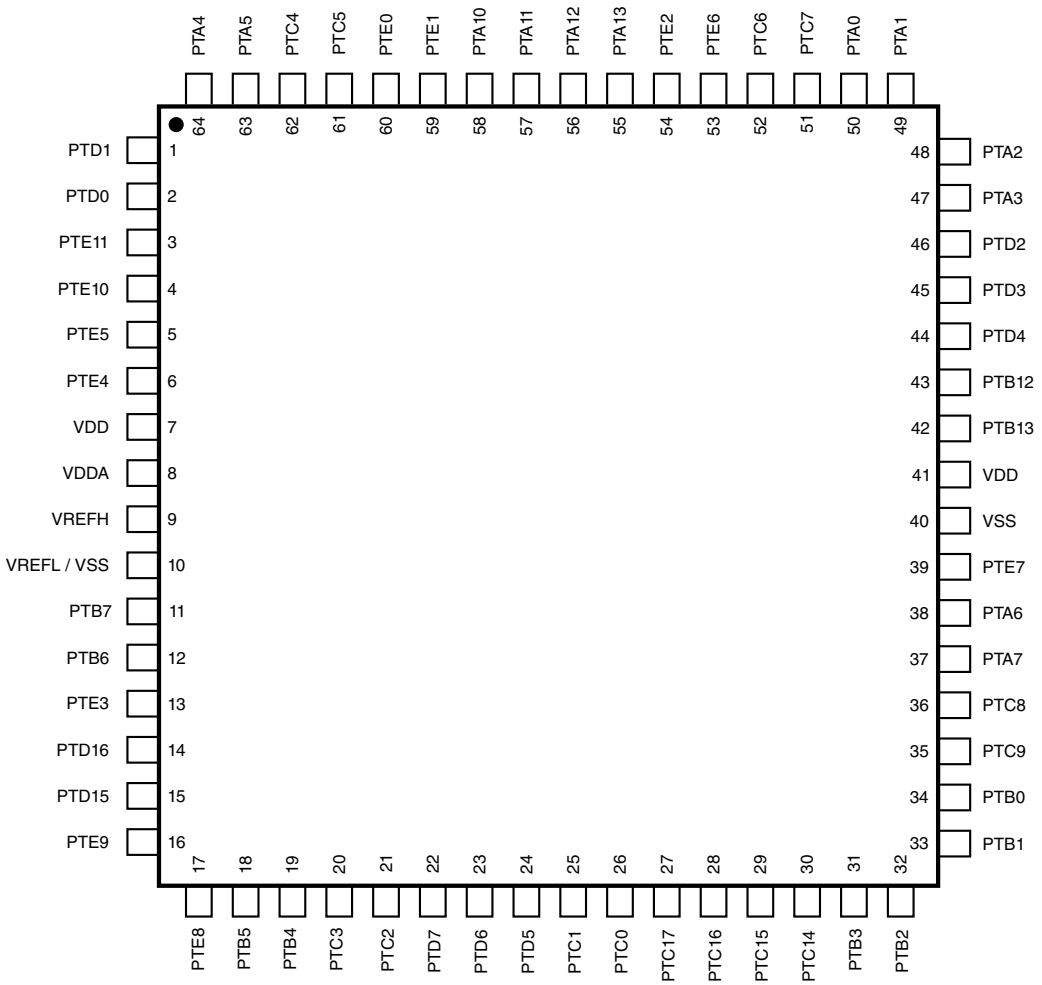


Figure 34-2. 64 LQFP Pinout Diagram

34.3 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.

34.3.1 Core Modules

Table 34-1. JTAG Signal Descriptions

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

Table 34-2. SWD Signal Descriptions

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

34.3.2 System Modules

Table 34-3. System Signal Descriptions

Chip signal name	Module signal name	Description	I/O
NMI_b	—	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
RESET_b	—	Reset bidirectional signal	I/O
VDD	—	MCU power	I
VSS	—	MCU ground	I

Table 34-4. 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
EWM_OUT_b	$\overline{\text{EWM\_out}}$	EWM reset out signal	O

34.3.3 Clock Modules

Table 34-5. OSC (in SCG) Signal Descriptions

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

Table 34-6. RTC Oscillator (OSC32) 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

34.3.4 Analog

Table 34-7. ADC*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
ADC*n*_SE[15:0]	AD[15:0]	Single-Ended Analog Channel Inputs	I
VREFH	V_{REFSH}	Voltage Reference Select High	I
VREFL	V_{REFSL}	Voltage Reference Select Low	I
VDDA	V_{DDA}	Analog Power Supply	I

Table 34-8. DAC0 Signal Descriptions

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

Table 34-9. ACMP*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
ACMP*n*_IN[6:0]	IN[6:0]	Analog voltage inputs	I
ACMP*n*_OUT	CMPO	Comparator output	O

34.3.5 Timer Modules

Table 34-10. LPTMR0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPTMR0_ALT[3:1]	LPTMR_ALT*n*	Pulse Counter Input pin	I

Table 34-11. RTC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
RTC_CLKOUT	RTC_CLKOUT	1 Hz square-wave output or 32 kHz clock	O

Table 34-12. FTM*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
FTM*n*_CH[7:0]	CHn	FTM channel (n), where n can be 7-0	I/O
FTM*n*_FLT[3:0]	FAULTj	Fault input (j), where j can be 3-0	I
TCLK[2:0]	EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I

34.3.6 Communication Interfaces

Table 34-13. CAN*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CAN*n*_RX	CAN Rx	CAN Receive Pin	I
CAN*n*_TX	CAN Tx	CAN Transmit Pin	O

Table 34-14. LPSPI*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPSPI*n*_SOUT	SOUT	Serial Data Out	O
LPSPI*n*_SIN	SIN	Serial Data In	I
LPSPI*n*_SCK	SCK	Serial Clock	I/O
LPSPI*n*_PCS[3:0]	PCS[3:0]	Peripheral Chip Select 0-3	I/O

Table 34-15. LPI2C*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPI2C*n*_SCL	SCL	Bidirectional serial clock line of the I2C system.	I/O
LPI2C*n*_SDA	SDA	Bidirectional serial data line of the I2C system.	I/O
LPI2C*n*_HREQ	HREQ	Host request, can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
LPI2C*n*_SCLS	SCLS	Secondary I2C clock line.	I/O
LPI2C*n*_SDAS	SDAS	Secondary I2C data line.	I/O

Table 34-16. LPUART*n* Signal Descriptions

Chip signal name	Module signal name	Description	I/O
LPUART*n*_TX	LPUART_TXD	Transmit data	I/O
LPUART*n*_RX	LPUART_RXD	Receive data	I
LPUART*n*_CTS	LPUART_CTS	Clear to send	I
LPUART*n*_RTS	LPUART_RTS	Request to send	O

Table 34-17. FlexIO Signal Descriptions

Chip signal name	Module signal name	Description	I/O
FXIO_D[7:0]	FXIO_D[7:0]	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

34.3.7 Human-Machine Interfaces (HMI)

Table 34-18. GPIO Signal Descriptions

Chip signal name	Module signal name	Description	I/O
PTA[17:0]	PORTA17–PORTA0	General-purpose input/output	I/O
PTB[17:0]	PORTB17–PORTB0	General-purpose input/output	I/O
PTC[17:0]	PORTC17–PORTC0	General-purpose input/output	I/O
PTD[17:0]	PORTD17–PORTD0	General-purpose input/output	I/O
PTE[16:0]	PORTE16–PORTE0	General-purpose input/output	I/O

Chapter 35
Port Control and Interrupts (PORT)

35.1 Chip-specific information for this module

35.1.1 I/O pin structure

The following figure shows the structure of normal I/O pin.

See Pin properties for properties on each pin.

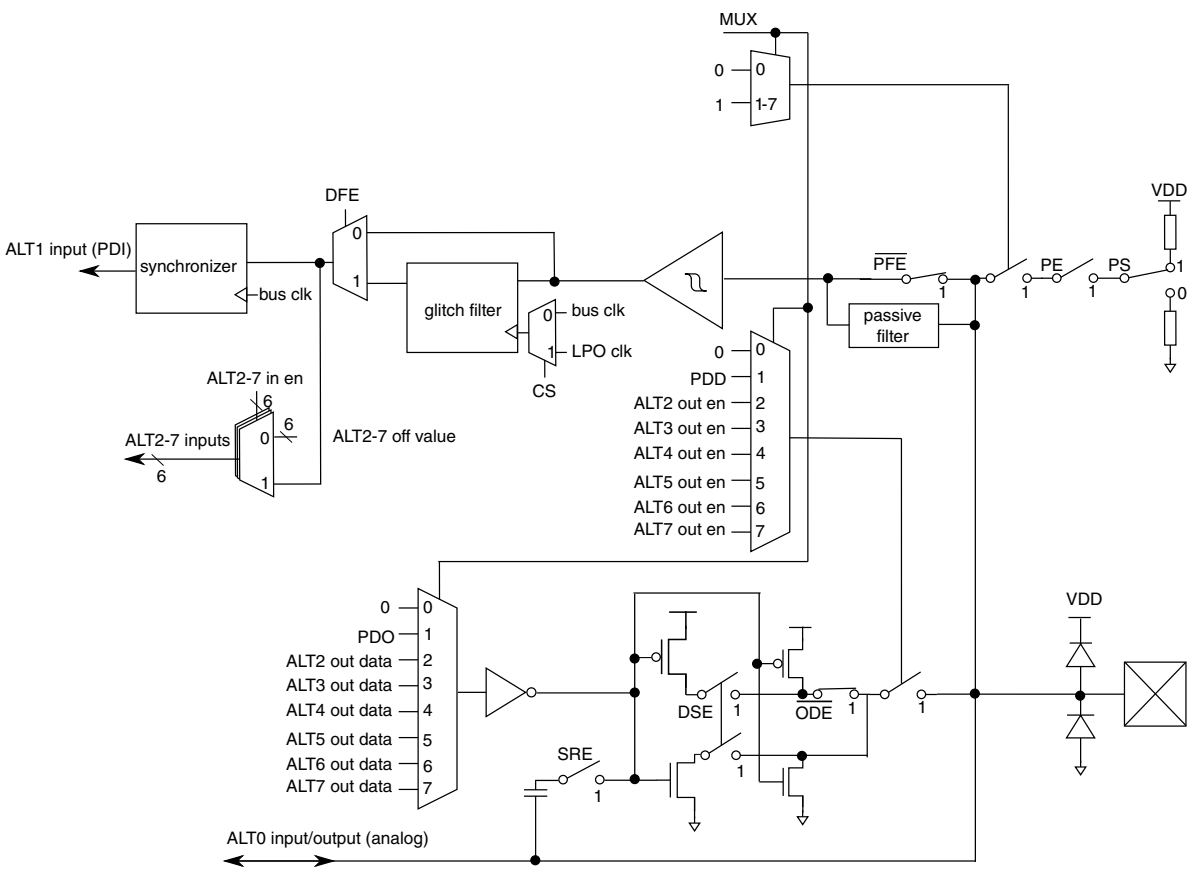


Figure 35-1. Normal I/O structure

35.1.2 Port control and interrupt module features

- 32-pin ports

NOTE

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

- Each 32-pin port is assigned one interrupt.

Table 35-1. Ports summary

Feature	Port A	Port B	Port C	Port D	Port E
Pull select control	Yes	Yes	Yes	Yes	Yes
Pull select at reset	PTA4/PTA5=Pull up, Others=No	No	PTC5=Pull up, Others=No	PTD3=Pull up, Others=No	No
Pull enable control	Yes	Yes	Yes	Yes	Yes
Pull enable at reset	PTA4/ PTA5=Enabled; Others=Disabled	Disabled	PTC4/ PTC5=Enabled; Others=Disabled	PTD3=Enabled; Others=Disabled	Disabled
Passive filter enable control	PTA5=Yes; Others=No	No	No	PTD3=Yes; Others=No	No
Passive filter enable at reset	PTA5=Enabled; Others=Disabled	Disabled	Disabled	Disabled	Disabled
Open drain enable control	Disabled	Disabled	Disabled	Disabled	Disabled
Open drain enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Drive strength enable control	No	PTB4/PTB5 only	No	PTD0/PTD1/ PTD15/PTD16 only	PTE0/PTE1 only
Drive strength enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Pin mux control	Yes	Yes	Yes	Yes	Yes
Pin mux at reset	PTA4/PTA5/ PTA10=ALT7; Others=ALT0	ALT0	PTC4/PTC5=ALT7; Others=ALT0	PTD3=ALT7; Others=ALT0	ALT0
Lock bit	Yes	Yes	Yes	Yes	Yes
Interrupt and DMA request	Yes	Yes	Yes	Yes	Yes
Digital glitch filter	Yes	Yes	Yes	Yes	Yes

35.1.3 Application-related Information

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.
3. The clock to the port control module can be gated on and off using the PCC_PORTx register. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing the corresponding module, set PCC_PORTx[CGC] to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution chapter.

35.2 Introduction

35.3 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.

35.3.1 Features

The PORT module has the following features:

- Pin interrupt
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wake-up in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
- 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 support
    - Individual drive strength field supporting high and low drive strength
    - Individual input passive filter field supporting enable and disable of the individual input passive filter
    - 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.

35.3.2 Modes of operation

35.3.2.1 Run mode

In Run mode, the PORT operates normally.

35.3.2.2 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.

35.3.2.3 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.

35.3.2.4 Debug mode

In Debug mode, PORT operates normally.

35.4 External signal description

The table found here describes the PORT external signal.

Table 35-2. 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.

35.5 Detailed signal description

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

Table 35-3. 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.

35.6 Memory map and register definition

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

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_9000	Pin Control Register n (PORTA_PCR0)	32	R/W	See section	35.6.1/788
4004_9004	Pin Control Register n (PORTA_PCR1)	32	R/W	See section	35.6.1/788
4004_9008	Pin Control Register n (PORTA_PCR2)	32	R/W	See section	35.6.1/788

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_900C	Pin Control Register n (PORTA_PCR3)	32	R/W	See section	35.6.1/788
4004_9010	Pin Control Register n (PORTA_PCR4)	32	R/W	See section	35.6.1/788
4004_9014	Pin Control Register n (PORTA_PCR5)	32	R/W	See section	35.6.1/788
4004_9018	Pin Control Register n (PORTA_PCR6)	32	R/W	See section	35.6.1/788
4004_901C	Pin Control Register n (PORTA_PCR7)	32	R/W	See section	35.6.1/788
4004_9020	Pin Control Register n (PORTA_PCR8)	32	R/W	See section	35.6.1/788
4004_9024	Pin Control Register n (PORTA_PCR9)	32	R/W	See section	35.6.1/788
4004_9028	Pin Control Register n (PORTA_PCR10)	32	R/W	See section	35.6.1/788
4004_902C	Pin Control Register n (PORTA_PCR11)	32	R/W	See section	35.6.1/788
4004_9030	Pin Control Register n (PORTA_PCR12)	32	R/W	See section	35.6.1/788
4004_9034	Pin Control Register n (PORTA_PCR13)	32	R/W	See section	35.6.1/788
4004_9038	Pin Control Register n (PORTA_PCR14)	32	R/W	See section	35.6.1/788
4004_903C	Pin Control Register n (PORTA_PCR15)	32	R/W	See section	35.6.1/788
4004_9040	Pin Control Register n (PORTA_PCR16)	32	R/W	See section	35.6.1/788
4004_9044	Pin Control Register n (PORTA_PCR17)	32	R/W	See section	35.6.1/788
4004_9048	Pin Control Register n (PORTA_PCR18)	32	R/W	See section	35.6.1/788
4004_904C	Pin Control Register n (PORTA_PCR19)	32	R/W	See section	35.6.1/788
4004_9050	Pin Control Register n (PORTA_PCR20)	32	R/W	See section	35.6.1/788
4004_9054	Pin Control Register n (PORTA_PCR21)	32	R/W	See section	35.6.1/788
4004_9058	Pin Control Register n (PORTA_PCR22)	32	R/W	See section	35.6.1/788
4004_905C	Pin Control Register n (PORTA_PCR23)	32	R/W	See section	35.6.1/788
4004_9060	Pin Control Register n (PORTA_PCR24)	32	R/W	See section	35.6.1/788
4004_9064	Pin Control Register n (PORTA_PCR25)	32	R/W	See section	35.6.1/788
4004_9068	Pin Control Register n (PORTA_PCR26)	32	R/W	See section	35.6.1/788
4004_906C	Pin Control Register n (PORTA_PCR27)	32	R/W	See section	35.6.1/788
4004_9070	Pin Control Register n (PORTA_PCR28)	32	R/W	See section	35.6.1/788
4004_9074	Pin Control Register n (PORTA_PCR29)	32	R/W	See section	35.6.1/788
4004_9078	Pin Control Register n (PORTA_PCR30)	32	R/W	See section	35.6.1/788
4004_907C	Pin Control Register n (PORTA_PCR31)	32	R/W	See section	35.6.1/788
4004_9080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads 0)	0000_0000h	35.6.2/791
4004_9084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always reads 0)	0000_0000h	35.6.3/791
4004_90A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	35.6.4/792
4004_90C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	35.6.5/792
4004_90C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	35.6.6/793
4004_90C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	35.6.7/793

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A000	Pin Control Register n (PORTB_PCR0)	32	R/W	See section	35.6.1/788
4004_A004	Pin Control Register n (PORTB_PCR1)	32	R/W	See section	35.6.1/788
4004_A008	Pin Control Register n (PORTB_PCR2)	32	R/W	See section	35.6.1/788
4004_A00C	Pin Control Register n (PORTB_PCR3)	32	R/W	See section	35.6.1/788
4004_A010	Pin Control Register n (PORTB_PCR4)	32	R/W	See section	35.6.1/788
4004_A014	Pin Control Register n (PORTB_PCR5)	32	R/W	See section	35.6.1/788
4004_A018	Pin Control Register n (PORTB_PCR6)	32	R/W	See section	35.6.1/788
4004_A01C	Pin Control Register n (PORTB_PCR7)	32	R/W	See section	35.6.1/788
4004_A020	Pin Control Register n (PORTB_PCR8)	32	R/W	See section	35.6.1/788
4004_A024	Pin Control Register n (PORTB_PCR9)	32	R/W	See section	35.6.1/788
4004_A028	Pin Control Register n (PORTB_PCR10)	32	R/W	See section	35.6.1/788
4004_A02C	Pin Control Register n (PORTB_PCR11)	32	R/W	See section	35.6.1/788
4004_A030	Pin Control Register n (PORTB_PCR12)	32	R/W	See section	35.6.1/788
4004_A034	Pin Control Register n (PORTB_PCR13)	32	R/W	See section	35.6.1/788
4004_A038	Pin Control Register n (PORTB_PCR14)	32	R/W	See section	35.6.1/788
4004_A03C	Pin Control Register n (PORTB_PCR15)	32	R/W	See section	35.6.1/788
4004_A040	Pin Control Register n (PORTB_PCR16)	32	R/W	See section	35.6.1/788
4004_A044	Pin Control Register n (PORTB_PCR17)	32	R/W	See section	35.6.1/788
4004_A048	Pin Control Register n (PORTB_PCR18)	32	R/W	See section	35.6.1/788
4004_A04C	Pin Control Register n (PORTB_PCR19)	32	R/W	See section	35.6.1/788
4004_A050	Pin Control Register n (PORTB_PCR20)	32	R/W	See section	35.6.1/788
4004_A054	Pin Control Register n (PORTB_PCR21)	32	R/W	See section	35.6.1/788
4004_A058	Pin Control Register n (PORTB_PCR22)	32	R/W	See section	35.6.1/788
4004_A05C	Pin Control Register n (PORTB_PCR23)	32	R/W	See section	35.6.1/788
4004_A060	Pin Control Register n (PORTB_PCR24)	32	R/W	See section	35.6.1/788
4004_A064	Pin Control Register n (PORTB_PCR25)	32	R/W	See section	35.6.1/788
4004_A068	Pin Control Register n (PORTB_PCR26)	32	R/W	See section	35.6.1/788
4004_A06C	Pin Control Register n (PORTB_PCR27)	32	R/W	See section	35.6.1/788
4004_A070	Pin Control Register n (PORTB_PCR28)	32	R/W	See section	35.6.1/788
4004_A074	Pin Control Register n (PORTB_PCR29)	32	R/W	See section	35.6.1/788
4004_A078	Pin Control Register n (PORTB_PCR30)	32	R/W	See section	35.6.1/788
4004_A07C	Pin Control Register n (PORTB_PCR31)	32	R/W	See section	35.6.1/788
4004_A080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads 0)	0000_0000h	35.6.2/791
4004_A084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads 0)	0000_0000h	35.6.3/791
4004_A0A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	35.6.4/792

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A0C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	35.6.5/792
4004_A0C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	35.6.6/793
4004_A0C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	35.6.7/793
4004_B000	Pin Control Register n (PORTC_PCR0)	32	R/W	See section	35.6.1/788
4004_B004	Pin Control Register n (PORTC_PCR1)	32	R/W	See section	35.6.1/788
4004_B008	Pin Control Register n (PORTC_PCR2)	32	R/W	See section	35.6.1/788
4004_B00C	Pin Control Register n (PORTC_PCR3)	32	R/W	See section	35.6.1/788
4004_B010	Pin Control Register n (PORTC_PCR4)	32	R/W	See section	35.6.1/788
4004_B014	Pin Control Register n (PORTC_PCR5)	32	R/W	See section	35.6.1/788
4004_B018	Pin Control Register n (PORTC_PCR6)	32	R/W	See section	35.6.1/788
4004_B01C	Pin Control Register n (PORTC_PCR7)	32	R/W	See section	35.6.1/788
4004_B020	Pin Control Register n (PORTC_PCR8)	32	R/W	See section	35.6.1/788
4004_B024	Pin Control Register n (PORTC_PCR9)	32	R/W	See section	35.6.1/788
4004_B028	Pin Control Register n (PORTC_PCR10)	32	R/W	See section	35.6.1/788
4004_B02C	Pin Control Register n (PORTC_PCR11)	32	R/W	See section	35.6.1/788
4004_B030	Pin Control Register n (PORTC_PCR12)	32	R/W	See section	35.6.1/788
4004_B034	Pin Control Register n (PORTC_PCR13)	32	R/W	See section	35.6.1/788
4004_B038	Pin Control Register n (PORTC_PCR14)	32	R/W	See section	35.6.1/788
4004_B03C	Pin Control Register n (PORTC_PCR15)	32	R/W	See section	35.6.1/788
4004_B040	Pin Control Register n (PORTC_PCR16)	32	R/W	See section	35.6.1/788
4004_B044	Pin Control Register n (PORTC_PCR17)	32	R/W	See section	35.6.1/788
4004_B048	Pin Control Register n (PORTC_PCR18)	32	R/W	See section	35.6.1/788
4004_B04C	Pin Control Register n (PORTC_PCR19)	32	R/W	See section	35.6.1/788
4004_B050	Pin Control Register n (PORTC_PCR20)	32	R/W	See section	35.6.1/788
4004_B054	Pin Control Register n (PORTC_PCR21)	32	R/W	See section	35.6.1/788
4004_B058	Pin Control Register n (PORTC_PCR22)	32	R/W	See section	35.6.1/788
4004_B05C	Pin Control Register n (PORTC_PCR23)	32	R/W	See section	35.6.1/788
4004_B060	Pin Control Register n (PORTC_PCR24)	32	R/W	See section	35.6.1/788
4004_B064	Pin Control Register n (PORTC_PCR25)	32	R/W	See section	35.6.1/788
4004_B068	Pin Control Register n (PORTC_PCR26)	32	R/W	See section	35.6.1/788
4004_B06C	Pin Control Register n (PORTC_PCR27)	32	R/W	See section	35.6.1/788
4004_B070	Pin Control Register n (PORTC_PCR28)	32	R/W	See section	35.6.1/788
4004_B074	Pin Control Register n (PORTC_PCR29)	32	R/W	See section	35.6.1/788
4004_B078	Pin Control Register n (PORTC_PCR30)	32	R/W	See section	35.6.1/788
4004_B07C	Pin Control Register n (PORTC_PCR31)	32	R/W	See section	35.6.1/788
4004_B080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads 0)	0000_0000h	35.6.2/791

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_B084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads 0)	0000_0000h	35.6.3/791
4004_B0A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	35.6.4/792
4004_B0C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	35.6.5/792
4004_B0C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	35.6.6/793
4004_B0C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	35.6.7/793
4004_C000	Pin Control Register n (PORTD_PCR0)	32	R/W	See section	35.6.1/788
4004_C004	Pin Control Register n (PORTD_PCR1)	32	R/W	See section	35.6.1/788
4004_C008	Pin Control Register n (PORTD_PCR2)	32	R/W	See section	35.6.1/788
4004_C00C	Pin Control Register n (PORTD_PCR3)	32	R/W	See section	35.6.1/788
4004_C010	Pin Control Register n (PORTD_PCR4)	32	R/W	See section	35.6.1/788
4004_C014	Pin Control Register n (PORTD_PCR5)	32	R/W	See section	35.6.1/788
4004_C018	Pin Control Register n (PORTD_PCR6)	32	R/W	See section	35.6.1/788
4004_C01C	Pin Control Register n (PORTD_PCR7)	32	R/W	See section	35.6.1/788
4004_C020	Pin Control Register n (PORTD_PCR8)	32	R/W	See section	35.6.1/788
4004_C024	Pin Control Register n (PORTD_PCR9)	32	R/W	See section	35.6.1/788
4004_C028	Pin Control Register n (PORTD_PCR10)	32	R/W	See section	35.6.1/788
4004_C02C	Pin Control Register n (PORTD_PCR11)	32	R/W	See section	35.6.1/788
4004_C030	Pin Control Register n (PORTD_PCR12)	32	R/W	See section	35.6.1/788
4004_C034	Pin Control Register n (PORTD_PCR13)	32	R/W	See section	35.6.1/788
4004_C038	Pin Control Register n (PORTD_PCR14)	32	R/W	See section	35.6.1/788
4004_C03C	Pin Control Register n (PORTD_PCR15)	32	R/W	See section	35.6.1/788
4004_C040	Pin Control Register n (PORTD_PCR16)	32	R/W	See section	35.6.1/788
4004_C044	Pin Control Register n (PORTD_PCR17)	32	R/W	See section	35.6.1/788
4004_C048	Pin Control Register n (PORTD_PCR18)	32	R/W	See section	35.6.1/788
4004_C04C	Pin Control Register n (PORTD_PCR19)	32	R/W	See section	35.6.1/788
4004_C050	Pin Control Register n (PORTD_PCR20)	32	R/W	See section	35.6.1/788
4004_C054	Pin Control Register n (PORTD_PCR21)	32	R/W	See section	35.6.1/788
4004_C058	Pin Control Register n (PORTD_PCR22)	32	R/W	See section	35.6.1/788
4004_C05C	Pin Control Register n (PORTD_PCR23)	32	R/W	See section	35.6.1/788
4004_C060	Pin Control Register n (PORTD_PCR24)	32	R/W	See section	35.6.1/788
4004_C064	Pin Control Register n (PORTD_PCR25)	32	R/W	See section	35.6.1/788
4004_C068	Pin Control Register n (PORTD_PCR26)	32	R/W	See section	35.6.1/788
4004_C06C	Pin Control Register n (PORTD_PCR27)	32	R/W	See section	35.6.1/788
4004_C070	Pin Control Register n (PORTD_PCR28)	32	R/W	See section	35.6.1/788
4004_C074	Pin Control Register n (PORTD_PCR29)	32	R/W	See section	35.6.1/788
4004_C078	Pin Control Register n (PORTD_PCR30)	32	R/W	See section	35.6.1/788

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C07C	Pin Control Register n (PORTD_PCR31)	32	R/W	See section	35.6.1/788
4004_C080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads 0)	0000_0000h	35.6.2/791
4004_C084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads 0)	0000_0000h	35.6.3/791
4004_C0A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	35.6.4/792
4004_C0C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	35.6.5/792
4004_C0C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	35.6.6/793
4004_C0C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	35.6.7/793
4004_D000	Pin Control Register n (PORTE_PCR0)	32	R/W	See section	35.6.1/788
4004_D004	Pin Control Register n (PORTE_PCR1)	32	R/W	See section	35.6.1/788
4004_D008	Pin Control Register n (PORTE_PCR2)	32	R/W	See section	35.6.1/788
4004_D00C	Pin Control Register n (PORTE_PCR3)	32	R/W	See section	35.6.1/788
4004_D010	Pin Control Register n (PORTE_PCR4)	32	R/W	See section	35.6.1/788
4004_D014	Pin Control Register n (PORTE_PCR5)	32	R/W	See section	35.6.1/788
4004_D018	Pin Control Register n (PORTE_PCR6)	32	R/W	See section	35.6.1/788
4004_D01C	Pin Control Register n (PORTE_PCR7)	32	R/W	See section	35.6.1/788
4004_D020	Pin Control Register n (PORTE_PCR8)	32	R/W	See section	35.6.1/788
4004_D024	Pin Control Register n (PORTE_PCR9)	32	R/W	See section	35.6.1/788
4004_D028	Pin Control Register n (PORTE_PCR10)	32	R/W	See section	35.6.1/788
4004_D02C	Pin Control Register n (PORTE_PCR11)	32	R/W	See section	35.6.1/788
4004_D030	Pin Control Register n (PORTE_PCR12)	32	R/W	See section	35.6.1/788
4004_D034	Pin Control Register n (PORTE_PCR13)	32	R/W	See section	35.6.1/788
4004_D038	Pin Control Register n (PORTE_PCR14)	32	R/W	See section	35.6.1/788
4004_D03C	Pin Control Register n (PORTE_PCR15)	32	R/W	See section	35.6.1/788
4004_D040	Pin Control Register n (PORTE_PCR16)	32	R/W	See section	35.6.1/788
4004_D044	Pin Control Register n (PORTE_PCR17)	32	R/W	See section	35.6.1/788
4004_D048	Pin Control Register n (PORTE_PCR18)	32	R/W	See section	35.6.1/788
4004_D04C	Pin Control Register n (PORTE_PCR19)	32	R/W	See section	35.6.1/788
4004_D050	Pin Control Register n (PORTE_PCR20)	32	R/W	See section	35.6.1/788
4004_D054	Pin Control Register n (PORTE_PCR21)	32	R/W	See section	35.6.1/788
4004_D058	Pin Control Register n (PORTE_PCR22)	32	R/W	See section	35.6.1/788
4004_D05C	Pin Control Register n (PORTE_PCR23)	32	R/W	See section	35.6.1/788
4004_D060	Pin Control Register n (PORTE_PCR24)	32	R/W	See section	35.6.1/788
4004_D064	Pin Control Register n (PORTE_PCR25)	32	R/W	See section	35.6.1/788
4004_D068	Pin Control Register n (PORTE_PCR26)	32	R/W	See section	35.6.1/788
4004_D06C	Pin Control Register n (PORTE_PCR27)	32	R/W	See section	35.6.1/788

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_D070	Pin Control Register n (PORTE_PCR28)	32	R/W	See section	35.6.1/788
4004_D074	Pin Control Register n (PORTE_PCR29)	32	R/W	See section	35.6.1/788
4004_D078	Pin Control Register n (PORTE_PCR30)	32	R/W	See section	35.6.1/788
4004_D07C	Pin Control Register n (PORTE_PCR31)	32	R/W	See section	35.6.1/788
4004_D080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads 0)	0000_0000h	35.6.2/791
4004_D084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads 0)	0000_0000h	35.6.3/791
4004_D0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	35.6.4/792
4004_D0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	35.6.5/792
4004_D0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	35.6.6/793
4004_D0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	35.6.7/793

35.6.1 Pin Control Register n (PORTx_PCR*n*)

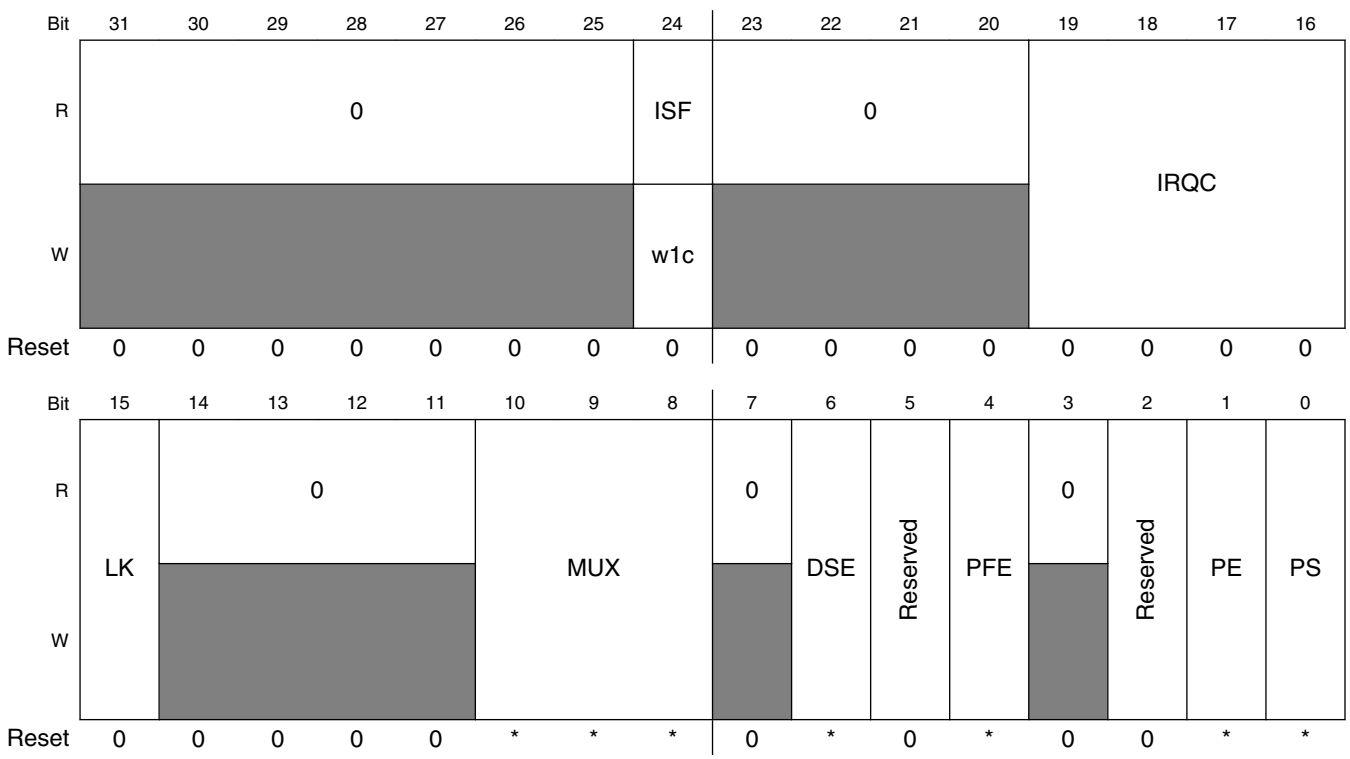
NOTE

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 that are not available in a reduced-pin package offering. Unbonded pins not available in a package are disabled by default to prevent them from consuming power.

Address: Base address + 0h offset + (4d × i), where i=0d to 31d



* Notes:

- MUX field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- DSE field: Varies by port. See the Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PFE 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 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 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 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).

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	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.
6 DSE	Drive Strength Enable Drive strength configuration is valid in all digital pin muxing modes. 0 Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. 1 High drive strength is configured on the corresponding pin, if pin is configured as a digital output.
5 Reserved	This field is reserved.
4 PFE	Passive Filter Enable Passive filter configuration is valid in all digital pin muxing modes. 0 Passive input filter is disabled on the corresponding pin. 1 Passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. Refer to the device data sheet for filter characteristics.
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 PE	Pull Enable 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 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.

35.6.2 Global Pin Control Low Register (PORTx_GPCLR)

Only 32-bit writes are supported to this register.

Address: Base address + 80h offset

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

PORTx_GPCLR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable 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.

35.6.3 Global Pin Control High Register (PORTx_GPCHR)

Only 32-bit writes are supported to this register.

Address: Base address + 84h offset

Bit	31	30	29	28	27	26	25	24	23	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.

35.6.4 Interrupt Status Flag Register (PORTx_ISFR)

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

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	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.

35.6.5 Digital Filter Enable Register (PORTx_DFER)

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

Address: Base address + C0h offset

Bit	31	30	29	28	27	26	25	24	23	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.

PORTx_DFER field descriptions (continued)

Field	Description
	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.

35.6.6 Digital Filter Clock Register (PORTx_DFCR)

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

Address: Base address + C4h 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															CS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_DFCR field descriptions

Field	Description
31–1 Reserved	This 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.

35.6.7 Digital Filter Width Register (PORTx_DFWR)

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

Address: Base address + C8h offset

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

PORTx_DFWR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FILT	Filter Length The digital filter configuration is valid in all digital pin muxing modes. Configures the maximum size of the glitches, in clock cycles, 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.

35.7 Functional description

35.7.1 Pin control

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

The upper half of the Pin Control register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred.

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

- Pullup or pulldown enable
- Drive strength
- Passive input filter enable
- Pin Muxing mode

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

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

The LK bit (bit 15 of Pin Control Register PCR*n*) 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.

35.7.2 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.

35.7.3 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.

35.7.4 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 maximum latency through a digital filter equals three filter clock cycles plus the filter width configuration register.

Chapter 36
General-Purpose Input/Output (GPIO)

36.1 Chip-specific information for this module

36.1.1 Instantiation Information

The number of GPIO signals available on the devices covered by this document are detailed in the "Ordering information" of the DataSheet.

See Pin properties for features of each pins.

Port control and interrupt module features are supported, each 32-pin port will support a single interrupt.

36.2 Introduction

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.

36.2.1 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.

36.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 36-1. Modes of operation

Modes of operation	Description
Run	The GPIO module operates normally.
Wait	The GPIO module operates normally.
Stop	The GPIO module is disabled.
Debug	The GPIO module operates normally.

36.2.3 GPIO signal descriptions

Table 36-2. GPIO signal descriptions

GPIO signal descriptions	Description	I/O
PORTA31–PORTA0	General-purpose input/output	I/O
PORTB31–PORTB0	General-purpose input/output	I/O
PORTC31–PORTC0	General-purpose input/output	I/O
PORTD31–PORTD0	General-purpose input/output	I/O
PORTE31–PORTE0	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.

36.2.3.1 Detailed signal description

Table 36-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description
PORTA31–PORTA0	I/O	General-purpose input/output

Table continues on the next page...

Table 36-3. GPIO interface-detailed signal descriptions (continued)

Signal	I/O	Description	
PORTB31–PORTB0 PORTC31–PORTC0 PORTD31–PORTD0 PORTE31–PORTE0		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
		Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

36.3 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)	32	R/W	0000_0000h	36.3.1/803
400F_F004	Port Set Output Register (GPIOA_PSOR)	32	W (always reads 0)	0000_0000h	36.3.2/803
400F_F008	Port Clear Output Register (GPIOA_PCOR)	32	W (always reads 0)	0000_0000h	36.3.3/804
400F_F00C	Port Toggle Output Register (GPIOA_PTOR)	32	W (always reads 0)	0000_0000h	36.3.4/804
400F_F010	Port Data Input Register (GPIOA_PDIR)	32	R	0000_0000h	36.3.5/805
400F_F014	Port Data Direction Register (GPIOA_PDDR)	32	R/W	0000_0000h	36.3.6/805

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F040	Port Data Output Register (GPIOB_PDOR)	32	R/W	0000_0000h	36.3.1/803
400F_F044	Port Set Output Register (GPIOB_PSOR)	32	W (always reads 0)	0000_0000h	36.3.2/803
400F_F048	Port Clear Output Register (GPIOB_PCOR)	32	W (always reads 0)	0000_0000h	36.3.3/804
400F_F04C	Port Toggle Output Register (GPIOB_PTOR)	32	W (always reads 0)	0000_0000h	36.3.4/804
400F_F050	Port Data Input Register (GPIOB_PDIR)	32	R	0000_0000h	36.3.5/805
400F_F054	Port Data Direction Register (GPIOB_PDDR)	32	R/W	0000_0000h	36.3.6/805
400F_F080	Port Data Output Register (GPIOC_PDOR)	32	R/W	0000_0000h	36.3.1/803
400F_F084	Port Set Output Register (GPIOC_PSOR)	32	W (always reads 0)	0000_0000h	36.3.2/803
400F_F088	Port Clear Output Register (GPIOC_PCOR)	32	W (always reads 0)	0000_0000h	36.3.3/804
400F_F08C	Port Toggle Output Register (GPIOC_PTOR)	32	W (always reads 0)	0000_0000h	36.3.4/804
400F_F090	Port Data Input Register (GPIOC_PDIR)	32	R	0000_0000h	36.3.5/805
400F_F094	Port Data Direction Register (GPIOC_PDDR)	32	R/W	0000_0000h	36.3.6/805
400F_F0C0	Port Data Output Register (GPIOD_PDOR)	32	R/W	0000_0000h	36.3.1/803
400F_F0C4	Port Set Output Register (GPIOD_PSOR)	32	W (always reads 0)	0000_0000h	36.3.2/803
400F_F0C8	Port Clear Output Register (GPIOD_PCOR)	32	W (always reads 0)	0000_0000h	36.3.3/804
400F_F0CC	Port Toggle Output Register (GPIOD_PTOR)	32	W (always reads 0)	0000_0000h	36.3.4/804
400F_F0D0	Port Data Input Register (GPIOD_PDIR)	32	R	0000_0000h	36.3.5/805
400F_F0D4	Port Data Direction Register (GPIOD_PDDR)	32	R/W	0000_0000h	36.3.6/805
400F_F100	Port Data Output Register (GPIOE_PDOR)	32	R/W	0000_0000h	36.3.1/803
400F_F104	Port Set Output Register (GPIOE_PSOR)	32	W (always reads 0)	0000_0000h	36.3.2/803
400F_F108	Port Clear Output Register (GPIOE_PCOR)	32	W (always reads 0)	0000_0000h	36.3.3/804

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F10C	Port Toggle Output Register (GPIOE_PTOR)	32	W (always reads 0)	0000_0000h	36.3.4/804
400F_F110	Port Data Input Register (GPIOE_PDIR)	32	R	0000_0000h	36.3.5/805
400F_F114	Port Data Direction Register (GPIOE_PDDR)	32	R/W	0000_0000h	36.3.6/805

36.3.1 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 unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 0h offset

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

GPIOx_PDOR field descriptions

Field	Description
PDO	Port Data Output Register bits for unbonded pins return a undefined value when read. 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.

36.3.2 Port Set Output Register (GPIOx_PSOR)

This register configures whether to set the fields of the PDOR.

Address: Base address + 4h offset

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

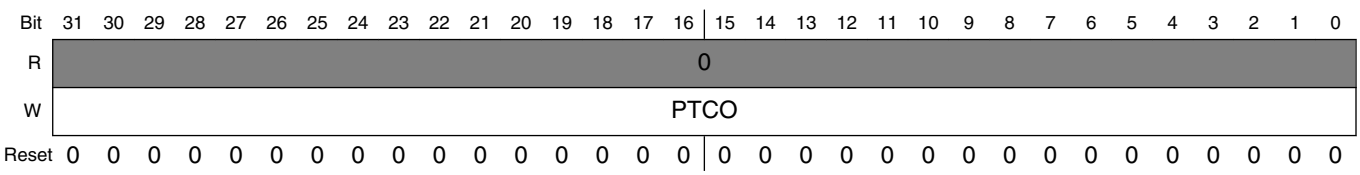
GPIOx_PSOR field descriptions

Field	Description
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.

36.3.3 Port Clear Output Register (GPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset

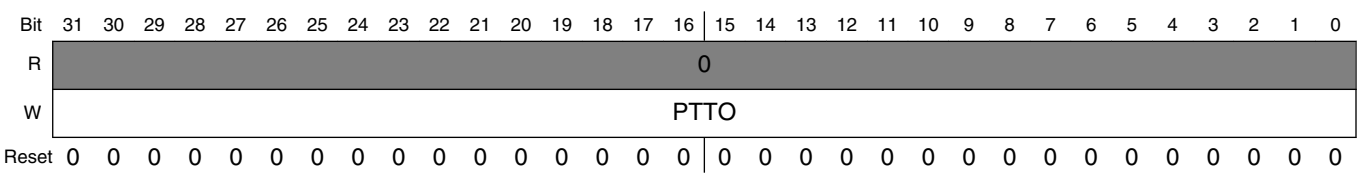


GPIOx_PCOR field descriptions

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.

36.3.4 Port Toggle Output Register (GPIOx_PTOR)

Address: Base address + Ch offset



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:

GPIOx_PTOR field descriptions (continued)

Field	Description
	0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

36.3.5 Port Data Input Register (GPIOx_PDIR)

NOTE

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

Address: Base address + 10h offset

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

GPIOx_PDIR field descriptions

Field	Description
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.

36.3.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Address: Base address + 14h offset

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

GPIOx_PDDR field descriptions

Field	Description
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.

36.4 Functional description

36.4.1 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.

36.4.2 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.

Chapter 37
Analog-to-Digital Converter (ADC)

37.1 Chip-specific information for this module

37.1.1 Instantiation information

Number of ADC	3
Number of result registers per ADC	8

37.1.1.1 Number of ADC channels

Each SAR ADC supports up to 16 external analog input channels, but the exact ADC channel number present on the device is different with packages as indicated in following table.

For details regarding a specific ADC channel available on a particular package, refer to the signal multiplexing chapter of this MCU.

Table 37-1. ADC external channels per package

ADC Module	100LQFP	64LQFP/QFP	80LQFP[1]
ADC0	16	16	16
ADC1	16	11	16
ADC2	16	11	12

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

37.1.1.2 ADC Connections/Channel Assignment

37.1.1.2.1 ADC0 channel assignment

The ADC0 channel assignments for the device are shown in following table. Reserved channels convert to an unknown value.

Table 37-2. ADC0 channel assignment

ADCH Value	Channel	Input
00000	AD0	PTA0/ADC0_SE0
00001	AD1	PTA1/ADC0_SE1
00010	AD2	PTA6/ADC0_SE2
00011	AD3	PTA7/ADC0_SE3
00100	AD4	PTB0/ADC0_SE4
00101	AD5	PTB1/ADC0_SE5
00110	AD6	PTB2/ADC0_SE6
00111	AD7	PTB3/ADC0_SE7
01000	AD8	PTC0/ADC0_SE8
01001	AD9	PTC1/ADC0_SE9
01010	AD10	PTC2/ADC0_SE10
01011	AD11	PTC3/ADC0_SE11
01100	AD12	PTC14/ADC0_SE12
01101	AD13	PTC15/ADC0_SE13
01110	AD14	PTC16/ADC0_SE14
01111	AD15	PTC17/ADC0_SE15
10000	AD16	Reserved
10001	AD17	Reserved
10010	AD18	Reserved
10011	AD19	Reserved
10100	AD20	Reserved
10101	AD21	Reserved
10110	AD22	Reserved
10111	AD23	12-bit DAC output
11000	AD24	Reserved
11001	AD25	Reserved
11010	AD26	Temperature Sensor
11011	AD27	Bandgap (1V reference voltage)
11100	AD28	Reserved
11101	AD29	VREFH
11110	AD30	VREFL
11111	Module disabled	None

37.1.1.2.2 ADC1 channel assignment

The ADC1 channel assignments for the device are shown in following table. Reserved channels convert to an unknown value.

Table 37-3. ADC1 channel assignment

ADCH Value	Channel	Input
00000	AD0	PTA2/ADC1_SE0
00001	AD1	PTA3/ADC1_SE1
00010	AD2	PTD2/ADC1_SE2
00011	AD3	PTD3/ADC1_SE3
00100	AD4	PTC6/ADC1_SE4
00101	AD5	PTC7/ADC1_SE5
00110	AD6	PTD4/ADC1_SE6
00111	AD7	PTB12/ADC1_SE7
01000	AD8	PTB13/ADC1_SE8
01001	AD9	PTB14/ADC1_SE9
01010	AD10	PTE2/ADC1_SE10
01011	AD11	PTE6/ADC1_SE11
01100	AD12	PTA15/ADC1_SE12
01101	AD13	PTA16/ADC1_SE13
01110	AD14	PTB15/ADC1_SE14
01111	AD15	PTB16/ADC1_SE15
10000	AD16	Reserved
10001	AD17	Reserved
10010	AD18	Reserved
10011	AD19	Reserved
10100	AD20	Reserved
10101	AD21	Reserved
10110	AD22	Reserved
10111	AD23	12-bit DAC output
11000	AD24	Reserved
11001	AD25	Reserved
11010	AD26	Temperature Sensor
11011	AD27	Bandgap (1V reference voltage)
11100	AD28	Reserved
11101	AD29	VREFH
11110	AD30	VREFL
11111	Module disabled	None

37.1.1.2.3 ADC2 channel assignment

The ADC2 channel assignments for the device are shown in following table. Reserved channels convert to an unknown value.

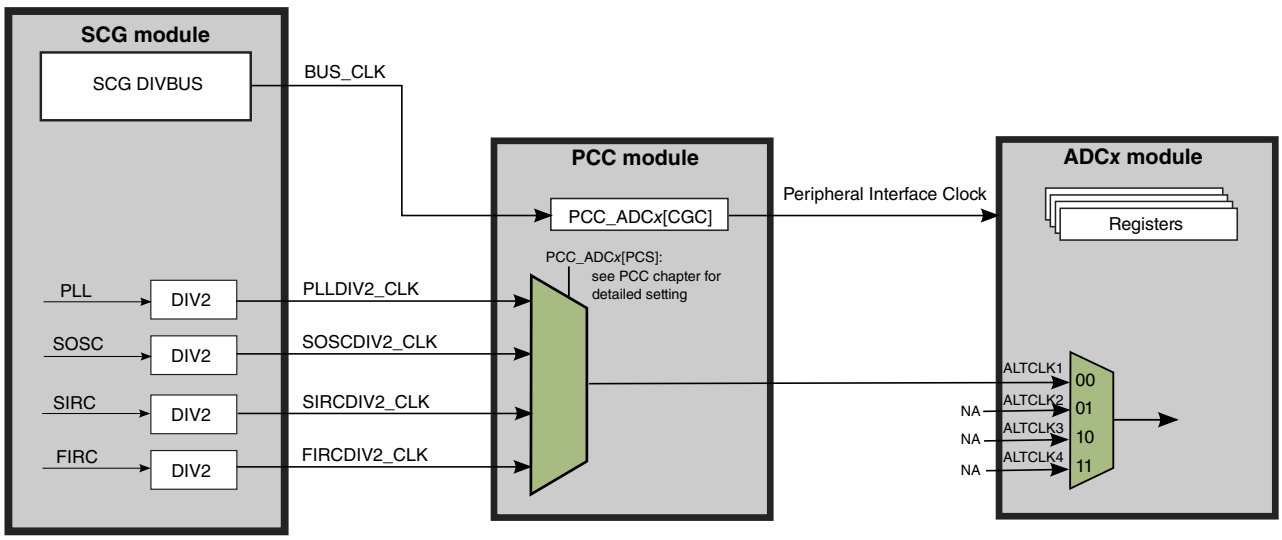
Table 37-4. ADC2 channel assignment

ADCH Value	Channel	Input
00000	AD0	PTD0/ADC2_SE0
00001	AD1	PTD1/ADC2_SE1
00010	AD2	PTE7/ADC2_SE2
00011	AD3	PTB17/ADC2_SE3
00100	AD4	PTA13/ADC2_SE4
00101	AD5	PTA12/ADC2_SE5
00110	AD6	PTE1/ADC2_SE6
00111	AD7	PTE0/ADC2_SE7
01000	AD8	PTB11/ADC2_SE8
01001	AD9	PTB10/ADC2_SE9
01010	AD10	PTB9/ADC2_SE10
01011	AD11	PTB8/ADC2_SE11
01100	AD12	PTE10/ADC2_SE12
01101	AD13	PTE11/ADC2_SE13
01110	AD14	PTC8/ADC2_SE14
01111	AD15	PTC9/ADC2_SE15
10000	AD16	Reserved
10001	AD17	Reserved
10010	AD18	Reserved
10011	AD19	Reserved
10100	AD20	Reserved
10101	AD21	Reserved
10110	AD22	Reserved
10111	AD23	12-bit DAC output
11000	AD24	Reserved
11001	AD25	Reserved
11010	AD26	Temperature Sensor
11011	AD27	Bandgap (1V reference voltage)
11100	AD28	Reserved
11101	AD29	VREFH
11110	AD30	VREFL
11111	Module disabled	None

37.1.2 ADC Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - ADC

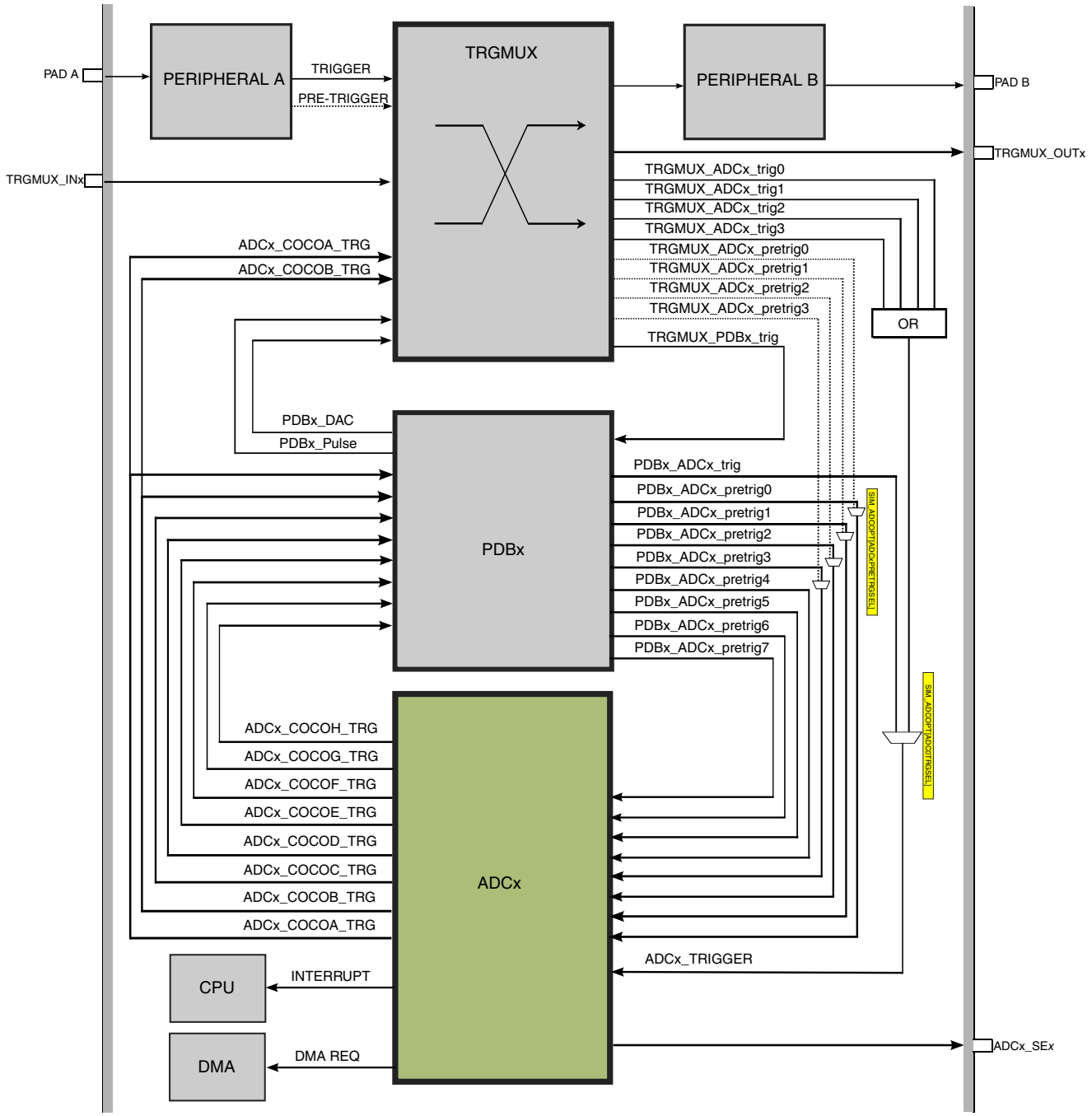


NOTE

ALTCLK2~4 are not connected on this chip.

37.1.3 Inter-connectivity Information

The ADC inter-connectivity is shown in following diagram.



37.1.4 Application-related Information

37.1.4.1 ADC Hardware Interleaved Channels

On this device, there are several special ADC channels which support hardware interleave between multiple ADCs. Taking ADC0_SE4 and ADC1_SE14 channels as an example, these two channels can work independently, but they can also be hardware interleaved as shown in the following diagram. In the hardware interleaved mode, a signal on the pin PTB0 can be sampled by both ADC0 and ADC1. The interleaved mode is enabled by SIM_CHIPCTL[ADC_INTERLEAVE_EN] bits.

The hardware interleave implementation on this device is as follows:

- ADC0_SE4 and ADC1_SE14 channels are interleaved on PTB0 pin
- ADC0_SE5 and ADC1_SE15 channels are interleaved on PTB1 pin
- ADC1_SE8 and ADC2_SE8 channels are interleaved on PTB13 pin
- ADC1_SE9 and ADC2_SE9 channels are interleaved on PTB14 pin

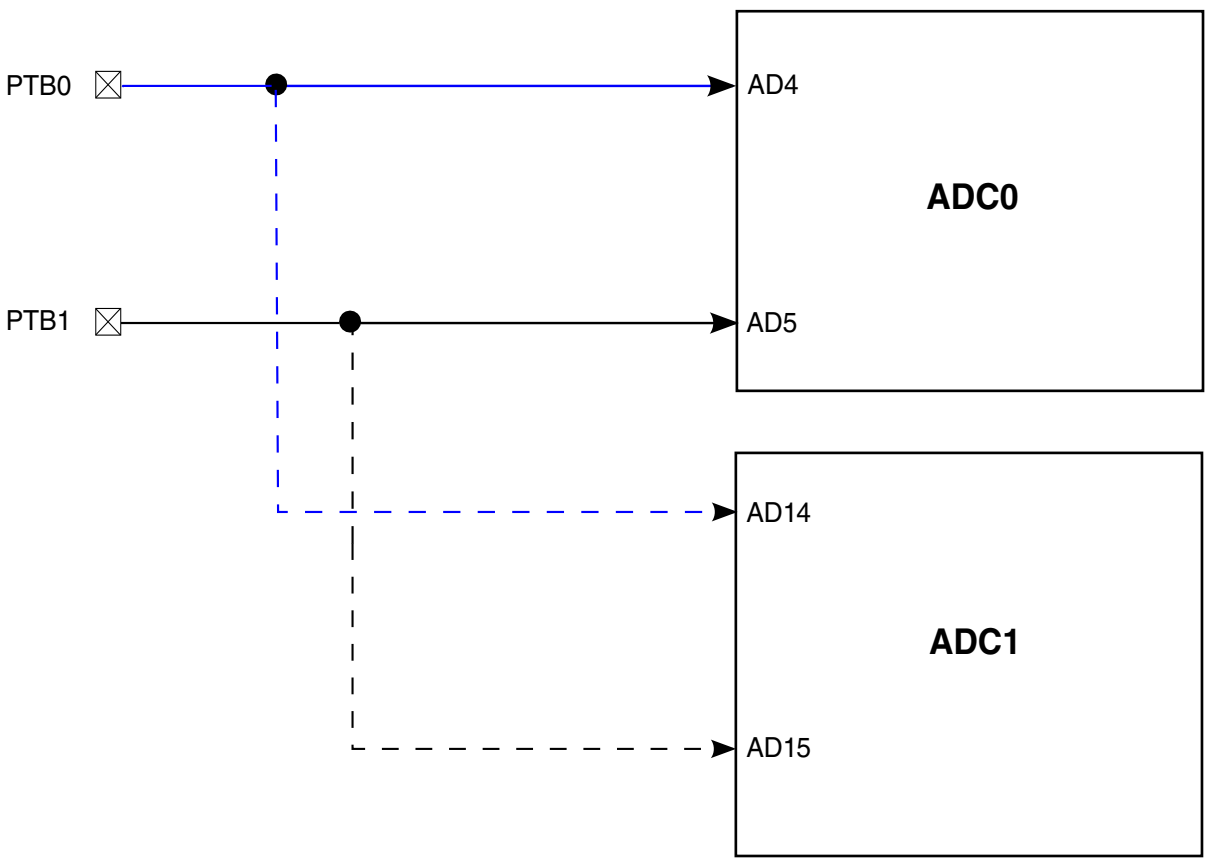


Figure 37-1. ADC0 and ADC1 hardware interleaved channels integration

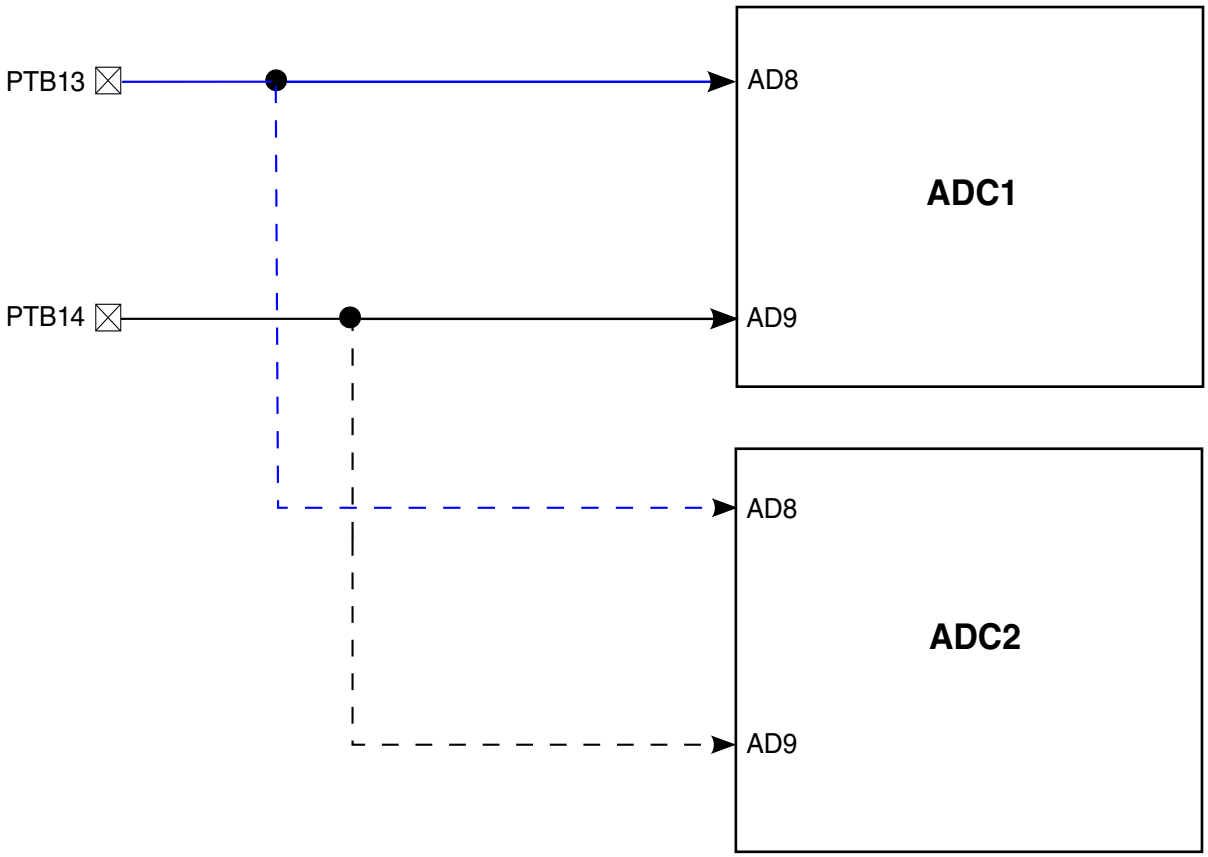


Figure 37-2. ADC1 and ADC2 hardware interleaved channels integration

37.1.4.2 ADC Reference Options

The ADC supports the following references:

- VREFH/VREFL - connected as the primary reference option

NOTE

VREFH pin on the PCB should use 3 bypass capacitors in the range: 1 µF, 100 nF and 1 nF. Capacitors should be placed to the VREFH pin as close as possible.

- Bandgap from PMC connected as the V_{ALT2} reference option. The V_{ALT2} input is also connected within the ADC module as ADC channel 27

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

37.1.4.3 ADC Trigger Sources

The ADC support multiple trigger sources. There is two kinds of trigger: pre-trigger and trigger. The pre-trigger precondition the ADC block and selects the specific data result register, before the ADC trigger is asserted. The trigger initiate the ADC conversion as soon as it's asserted. The trigger and pre-trigger sources are described as following:

- Hardware pre-triggers/triggers are connected through PDB and TRGMUX. The pre-triggers can also be controlled by software to provide flexible trigger schemes (by controlling SIM_ADCOPT[ADCxSWPRETRG] registers). Besides the hardware triggers through ADHWT, the ADC module itself also supports software trigger mode by setting SC2[ADTRG]=0. Following a write to SC1A register, a conversion is initiated.
- 3×PDB can generate triggers and pre-triggers for 3×ADC, each PDB channel will have up to 8 pre-triggers for ADC channel control, which provides an automatically trigger scheme so that the CPU involvement is not necessary.
- TRGMUX can provide triggers for each ADC, while the pre-triggers need to be controlled by software to determine relative priority. It should not trigger the ADC again before a single conversion has not completed.

The following triggers are via the TRGMUX:

- CMP out to trigger each ADC
- LPIT capable to trigger each ADC, but LPIT only supports up to 4 pre-triggers, which is limited to be used only on the ADHWTSA ~ ADHWTSD of each ADC.
- RTC capable to trigger each ADC
- LPTMR capable to trigger each ADC
- Software trigger capable to trigger each ADC

NOTE

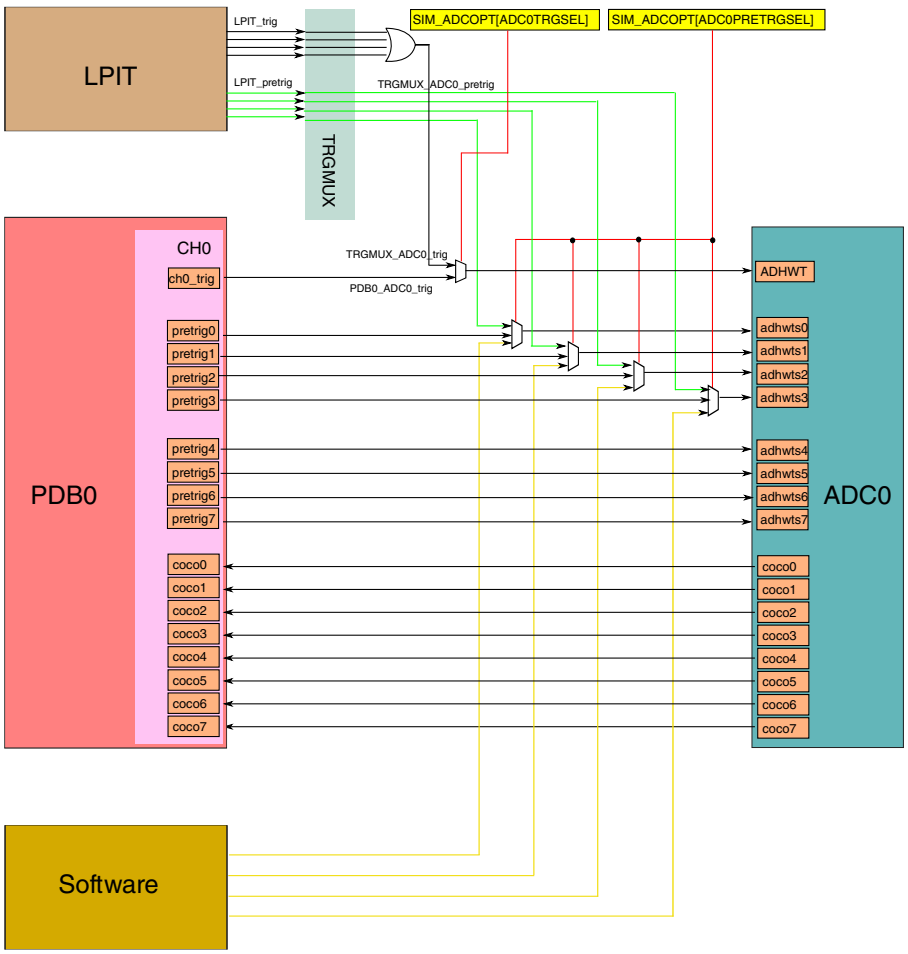
The software trigger/pre-trigger through TRGMUX, the ADC's own software trigger mode and the software pre-trigger controlled by SIM are different concepts.

Following specification and diagram are just giving an example to help understanding the ADC trigger scheme. Generally, the ADC support two kind of hardware triggering scheme:

- The default hardware triggering scheme is using PDB to trigger ADC (suggested).
- Another optional hardware triggering scheme is using TRGMUX.
- SIM_ADCOPT[ADCxTRGSEL] bit is used to control the ADC triggering source/ scheme.
  - When ADCxTRGSEL=0, the ADC pre-trigger is coming from PDB directly.
  - When ADCxTRGSEL=1, the ADC pre-trigger is coming from TRGMUX, e.g. LPIT.

NOTE

For PDB triggering, each PDB's ADC channel supports up to 8 pre-triggers. But for TRGMUX triggering, only up to 4 pre-triggers are supported.



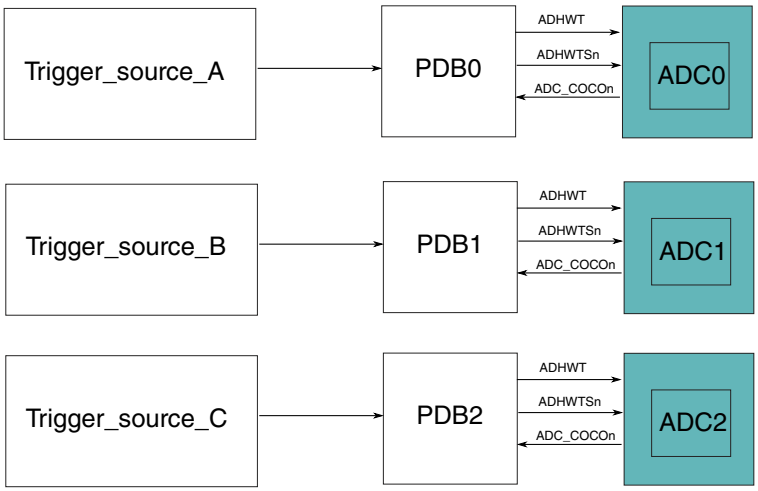
PDB triggering scheme:

PDB triggering scheme is the default and suggested trigger method for ADC. One ADC and one PDB work as one pair, the implementation on this device is: PDB0-ADC0, PDB1-ADC1, PDB2-ADC2. Here we take PDB0-ADC0 as an example to specify the triggering scheme.

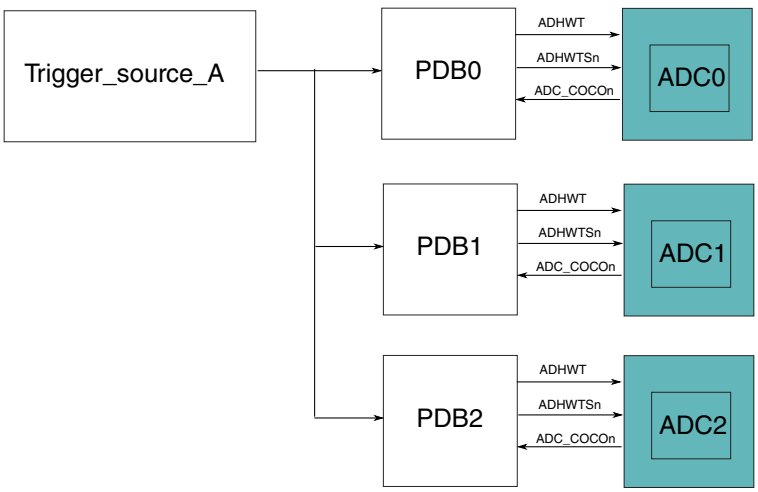
- Set SIM_ADCOPT[ADC*x*TRGSEL]=0. PDB0 channel0 is selected as ADC trigger source.
- Set SIM_ADCOPT[ADCxPRETRGSEL]=00. PDB0 pre-triggers will connect directly to ADC0 ADHWTS ports to control the channels.
- The ADC0 COCO signals are directly feed-backed to PDB0 to deactivate the PDB lock state.

Following are typical case for ADC triggering using PDB:

Case 1:



Case 2:



TRGMUX triggering scheme:

TRGMUX supports many trigger sources, here we take LPIT as an example (typical), but the trigger source can also be others which mentioned above. LPIT supports up to 4 channels, each channel have a trigger and pre-trigger.

- Set SIM_ADCOPT[ADC*x*TRGSEL]=1. TRGMUX out is selected as ADC trigger source.
- Configure TRGMUX to select LPIT triggers as ADC trigger and pre-trigger source.
- TRGMUX only supports up to 4 pre-triggers for each ADC (pre-trigger0 ~ pre-trigger3), the other pre-triggers could not be used with TRGMUX.
- Set SIM_ADCOPT[ADCxPRETRGSEL]=01. LPIT pre-triggers will connect directly to ADC0 ADHWTS ports to control the channels.
- ADC COCO is not required in this case. Software need to take care of the intermission time between each ADC conversion.
- With TRGMUX, a single LPIT could be used to trigger 3 ADCs at same time. This is one of the benefits for TRGMUX triggering, compared with PDB triggering.

NOTE

For other trigger sources other than PDB and LPIT, software engagement is required to configure ADC pre-trigger selection. That means it must select pre-trigger source from software (it is required SIM_ADCOPT[ADCxPRETRGSEL] is set to 10 in this case, to make sure that software pre-triggers connect directly to ADC0 ADHWTS ports), and which ADC channel to use (by setting ADCxSWPRETRG).

Software triggering scheme:

It also supports to configure ADC pre-trigger/trigger by software.

- By setting SC2[ADTRG]=0, ADC software trigger mode is selected. A conversion is initiated following a write to SC1A register.

NOTE

ADC software trigger mode only support SC1A and data register A.

- Configure SC2[ADTRG]=1, ADC is in hardware triggering mode. By setting SIM_ADCOPT[ADCxSWPRETRG], the pre-trigger for ADC is selected. The software trigger trough TRGMUX can trigger the ADC conversion. This mechanism supports multiple data registers.

37.2 Introduction

The 12-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

37.2.1 Features

Following are the features of the ADC module:

- Linear successive approximation algorithm with up to 12-bit resolution
- Up to 16 single-ended external analog inputs
- Output modes:

    - single-ended 12-bit, 10-bit, and 8-bit modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion modes
- 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
- 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

37.2.2 Block diagram

The following figure is the ADC module block diagram.

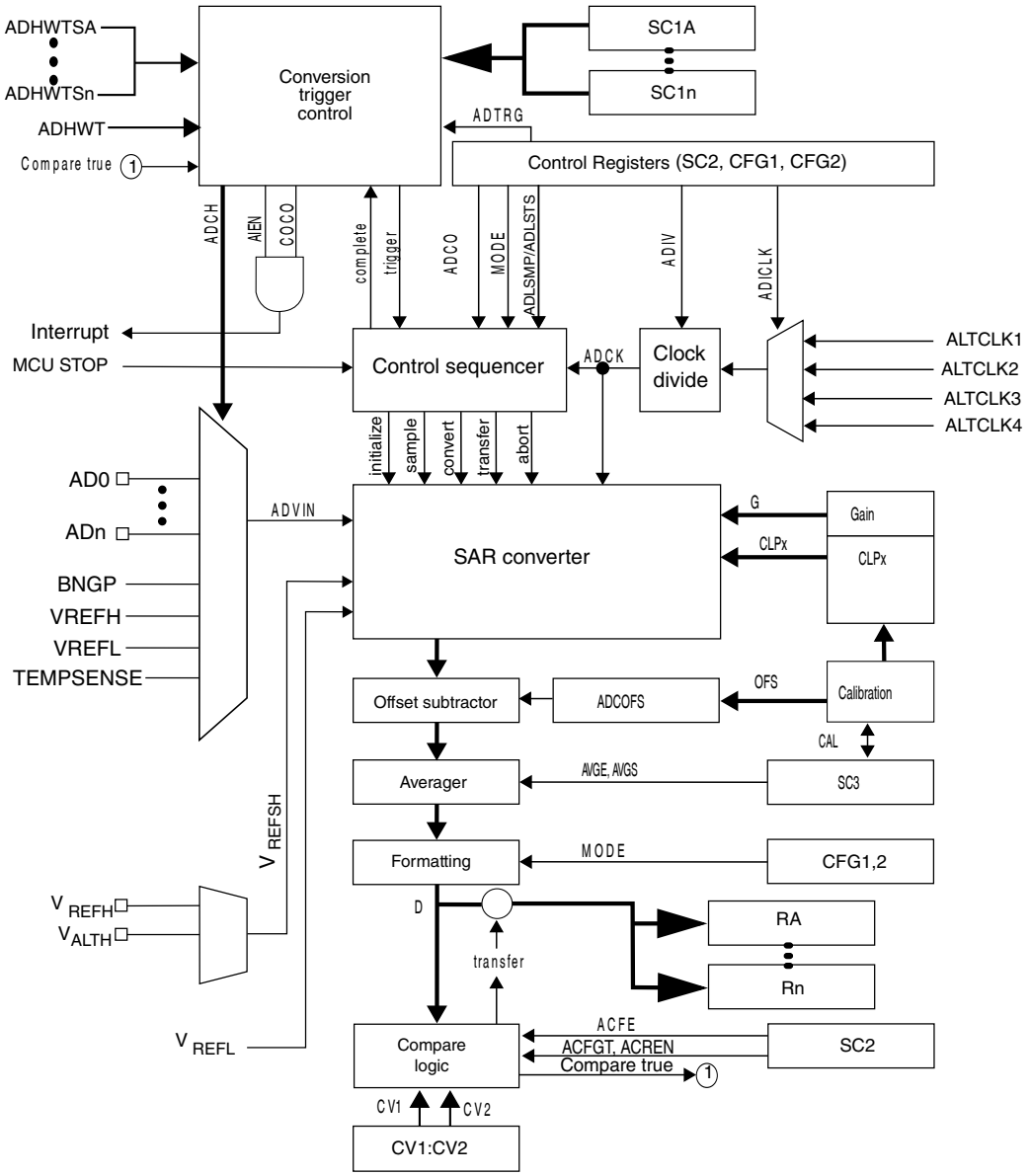


Figure 37-3. ADC block diagram

37.3 ADC signal descriptions

Each ADC module supports up to 16 single-ended inputs.

The ADC also requires four supply/reference/ground connections.

NOTE

For the number of channels supported on this device, see the chip-specific ADC information.

The ADC does not produce any output signals.

Table 37-5. ADC input signal descriptions

Signal	Description
AD*n*	Single-Ended Analog Channel Inputs
V_{REFSH}	Voltage Reference Select High
V_{REFSL}	Voltage Reference Select Low
V_{DDA}	Analog Power Supply
V_{SSA}	Analog Ground

37.3.1 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.

37.3.2 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}.

37.3.3 Voltage Reference Select

V_{REFSH} and V_{REFSL} are the high and low reference voltages for the ADC module.

The ADC can be configured to accept one of the voltage reference pairs for V_{REFSH} and V_{REFSL} by configuring V_{REFSH} as V_{REFH} or $_{VALTH}$. 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}) alternate (V_{ALTLH} and V_{REFL}). These voltage references are selected using SC2[REFSEL]. The alternate voltage reference, V_{ALTH} may select additional external pin or internal source depending on MCU configuration. See the chip configuration information on the Voltage References specific to this MCU.

In some packages, V_{REFH} is connected in the package to V_{DDA} and V_{REFL} to V_{SSA}. If externally available, the positive reference(s) may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential. V_{REFH} must never exceed V_{DDA}. Connect the ground references to the same voltage potential as V_{SSA}.

37.3.4 Analog Channel Inputs (ADx)

The ADC module supports up to 16 analog inputs. An analog input is selected for conversion through the SC1[ADCH] channel select field.

37.4 Memory map and register definitions

This section describes the ADC registers.

NOTE

The reset values of ADC Calibration and Gain registers are loaded from IFR.

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_7000	ADC Status and Control Register 1 (ADC1_SC1A)	32	R/W	0000_001Fh	37.4.1/829
4002_7004	ADC Status and Control Register 1 (ADC1_SC1B)	32	R/W	0000_001Fh	37.4.1/829
4002_7008	ADC Status and Control Register 1 (ADC1_SC1C)	32	R/W	0000_001Fh	37.4.1/829
4002_700C	ADC Status and Control Register 1 (ADC1_SC1D)	32	R/W	0000_001Fh	37.4.1/829
4002_7010	ADC Status and Control Register 1 (ADC1_SC1E)	32	R/W	0000_001Fh	37.4.1/829
4002_7014	ADC Status and Control Register 1 (ADC1_SC1F)	32	R/W	0000_001Fh	37.4.1/829
4002_7018	ADC Status and Control Register 1 (ADC1_SC1G)	32	R/W	0000_001Fh	37.4.1/829
4002_701C	ADC Status and Control Register 1 (ADC1_SC1H)	32	R/W	0000_001Fh	37.4.1/829
4002_7040	ADC Configuration Register 1 (ADC1_CFG1)	32	R/W	0000_0000h	37.4.2/832
4002_7044	ADC Configuration Register 2 (ADC1_CFG2)	32	R/W	0000_000Ch	37.4.3/833
4002_7048	ADC Data Result Registers (ADC1_RA)	32	R	0000_0000h	37.4.4/833
4002_704C	ADC Data Result Registers (ADC1_RB)	32	R	0000_0000h	37.4.4/833
4002_7050	ADC Data Result Registers (ADC1_RC)	32	R	0000_0000h	37.4.4/833
4002_7054	ADC Data Result Registers (ADC1_RD)	32	R	0000_0000h	37.4.4/833
4002_7058	ADC Data Result Registers (ADC1_RE)	32	R	0000_0000h	37.4.4/833
4002_705C	ADC Data Result Registers (ADC1_RF)	32	R	0000_0000h	37.4.4/833
4002_7060	ADC Data Result Registers (ADC1_RG)	32	R	0000_0000h	37.4.4/833

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_7064	ADC Data Result Registers (ADC1_RH)	32	R	0000_0000h	37.4.4/833
4002_7088	Compare Value Registers (ADC1_CV1)	32	R/W	0000_0000h	37.4.5/834
4002_708C	Compare Value Registers (ADC1_CV2)	32	R/W	0000_0000h	37.4.5/834
4002_7090	Status and Control Register 2 (ADC1_SC2)	32	R/W	0000_0000h	37.4.6/835
4002_7094	Status and Control Register 3 (ADC1_SC3)	32	R/W	0000_0000h	37.4.7/837
4002_7098	BASE Offset Register (ADC1_BASE_OFS)	32	R/W	0000_0040h	37.4.8/838
4002_709C	ADC Offset Correction Register (ADC1_OFS)	32	R/W	0000_0000h	37.4.9/838
4002_70A0	USER Offset Correction Register (ADC1_USR_OFS)	32	R/W	0000_0000h	37.4.10/ 839
4002_70A4	ADC X Offset Correction Register (ADC1_XOFS)	32	R/W	0000_0030h	37.4.11/ 840
4002_70A8	ADC Y Offset Correction Register (ADC1_YOFS)	32	R/W	0000_0037h	37.4.12/ 840
4002_70AC	ADC Gain Register (ADC1_G)	32	R/W	0000_02F0h	37.4.13/ 840
4002_70B0	ADC User Gain Register (ADC1_UG)	32	R/W	0000_0004h	37.4.14/ 841
4002_70B4	ADC General Calibration Value Register S (ADC1_CLPS)	32	R/W	See section	37.4.15/ 841
4002_70B8	ADC Plus-Side General Calibration Value Register 3 (ADC1_CLP3)	32	R/W	See section	37.4.16/ 842
4002_70BC	ADC Plus-Side General Calibration Value Register 2 (ADC1_CLP2)	32	R/W	See section	37.4.17/ 843
4002_70C0	ADC Plus-Side General Calibration Value Register 1 (ADC1_CLP1)	32	R/W	See section	37.4.18/ 843
4002_70C4	ADC Plus-Side General Calibration Value Register 0 (ADC1_CLP0)	32	R/W	See section	37.4.19/ 844
4002_70C8	ADC Plus-Side General Calibration Value Register X (ADC1_CLPX)	32	R/W	See section	37.4.20/ 844
4002_70CC	ADC Plus-Side General Calibration Value Register 9 (ADC1_CLP9)	32	R/W	See section	37.4.21/ 845
4002_70D0	ADC General Calibration Offset Value Register S (ADC1_CLPS_OFS)	32	R/W	0000_0000h	37.4.22/ 846
4002_70D4	ADC Plus-Side General Calibration Offset Value Register 3 (ADC1_CLP3_OFS)	32	R/W	0000_0000h	37.4.23/ 846
4002_70D8	ADC Plus-Side General Calibration Offset Value Register 2 (ADC1_CLP2_OFS)	32	R/W	0000_0000h	37.4.24/ 847
4002_70DC	ADC Plus-Side General Calibration Offset Value Register 1 (ADC1_CLP1_OFS)	32	R/W	0000_0000h	37.4.25/ 847
4002_70E0	ADC Plus-Side General Calibration Offset Value Register 0 (ADC1_CLP0_OFS)	32	R/W	0000_0000h	37.4.26/ 847
4002_70E4	ADC Plus-Side General Calibration Offset Value Register X (ADC1_CLPX_OFS)	32	R/W	0000_0440h	37.4.27/ 848

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_70E8	ADC Plus-Side General Calibration Offset Value Register 9 (ADC1_CLP9_OFS)	32	R/W	0000_0240h	37.4.28/ 848
4003_B000	ADC Status and Control Register 1 (ADC0_SC1A)	32	R/W	0000_001Fh	37.4.1/829
4003_B004	ADC Status and Control Register 1 (ADC0_SC1B)	32	R/W	0000_001Fh	37.4.1/829
4003_B008	ADC Status and Control Register 1 (ADC0_SC1C)	32	R/W	0000_001Fh	37.4.1/829
4003_B00C	ADC Status and Control Register 1 (ADC0_SC1D)	32	R/W	0000_001Fh	37.4.1/829
4003_B010	ADC Status and Control Register 1 (ADC0_SC1E)	32	R/W	0000_001Fh	37.4.1/829
4003_B014	ADC Status and Control Register 1 (ADC0_SC1F)	32	R/W	0000_001Fh	37.4.1/829
4003_B018	ADC Status and Control Register 1 (ADC0_SC1G)	32	R/W	0000_001Fh	37.4.1/829
4003_B01C	ADC Status and Control Register 1 (ADC0_SC1H)	32	R/W	0000_001Fh	37.4.1/829
4003_B040	ADC Configuration Register 1 (ADC0_CFG1)	32	R/W	0000_0000h	37.4.2/832
4003_B044	ADC Configuration Register 2 (ADC0_CFG2)	32	R/W	0000_000Ch	37.4.3/833
4003_B048	ADC Data Result Registers (ADC0_RA)	32	R	0000_0000h	37.4.4/833
4003_B04C	ADC Data Result Registers (ADC0_RB)	32	R	0000_0000h	37.4.4/833
4003_B050	ADC Data Result Registers (ADC0_RC)	32	R	0000_0000h	37.4.4/833
4003_B054	ADC Data Result Registers (ADC0_RD)	32	R	0000_0000h	37.4.4/833
4003_B058	ADC Data Result Registers (ADC0_RE)	32	R	0000_0000h	37.4.4/833
4003_B05C	ADC Data Result Registers (ADC0_RF)	32	R	0000_0000h	37.4.4/833
4003_B060	ADC Data Result Registers (ADC0_RG)	32	R	0000_0000h	37.4.4/833
4003_B064	ADC Data Result Registers (ADC0_RH)	32	R	0000_0000h	37.4.4/833
4003_B088	Compare Value Registers (ADC0_CV1)	32	R/W	0000_0000h	37.4.5/834
4003_B08C	Compare Value Registers (ADC0_CV2)	32	R/W	0000_0000h	37.4.5/834
4003_B090	Status and Control Register 2 (ADC0_SC2)	32	R/W	0000_0000h	37.4.6/835
4003_B094	Status and Control Register 3 (ADC0_SC3)	32	R/W	0000_0000h	37.4.7/837
4003_B098	BASE Offset Register (ADC0_BASE_OFS)	32	R/W	0000_0040h	37.4.8/838
4003_B09C	ADC Offset Correction Register (ADC0_OFS)	32	R/W	0000_0000h	37.4.9/838
4003_B0A0	USER Offset Correction Register (ADC0_USR_OFS)	32	R/W	0000_0000h	37.4.10/ 839
4003_B0A4	ADC X Offset Correction Register (ADC0_XOFS)	32	R/W	0000_0030h	37.4.11/ 840
4003_B0A8	ADC Y Offset Correction Register (ADC0_YOFS)	32	R/W	0000_0037h	37.4.12/ 840
4003_B0AC	ADC Gain Register (ADC0_G)	32	R/W	0000_02F0h	37.4.13/ 840
4003_B0B0	ADC User Gain Register (ADC0_UG)	32	R/W	0000_0004h	37.4.14/ 841
4003_B0B4	ADC General Calibration Value Register S (ADC0_CLPS)	32	R/W	See section	37.4.15/ 841
4003_B0B8	ADC Plus-Side General Calibration Value Register 3 (ADC0_CLP3)	32	R/W	See section	37.4.16/ 842

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_B0BC	ADC Plus-Side General Calibration Value Register 2 (ADC0_CLP2)	32	R/W	See section	37.4.17/ 843
4003_B0C0	ADC Plus-Side General Calibration Value Register 1 (ADC0_CLP1)	32	R/W	See section	37.4.18/ 843
4003_B0C4	ADC Plus-Side General Calibration Value Register 0 (ADC0_CLP0)	32	R/W	See section	37.4.19/ 844
4003_B0C8	ADC Plus-Side General Calibration Value Register X (ADC0_CLPX)	32	R/W	See section	37.4.20/ 844
4003_B0CC	ADC Plus-Side General Calibration Value Register 9 (ADC0_CLP9)	32	R/W	See section	37.4.21/ 845
4003_B0D0	ADC General Calibration Offset Value Register S (ADC0_CLPS_OFS)	32	R/W	0000_0000h	37.4.22/ 846
4003_B0D4	ADC Plus-Side General Calibration Offset Value Register 3 (ADC0_CLP3_OFS)	32	R/W	0000_0000h	37.4.23/ 846
4003_B0D8	ADC Plus-Side General Calibration Offset Value Register 2 (ADC0_CLP2_OFS)	32	R/W	0000_0000h	37.4.24/ 847
4003_B0DC	ADC Plus-Side General Calibration Offset Value Register 1 (ADC0_CLP1_OFS)	32	R/W	0000_0000h	37.4.25/ 847
4003_B0E0	ADC Plus-Side General Calibration Offset Value Register 0 (ADC0_CLP0_OFS)	32	R/W	0000_0000h	37.4.26/ 847
4003_B0E4	ADC Plus-Side General Calibration Offset Value Register X (ADC0_CLPX_OFS)	32	R/W	0000_0440h	37.4.27/ 848
4003_B0E8	ADC Plus-Side General Calibration Offset Value Register 9 (ADC0_CLP9_OFS)	32	R/W	0000_0240h	37.4.28/ 848
4003_C000	ADC Status and Control Register 1 (ADC2_SC1A)	32	R/W	0000_001Fh	37.4.1/829
4003_C004	ADC Status and Control Register 1 (ADC2_SC1B)	32	R/W	0000_001Fh	37.4.1/829
4003_C008	ADC Status and Control Register 1 (ADC2_SC1C)	32	R/W	0000_001Fh	37.4.1/829
4003_C00C	ADC Status and Control Register 1 (ADC2_SC1D)	32	R/W	0000_001Fh	37.4.1/829
4003_C010	ADC Status and Control Register 1 (ADC2_SC1E)	32	R/W	0000_001Fh	37.4.1/829
4003_C014	ADC Status and Control Register 1 (ADC2_SC1F)	32	R/W	0000_001Fh	37.4.1/829
4003_C018	ADC Status and Control Register 1 (ADC2_SC1G)	32	R/W	0000_001Fh	37.4.1/829
4003_C01C	ADC Status and Control Register 1 (ADC2_SC1H)	32	R/W	0000_001Fh	37.4.1/829
4003_C040	ADC Configuration Register 1 (ADC2_CFG1)	32	R/W	0000_0000h	37.4.2/832
4003_C044	ADC Configuration Register 2 (ADC2_CFG2)	32	R/W	0000_000Ch	37.4.3/833
4003_C048	ADC Data Result Registers (ADC2_RA)	32	R	0000_0000h	37.4.4/833
4003_C04C	ADC Data Result Registers (ADC2_RB)	32	R	0000_0000h	37.4.4/833
4003_C050	ADC Data Result Registers (ADC2_RC)	32	R	0000_0000h	37.4.4/833
4003_C054	ADC Data Result Registers (ADC2_RD)	32	R	0000_0000h	37.4.4/833
4003_C058	ADC Data Result Registers (ADC2_RE)	32	R	0000_0000h	37.4.4/833
4003_C05C	ADC Data Result Registers (ADC2_RF)	32	R	0000_0000h	37.4.4/833
4003_C060	ADC Data Result Registers (ADC2_RG)	32	R	0000_0000h	37.4.4/833

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_C064	ADC Data Result Registers (ADC2_RH)	32	R	0000_0000h	37.4.4/833
4003_C088	Compare Value Registers (ADC2_CV1)	32	R/W	0000_0000h	37.4.5/834
4003_C08C	Compare Value Registers (ADC2_CV2)	32	R/W	0000_0000h	37.4.5/834
4003_C090	Status and Control Register 2 (ADC2_SC2)	32	R/W	0000_0000h	37.4.6/835
4003_C094	Status and Control Register 3 (ADC2_SC3)	32	R/W	0000_0000h	37.4.7/837
4003_C098	BASE Offset Register (ADC2_BASE_OFS)	32	R/W	0000_0040h	37.4.8/838
4003_C09C	ADC Offset Correction Register (ADC2_OFS)	32	R/W	0000_0000h	37.4.9/838
4003_C0A0	USER Offset Correction Register (ADC2_USR_OFS)	32	R/W	0000_0000h	37.4.10/ 839
4003_C0A4	ADC X Offset Correction Register (ADC2_XOFS)	32	R/W	0000_0030h	37.4.11/ 840
4003_C0A8	ADC Y Offset Correction Register (ADC2_YOFS)	32	R/W	0000_0037h	37.4.12/ 840
4003_C0AC	ADC Gain Register (ADC2_G)	32	R/W	0000_02F0h	37.4.13/ 840
4003_C0B0	ADC User Gain Register (ADC2_UG)	32	R/W	0000_0004h	37.4.14/ 841
4003_C0B4	ADC General Calibration Value Register S (ADC2_CLPS)	32	R/W	See section	37.4.15/ 841
4003_C0B8	ADC Plus-Side General Calibration Value Register 3 (ADC2_CLP3)	32	R/W	See section	37.4.16/ 842
4003_C0BC	ADC Plus-Side General Calibration Value Register 2 (ADC2_CLP2)	32	R/W	See section	37.4.17/ 843
4003_C0C0	ADC Plus-Side General Calibration Value Register 1 (ADC2_CLP1)	32	R/W	See section	37.4.18/ 843
4003_C0C4	ADC Plus-Side General Calibration Value Register 0 (ADC2_CLP0)	32	R/W	See section	37.4.19/ 844
4003_C0C8	ADC Plus-Side General Calibration Value Register X (ADC2_CLPX)	32	R/W	See section	37.4.20/ 844
4003_C0CC	ADC Plus-Side General Calibration Value Register 9 (ADC2_CLP9)	32	R/W	See section	37.4.21/ 845
4003_C0D0	ADC General Calibration Offset Value Register S (ADC2_CLPS_OFS)	32	R/W	0000_0000h	37.4.22/ 846
4003_C0D4	ADC Plus-Side General Calibration Offset Value Register 3 (ADC2_CLP3_OFS)	32	R/W	0000_0000h	37.4.23/ 846
4003_C0D8	ADC Plus-Side General Calibration Offset Value Register 2 (ADC2_CLP2_OFS)	32	R/W	0000_0000h	37.4.24/ 847
4003_C0DC	ADC Plus-Side General Calibration Offset Value Register 1 (ADC2_CLP1_OFS)	32	R/W	0000_0000h	37.4.25/ 847
4003_C0E0	ADC Plus-Side General Calibration Offset Value Register 0 (ADC2_CLP0_OFS)	32	R/W	0000_0000h	37.4.26/ 847
4003_C0E4	ADC Plus-Side General Calibration Offset Value Register X (ADC2_CLPX_OFS)	32	R/W	0000_0440h	37.4.27/ 848

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_C0E8	ADC Plus-Side General Calibration Offset Value Register 9 (ADC2_CLP9_OFS)	32	R/W	0000_0240h	37.4.28/ 848

37.4.1 ADC Status and Control Register 1 (ADC*x*_SC1*n*)

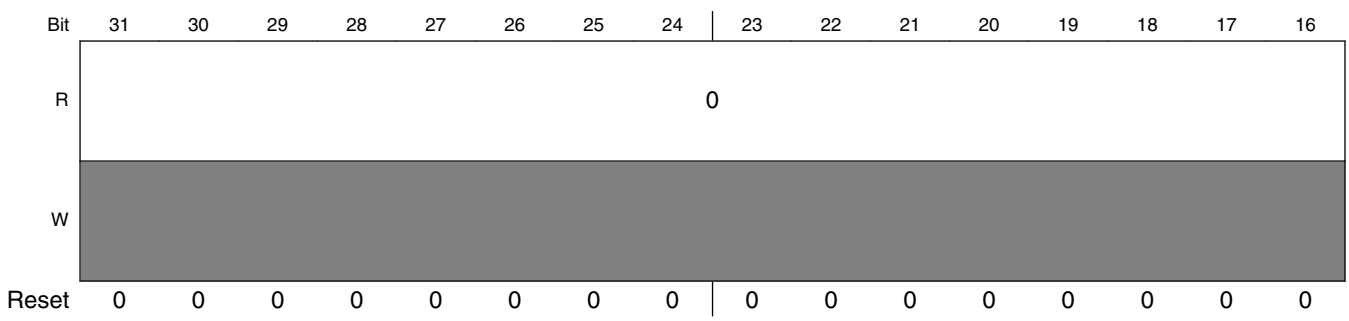
SC1A is used for both software and hardware trigger modes of operation.

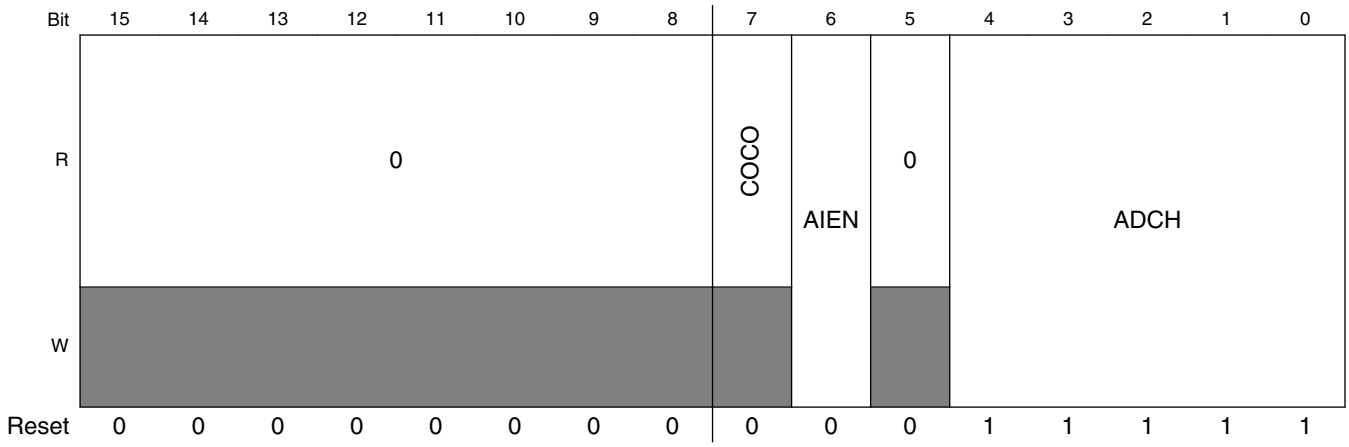
At any one point in time, only one of the SC1*n* registers is actively controlling ADC sequential conversions. Updating SC1A while SC1*n* is actively controlling a conversion is allowed, and vice versa for any of the SC1*n* registers specific to this MCU.

Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode (when SC2[ADTRG] = 0), writes to SC1A initiate a new conversion. This is valid for all values of SC1A[ADCH] other than 11111 (module disabled)."

Writing any of the SC1*n* registers while that specific SC1*n* register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1*n* registers are used for software trigger operation and therefore writes to the SC1B–SC1*n* registers do not initiate a new conversion.

Address: Base address + 0h offset + (4d × i), where i=0d to 7d





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 one or more of the following is true: • The compare function is disabled • SC2[ACFE]=0 and the hardware average function is disabled • SC3[AVGE]=0 If the compare result is true, then COCO is set upon completion of a conversion if one or more of the following is true: • The compare function is enabled • SC2[ACFE]=1 COCO is set upon completion of the selected number of conversions (determined by AVGS) if one or more of the following is true: • The hardware average function is enabled • SC3[AVGE]=1 COCO in SC1A is also set at the completion of a calibration sequence. COCO is cleared when one of the following is true: • The respective SC1n register is written • 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.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

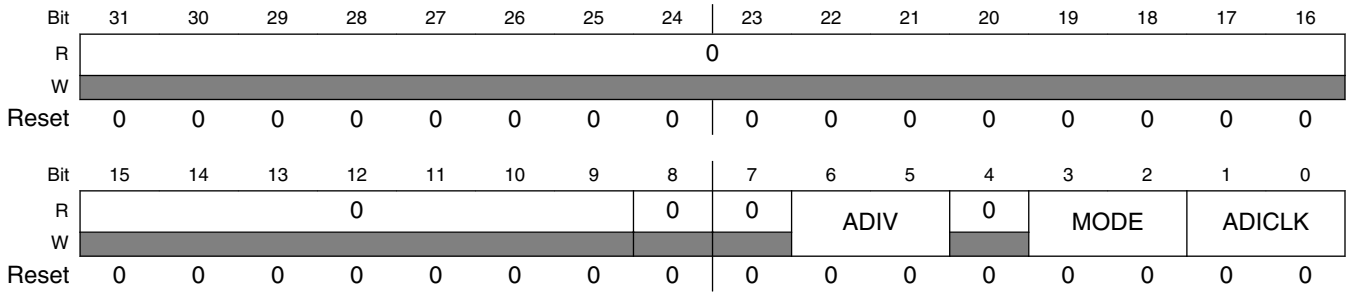
ADC*x*_SC1*n* field descriptions (continued)

Field	Description
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 chip. For the actual ADC channel assignments for your device, see the chip-specific information. The successive approximation converter subsystem is turned off when the channel bits are all set (i.e. ADCH set to all 1s). 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 Exernal channel 0 is selected as input. 00001 Exernal channel 1 is selected as input. 00010 Exernal channel 2 is selected as input. 00011 Exernal channel 3 is selected as input. 00100 Exernal channel 4 is selected as input. 00101 Exernal channel 5 is selected as input. 00110 Exernal channel 6 is selected as input. 00111 Exernal channel 7 is selected as input. 01000 Exernal channel 8 is selected as input. 01001 Exernal channel 9 is selected as input. 01010 Exernal channel 10 is selected as input. 01011 Exernal channel 11 is selected as input. 01100 Exernal channel 12 is selected as input. 01101 Exernal channel 13 is selected as input. 01110 Exernal channel 14 is selected as input. 01111 Exernal channel 15 is selected as input. 10000 Reserved 10001 Reserved 10010 Exernal channel 18 is selected as input. 10011 Exernal channel 19 is selected as input. 10100 Reserved. 10101 Internal channel 0 is selected as input. 10110 Internal channel 1 is selected as input. 10111 Internal channel 2 is selected as input. 11000 Reserved 11001 Reserved 11010 Temp Sensor 11011 Band Gap 11100 Internal channel 3 is selected as input. 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

37.4.2 ADC Configuration Register 1 (ADCx_CFG1)

Configuration Register 1 (CFG1) selects the mode of operation, clock source, clock divide.

Address: Base address + 40h offset



ADCx_CFG1 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 MODE	Conversion mode selection Selects the ADC resolution. 00 8-bit conversion. 01 12-bit conversion. 10 10-bit conversion. 11 Reserved
ADICLK	Input Clock Select Selects the input clock source to generate the internal clock, ADCK. See the clock distribution/clocking chapter of your device for details on which alternate clocks are supported. 00 Alternate clock 1 (ADC_ALTCLK1) 01 Alternate clock 2 (ADC_ALTCLK2)

Table continues on the next page...

ADCx_CFG1 field descriptions (continued)

Field	Description
	10 Alternate clock 3 (ADC_ALTCLK3) 11 Alternate clock 4 (ADC_ALTCLK4)

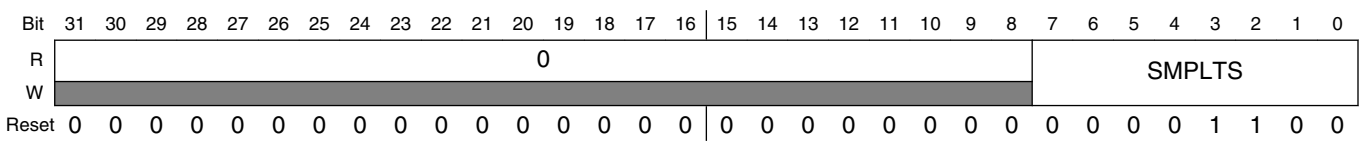
37.4.3 ADC Configuration Register 2 (ADCx_CFG2)

Configuration Register 2 (CFG2) selects the long sample time duration during long sample mode.

NOTE

Writing 0 is not supported on this register.

Address: Base address + 44h offset



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.
SMPLTS	Sample Time Select Selects a sample time of 2 to 256 ADCK clock cycles. The value written to this register field is the desired sample time minus 1. A sample time of 1 is not supported. Allows higher impedance inputs to be accurately sampled or conversion speed to be maximized 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.

37.4.4 ADC Data Result Registers (ADCx_R*n*)

R*n* contains the result of an ADC conversion of the channel selected by the corresponding SC1A:SC1*n*. For every status and channel control register, there is a corresponding data result register.

Unused bits in R*n* are cleared.

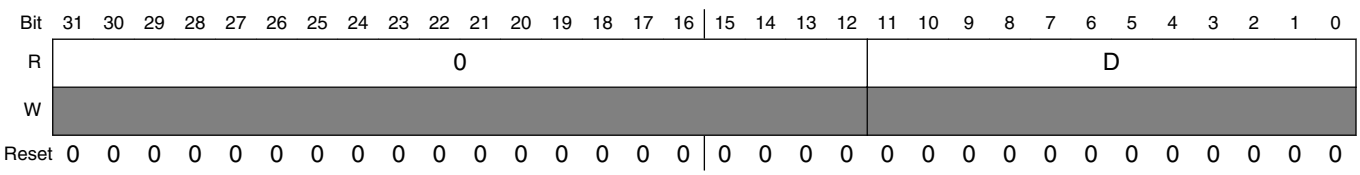
The following table describes the behavior of the data result registers in the different modes of operation.

Table 37-6. Data result register description

Conversion mode	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
12-bit single-ended	D												Unsigned right-justified
10-bit single-ended	0		D										
8-bit single-ended	0				D								

D: Data. The data result registers are read-only; writing to these registers generates a transfer error.

Address: Base address + 48h offset + (4d × i), where i=0d to 7d



ADC*x*_R*n* field descriptions

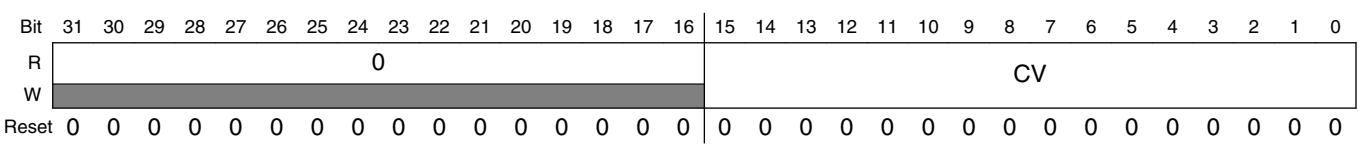
Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
D	Data result

37.4.5 Compare Value Registers (ADC*x*_CV*n*)

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. Therefore, the compare function uses only the CVn fields that are related to the ADC mode of operation.

CV2 is used only when the compare range function is enabled, that is, SC2[ACREN]=1.

Address: Base address + 88h offset + (4d × i), where i=0d to 1d



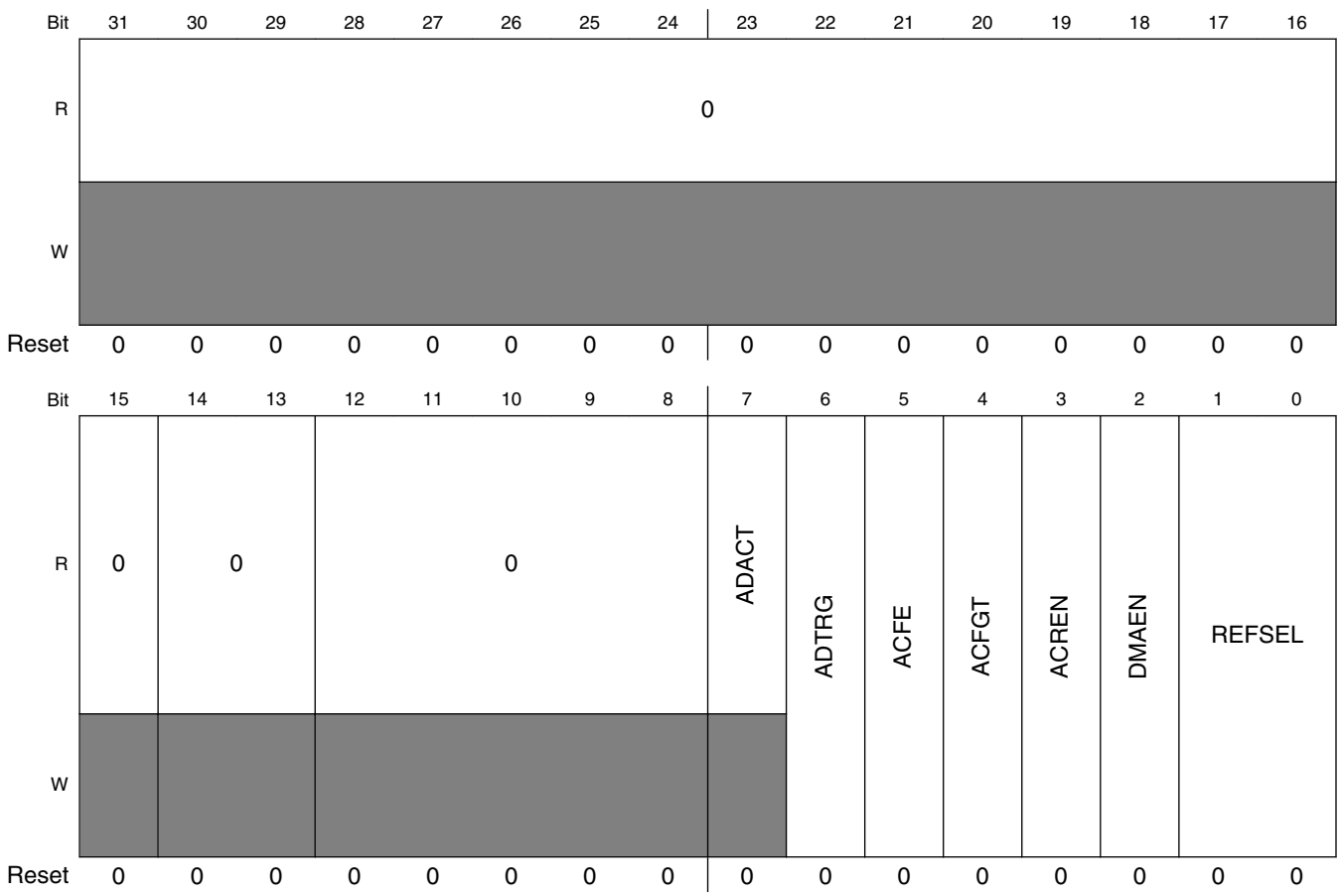
ADCx_CVn field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CV	Compare Value.

37.4.6 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: Base address + 90h offset



ADCx_SC2 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

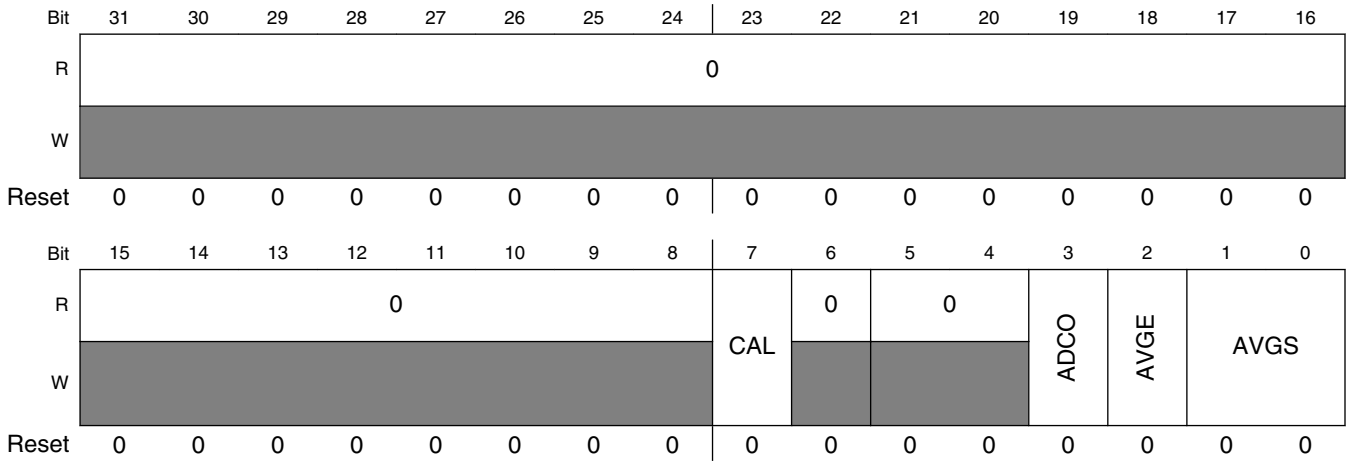
ADCx_SC2 field descriptions (continued)

Field	Description
14–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 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 triggers can be selected: • 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 CV1 and CV2 based upon the value of ACREN. ACFE must be set for ACFGT to have any effect. See Table 37-8 "Compare modes" for further details.
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. See Table 37-8 "Compare modes" for further details.
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 , which is indicated when any SC1n[COCO] flag 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} 01 Alternate reference voltage, that is, V_{ALTH}. This voltage may be additional external pin or internal source depending on the MCU configuration. See the chip configuration information for details specific to this MCU. 10 Reserved 11 Reserved

37.4.7 Status and Control Register 3 (ADCx_SC3)

The Status and Control Register 3 (SC3) controls the calibration, continuous conversion, and hardware averaging functions of the ADC module.

Address: Base address + 94h offset



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 When CAL=1, the ADC begins the calibration sequence. This field stays set while the calibration is in progress and is cleared when the calibration sequence is completed. After it is started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid. Setting CAL will abort any current conversion. **NOTE:** For calibration, it is mandatory to use averaging and average number 32. **NOTE:** If several ADCs are on a device, they should be calibrated sequentially. No parallel calibrations of ADCs are allowed because they will disturb each other.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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 will be performed (or one set of conversions, if AVGE is set) after a conversion is initiated. 1 Continuous conversions will be performed (or continuous sets of conversions, if AVGE is set) after a conversion is initiated.
2 AVGE	Hardware Average Enable

Table continues on the next page...

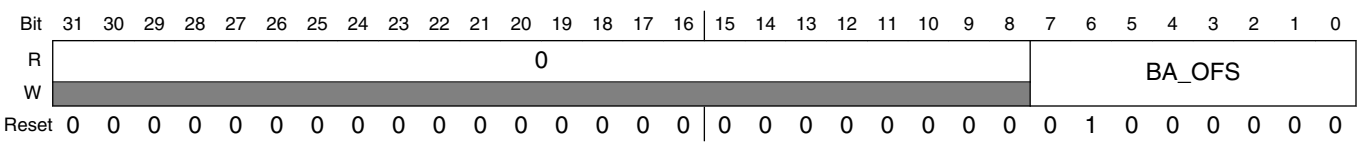
ADCx_SC3 field descriptions (continued)

Field	Description
	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.

37.4.8 BASE Offset Register (ADCx_BASE_OFS)

The BASE Offset Register (BASE_OFS) contains the offset value used by the calibration algorithm to determine the Offset Calibration Value (OFS).

Address: Base address + 98h offset



ADCx_BASE_OFS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
BA_OFS	Base Offset Error Correction Value

37.4.9 ADC Offset Correction Register (ADCx_OFS)

The ADC Offset Correction Register (OFS) contains the calibration-generated offset error correction value (OFS). The value in BA_OFS is used in the calibration algorithm to calculate the offset correction value that gets stored in the OFS register. 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.

NOTE

If offset register is set to a negative value and it is lower than or equal to 0xFFF8, the ADC will not result code 0. If offset register is set to a negative value and it is lower than or equal to 0xFFF0, the ADC will not result code 1.

Address: Base address + 9Ch offset

Bit	31	30	29	28	27	26	25	24	23	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	0	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

37.4.10 USER Offset Correction Register (ADCx_USR_OFS)

The ADC USER Offset Correction Register (USR_OFS) contains the user defined offset error correction value used in the conversion result error correction algorithm.

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	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																								USR_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	0	0	0

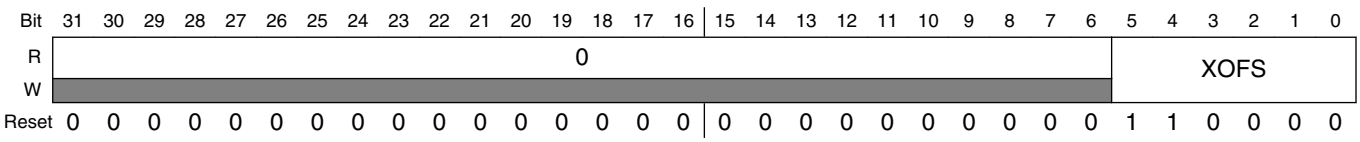
ADCx_USR_OFS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
USR_OFS	USER Offset Error Correction Value

37.4.11 ADC X Offset Correction Register (ADCx_XOFS)

The ADC X Offset Correction Register (XOFS) contains the X offset used in the conversion result error correction algorithm.

Address: Base address + A4h offset



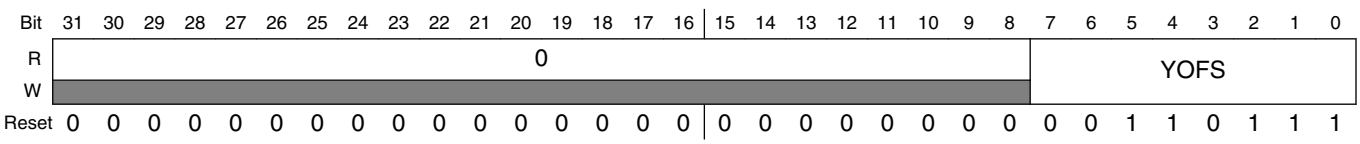
ADCx_XOFS field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
XOFS	X offset error correction value

37.4.12 ADC Y Offset Correction Register (ADCx_YOFS)

The ADC Y Offset Correction Register (YOFS) contains the Y offset used in the conversion result error correction algorithm.

Address: Base address + A8h offset



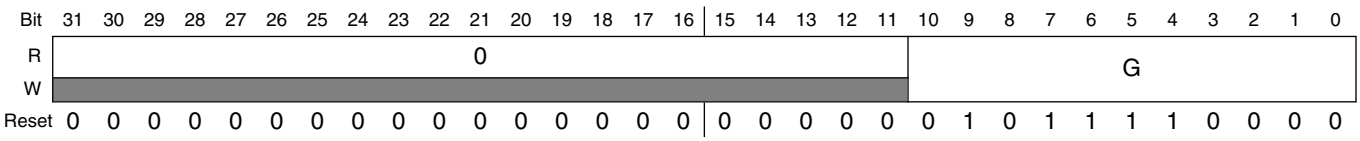
ADCx_YOFS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
YOFS	Y offset error correction value

37.4.13 ADC Gain Register (ADCx_G)

The Gain Register (G) contains the gain error correction for the overall conversion. G, a 11-bit real number in binary format, is the gain adjustment factor. This register value is determined and uploaded by the calibration algorithm.

Address: Base address + ACh offset



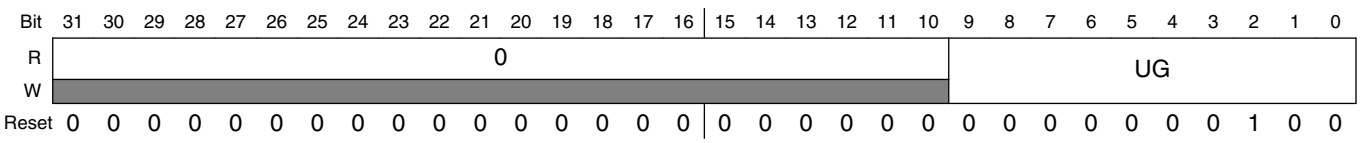
ADCx_G field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
G	Gain error adjustment factor for the overall conversion

37.4.14 ADC User Gain Register (ADCx_UG)

The User Gain Register (UG) contains the user gain error correction. It allows you to adjust the final calibration gain value.

Address: Base address + B0h offset



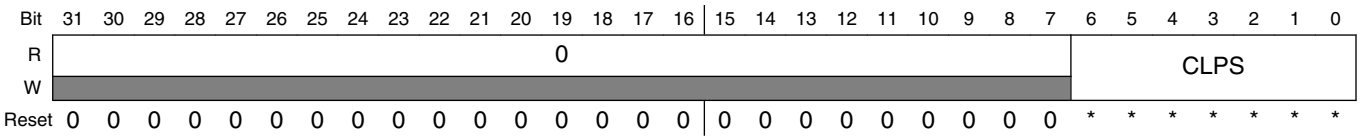
ADCx_UG field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
UG	User gain error correction value

37.4.15 ADC General Calibration Value Register S (ADCx_CLPS)

The General Calibration Value Registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven signed calibration values of varying widths in two's complement format. 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.

Address: Base address + B4h offset



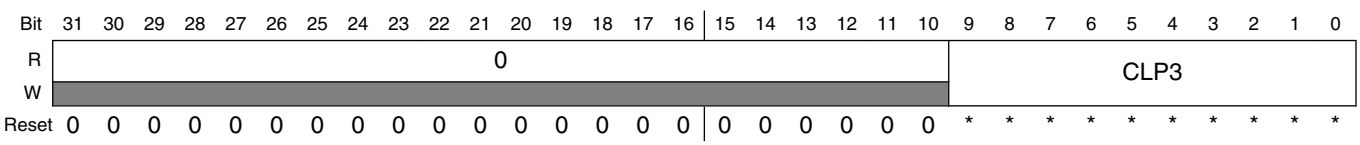
* Notes:
- CLPS field: Reset values are loaded out of IFR.

ADCx_CLPS field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPS	Calibration Value

37.4.16 ADC Plus-Side General Calibration Value Register 3 (ADCx_CLP3)

Address: Base address + B8h offset



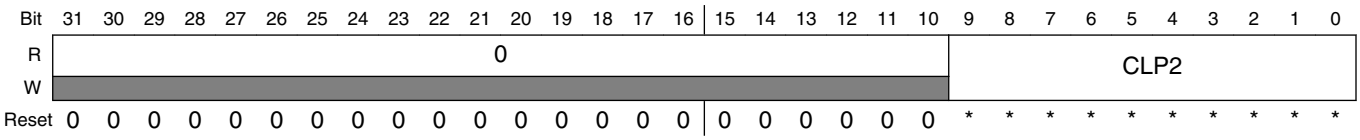
* Notes:
- CLP3 field: Reset values are loaded out of IFR.

ADCx_CLP3 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP3	Calibration Value

37.4.17 ADC Plus-Side General Calibration Value Register 2 (ADCx_CLP2)

Address: Base address + BCh offset



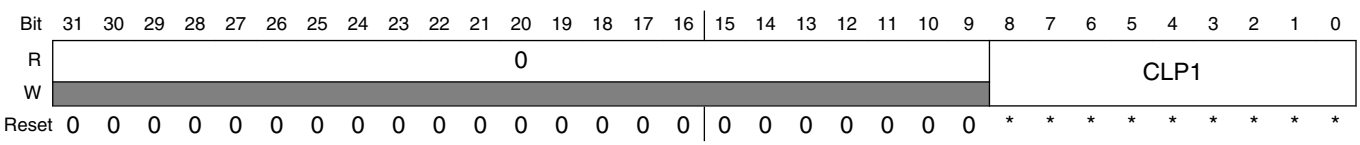
* Notes:
- CLP2 field: Reset values are loaded out of IFR.

ADCx_CLP2 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP2	Calibration Value

37.4.18 ADC Plus-Side General Calibration Value Register 1 (ADCx_CLP1)

Address: Base address + C0h offset



* Notes:
- CLP1 field: Reset values are loaded out of IFR.

ADCx_CLP1 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP1	Calibration Value

37.4.19 ADC Plus-Side General Calibration Value Register 0 (ADCx_CLP0)

Address: Base address + C4h offset

Bit	31	30	29	28	27	26	25	24	23	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	*	*	*	*	*	*	*	*

* Notes:
- CLP0 field: Reset values are loaded out of IFR.

ADCx_CLP0 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP0	Calibration Value

37.4.20 ADC Plus-Side General Calibration Value Register X (ADCx_CLPX)

Address: Base address + C8h 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	CLPX						
W																
Reset	0	0	0	0	0	0	0	0	0	*	*	*	*	*	*	*

* Notes:
- CLPX field: Reset values are loaded out of IFR.

ADCx_CLPX field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

ADCx_CLPX field descriptions (continued)

Field	Description
7 Reserved	This field is reserved.
CLPX	Calibration Value

37.4.21 ADC Plus-Side General Calibration Value Register 9 (ADCx_CLP9)

Address: Base address + CCh 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	CLP9						
W																
Reset	0	0	0	0	0	0	0	0	0	*	*	*	*	*	*	*

* Notes:
- CLP9 field: Reset values are loaded out of IFR.

ADCx_CLP9 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved.
CLP9	Calibration Value

37.4.22 ADC General Calibration Offset Value Register S (ADCx_CLPS_OFS)

Address: Base address + D0h offset

Bit	31	30	29	28	27	26	25	24	23	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_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	0	0	0

ADCx_CLPS_OFS field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPS_OFS	CLPS Offset Capacitor offset correction value

37.4.23 ADC Plus-Side General Calibration Offset Value Register 3 (ADCx_CLP3_OFS)

Address: Base address + D4h offset

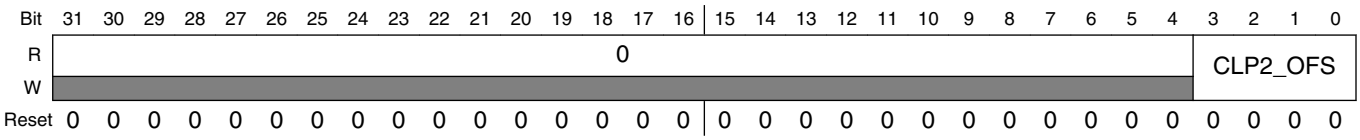
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																												CLP3_OFS			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CLP3_OFS field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP3_OFS	CLP3 Offset Capacitor offset correction value

37.4.24 ADC Plus-Side General Calibration Offset Value Register 2 (ADCx_CLP2_OFS)

Address: Base address + D8h offset

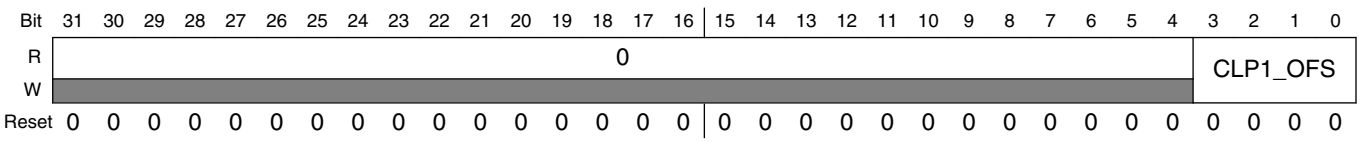


ADCx_CLP2_OFS field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP2_OFS	CLP2 Offset Capacitor offset correction value

37.4.25 ADC Plus-Side General Calibration Offset Value Register 1 (ADCx_CLP1_OFS)

Address: Base address + DCh offset

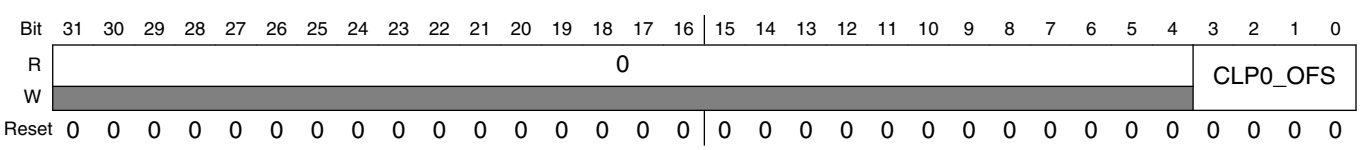


ADCx_CLP1_OFS field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP1_OFS	CLP1 Offset Capacitor offset correction value

37.4.26 ADC Plus-Side General Calibration Offset Value Register 0 (ADCx_CLP0_OFS)

Address: Base address + E0h offset

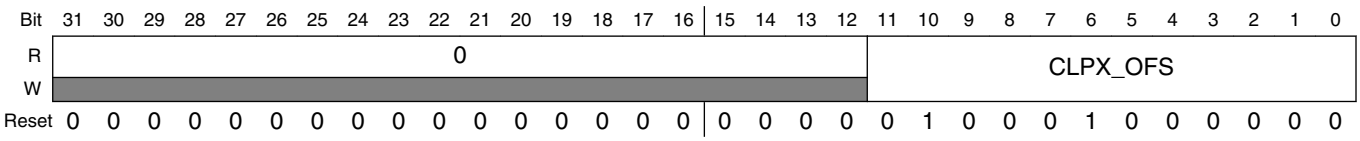


ADCx_CLP0_OFS field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP0_OFS	CLP0 Offset Capacitor offset correction value

37.4.27 ADC Plus-Side General Calibration Offset Value Register X (ADCx_CLPX_OFS)

Address: Base address + E4h offset

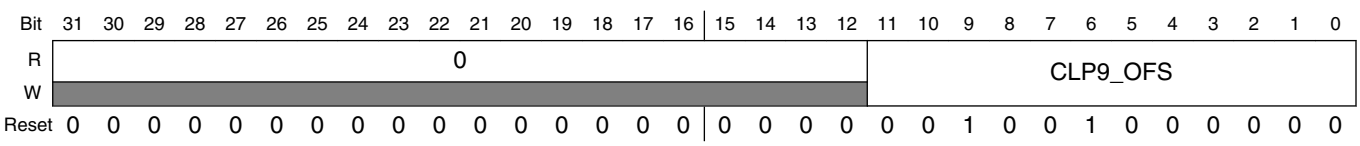


ADCx_CLPX_OFS field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPX_OFS	CLPX Offset Capacitor offset correction value

37.4.28 ADC Plus-Side General Calibration Offset Value Register 9 (ADCx_CLP9_OFS)

Address: Base address + E8h offset



ADCx_CLP9_OFS field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP9_OFS	CLP9 Offset Capacitor offset correction value

37.5 Functional description

The ADC module is disabled during reset, 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 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.

37.5.1 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].

- ALTCLK*x*: As defined for this MCU. See the chip configuration information. Conversions are possible using ALTCLK*x* as the input clock source while the MCU is in Normal Stop mode. ALTCLK1 is the default selection following reset.

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. The ADC bus clock frequency must be greater than or equal to the ADC ALT clock frequency. Please refer to the device datasheet for the ADC specifications.

37.5.2 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}). These voltage references are selected using SC2[REFSEL]. The alternate V_{ALTH} voltage reference may select additional external pin or internal source depending on MCU configuration. See the chip configuration information for the voltage references specific to this MCU.

37.5.3 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, ADHWTSn, has occurred. If a conversion is in progress when a rising edge of a trigger occurs, the rising edge is ignored. In continuous conversionn 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 an 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 will cause unpredictable 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.

37.5.4 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

37.5.4.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, 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 that are 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 cause unpredictable 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.

37.5.4.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, Rn, as indicated in the following table.

Table 37-7. Indication of conversion completion

Compare functions	Hardware averaging	Conversion status	Is SC1n[COCO] set to 1, and is the conversion result transferred into the data result registers?
Disabled	Disabled	Not completed	No
Disabled	Disabled	Completed	Yes
Disabled	Enabled	Not completed	No
Disabled	Enabled	Completed	Yes, if the last of the selected number of conversions is completed
Enabled	Disabled	Not completed	No
Enabled	Disabled	Completed	Yes, if the compare condition is true
Enabled	Enabled	Not completed	No
Enabled	Enabled	Completed	Yes, 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.

37.5.4.3 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.

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, RA and Rn return to their reset states.

37.5.4.4 Power control

The ADC module remains in its Idle state until a conversion is initiated. The Idle state implies that ADC conversion routine is held in reset.

37.5.4.5 Sample time and total conversion time

The total conversion time depends upon:

- The sample time as determined by CFG2[SMPLTS]
- The MCU bus frequency
- The conversion mode, as determined by CFG1[MODE]
- The frequency of the conversion clock, that is, f_{ADCK}.

After the module becomes active, sampling of the input begins.

1. CFG2[SMPLTS] selects 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.

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]. To calculate total conversion time the following formula is applied:

ADC TOTAL CONVERSION TIME = Sample Phase Time (set by SMPLTS+1) + Hold Phase (1 ADC Cycle) + Compare Phase Time (8-bit Mode=20 ADC Cycles, 10-bit Mode=24 ADC Cycles, 12-bit Mode=28 ADC Cycles) + Single or First continuous time adder (5 ADC cycles + 5 bus clock cycles)

37.5.4.6 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 after 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 mode. The ADC interrupt wakes the MCU when the hardware average is completed if SC1n[AIEN] is set.

37.5.5 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 37-8. Compare modes

SC2[ACFGT]	SC2[ACREN]	CV1 relative to CV2	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.

Table continues on the next page...

Table 37-8. Compare modes (continued)

SC2[ACFGT]	SC2[ACREN]	CV1 relative to CV2	Function	Compare mode description
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 mode. The ADC interrupt wakes the MCU when the compare condition is met.

37.5.6 Calibration function

The ADC is equipped with a calibration mechanism to provide high accuracy as specified in the data sheet. **It is mandatory to calibrate the ADC after power up or reset. Not doing this can result in ADC conversion results with lower than specified accuracy.**

In order to calibrate the ADC correctly, the following has to be done:

- On startup, wait until the reference voltage (VREFH) has stabilized.
- ADC has to be recalibrated after each system reset.
- Calibrate only one ADC instance at a time. So, when calibrating instance ADC0, the instances ADC1, ADC2, etc. are required to be idle.
- Set ADCK (ADC clock) to half the maximum specified frequency.

- Before starting calibration, the calibration registers (CLPS, CLP3, CLP2, CLP1, CLP0, CLPX, and CLP9) must be cleared by writing 0x0 into them.
- Start ADC calibration by writing ADC_SC3 register with: CAL=1, AVGE=1, AVGS=11.
- Wait for calibration to finish. This will be indicated by conversion complete flag (COCO in ADC_SC1n).
- Now you can run ADC conversions with high accuracy in your application. Please make sure to reconfigure the ADCK clock speed and reconfigure AVGE and AVGS to your desired settings. (Maximum clock speed and no use of hardware averaging is possible.)

The total calibration conversion time is: 12 * (# of AVERAGE * [Sample time (sample +1) + 1 cycle for hold + 34 cycles for compare phase]) + 1st conversion synchronization (~5 ADC cycles + 5 IPG clocks).

For high accuracy of the ADC (as specified in data sheet) on your application board (PCB), the following requirements should be met:

- Bypass caps between VREFH and VREFL. Suggested cap sizes: 1 nF, 100 nF, 10 µF.
- Place caps on PCB as close as possible to the device pins VREFH and VREFL.
- Bypass caps between VDDA and VSSA. Suggested cap sizes: 1 nF, 100 nF, 10 µF.
- Place caps on PCB as close as possible to the device pins VDDA and VSSA.
- Routing of VDDA, VSSA, VREFH, and VREFL on PCB:
  - Low impedance between the bypass caps and the MCU pins.
  - Keep routing distant from noisy signal routes like switching I/Os.

37.5.7 User-defined offset function

OFS is a two's-complement, left-justified register that contains the calibration-generated offset error correction 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.

The formatting of 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.

OFS is automatically set according to calibration requirements after the self-calibration sequence is done, that is, SC3[CAL] is cleared. You can write to OFS to override the calibration result if desired. If you write an OFS value that is different from the calibration value, the ADC error specifications may not be met. You should store the value generated by the calibration function in memory before overwriting with a user-specified value.

Note

There is an effective limit to the values of offset that you can set. If the magnitude of the offset is too high, the results of the conversions will cap off at the limits.

You can use the offset calibration function to remove application offsets or DC bias values. USR_OFS may be written with a number in two'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 two'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.

37.5.8 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 Alternate Clock sources 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.

37.5.9 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.

37.5.9.1 Normal Stop mode with Alternate clock sources enabled

If Alternate clock source selected for the conversion clock is enabled, 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.

37.6 Usage Guide

37.6.1 ADC module initialization sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is as below:

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.
3. Update SC2 to select the conversion trigger, hardware or software, and compare function options, if enabled.

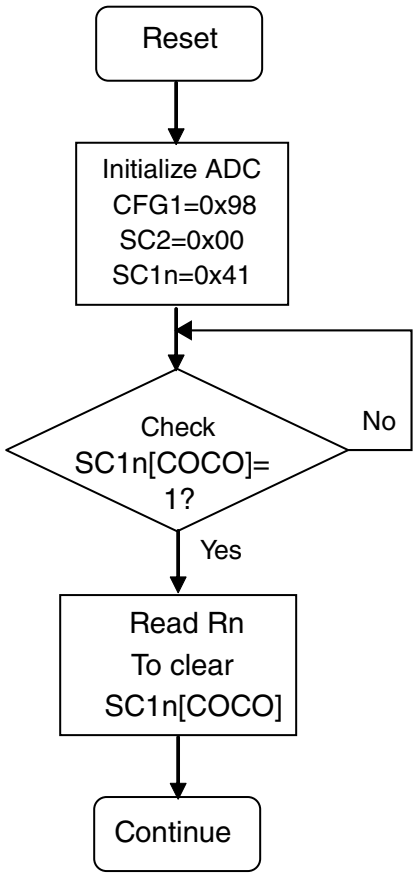
4. Update SC3 to select whether conversions will be continuous or completed only once (ADCO) and whether to perform hardware averaging.
5. Update SC1:SC1*n* registers to enable or disable conversion complete interrupts.

Also, select the input channel which can be used to perform conversions.

37.6.2 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.

```
ADC_CFG1 = ADC_CFG1_ADLPC_MASK |
ADC_CFG1_ADLSMP_MASK | ADC_CFG1_MODE(0x10);
// 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.
ADC_SC2 = 0x00;
// 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 ).
ADC_SC1A = ADC_SC1_AIEN_MASK | ADC_SC1_ADCH(0x1);
// 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.
ADC_RA = 0xxx
// Holds results of conversion.
ADC_CV = 0xxx
// Holds compare value when compare function enabled.
```



37.6.3 Calibration

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. Not doing this can result in ADC conversion results with lower than specified accuracy.

In order to calibrate ADC correctly the following steps have to be done:

- On startup, wait until reference voltage (VREFH/VREFL) has stabilized, use 3 bypass capacitance in the range: 1 μF, 100 nF and 1 nF.
- Calibrate only one ADC instance at a time, no parallel calibration of ADCs because they will disturb each other.
- Set ADCK (ADC clock) to half the maximum specified frequency, e.g. 25 MHz.
- Start ADC calibration by writing ADC_SC3 register with: CAL=1, AVGE=1, AVGS=11.

- Wait for calibration to finish. This will be indicated by conversion complete flag (COCO in ADC_SC1A).
- Run ADC conversions with high accuracy in your application. Make sure to re-configure ADCK clock speed and to re-configure AVGE and AVGS to the desired settings.

For more detailed information about calibration guidelines, refer to the application note AN5314: ADC Calibration on Kinetis E+ Microcontrollers.

NOTE

In the OFS, CLPX and CLP9 registers, the calibration values are signed numbers (in 2's complement format).

37.6.4 Application hints

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, refer to the application note AN5250: How to Increase the Analog-to-Digital Converter Accuracy in an Application.

37.6.5 DMA Support on ADC

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

For most cases, the DMA request can be directly triggered from ADC conversion completion. The device also support another way to trigger DMA via TRGMUX module. The TRGMUX will provide user a more flexible DMA triggering scheme using software based on different application requirements, for example, the DMA can be triggered after multiple ADC conversion completion instead of every ADC conversion completion.

37.6.6 ADC low-power modes

The ADC will be available in STOP, VLPR, VLPW, and VLPS mode.

NOTE

When in VLPx mode, the ADC clock source is only limited to OSC and SIRC.

37.6.7 ADC Trigger Concept – Use Case

FTM module support counter init trigger and channel match trigger, these triggers could be used as trigger input of PDB, PDB then be used to trigger other modules like ADC. Each ADC channel in PDB module supports up to 8 pre-triggers, which could be used as ADC hardware channel selection to precondition the ADC block prior to actual trigger. The ADC trigger is initiated after pre-trigger to trigger ADC conversion. The waveforms shown in following diagram illustrate the pre-trigger and trigger output of PDB to ADC. Every time when one PDB pre-trigger and trigger output starts an ADC conversion, an internal lock associated with the corresponding pre-trigger is activated. This lock becomes inactive when receiving COCO signal from ADC.

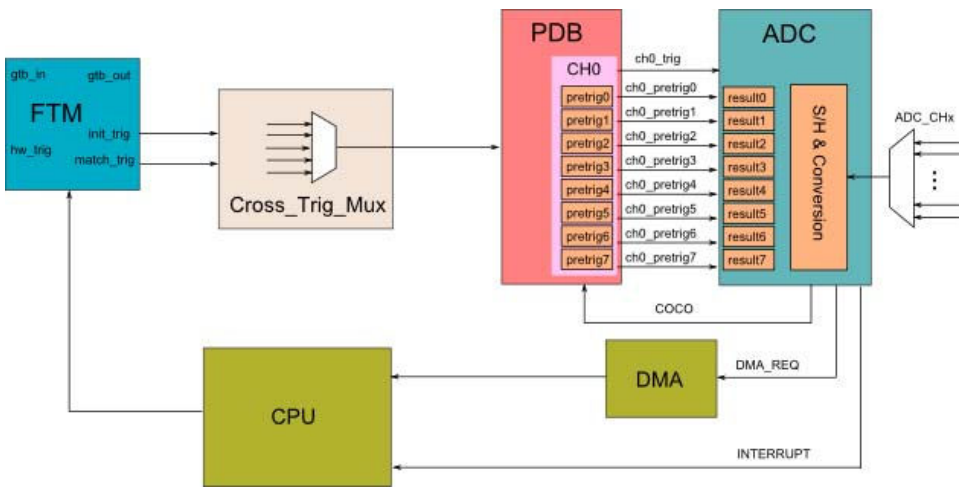


Figure 37-4. PWM Load Diagnosis – ADC Trigger Concept (block diagram)

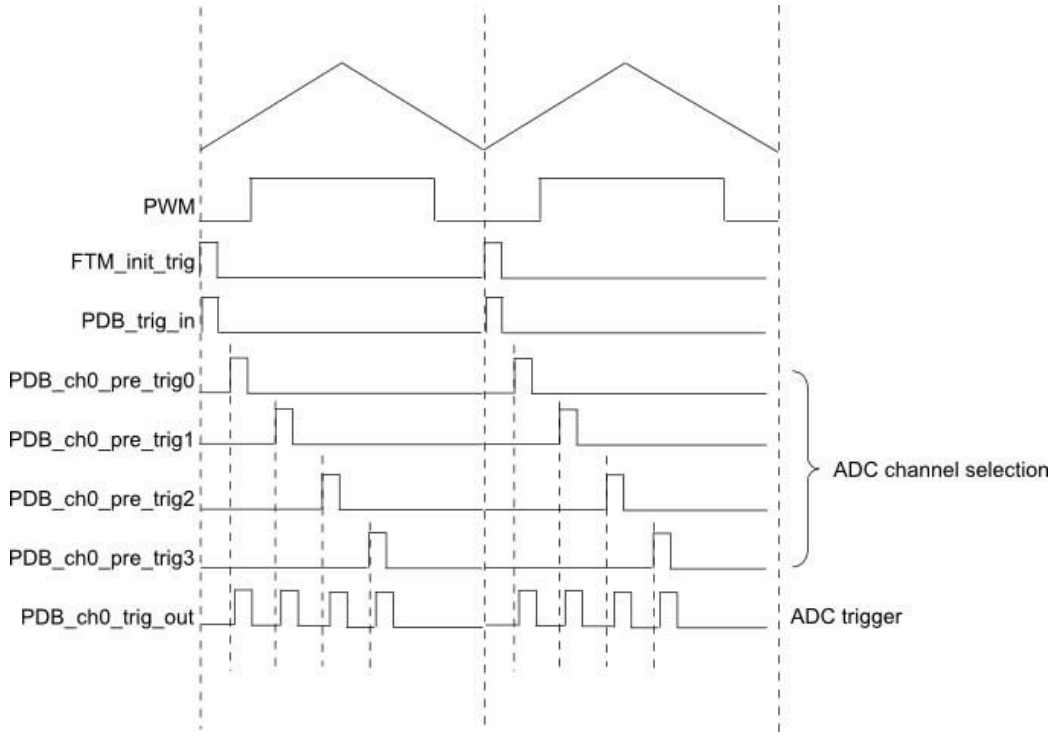


Figure 37-5. PWM Load Diagnosis – ADC Trigger Concept 1 (Timing)

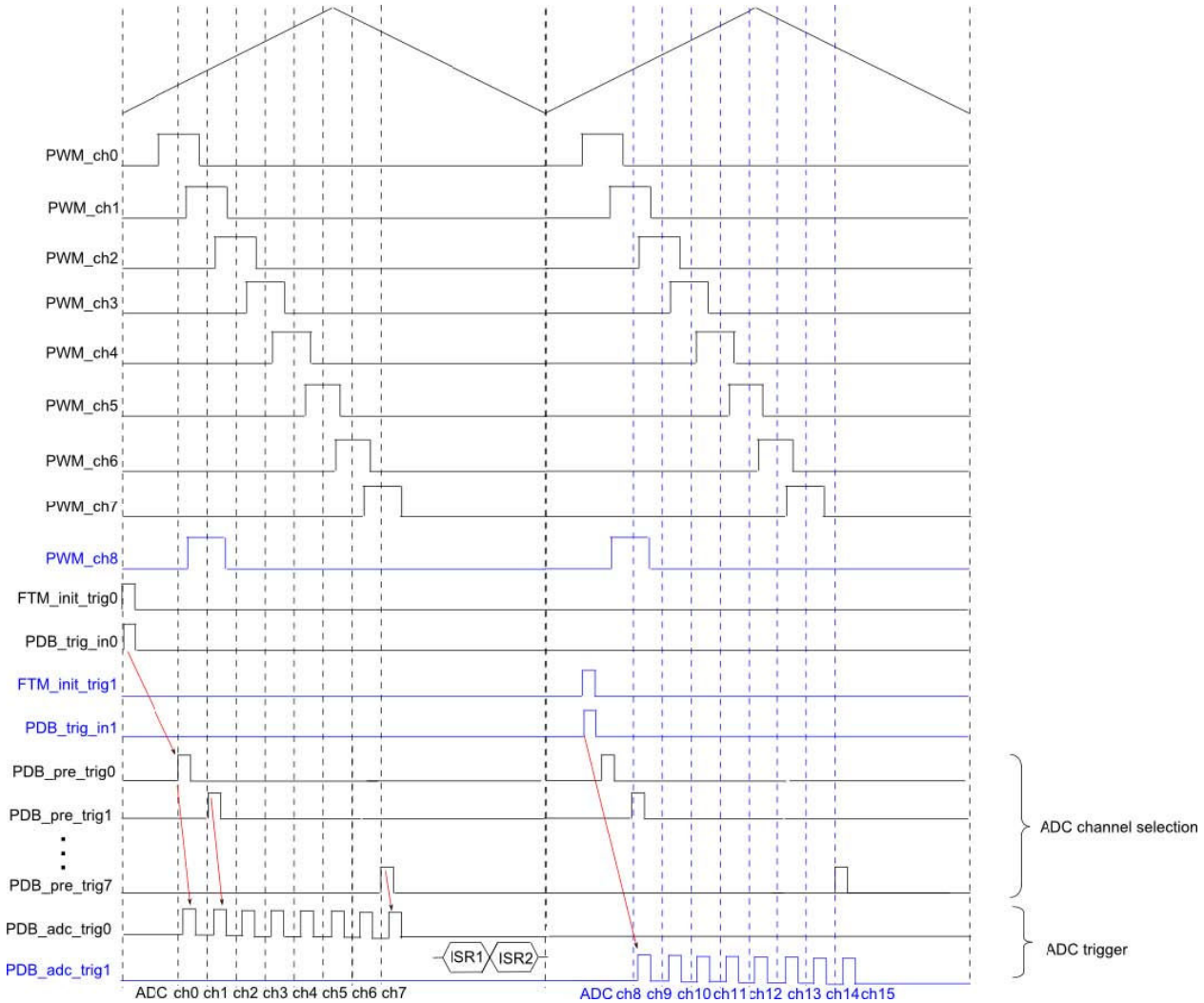


Figure 37-6. PWM Load Diagnosis – ADC Trigger Concept 2 (Timing)

37.6.8 ADC self-test and calibration scheme

ADC calibration needs to be initiated by setting the ADCx_SC3[CAL] bit.

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. Not doing this can result in ADC conversion results with lower than specified accuracy. Calibration needs to be initiated manually by setting the CAL bit. For more details, please refer to "Calibration" section.

Chapter 38
Comparator (CMP)

38.1 Chip-specific information for this module

38.1.1 Instantiation information

Number of CMP	2
8-bit DAC sub-block	Each CMP has its own independent 8-bit DAC.
Analog inputs	Each CMP supports up to 7 analog inputs from external pins.
Internal reference	Each CMP is able to convert an internal reference from the bandgap (1 V reference voltage).
Round-robin mode	Each CMP supports the round-robin sampling scheme.[1]

1. In summary, this allow the CMP to operate independently in STOP and VLPS mode, whilst being triggered periodically to sample up to 7 inputs. Only if an input changes state is a full wakeup generated.

38.1.1.1 CMP input connections

The following table shows the input connections to the CMP.

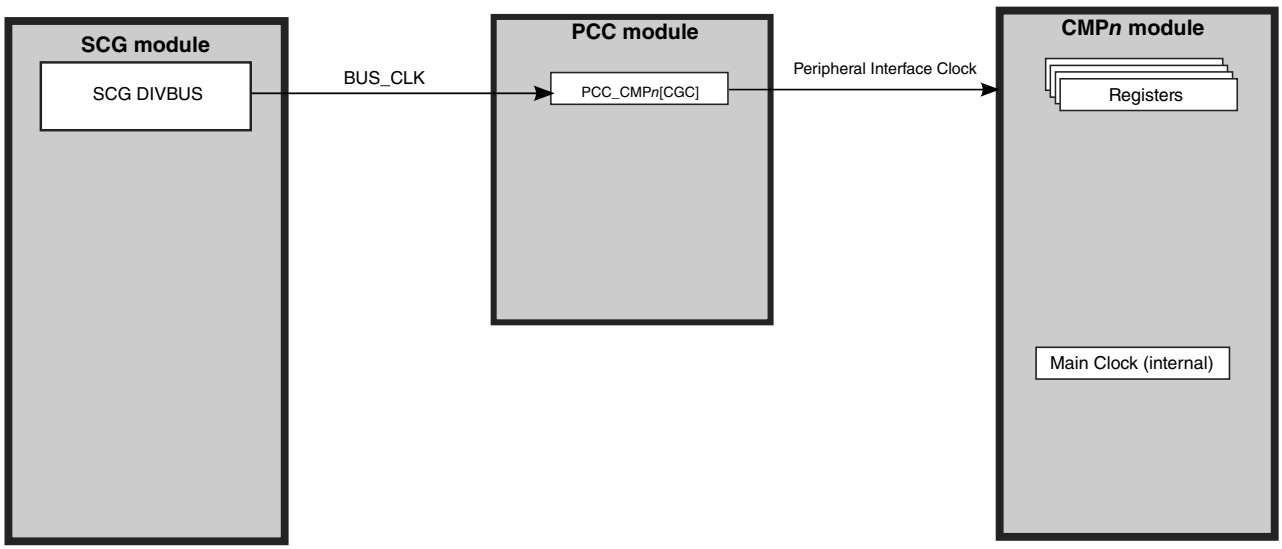
Table 38-1. CMP input connections

CMP Inputs	CMP0	CMP1	CMP2
IN0	ACMP0_IN0	ACMP1_IN0	ACMP2_IN0
IN1	ACMP0_IN1	ACMP1_IN1	ACMP2_IN1
IN2	ACMP0_IN2	ACMP1_IN2	ACMP2_IN2
IN3	ACMP0_IN3	ACMP1_IN3	ACMP2_IN3
IN4	ACMP0_IN4	ACMP1_IN4	ACMP2_IN4
IN5	ACMP0_IN5	ACMP1_IN5	ACMP2_IN5
IN6	ACMP0_IN6	ACMP1_IN6	ACMP2_IN6
IN7	12-bit DAC output	12-bit DAC output	12-bit DAC output

38.1.2 CMP Clocking Information

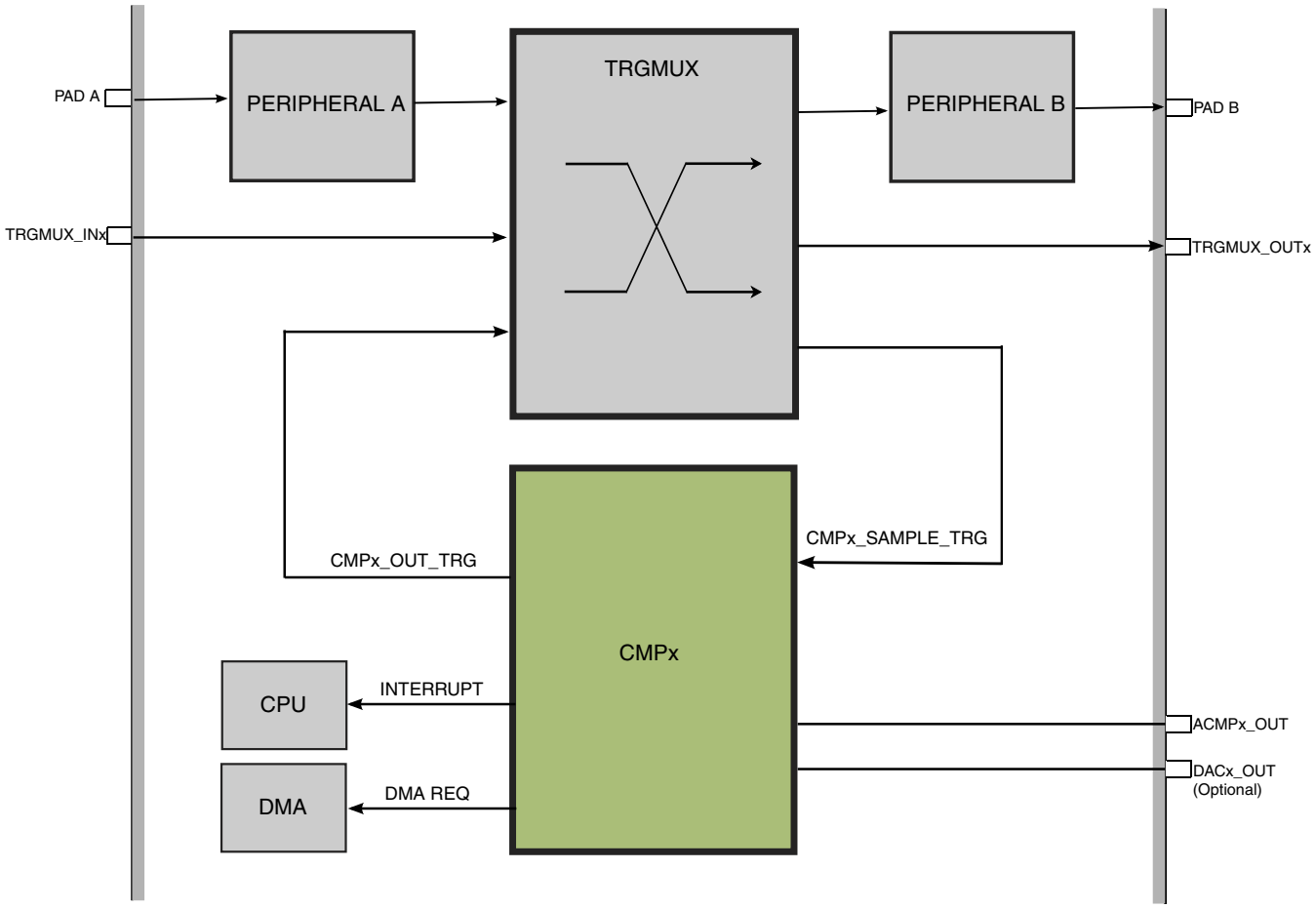
The CMP clocking input is as below.

Peripheral Clocking - CMP



38.1.3 Inter-connectivity Information

The CMP inter-connectivity is shown in following diagram.



38.1.4 Application-related Information

38.1.4.1 CMP external references

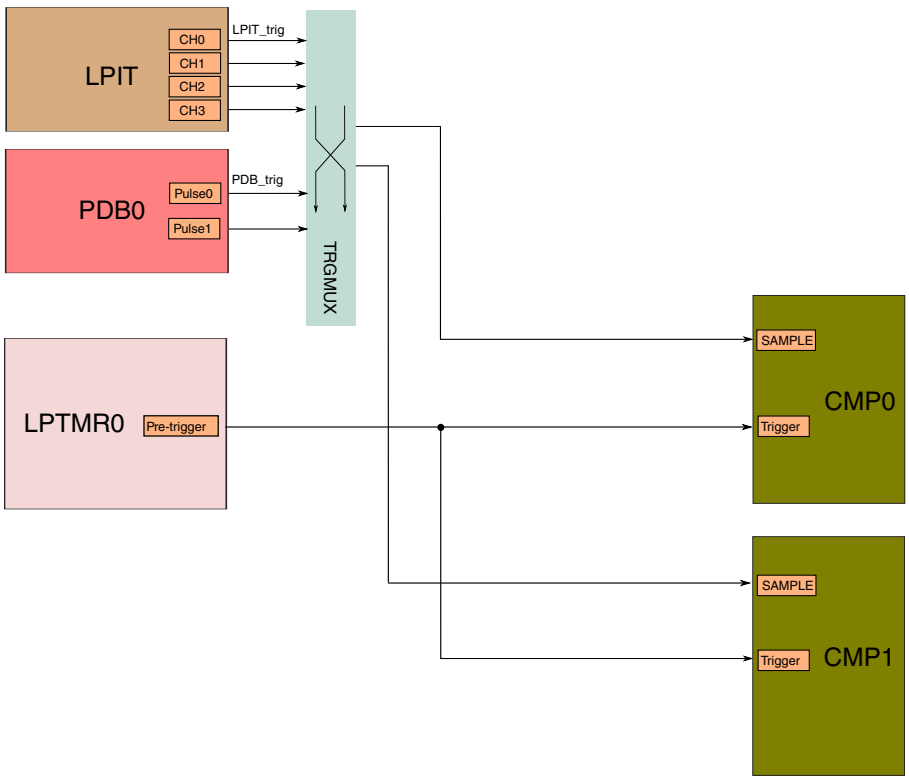
The CMP could get external reference through the tightly integrated 8-bit DAC sub-block. The 8-bit DAC sub-block supports selection of two references. For this device, the references are connected as follows:

- VDDA -- connected to V_{in1} of CMP
- PMC bandgap buffer out (1V reference voltage) -- connected to V_{in2} of CMP

This device have an on-chip 12-bit DAC. The CMP also support reference from the output of 12-bit DAC. In case of 12-bit DAC output is used as CMP reference, it will occupy one of the external inputs (IN7) of the CMP module.

38.1.4.2 External window/sample input

PDB and LPIT could be used to generate pulse output which can be used as sampling windows of CMP block via TRGMUX.



38.1.4.3 CMP trigger mode

The CMP and 8-bit DAC sub-block supports trigger mode operation when the chip is in STOP or VLPS mode. When trigger mode is enabled, the trigger source will provide a low power clock and the triggers to the CMP. The trigger event will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output.

In this device, control for this two-staged sequencing is provided from, for example, LPTMR. The LPTMR provides a single trigger output to all implemented comparators. Through configuration of the CMPx_C2[RRE] bits the trigger can be used to trigger a single comparator or multiple comparators concurrently. The LPTMR only offers single wire trigger to CMP. And the configuration must be done by LPTMR itself (round robin) before entering low power mode.

38.2 Introduction

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 8-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. The 256-tap resistor ladder network divides the supply reference V_{in} into 256 voltage levels. A 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}. The DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

38.3 Features

The following subsections list the features of the CMP, the DAC, and the ANMUX.

38.3.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
    - Sampled
    - Windowed, which is ideal for certain PWM zero-crossing-detection applications
    - Digitally filtered:

    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:
    - Shorter propagation delay at the expense of higher power
    - Low power, with longer propagation delay
- DMA transfer support
    - A comparison event can be selected to trigger a DMA transfer
- Functional in all power modes available on this MCU
- The window and filter functions are not available in STOP modes
- The comparator can be triggered by other peripherals to work for only a small fraction of the time

38.3.2 8-bit DAC key features

The DAC has the following features:

- 8-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

38.3.3 ANMUX key features

The ANMUX has the following features:

- Two 8-to-1 channel MUXes
- Operational over the entire supply range

38.4 CMP, DAC, and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

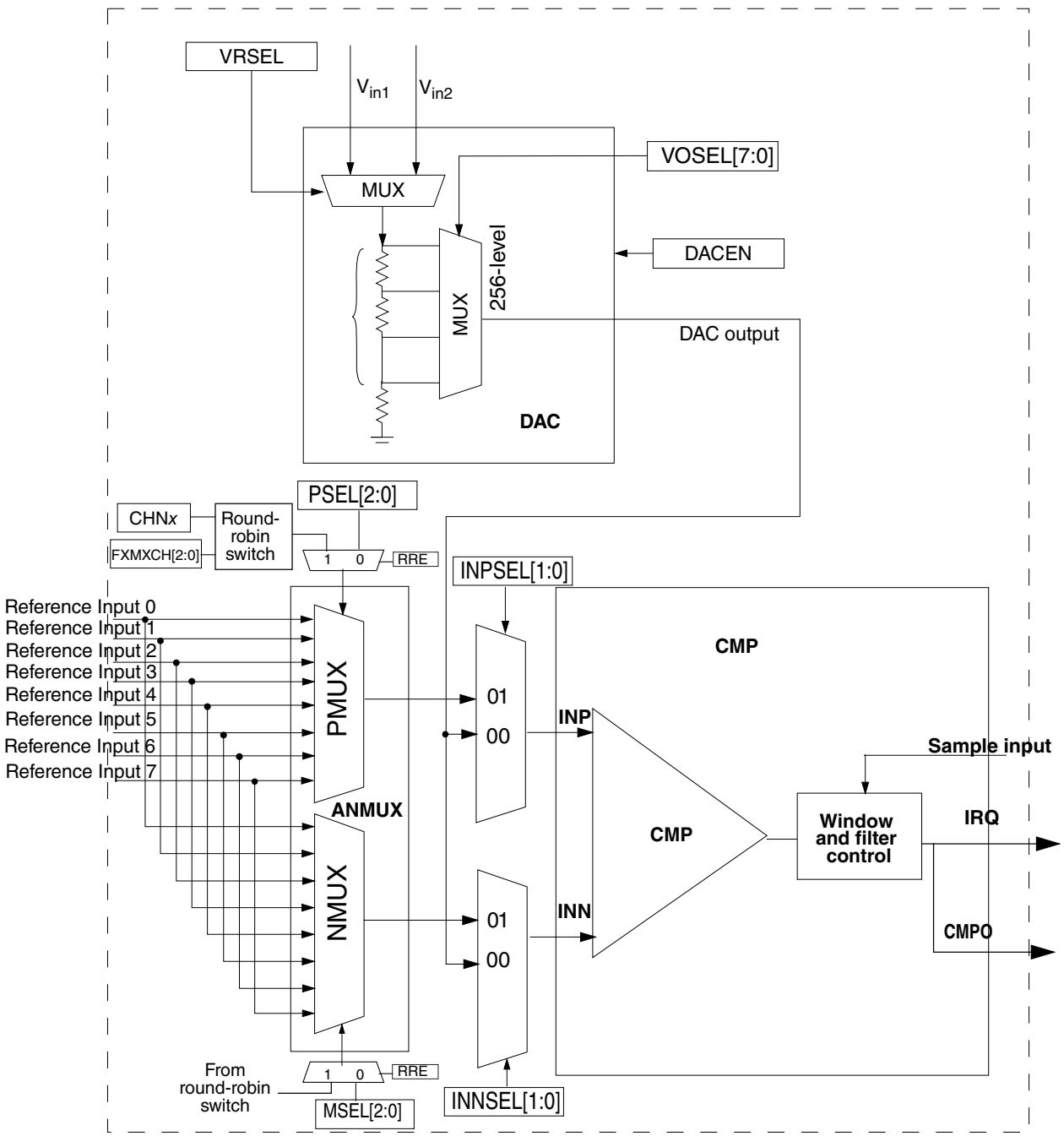


Figure 38-1. CMP high level diagram

38.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

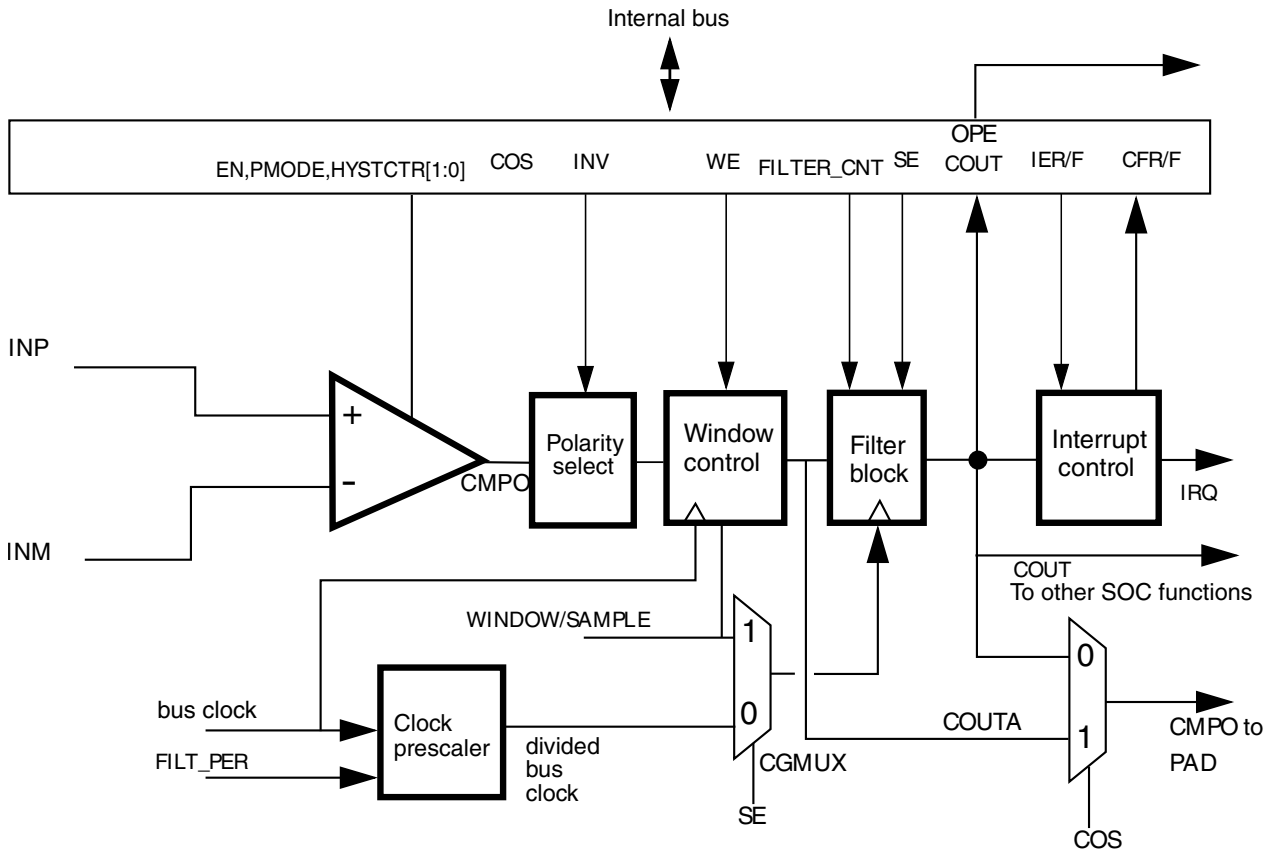


Figure 38-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when C0[WE] = 0.
- If C0[WE] = 1, the comparator output is sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.

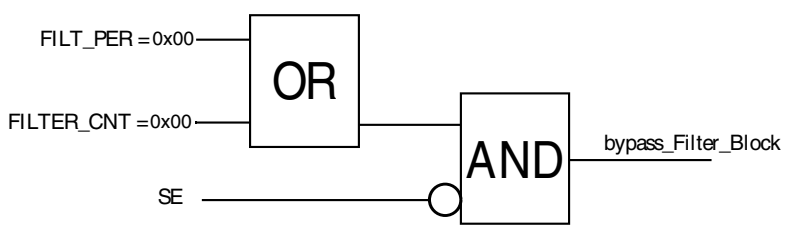


Figure 38-3. Filter block bypass logic

- The Filter block acts as a simple sampler if the filter is bypassed and C0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and C0[FILTER_CNT] is set greater than 0x01.
  - If C0[SE] = 1, the external SAMPLE input is used as the sampling clock.
  - IF C0[SE] = 0, the divided bus clock is used as the sampling clock.
- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which crosses clock domain boundaries, must be resynchronized to the bus clock.
- C0[WE] and C0[SE] are mutually exclusive.
- If enabled, the filter clock and the sample period must be at least 4 times slower than the system clock to the comparator.

38.6 CMP pin descriptions

This section provides the comparator pin descriptions. The external inputs IN[7:0] are muxed by CMP_C1[PSEL] and CMP_C1[MSEL] beforehand and multiplexed output will then go to the second stage of multiplex with the input of 8-bit DAC and other two internal reserved test signals, determined by CMP_C1[INPSEL] and CMP_C1[INNSEL]. The output of the second multiplex will finally go to the positive and negative ports of the comparator respectively.

Table 38-2. CMP signal descriptions

Signal	Description	I/O
IN[7:0]	Analog voltage inputs	I

NOTE

If comparing one input channel with the DAC output, and if there is injection or over-voltage in the input channels, the DAC output may be corrupted. For such case, the software workaround is to configure the DAC side SEL[2:0] same as the non-DAC side, i.e. configuration of MSEL and PSEL register bits must be the same.

38.6.1 External pins

The CMP has two analog inputs: INP and INM. Each of these pins can accept an input voltage that varies across the full operating range of the MCU. If the module is not enabled, each pin can be used as a digital input or output. Consult the specific MCU documentation to determine what functions are shared with these analog inputs.

The user can select either filtered or unfiltered comparator outputs for use on an external I/O pad.

38.7 CMP functional modes

There are three main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, C0[FILTER_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using C0[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting C0[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 38-3. Comparator sample/filter controls

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode**

Table continues on the next page...

Table 38-3. Comparator sample/filter controls (continued)

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
2B	1	0	0	X	0x00	See the Continuous mode (#s 2A & 2B).
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode**
3B	1	0	0	0x01	> 0x00	See the Sampled, Non-Filtered mode (#s 3A & 3B).
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode**
4B	1	0	0	> 0x01	> 0x04	See the Sampled, Filtered mode (#s 4A & 4B).
5A	1	1	0	0x00	X	**Windowed mode**
5B	1	1	0	X	0x00	Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA. See the Windowed mode (#s 5A & 5B).
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 C0[FPR] to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. See the Windowed/Filtered mode (#7).
All other combinations of C0[EN], C0[WE], C0[SE], C0[FILTER_CNT], and C0[FPR] are illegal.						

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings must be changed only after setting C0[SE]=0, C0[FPR] =0 and C0[FILTER_CNT]=0x00. This resets the filter to a known state.

38.7.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

38.7.2 Continuous mode (#s 2A & 2B)

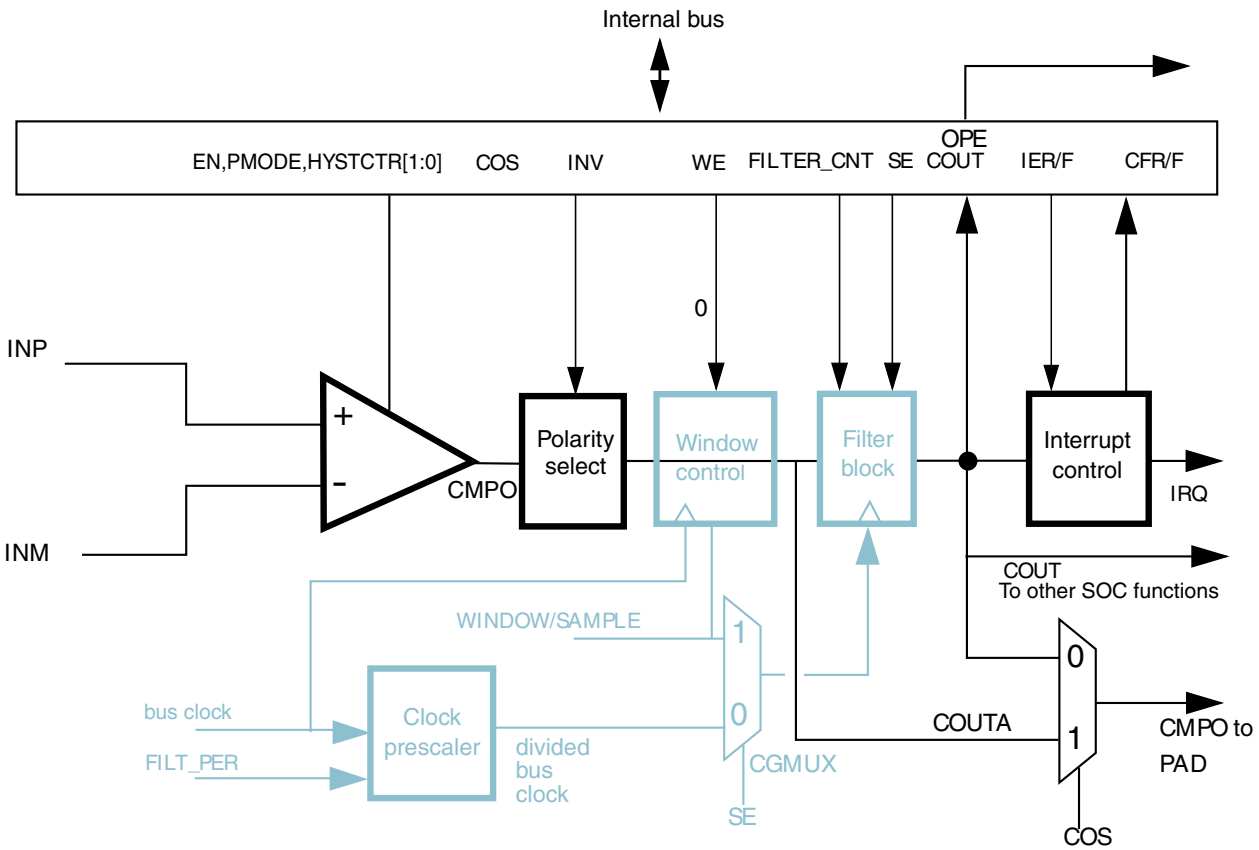


Figure 38-4. Comparator operation in Continuous mode

NOTE

See the chip configuration section for the source of sample/ window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed (as the grey-colored parts in the figure). C0[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations that result in disabling the filter block, see Figure 38-3.

38.7.3 Sampled, Non-Filtered mode (#s 3A & 3B)

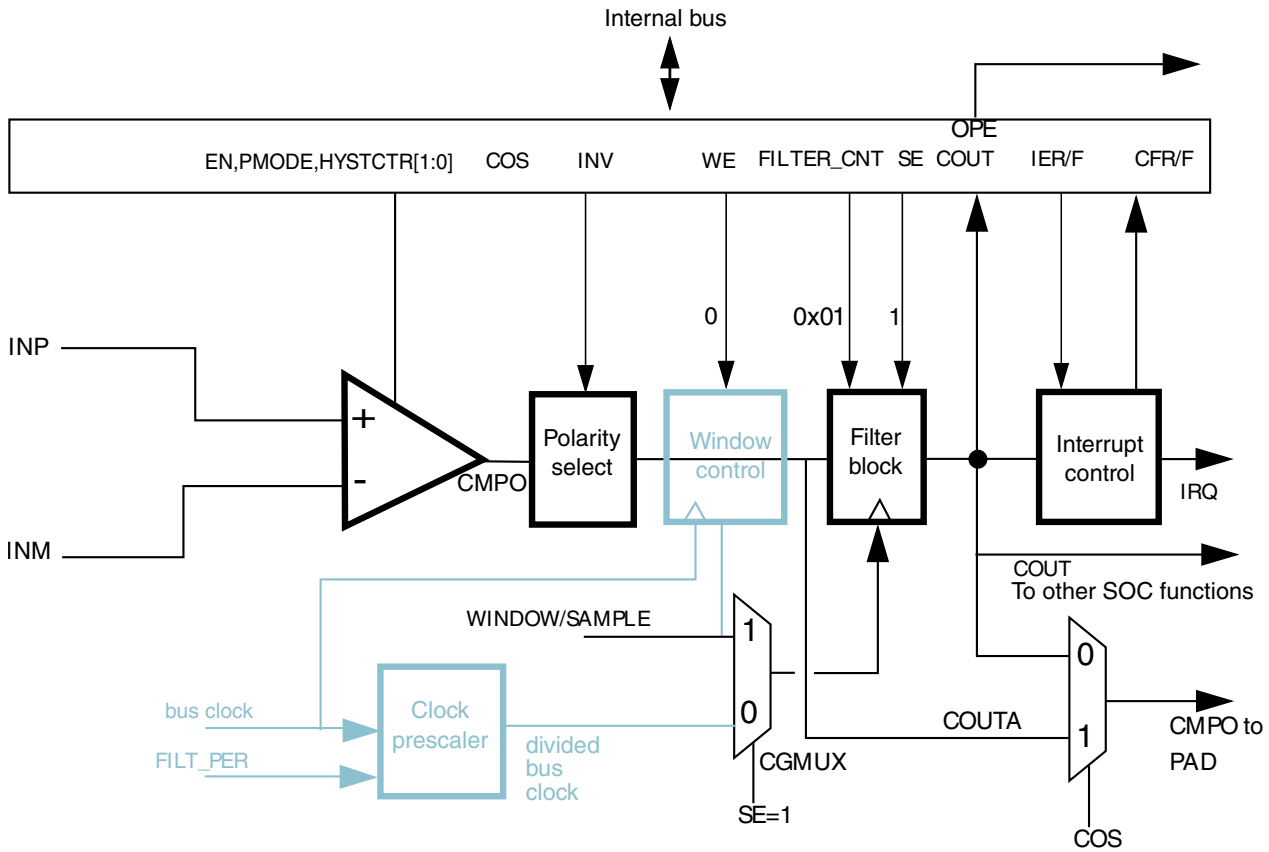


Figure 38-5. Sampled, Non-Filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

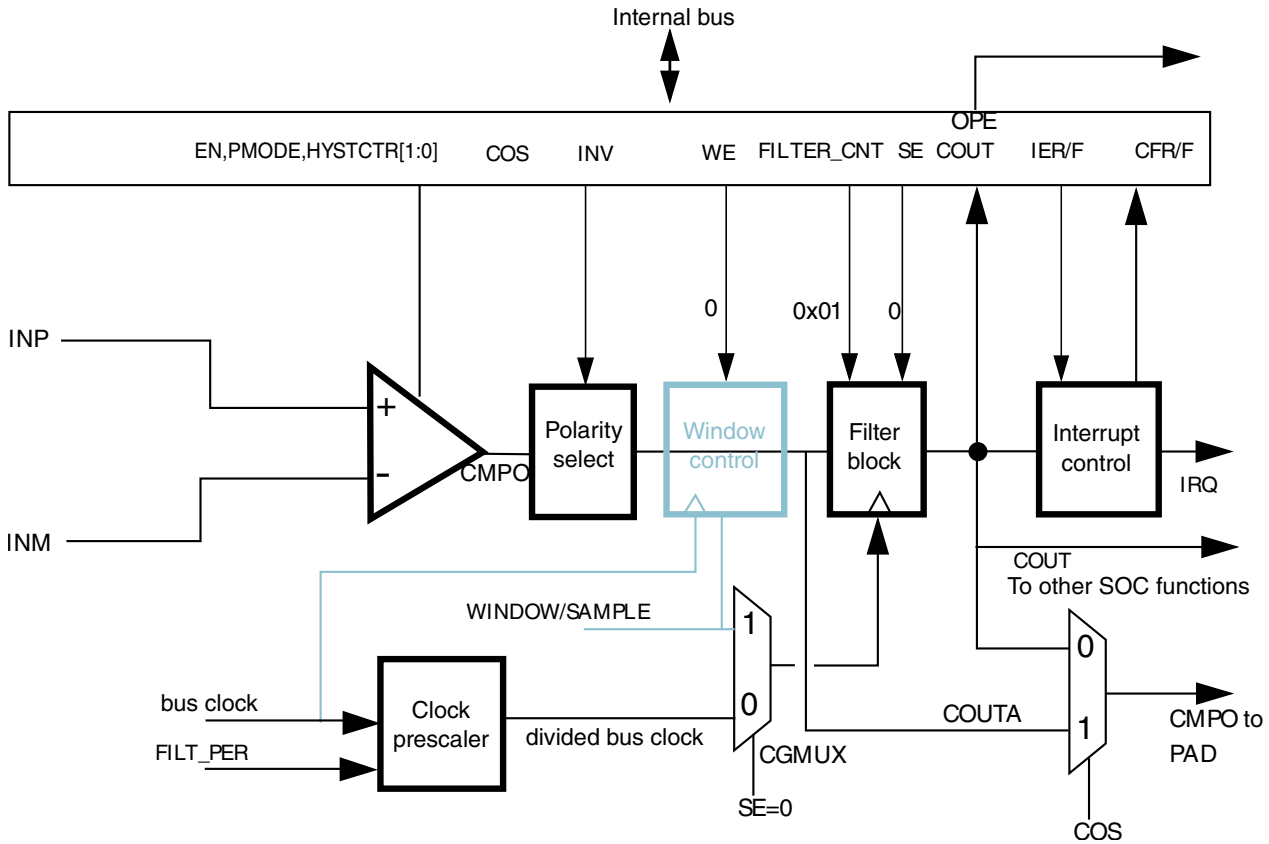


Figure 38-6. Sampled, Non-Filtered (# 3B): sampling interval internally derived

The following figure illustrates comparator operation in this mode, assuming the polarity select is set to non-inverting state.

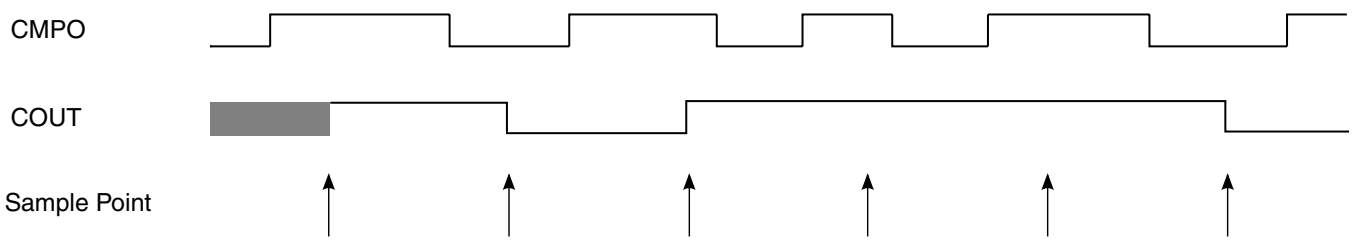


Figure 38-7. Sampled, Non-Filtered Mode Timing Diagram

38.7.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, C0[FILTER_CNT]>1, which activates filter operation.

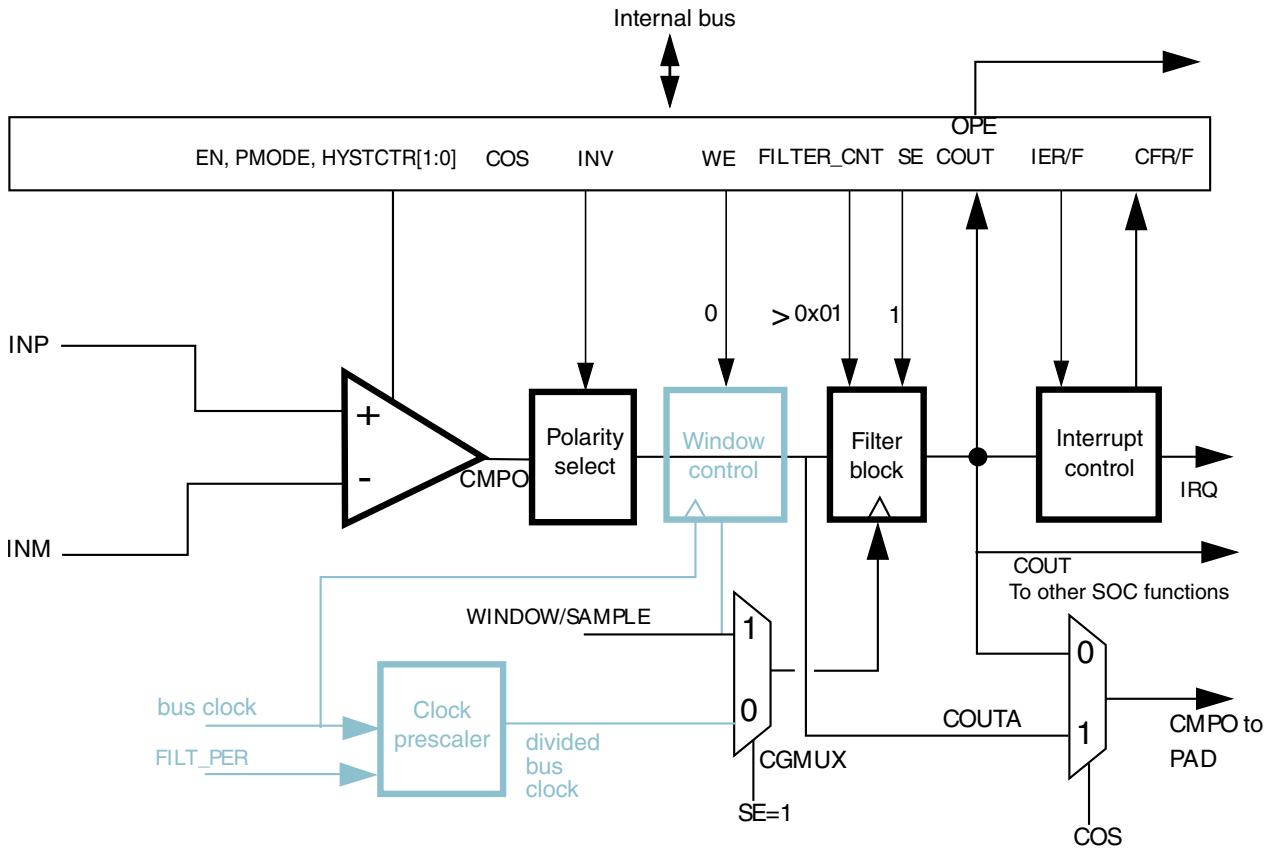


Figure 38-8. Sampled, Filtered (# 4A): sampling point externally driven

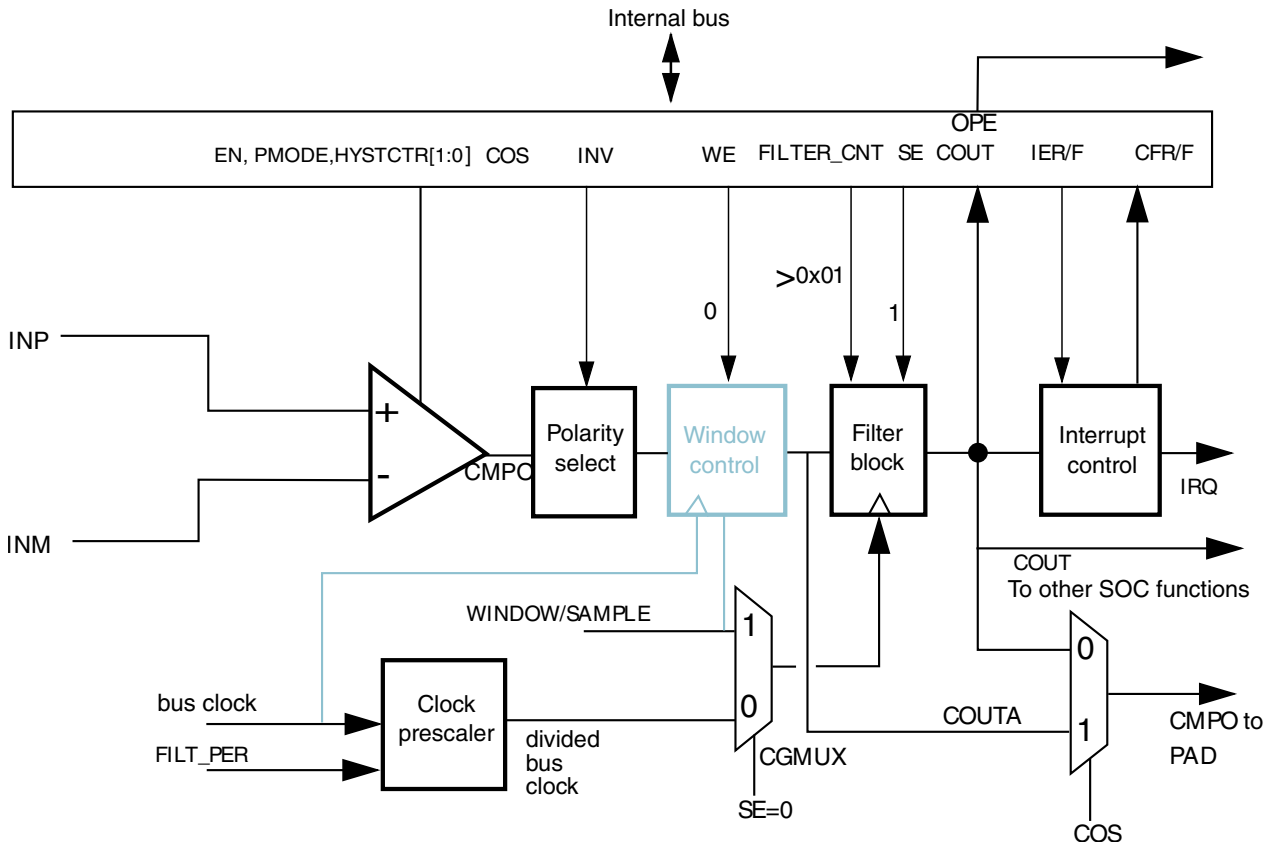


Figure 38-9. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, C0[FILTER_CNT]>1, which activates filter operation.

38.7.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

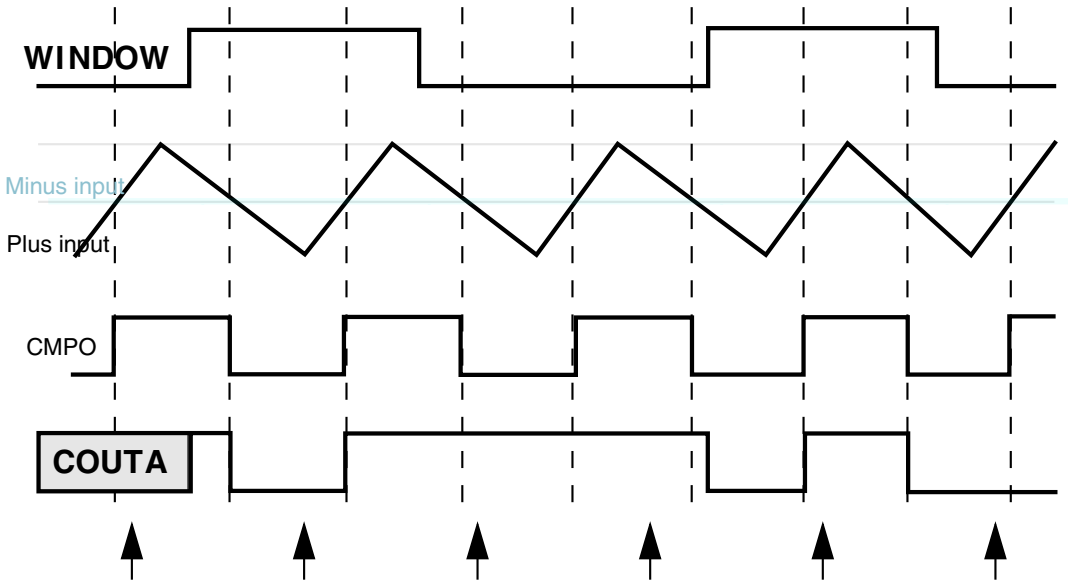


Figure 38-10. Windowed mode timing diagram

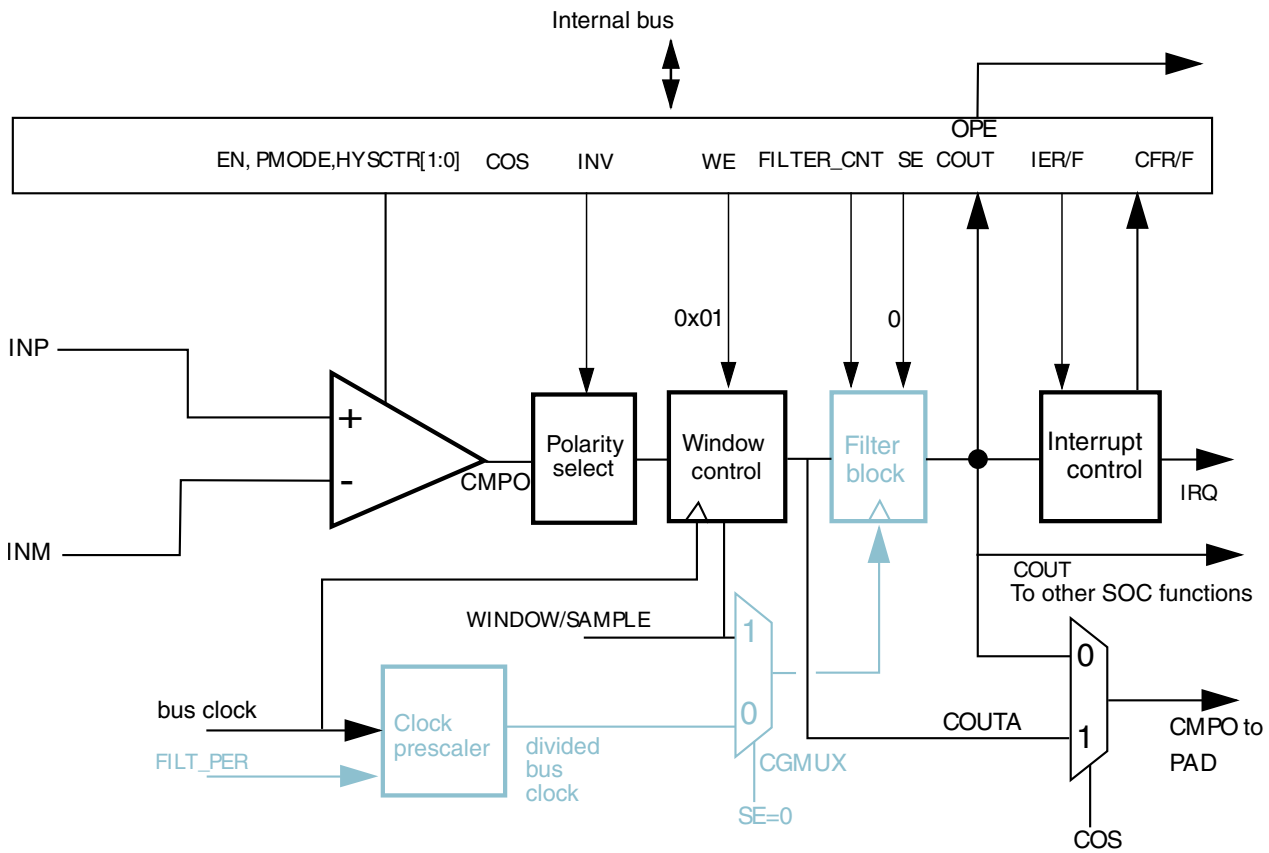


Figure 38-11. Windowed mode

For control configurations which result in disabling the filter block, see Figure 38-3.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

NOTE

The sample input must be high for ≥ 2.5 CMP bus clock cycles to ensure no sampling event is missed.

38.7.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 38-10, 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.

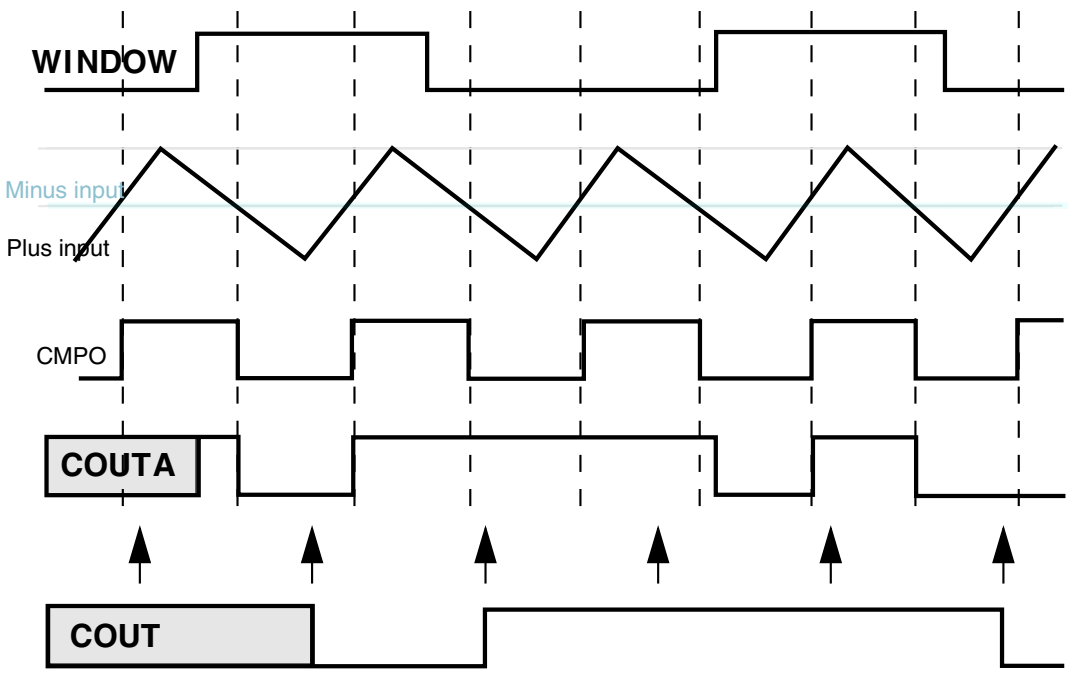


Figure 38-12. Windowed/resampled mode operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of C0[FILTER_CNT] must be 1.

NOTE

The sample input must be high for ≥ 2.5 CMP bus clock cycles to ensure no sampling event is missed.

38.7.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + [(C0[FILTER_CNT] × C0[FPR]) + 1] × bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

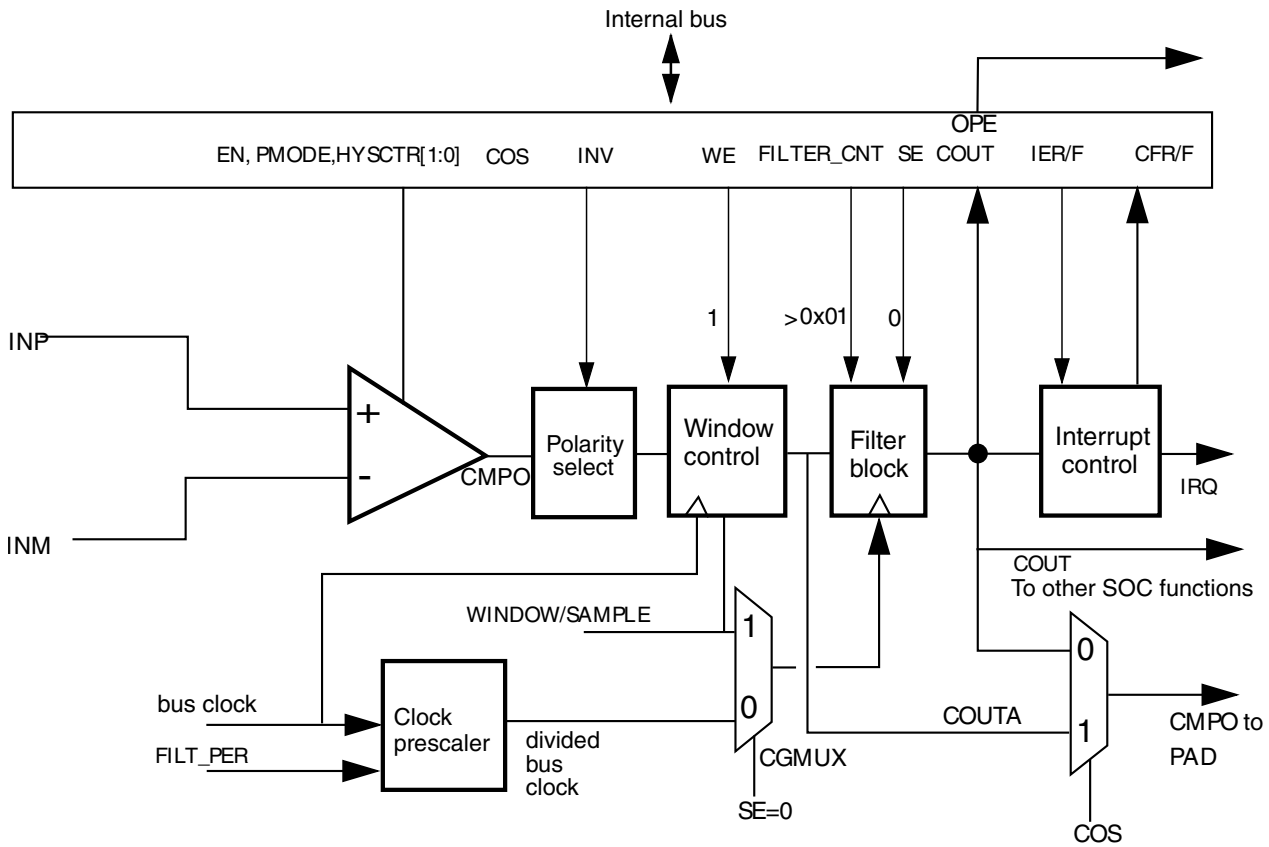


Figure 38-13. Windowed/Filtered mode

The following figure shows the operation timing for this mode, considering uncertainty is introduced by the internal synchronization for the filter block.

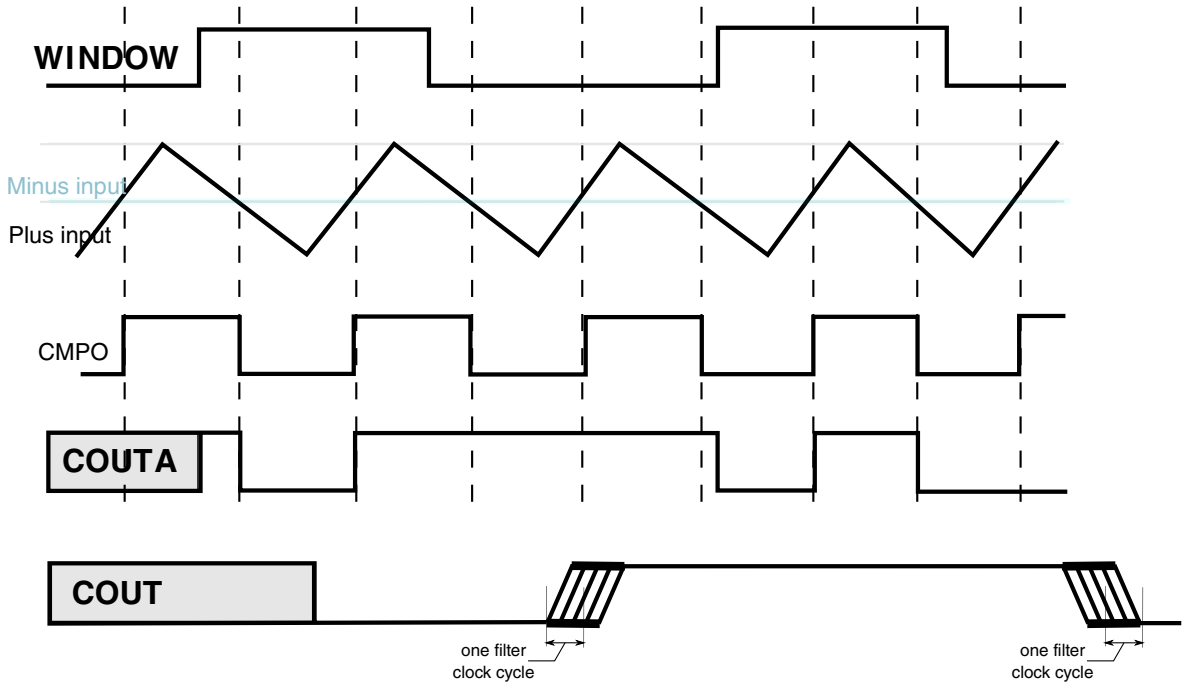


Figure 38-14. Windowed/Filtered mode operation

38.8 Memory map/register definitions

CMP memory map

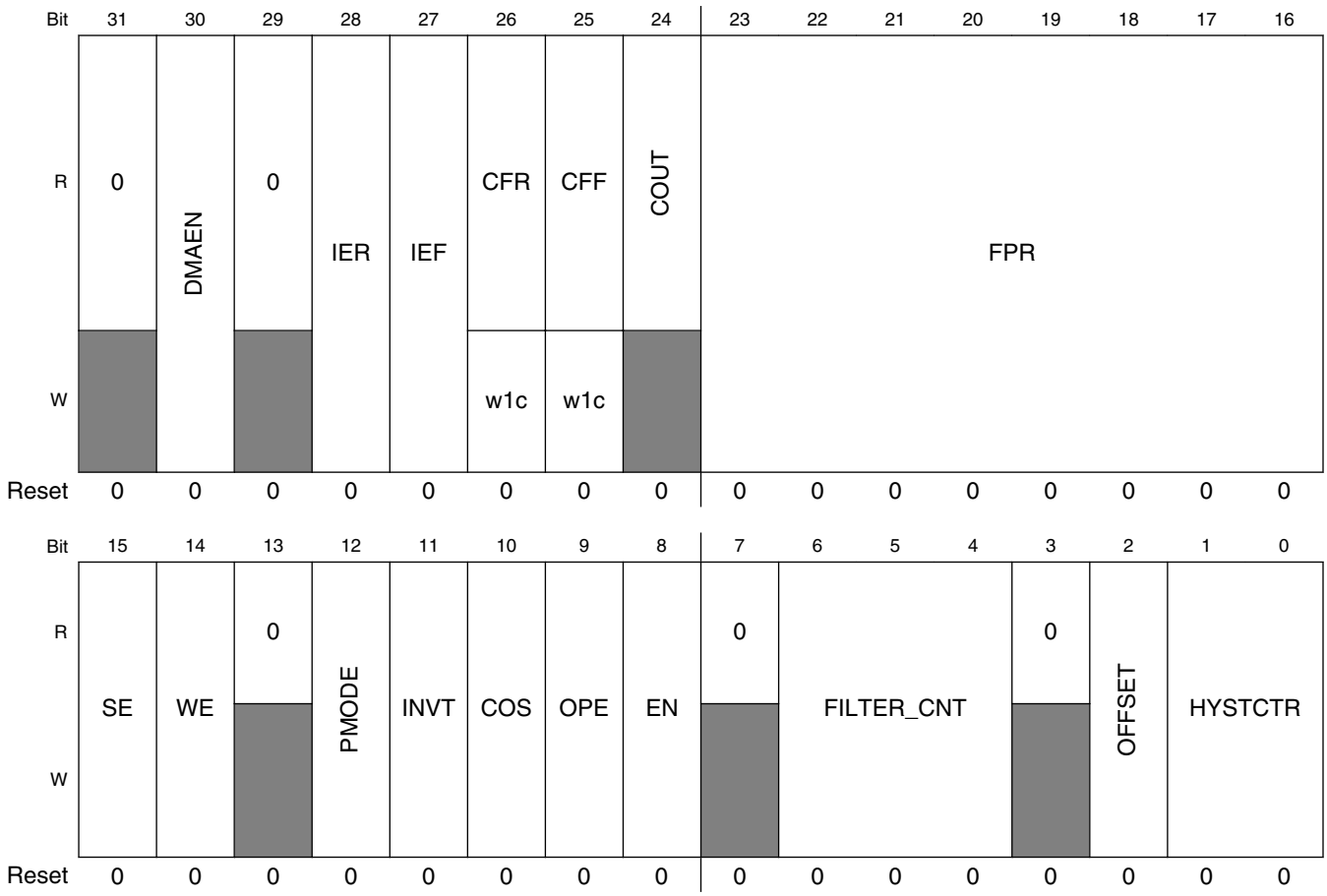
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_3000	CMP Control Register 0 (CMP0_C0)	32	R/W	0000_0000h	38.8.1/886
4007_3004	CMP Control Register 1 (CMP0_C1)	32	R/W	0000_0000h	38.8.2/890
4007_3008	CMP Control Register 2 (CMP0_C2)	32	R/W	0000_0000h	38.8.3/893
4007_4000	CMP Control Register 0 (CMP1_C0)	32	R/W	0000_0000h	38.8.1/886
4007_4004	CMP Control Register 1 (CMP1_C1)	32	R/W	0000_0000h	38.8.2/890
4007_4008	CMP Control Register 2 (CMP1_C2)	32	R/W	0000_0000h	38.8.3/893
4007_5000	CMP Control Register 0 (CMP2_C0)	32	R/W	0000_0000h	38.8.1/886
4007_5004	CMP Control Register 1 (CMP2_C1)	32	R/W	0000_0000h	38.8.2/890
4007_5008	CMP Control Register 2 (CMP2_C2)	32	R/W	0000_0000h	38.8.3/893

38.8.1 CMP Control Register 0 (CMPx_C0)

Access:

- Supervisor read/write
- User read/write

Address: Base address + 0h offset



CMPx_C0 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 DMAEN	DMA Enable 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.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
27 IEF	Comparator Interrupt Enable Falling

Table continues on the next page...

CMPx_C0 field descriptions (continued)

Field	Description
	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.
26 CFR	Analog Comparator Flag Rising Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is level sensitive 0 A rising edge has not been detected on COUT. 1 A rising edge on COUT has occurred.
25 CFF	Analog Comparator Flag Falling Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is level sensitive . 0 A falling edge has not been detected on COUT. 1 A falling edge on COUT has occurred.
24 COUT	Analog Comparator Output Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as C0[INVT] when the Analog Comparator module is disabled, that is, when C0[EN] = 0. Writes to this field are ignored.
23–16 FPR	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when C0[SE] = 0. Setting FPR to 0x0 disables the filter. Filter programming and latency details are provided in the CMP functional description. This field has no effect when C0[SE]= 1. In that case, the external SAMPLE signal is used to determine the sampling period.
15 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved. 0 Sampling mode is not selected. 1 Sampling mode is selected.
14 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved. 0 Windowing mode is not selected. 1 Windowing mode is selected.
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode is selected. 1 High Speed (HS) comparison mode is selected, in VLPx mode, or Stop mode switched to Low Speed (LS) mode.

Table continues on the next page...

CMPx_C0 field descriptions (continued)

Field	Description
11 INVT	Comparator invert This bit allows selecting the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as C0[COUT]) when C0[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
10 COS	Comparator Output Select 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
9 OPE	Comparator Output Pin Enable The OPE bit enables the path from the comparator output to a selected pin. 0 When OPE is 0, the comparator output (after window/filter settings dependent on software configuration) is not available to a packaged pin. 1 When OPE is 1, and if the software has configured the comparator to own a packaged pin, the comparator is available in a packaged pin.
8 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, the analog part remains in the off state, and consumes no power. 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.
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 This field specifies the number of consecutive samples that must agree prior to the comparator output filter accepting a new output state. For information regarding filter programming and latency, please see the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 OFFSET	Comparator hard block offset control. See chip data sheet to get the actual offset value with each level

Table continues on the next page...

CMPx_C0 field descriptions (continued)

Field	Description
	NOTE: • If OFFSET = 1, then there will be no hysteresis in the case of INP crossing INN in the positive direction (or INN crossing INP in the negative direction). A Half Hysteresis value still exists for INP crossing INN in the falling direction. • If OFFSET = 0, then the hysteresis selected by HYSTCTR is valid for both directions. 0 The comparator hard block output has level 0 offset internally. 1 The comparator hard block output has level 1 offset internally.
HYSTCTR	Comparator hard block hysteresis control. See chip data sheet to get the actual hysteresis value with each level 00 The hard block output has level 0 hysteresis internally. 01 The hard block output has level 1 hysteresis internally. 10 The hard block output has level 2 hysteresis internally. 11 The hard block output has level 3 hysteresis internally.

38.8.2 CMP Control Register 1 (CMPx_C1)

Access:

- Supervisor read/write
- User read/write

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0	INPSEL		0	INNSEL		CHN7	CHN6	CHN5	CHN4	CHN3	CHN2	CHN1	CHN0
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	DACEN	VRSEL	PSEL			MSEL			VOSEL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMPx_C1 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

CMPx_C1 field descriptions (continued)

Field	Description
28–27 INPSEL	Selection of the input to the positive port of the comparator Determines which input is selected for the plus input of the comparator. NOTE: These selections is used to select the final positive input to the comparator. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMPx_C1 register must have different values. 00 IN0, from the 8-bit DAC output 01 IN1, from the analog 8-1 mux 10 Reserved 11 Reserved
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 INNSEL	Selection of the input to the negative port of the comparator Determines which input is selected for the plus input of the comparator. NOTE: These selections is used to select the final negative input to the comparator. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMPx_C1 register must have different values. 00 IN0, from the 8-bit DAC output 01 IN1, from the analog 8-1 mux 10 Reserved 11 Reserved
23 CHN7	Channel 7 input enable Channel 7 of the input enable for the round-robin checker. If CHN7 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
22 CHN6	Channel 6 input enable Channel 6 of the input enable for the round-robin checker. If CHN6 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
21 CHN5	Channel 5 input enable Channel 5 of the input enable for the round-robin checker. If CHN5 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
20 CHN4	Channel 4 input enable Channel 4 of the input enable for the round-robin checker. If CHN4 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
19 CHN3	Channel 3 input enable Channel 3 of the input enable for the round-robin checker. If CHN3 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.

Table continues on the next page...

CMPx_C1 field descriptions (continued)

Field	Description
18 CHN2	Channel 2 input enable Channel 2 of the input enable for the round-robin checker. If CHN2 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
17 CHN1	Channel 1 input enable Channel 1 of the input enable for the round-robin checker. If CHN1 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
16 CHN0	Channel 0 input enable Channel 0 of the input enable for the round-robin checker. If CHN0 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
15 DACEN	DAC Enable This bit is used to enable the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
14 VRSEL	Supply Voltage Reference Source Select 0 Vin1 is selected as resistor ladder network supply reference Vin. 1 Vin2 is selected as resistor ladder network supply reference Vin.
13–11 PSEL	Plus Input MUX Control Determines which input is selected for the plus mux. NOTE: These bits are used to select the external 8 inputs for the plus mux, the actual input to the positive port of the comparator is selected between this mux out and other inputs finally, see the definition in INPSEL. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMPx_C1 register must have different values. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7
10–8 MSEL	Minus Input MUX Control Determines which input is selected for the minus mux. NOTE: These bits are used to select the external 8 inputs for the minus mux, the actual input to the negative port of the comparator is selected between this mux out and other inputs finally, see the definition in INNSEL. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMPx_C1 register must have different values.

Table continues on the next page...

CMPx_C1 field descriptions (continued)

Field	Description
	000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7
VOSEL	DAC Output Voltage Select This bit selects an output voltage from one of 256 distinct levels. DACO = (Vin/256) × (VOSEL[7:0] + 1), so the DACO range is from Vin/256 to Vin.

38.8.3 CMP Control Register 2 (CMPx_C2)

Access:

- Supervisor read/write
- User read/write

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RRE	RRIE	FXMP	0	FXMXCH			0	CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

CMPx_C2 field descriptions

Field	Description
31 RRE	Round-Robin Enable This bit enables the round-robin operation. 0 Round-robin operation is disabled. 1 Round-robin operation is enabled.
30 RRIE	Round-Robin interrupt enable This bit enables the interrupt/wake-up when the comparison result changes for a given channel. 0 The round-robin interrupt is disabled. 1 The round-robin interrupt is enabled when a comparison result changes from the last sample.
29 FXMP	Fixed MUX Port This bit is used to fix the analog mux port for the round-robin mode. 0 The Plus port is fixed. Only the inputs to the Minus port are swept in each round. 1 The Minus port is fixed. Only the inputs to the Plus port are swept in each round.
28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–25 FXMXCH	Fixed channel selection This field indicates which channel in the mux port is fixed in a given round-robin mode. 000 Channel 0 is selected as the fixed reference input for the fixed mux port. 001 Channel 1 is selected as the fixed reference input for the fixed mux port. 010 Channel 2 is selected as the fixed reference input for the fixed mux port. 011 Channel 3 is selected as the fixed reference input for the fixed mux port. 100 Channel 4 is selected as the fixed reference input for the fixed mux port. 101 Channel 5 is selected as the fixed reference input for the fixed mux port. 110 Channel 6 is selected as the fixed reference input for the fixed mux port. 111 Channel 7 is selected as the fixed reference input for the fixed mux port.
24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 CH7F	Channel 7 input changed flag. This bit is set if the channel 7 input changed from the last comparison with the fixed mux port.
22 CH6F	Channel 6 input changed flag. This bit is set if the channel 6 input changed from the last comparison with the fixed mux port.
21 CH5F	Channel 5 input changed flag. This bit is set if the channel 5 input changed from the last comparison with the fixed mux port.
20 CH4F	Channel 4 input changed flag. This bit is set if the channel 4 input changed from the last comparison with the fixed mux port.
19 CH3F	Channel 3 input changed flag. This bit is set if the channel 3 input changed from the last comparison with the fixed mux port.
18 CH2F	Channel 2 input changed flag. This bit is set if the channel 2 input changed from the last comparison with the fixed mux port.
17 CH1F	Channel 1 input changed flag. This bit is set if the channel 1 input changed from the last comparison with the fixed mux port.

Table continues on the next page...

CMPx_C2 field descriptions (continued)

Field	Description
16 CH0F	Channel 0 input changed flag. This bit is set if the channel 0 input changed from the last comparison with the fixed mux port.
15–14 NSAM	Number of sample clocks For a given channel, this field specifies how many round-robin clock cycles later the sample takes place. 00 The comparison result is sampled as soon as the active channel is scanned in one round-robin clock. 01 The sampling takes place 1 round-robin clock cycle after the next cycle of the round-robin clock. 10 The sampling takes place 2 round-robin clock cycles after the next cycle of the round-robin clock. 11 The sampling takes place 3 round-robin clock cycles after the next cycle of the round-robin clock.
13–8 INITMOD	Comparator and DAC initialization delay modulus. These values specify the round robin clock cycles used to determine the comparator and DAC initialization delays specified by the datasheet. For example the initialization delay is 80us and the round robin clock is 100kHz, then INITMOD should be set to 80us/10us = 8. 000000 The modulus is set to 64 (same with 111111). other values Initialization delay is set to INITMOD × round robin clock period
ACOn	The result of the input comparison for channel *n* . This field stores the latest comparison result of the input channel *n* with the fixed mux port. Reading this bit returns the latest comparison result. Writing this field defines the pre-set state of channel *n*.

38.9 CMP functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM. CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting C0[INVT] = 1.

C0[IER] and C0[IEF] are used to select the condition that causes the CMP module to assert an interrupt to the processor. C0[CFF] is set on a falling edge, and C0[CFR] is set on a rising edge of the comparator output. The optionally filtered CMPO can be read directly through C0[COUT].

38.9.1 Initialization

A typical startup sequence is as follows.

The time required to stabilize COUT is the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter. See the datasheet for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the Low-pass filter section.

During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and C0[CFR]/C0[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT initially equals 0 until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

38.9.2 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by 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.

38.9.2.1 Enabling filter modes

Filter modes can be enabled by:

- Setting C0[FILTER_CNT] > 0x01 and
- Setting C0[FPR] to a nonzero value or setting C0[SE]=1

If using the divided bus clock to drive the filter, it samples COUTA every C0[FPR] bus clock cycles.

The filter output is at logic 0 when first initalized, and subsequently changes when all the consecutive C0[FILTER_CNT] samples agree that the output value has changed. In other words, C0[COUT] is 0 for some initial period, even when COUTA is at logic 1.

Setting all of C0[SE], C0[FPR] and C0[FILTER_CNT] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching C0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If C0[SE]=1, the filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive C0[FILTER_CNT] samples agree that the output value has changed.

38.9.2.2 Latency issues

The value of C0[FPR] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of C0[FILTER_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of C0[FILTER_CNT].

The values of C0[FPR] or SAMPLE period and C0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of C0[FILTER_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

Table 38-4. Comparator sample/filter maximum latencies

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$

Table continues on the next page...

Table 38-4. Comparator sample/filter maximum latencies (continued)

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
3B	1	0	0	0x01	> 0x00		T_{PD} + (C0[FPR] * T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (C0[FILTER_CNT] * T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (C0[FILTER_CNT] * C0[FPR] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	T_{PD} + T_{per}
5B	1	1	0	X	0x00		T_{PD} + T_{per}
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (C0[FPR] * T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (C0[FILTER_CNT] * C0[FPR] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

38.10 Interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both. Assuming the CMP DMA enable bit is not set, the following table gives the conditions in which the interrupt request is asserted and deasserted.

Table 38-5. CMP interrupt generations

When	Then
C0[IER] and C0[CFR] are set	The interrupt request is asserted
C0[IEF] and C0[CFF] are set	The interrupt request is asserted
C0[IER] and C0[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
C0[IEF] and C0[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

38.11 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting C0[DMAEN] and the interrupt is enabled by setting C0[IER], C0[IEF], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it

sends a transfer completing indicator signal that deasserts the DMA transfer request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

38.12 DAC functional description

This section provides DAC functional description.

38.12.1 Digital-to-analog converter block diagram

The following figure shows the block diagram of the DAC module. It contains a 256-tap resistor ladder network and a 256-to-1 multiplexer, which selects an output voltage from one of 256 distinct levels that outputs from DACO. It is controlled through the Control register 1 (CMP_C1). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down or disabled when not in use. When in the Disabled mode, DACO is connected to the analog ground.

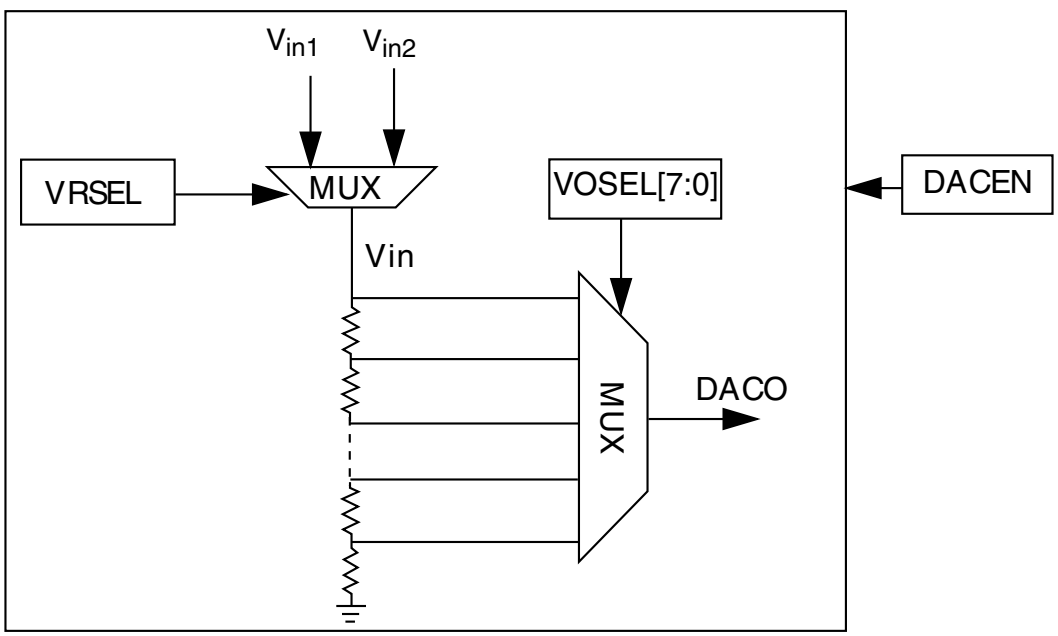


Figure 38-15. 8-bit DAC block diagram

38.12.2 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

38.12.3 DAC clocks

This module has a single clock input, the bus clock.

38.12.4 DAC interrupts

This module has no interrupts.

38.13 Trigger mode

The CMP and the 8-bit DAC are designed to support the trigger mode operation, which is enabled when the MCU enters STOP modes with C2[RRE] and C0[EN] are set.

With this mode enabled, the trigger events that include the operation clock and a trigger start signal will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output. A fixed channel for either the plus-side mux or the minus-side mux is selected by software via C2[FXMP] and C2[FXMXCH]. It is a mandatory request that the round-robin cycling period must be set longer than the time that all the active channels complete the specified comparison cycles set by C2[NSAM].

The active channels selected by C1[CHN*n*] are then routed to the non-fixed channel mux and compared with the reference input in a round-robin manner. In order to meet the comparator stabilization time, after the configurable number of operation clocks defined by C2[NSAM], the comparison result is sampled for the selected channel. A software pre-programmed state for each channel is configured by writing to C2[ACO*n*] field. After all the active channels are sampled, if the comparison result changes from its pre-programmed state, the corresponding flag in C2[CH*n*F] is set. If C2[RRIE] is set, an asynchronous reset is asserted to bring the MCU out of STOP mode.

NOTE

These flags do not support generating a DMA transfer event.

This mode is active when the MCU is in STOP mode, so none of the window/filter functions are available. A basic assumption of this mode is that the selected inputs are changing at a much slower rate than the operation clock. It is suggested to configure the comparator in low power comparison mode as well. In programming the C2[INITMOD] registers, the INITMOD × round-robin clock period must be longer than the initialization delay, which can be referred from the chip datasheet.

The following diagram shows the basic flow of this mode. In the diagram, C1[CHN1], C1[CHN3], and C1[CHN7] are set, so channels #1, #3, and #7 are selected for round-robin. C2[NSAM] is set to 2'b01, so one clock later the comparison result of the selected channel is sampled. When channel #7 is compared, the result is sampled, and round-robin ends. If any of the comparison results from channel #1, #3, or #7 changed from their programmed value (written to C2[ACO1], C2[ACO3], and C2[ACO7]), an interrupt is generated to wake up the MCU from the STOP mode. Software can then poll the C2[CH*n*F] to see which channel input(s) changed value during the STOP mode.

NOTE

In round-robin mode, it should be ensured that the RTC_CLK period is greater than the comparison time corresponding to the value of C0[PMODE]. It is also required to **not** select the internal reserved channels for round-robin by INPSEL and INNSEL.

NOTE

In round-robin mode, it is suggested to always configure the DAC output as the fixed port reference.

NOTE

In round-robin mode, current injection or over-voltage is not supported on the input channels.

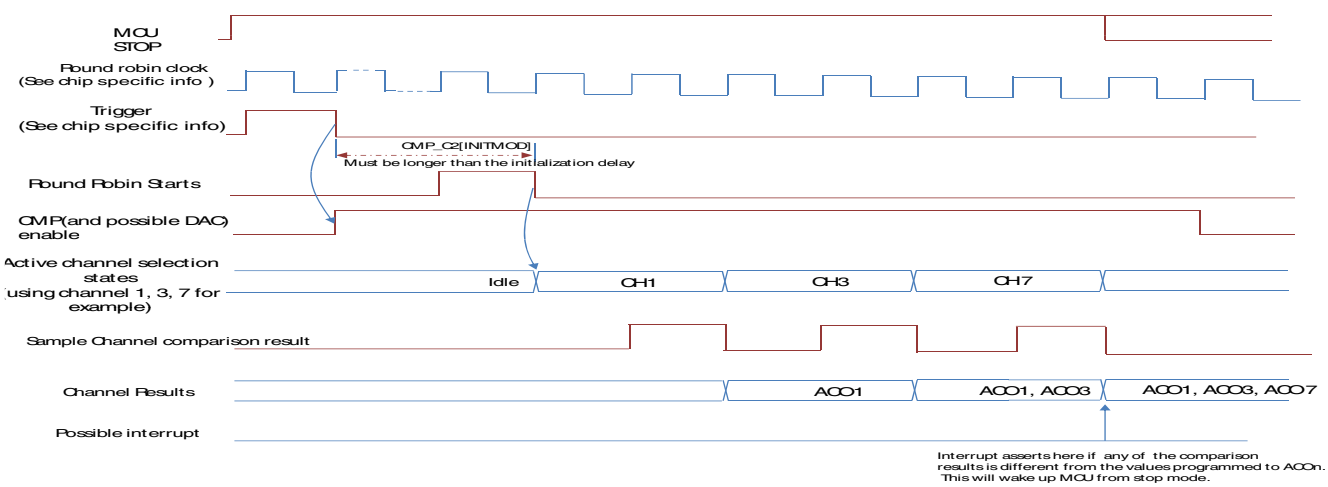


Figure 38-16. Trigger mode

The following table shows the channel decoding in both functional mode and trigger mode. Other cases not listed in the table are illegal.

Table 38-6. CMP channel decoding in functional mode and trigger mode

Mode	RRE	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INNSEL[1:0]	FXMP	FXMXCH[2:0]	CHNx	INP	INN	CMP Behavior
Functional Mode	0	x[1]	0~7	0	1	x	x	x	DAC	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with DAC
		0~7	x	1	0	x	x	x	Channel decoded from PSEL[2:0]	DAC	Channel 0~7 can be compared with DAC
		0~7	0~7	1	1	x	x	x	Channel decoded from PSEL[2:0]	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with channel 0~7[2]
Trigger Mode	1	x	x	0	1	0	x	0~7	DAC	Channel sweep (CHNx)	Channel 0~7 can be swept with DAC
		x	x	1	0	1	x	0~7	Channel sweep (CHNx)	DAC	Channel 0~7 can be swept with DAC
		x	x	1	1	0	0~7	0~7	Channel fixed by FXMXCH[2:0]	Channel sweep (CHNx)	Channel 0~7 can be swept with a fixed channel (0~7)[3]
		x	x	1	1	1	0~7	0~7	Channel sweep (CHNx)	Channel fixed by FXMXCH[2:0]	Channel 0~7 can be swept with a fixed channel (0~7)[3]

1. "x" means "do not care".
2. PSEL should not be set the same as MSEL.
3. Channel in the sweep side should not be the same as the fixed side.

38.14 Usage Guide

38.14.1 Zero Crossing Detection

A zero-crossing is a point where the sign of a signal's mathematical function changes (e.g. from positive to negative), represented by a crossing of the axis (zero value) in the graph of the signal function. It is a commonly used in electronics application especially for systems which send digital data over AC circuits.

When in some cases, the "Zero point" could be other voltage than actual 0 V. This "Zero point" would be used to judge whether the indicated voltage level is reached. In this situation, the internal DAC could generate the reference voltage level for "Zero point" to make the comparison with the other input channel of CMP module, and then output the result of logic "0" and "1".

To enable the internal DAC and set it as the comparator's input of minus side, the code could be as follow:

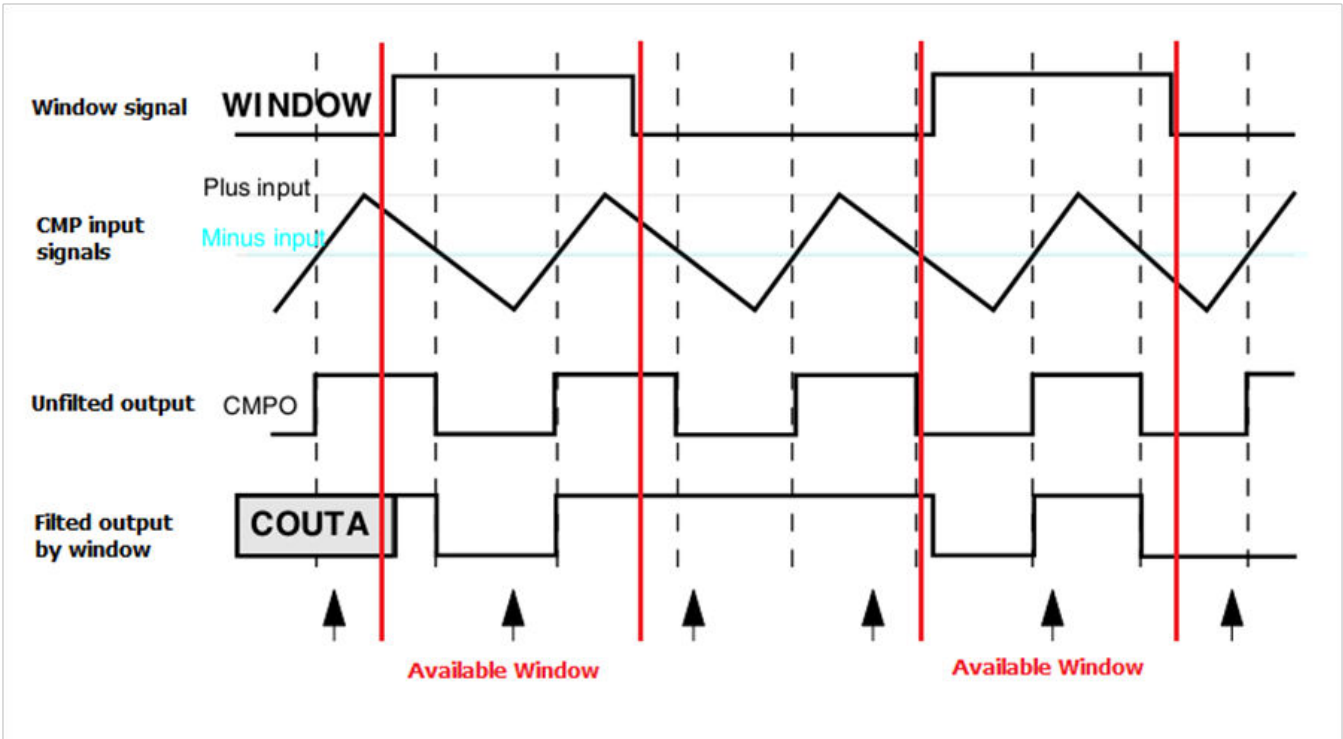
```
/* Set internal DAC as minus input. */
CMPx_C1 &= ~CMP_C1_ INNSEL_MASK;

/* Set input channel 3 as plus input. */
CMPx_C1 = (CMPx_C1 & ~(CMP_C1_ INPSEL_MASK | CMP_C1_PSEL_MASK))
          | CMP_C1_INPSEL(1) | CMP_C1_PSEL(3);
```

Then, the CMP output interrupts with their flags would be used to indicate the event of Zero Crossing Detection.

38.14.2 Window Mode

This mode could be used to create a kind of filter for input signal. When enabling the window mode, the compare would only launch the comparison in available window, which could be generated by some timer modules (e.g. PDB or LPIT). And output of CMP in unavailable window would be hold.



To enable the window mode for CMP, the code could be as follows:

```
/* Enable the window mode and disable the sample mode. */
CMPx_C0 = (CMPx_C0 & ~ CMP_C0_SE_MASK ) | CMP_C0_WE_MASK;
```

Then enable the window's generator (to produce the WINDOW signal) of related module.

For detailed information about CMP's window feature, please see to section "Windowed mode" in this chapter.

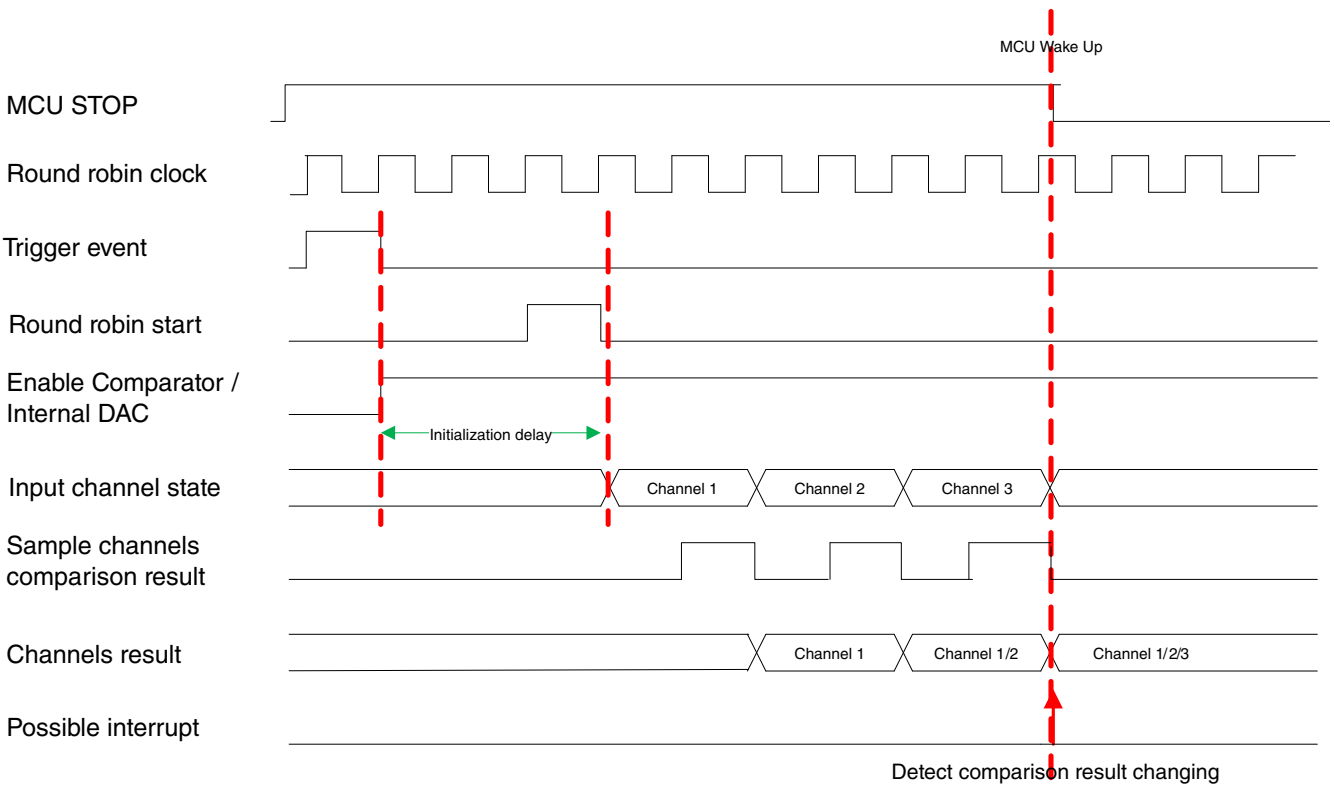
38.14.3 Round Robin Mode

This mode compares multiple input channels with the reference input channel (fixed) in a round-robin manner. It is commonly used to provide a trigger mode to wake up the MCU in STOP mode.

This mode needs some trigger events to work. The trigger events include the operation clock and a trigger start signal which can be provided by other module (e.g. LPTMR).

Round robin mode works as follows:

1. The trigger start signal will enable the comparator and internal DAC in the initialization delay period;
2. The comparator will then compare the multiple input channels with the reference input channel in turn under the operation clock until all input channels complete comparison;
3. If current comparison result is different with the pre-set state or the previous comparison result and round robin interrupt is enabled, an interrupt will generate to bring the MCU out of STOP mode.



The code snippet to enable the round robin mode is:

```
/* Set the positive port input from DAC and negative port input from minus mux input */
/* Plus mux input must be different from minus mux input even though they aren't functional
in round robin mode. */
CMPx_C1 = ((CMPx_C1 & (~(CMP_C1_INPSEL_MASK | CMP_C1_INNSEL_MASK | CMP_C1_PSEL_MASK |
CMP_C1_MSEL_MASK)))
          | (CMP_C1_INPSEL(0) | CMP_C1_INNSEL(1) | CMP_C1_PSEL(0) | CMP_C1_MSEL(1)));

/* Set following round robin attribute:
positive port as fixed port.
All channel0~7 as the round robin checker channel in non-fixed port.
The comparison result is sampled as soon as the active channel is scanned in one round-robin
clock.
The initialization delay modulus is set to 64.
Enable round robin mode.
```

```
Enable round robin interrupt.
*/
CMPx_C1 = ((CMPx_C1 & (~(CMP_C1_CHN0_MASK | CMP_C1_CHN1_MASK | CMP_C1_CHN2_MASK |
CMP_C1_CHN3_MASK |
           CMP_C1_CHN4_MASK | CMP_C1_CHN5_MASK | CMP_C1_CHN6_MASK | CMP_C1_CHN7_MASK)))
           |(0xFF << CMP_C1_CHN0_SHIFT));

CMPx_C2 = ((CMPx_C2 & (~(CMP_C2_FXMP_MASK | CMP_C2_FXMXCH_MASK | CMP_C2_NSAM_MASK
          | CMP_C2_INITMOD_MASK | CMP_C2_CHnF_MASK))) | (CMP_C2_FXMP(0)
          | CMP_C2_FXMXCH(0) | CMP_C2_NSAM(0)
          | CMP_C2_INITMOD(0) | CMP_C2_RRE_MASK | CMP_C2_RRIE_MASK));

/* Set all the pre-state of round robin checker channel0~7 to 1. */
CMPx ->C2 = ((CMPx ->C2 & (~CMP_C2_ACOn_MASK | CMP_C2_CHnF_MASK)) | (0xFF <<
CMP_C2_ACOn_SHIFT));

/* Set round robin comparison trigger. See the chip configuration about the available
trigger in the SoC. */

/* Set SoC enter into STOP mode. See the power management chapter. */

/* Change the voltage of input channel to wake up the SoC. */
```

Chapter 39
12-bit Digital-to-Analog Converter (DAC)

39.1 Chip-specific information for this module

39.1.1 Instantiation information

This device contains one 12-bit digital-to-analog converter (DAC) with programmable reference generator output. The DAC includes a 16 words FIFO for DMA support.

39.1.1.1 12-bit DAC Reference

For this device, VREFH and VDDA are selectable as the DAC reference. VREFH is connected to the DACREF_1 input and VDDA is connected to the DACREF_2 input. Use DAC0_STATCTRL[DACRFS] to select the required reference.

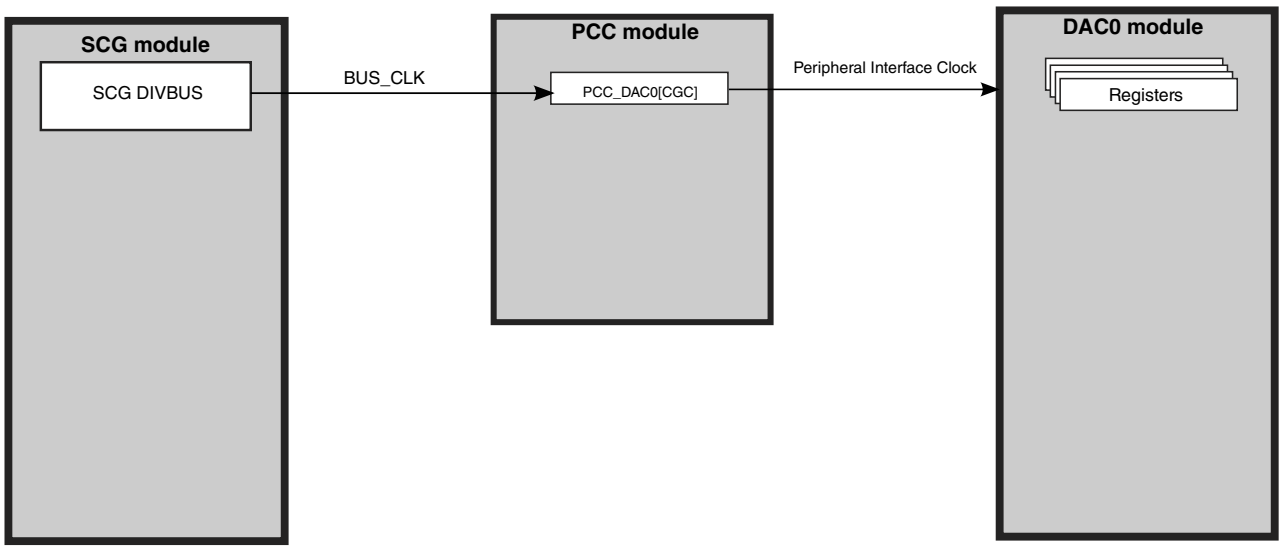
NOTE

If the DAC and ADC use the same reference simultaneously, some degradation of ADC accuracy is to be expected due to DAC switching.

39.1.2 DAC Clocking Information

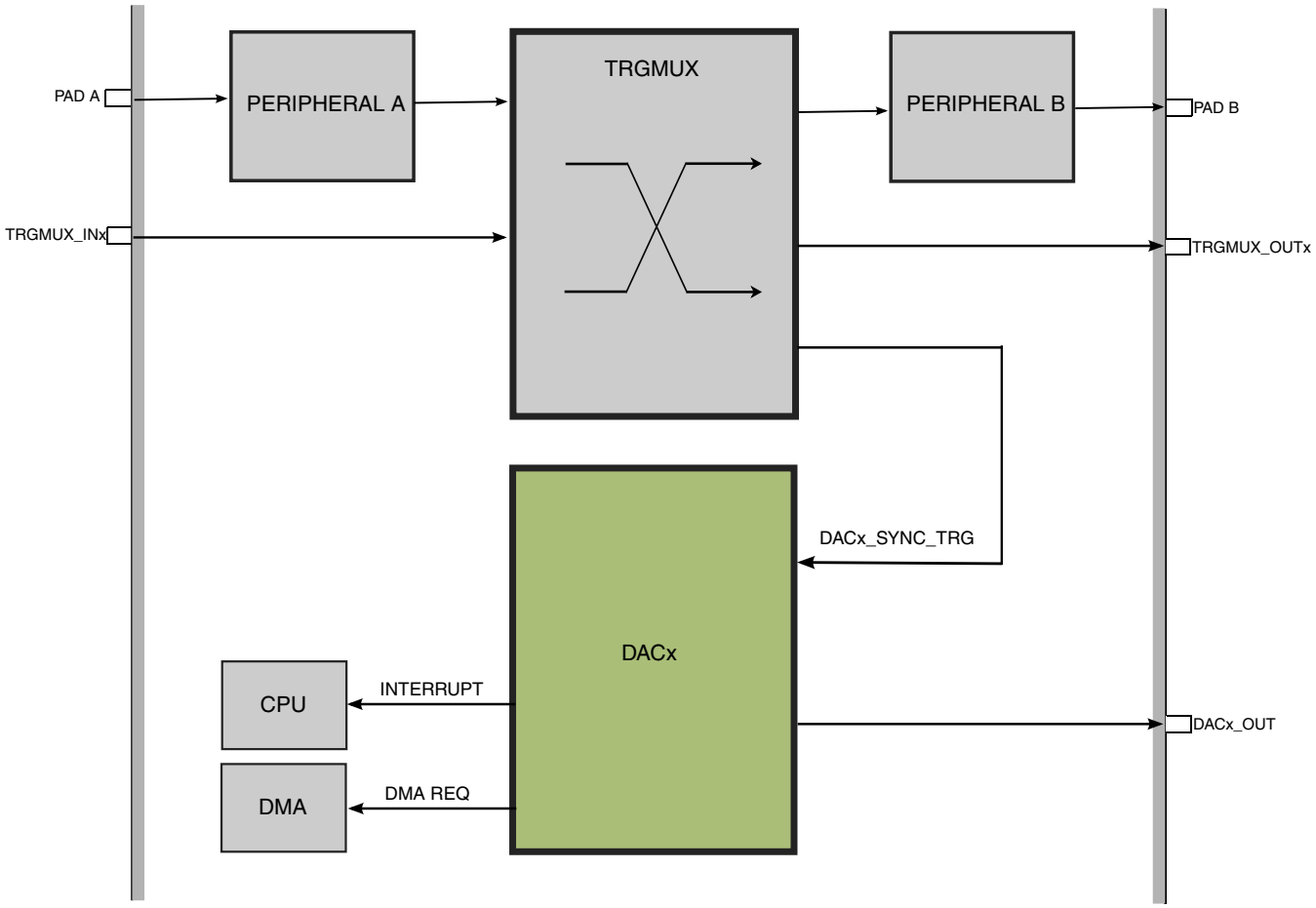
The DAC clocking input is as below.

Peripheral Clocking - DAC



39.1.3 Inter-connectivity Information

The DAC inter-connectivity is shown in following diagram.



39.2 Introduction

The 12-bit digital-to-analog converter (DAC) is a low-power, general-purpose DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator, op-amps, or ADC.

39.3 Features

The features of the DAC module include:

- 2.7 V to 5.5 V operation.
- On-chip programmable reference generator output. The voltage output range is from 1⁄4096 V_{in} to V_{in}, and the step is 1⁄4096 V_{in}, where V_{in} is the input voltage.

- V_{in} can be selected from two reference sources
- Static operation in Normal Stop mode
- 16-word data buffer supported with configurable watermark and multiple operation modes
- DMA support

39.4 Block diagram

The block diagram of the DAC module is as follows:

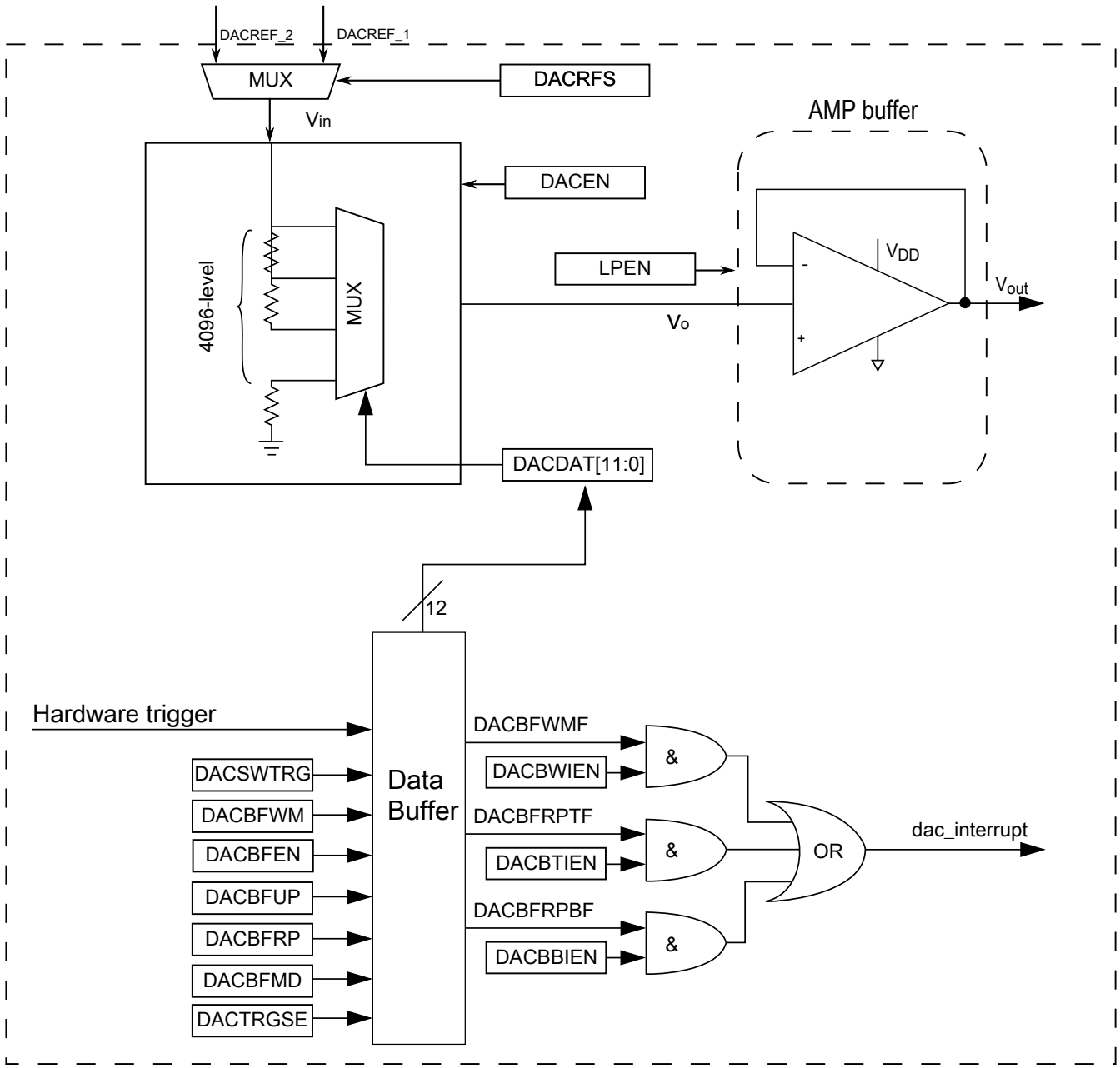


Figure 39-1. DAC block diagram

39.5 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

DAC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_F000	DAC Data Register (DAC0_DAT0)	32	R/W	0000_0000h	39.5.1/912
4003_F004	DAC Data Register (DAC0_DAT1)	32	R/W	0000_0000h	39.5.1/912
4003_F008	DAC Data Register (DAC0_DAT2)	32	R/W	0000_0000h	39.5.1/912
4003_F00C	DAC Data Register (DAC0_DAT3)	32	R/W	0000_0000h	39.5.1/912
4003_F010	DAC Data Register (DAC0_DAT4)	32	R/W	0000_0000h	39.5.1/912
4003_F014	DAC Data Register (DAC0_DAT5)	32	R/W	0000_0000h	39.5.1/912
4003_F018	DAC Data Register (DAC0_DAT6)	32	R/W	0000_0000h	39.5.1/912
4003_F01C	DAC Data Register (DAC0_DAT7)	32	R/W	0000_0000h	39.5.1/912
4003_F020	DAC Status and Control Register (DAC0_STATCTRL)	32	R/W	See section	39.5.2/913

39.5.1 DAC Data Register (DAC*x*_DAT*n*)

Address: 4003_F000h base + 0h 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	0				DATA1												0				DATA0											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAC*x*_DAT*n* field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–16 DATA1	DATA1 When the DAC buffer is not enabled, this field does not control the output voltage. When the DAC buffer is enabled, this field is mapped to the 16-word buffer.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA0	DATA0 When the DAC buffer is not enabled, this field controls the output voltage based on the following formula: $V_{out} = V_{in} * (1 + DAC\_DATn[DATA0])/4096$ When the DAC buffer is enabled, this field is mapped to the 16-word buffer.

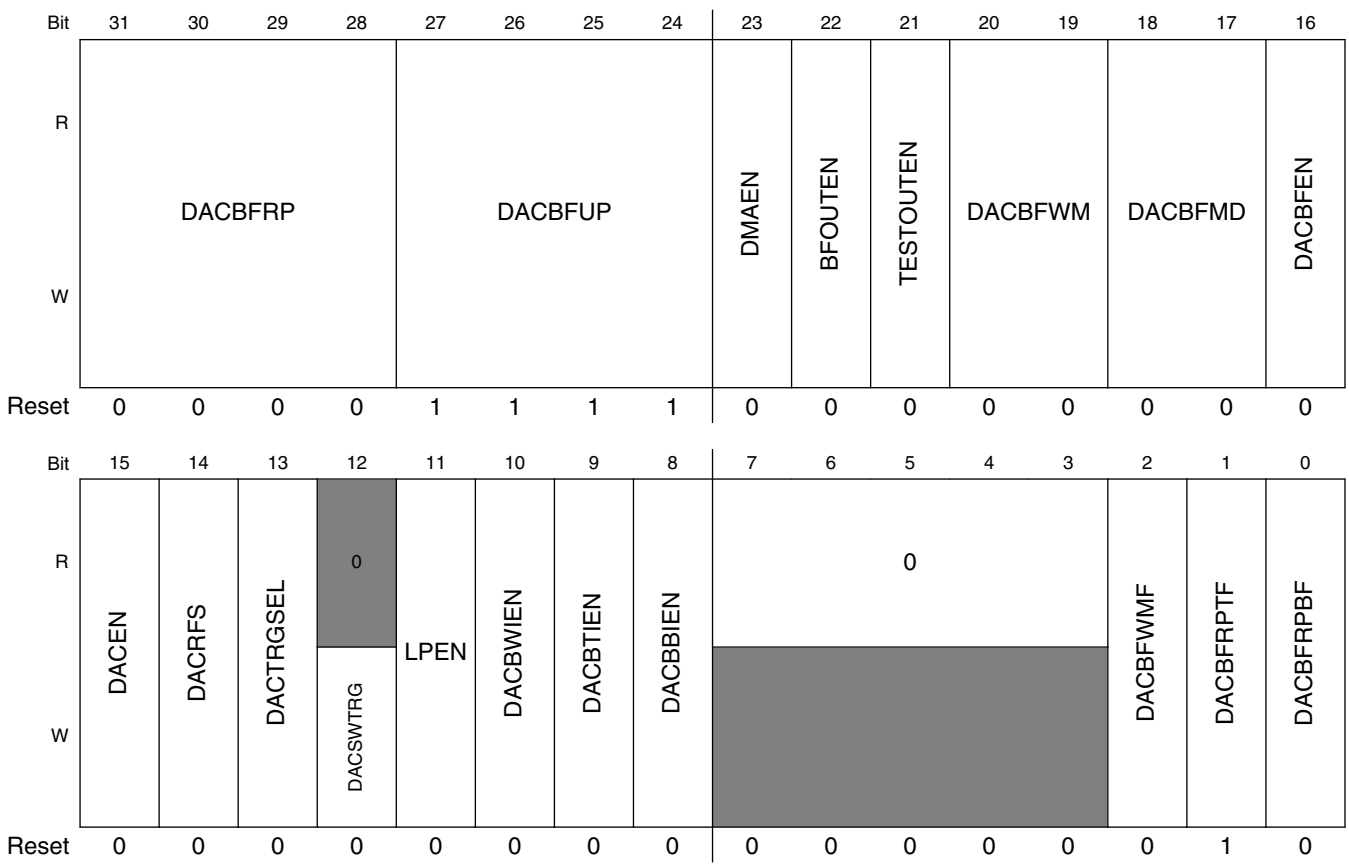
39.5.2 DAC Status and Control Register (DACx_STATCTRL)

If DMA is enabled, the flags can be cleared automatically by DMA when the DMA request is done. Writing 0 to a field clears it whereas writing 1 has no effect. After reset, DACBFRPTF is set and can be cleared by software, if needed. The flags are set only when the data buffer status is changed.

NOTE

In FIFO mode, when only one data remains, any DAC trigger does not increase the read pointer, which mean the FIFO is never empty. This is to avoid the buffer under-run. When FIFO is full, any write to the FIFO buffer is ignored and read/write pointers remain unchanged.

Address: 4003_F000h base + 20h offset = 4003_F020h



DACx_STATCTRL field descriptions

Field	Description
31–28 DACBFRP	DAC Buffer Read Pointer

Table continues on the next page...

DACx_STATCTRL field descriptions (continued)

Field	Description
	FIFO mode, it is the FIFO read pointer. It is writable in FIFO mode. User can configure it to same address to reset FIFO as empty.
27–24 DACBFUP	DAC Buffer Upper Limit In FIFO mode it is the FIFO write pointer. User cannot set Buffer Up limit in FIFO mode. In Normal mode its reset value is MAX. When the module is configured to FIFO mode, this register becomes Write_Pointer. It is writable and user can configure it to the same address to reset FIFO as empty. **NOTE:** After FIFO mode enabled, user needs to reset FIFO by writing the pointer to the same address with Read_Pointer.
23 DMAEN	DMA Enable Select 0 DMA is disabled. 1 DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.
22 BFOUTEN	DAC output buffer enable 0 Disable DAC output buffer 1 Enable DAC output buffer
21 TESTOUTEN	DAC test output enable 0 Disable DAC test output 1 Enable DAC test output
20–19 DACBFWM	DAC Buffer Watermark Select In normal mode it controls when DACBFWMF is set. When the DAC buffer read pointer reaches the word defined by this field, which is 1–4 words away from the upper limit (DACBUP), DACBFWMF is set. This allows user configuration of the watermark interrupt. In FIFO mode, it is FIFO watermark select field. 00 In normal mode, 1 word . In FIFO mode, 2 or less than 2 data remaining in FIFO will set watermark status bit. 01 In normal mode, 2 words . In FIFO mode, Max/4 or less than Max/4 data remaining in FIFO will set watermark status bit. 10 In normal mode, 3 words . In FIFO mode, Max/2 or less than Max/2 data remaining in FIFO will set watermark status bit. 11 In normal mode, 4 words . In FIFO mode, Max-2 or less than Max-2 data remaining in FIFO will set watermark status bit. **NOTE:** Max is equal to the FIFO depth.
18–17 DACBFMD	DAC Buffer Work Mode Select 00 Normal mode 01 Swing mode 10 One-Time Scan mode 11 FIFO mode
16 DACBFEN	DAC Buffer Enable 0 Buffer read pointer is disabled. The converted data is always the first word of the buffer. 1 Buffer read pointer is enabled. The converted data is the word that the read pointer points to. It means converted data can be from any word of the buffer.

Table continues on the next page...

DACx_STATCTRL field descriptions (continued)

Field	Description
15 DACEN	DAC Enable Starts the Programmable Reference Generator operation. 0 The DAC system is disabled. 1 The DAC system is enabled.
14 DACRFS	DAC Reference Select 0 The DAC selects DACREF_1 as the reference voltage. 1 The DAC selects DACREF_2 as the reference voltage.
13 DACTRGSEL	DAC Trigger Select 0 The DAC hardware trigger is selected. 1 The DAC software trigger is selected.
12 DACSWTRG	DAC Software Trigger Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and buffer is enabled, writing 1 to this field will advance the buffer read pointer once. 0 The DAC soft trigger is not valid. 1 The DAC soft trigger is valid.
11 LPEN	DAC Low Power Control **NOTE:** See the 12-bit DAC electrical characteristics of the device data sheet for details on the impact of the modes below. 0 High-Power mode 1 Low-Power mode
10 DACBWIEN	DAC Buffer Watermark Interrupt Enable 0 The DAC buffer watermark interrupt is disabled. 1 The DAC buffer watermark interrupt is enabled.
9 DACBTIEN	DAC Buffer Read Pointer Top Flag Interrupt Enable 0 The DAC buffer read pointer top flag interrupt is disabled. 1 The DAC buffer read pointer top flag interrupt is enabled.
8 DACBBIEN	DAC Buffer Read Pointer Bottom Flag Interrupt Enable 0 The DAC buffer read pointer bottom flag interrupt is disabled. 1 The DAC buffer read pointer bottom flag interrupt is enabled.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 DACBFWMF	DAC Buffer Watermark Flag In FIFO mode it is FIFO watermark status flag. This bit is set if the remaining FIFO data is less than the watermark setting. It is cleared automatically by writing data into FIFO by DMA or CPU. Write to this bit is ignored in FIFO mode. 0 The DAC buffer read pointer has not reached the watermark level. 1 The DAC buffer read pointer has reached the watermark level.

Table continues on the next page...

DACx_STATCTRL field descriptions (continued)

Field	Description
1 DACBFRPTF	DAC Buffer Read Pointer Top Position Flag In FIFO mode it is FIFO nearly empty flag. It is set when only one data remains in FIFO. Any DAC trigger does not increase the Read Pointer if this bit is set to avoid any possible glitch or abrupt change at DAC output. It is cleared automatically if FIFO is not empty. 0 The DAC buffer read pointer is not zero. 1 The DAC buffer read pointer is zero.
0 DACBFRPBF	DAC Buffer Read Pointer Bottom Position Flag In FIFO mode it is FIFO FULL status bit. It means FIFO read pointer equals Write Pointer because of Write Pointer increase. If this bit is set, any write to FIFO from either DMA or CPU is ignored by DAC. It is cleared if there is any DAC trigger making the DAC read pointer increase. Write to this bit is ignored in FIFO mode. 0 The DAC buffer read pointer is not equal to DACBFUP. 1 The DAC buffer read pointer is equal to DACBFUP.

39.6 Functional description

The 12-bit DAC module can select one of the two reference inputs—DACREF_1 and DACREF_2 as the DAC reference voltage, V_{in} by STATCTRL [DACRFS]. See the chip-specific DAC information to determine the source options for DACREF_1 and DACREF_2.

When the DAC is enabled, it converts the data in DACDAT0[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

39.6.1 DAC data buffer operation

When the DAC is enabled and the buffer is not enabled, the DAC module always converts the data in DAT0 to the analog output voltage.

When both the DAC and the buffer are enabled, the DAC converts the data in the data buffer to analog output voltage. The data buffer read pointer advances to the next word whenever a hardware or software trigger event occurs.

The data buffer can be configured to operate in Normal mode, Swing mode, One-Time Scan mode or FIFO mode. When the buffer operation is switched from one mode to another, the read pointer does not change. The read pointer can be set to any value between 0 and STATCTRL[DACBFUP] by writing STATCTRL[DACBFRP].

39.6.1.1 DAC data buffer interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer. STATCTRL[DACBFRPBF] is set when the DAC buffer read pointer reaches the DAC buffer upper limit, that is, STATCTRL[DACBFRP] = STATCTRL[DACBFUP]. STATCTRL[DACBFRPTF] is set when the DAC read pointer is equal to the start position, 0. Finally, STATCTRL[DACBFWMF] is set when the DAC buffer read pointer has reached the position defined by STATCTRL[DACBFWM]. STATCTRL[DACBFWM] can be used to generate an interrupt when the DAC buffer read pointer is between 1 to 4 words from STATCTRL[DACBFUP].

39.6.1.2 Modes of DAC data buffer operation

The following table describes the different modes of data buffer operation for the DAC module.

Table 39-1. Modes of DAC data buffer operation

Modes	Description
Buffer Normal mode	This is the default mode. The buffer works as a circular buffer. The read pointer increases by one, every time the trigger occurs. When the read pointer reaches the upper limit, it goes to 0 directly in the next trigger event.
Buffer Swing mode	This mode is similar to the normal mode. However, when the read pointer reaches the upper limit, it does not go to 0. It will descend by 1 in the next trigger events until 0 is reached.
Buffer One-time Scan mode	The read pointer increases by 1 every time the trigger occurs. When it reaches the upper limit, it stops there. If read pointer is reset to the address other than the upper limit, it will increase to the upper address and stop there again. **NOTE:** If the software set the read pointer to the upper limit, the read pointer will not advance in this mode.
FIFO Mode	In FIFO mode, the buffer is organized as a FIFO. For a valid write to any DACDATx, the data is put into the FIFO, and the write pointer is automatically incremented. The module is connected internally to a 32bit interface. For any 16bit or 8bit FIFO access, address bit[1] needs to be 0; otherwise, the write is ignored. For any 32bit FIFO access, the Write_Pointer needs to be an EVEN number; otherwise, the write is ignored. **NOTE:** A successful 32bit FIFO write will increase the write pointer by 2. Any write will cause the FIFO over-flow will be ignored, the cases includes: 1.FIFO is full, the write will be ignored. 2.FIFO is nearly full (FIFO_SIZE-1), 32bit write will be ignored. **NOTE:** For 8bit write, address bit[0] determine which byte lane will be written to the FIFO according to little endian alignment. Only both byte lanes are written

Table 39-1. Modes of DAC data buffer operation

Modes	Description
	will the write pointer increase. User need to make sure 8bit access happened in pair and both upper & lower bytes are written. There is no requirement on which byte write first. In FIFO mode, there is no change to read access of DACDATx (from normal mode), read to DACDATx will return the DATA addressed by the access address to the data buffer, and both write pointer and read pointer in FIFO mode will NOT be changed by read access. FIFO write can be happened when DAC is not enabled for 1st data conversion enable. But FIFO mode need to work at buffer Enabled at DACC1[DACBFEN]. In FIFO mode, the DATA BUF will be organized as FIFO.

39.6.2 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

39.6.3 Resets

During reset, the DAC is configured in the default mode and is disabled.

39.6.4 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

Table 39-2. Modes of operation

Modes of operation	Description
Wait mode	The DAC will operate normally, if enabled.
Stop mode	If enabled, the DAC module continues to operate in Normal Stop mode and the output voltage will hold the value before stop. In low-power stop modes, the DAC is fully shut down.

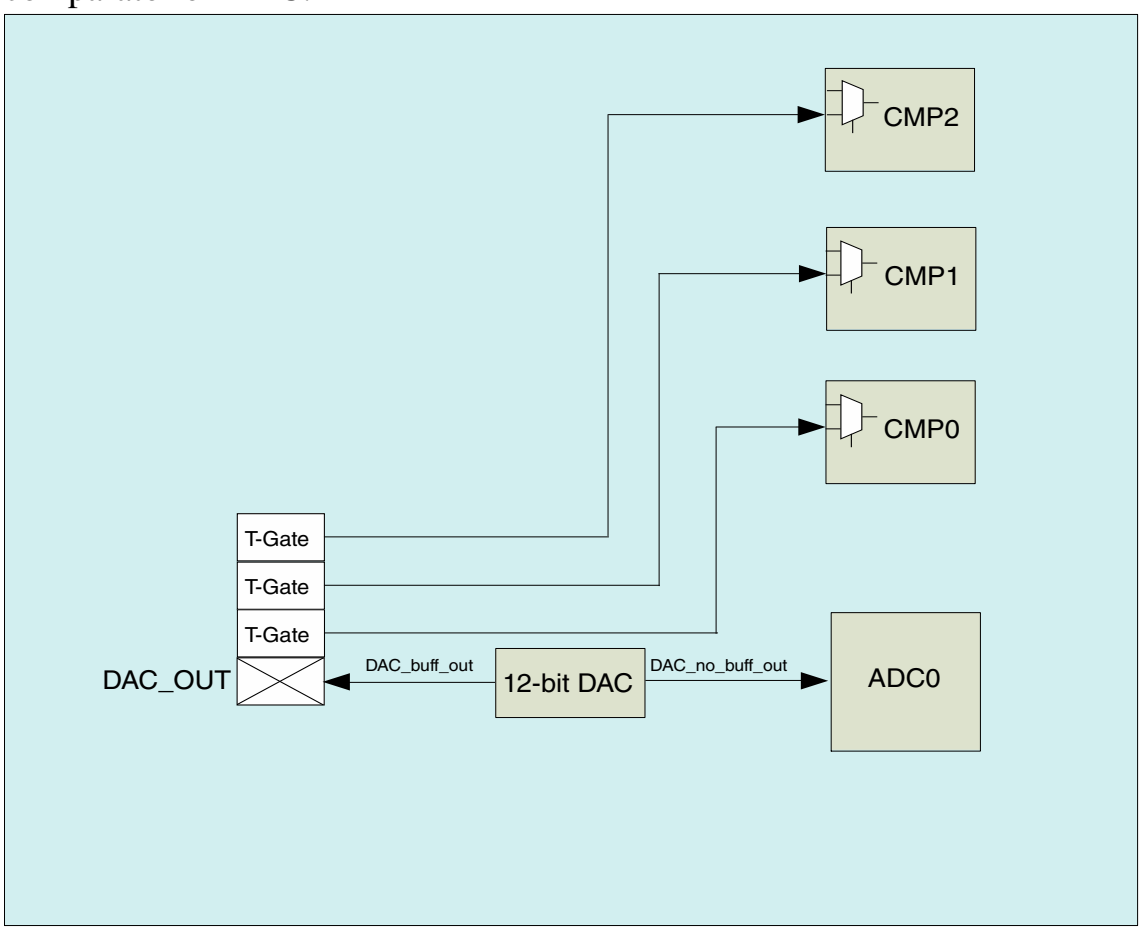
NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

39.7 Usage Guide

39.7.1 12-bit DAC Output

The output of the DAC can be placed on an external pin or set as input to the analog comparator or ADC.



39.7.2 12-bit DAC Reference

For this device, VREFH and VDDA are selectable as the DAC reference. VREFH is connected to the DACREF_1 input and VDDA is connected to the DACREF_2 input. Use DAC0_STATCTRL[DACRFS] to select the required reference.

NOTE

If the DAC and ADC use the same reference simultaneously, some degradation of ADC accuracy is to be expected due to DAC switching.

39.7.3 12-bit DAC FIFO

There are applications requiring the DAC outputs a waveform with specific DAC refresh rate. If the waveform frequency is high, it could mean considerable amount of CPU load and thus the CPU may need to run at much higher frequency.

This device includes an on-chip DMA controller that can be used to transfer the data to the DAC module without impacting CPU performance. To allow this, the DAC module needs a FIFO buffer with configurable watermark level (interrupt on FIFO full, FIFO empty, and as indicated by watermark register) that allows to generate DMA request/ acknowledge signals based on watermark level settings. The FIFO depth implemented on this device is 16 words.

Chapter 40
Programmable Delay Block (PDB)

40.1 Chip-specific information for this module

40.1.1 Instantiation Information

40.1.1.1 PDB Configuration

There are 3 PDB modules on this device.

Number of ADC channel	1
Number of pre-triggers per ADC channel	8
Number of DAC triggers	1
Number of Pulse Out	1

40.1.1.2 PDB Input Trigger Connections

On this device, the PDB trigger source selection is implemented through the TRGMUX module. For each PDB unit, there is only one trigger input from TRGMUX, but it supports different trigger sources. The internal trigger mux inside PDB is not used any more.

PDB Trigger	PDB Input
0000	TRGMUX_PDB0_EXTRG

40.1.2 PDB Clock Options

The PDB module is clocked by the system clock (SYS_CLK). The SYS_CLK could run up to CPU frequency which provides higher timing resolution and more precise delay control with the PDB counter.

40.1.3 PDB Module Interconnections

Besides the specific PDB to ADC triggering scheme (refer to the "ADC trigger source" section), the PDB channel triggers can also work as trigger source of TRGMUX module, which can be used to trigger other modules besides ADC.

Following are PDB module inter-connectivities:

PDBx Trigger outputs	Interconnectivity
ADC Channel 0 triggers	ADC trigger connects to both ADC and TRGMUX
DAC triggers	DAC trigger connects to TRGMUX
Pulse-out	Pulse-out connects to TRGMUX

40.1.3.1 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, the following PDB back-to-back operation acknowledgment connections are implemented based on SIM_CHIPCTL[PDB_BB_SEL] bit setting.

When SIM_CHIPCTL[PDB_BB_SEL]=0:

The PDB back-to-back operation acknowledgement connections are implemented inside each PDB unit as a ring. The following list is an example for PDB0.

- PDB0 channel 0 pre-trigger 0 acknowledgement input: ADC0SC1H_COCO
- PDB0 channel 0 pre-trigger 1 acknowledgement input: ADC0SC1A_COCO
- PDB0 channel 0 pre-trigger 2 acknowledgement input: ADC0SC1B_COCO
- PDB0 channel 0 pre-trigger 3 acknowledgement input: ADC0SC1C_COCO
- PDB0 channel 0 pre-trigger 4 acknowledgement input: ADC0SC1D_COCO
- PDB0 channel 0 pre-trigger 5 acknowledgement input: ADC0SC1E_COCO
- PDB0 channel 0 pre-trigger 6 acknowledgement input: ADC0SC1F_COCO
- PDB0 channel 0 pre-trigger 7 acknowledgement input: ADC0SC1G_COCO

The back-to-back chain diagram is as follows:

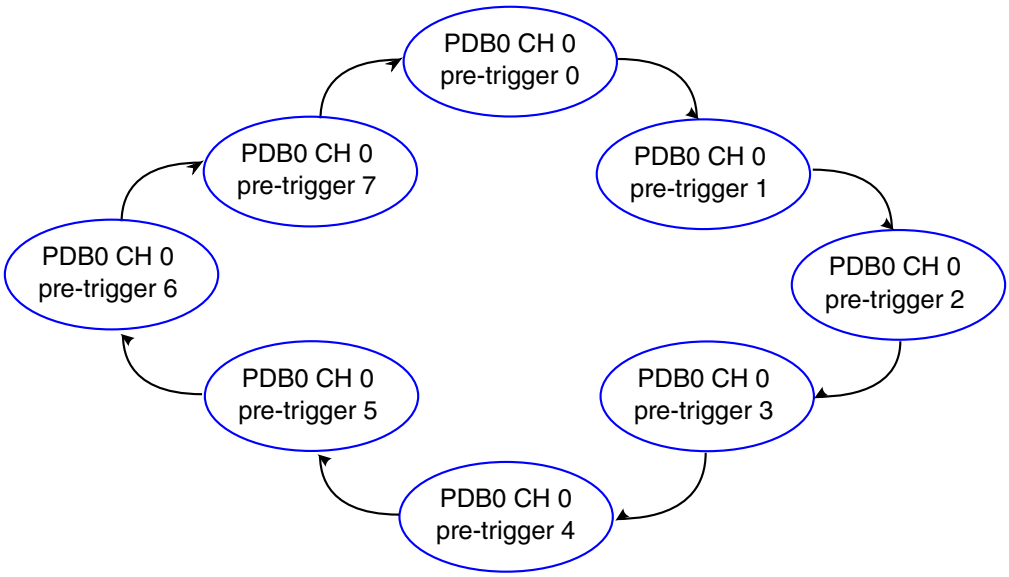


Figure 40-1. PDB0 back-to-back chain

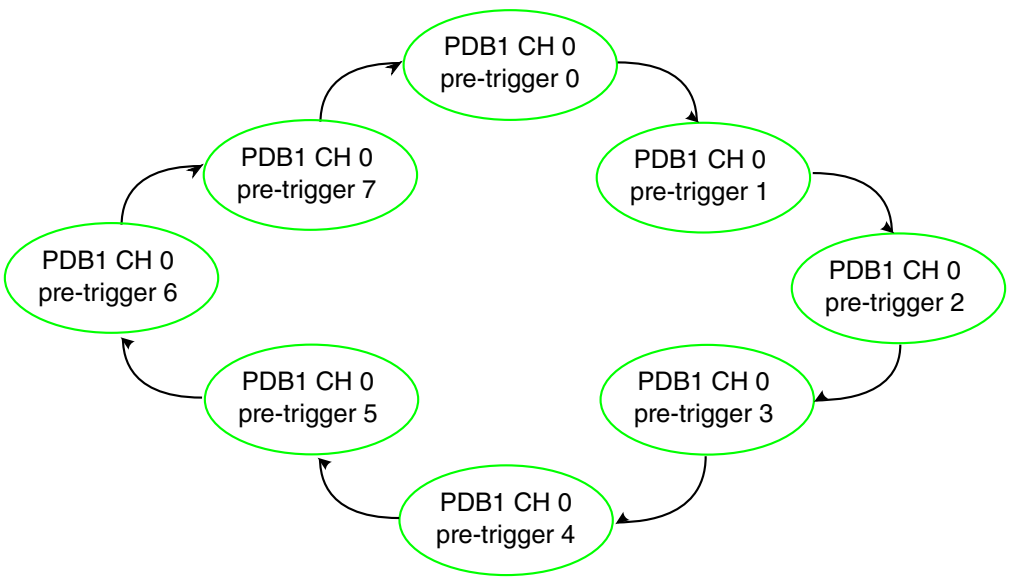


Figure 40-2. PDB1 back-to-back chain

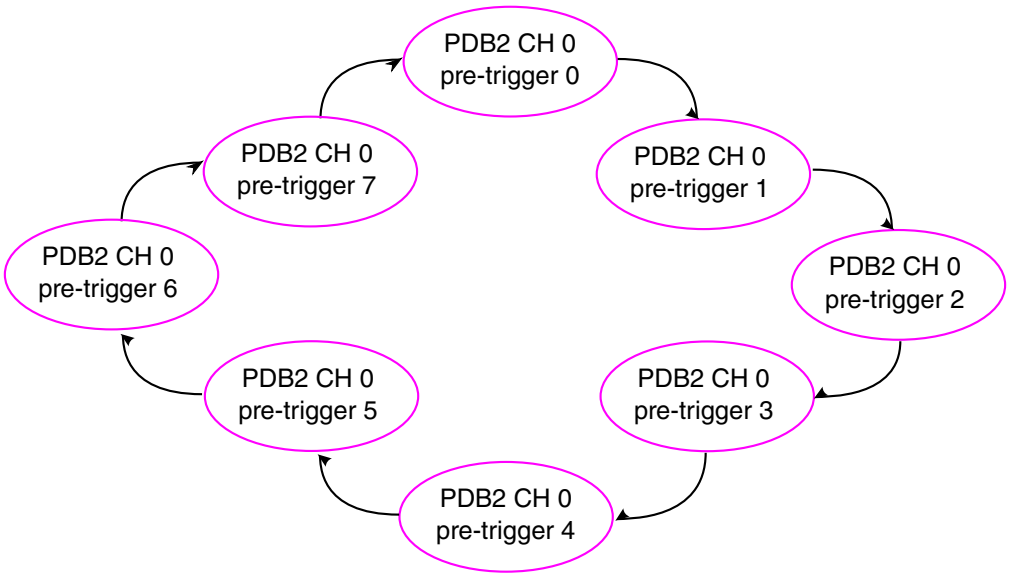


Figure 40-3. PDB2 back-to-back chain

When SIM_CHIPCTL[PDB_BB_SEL]=1:

The PDB back-to-back operation acknowledgement connections are implemented with all of the PDB units as a ring. The chain diagram is as follows:

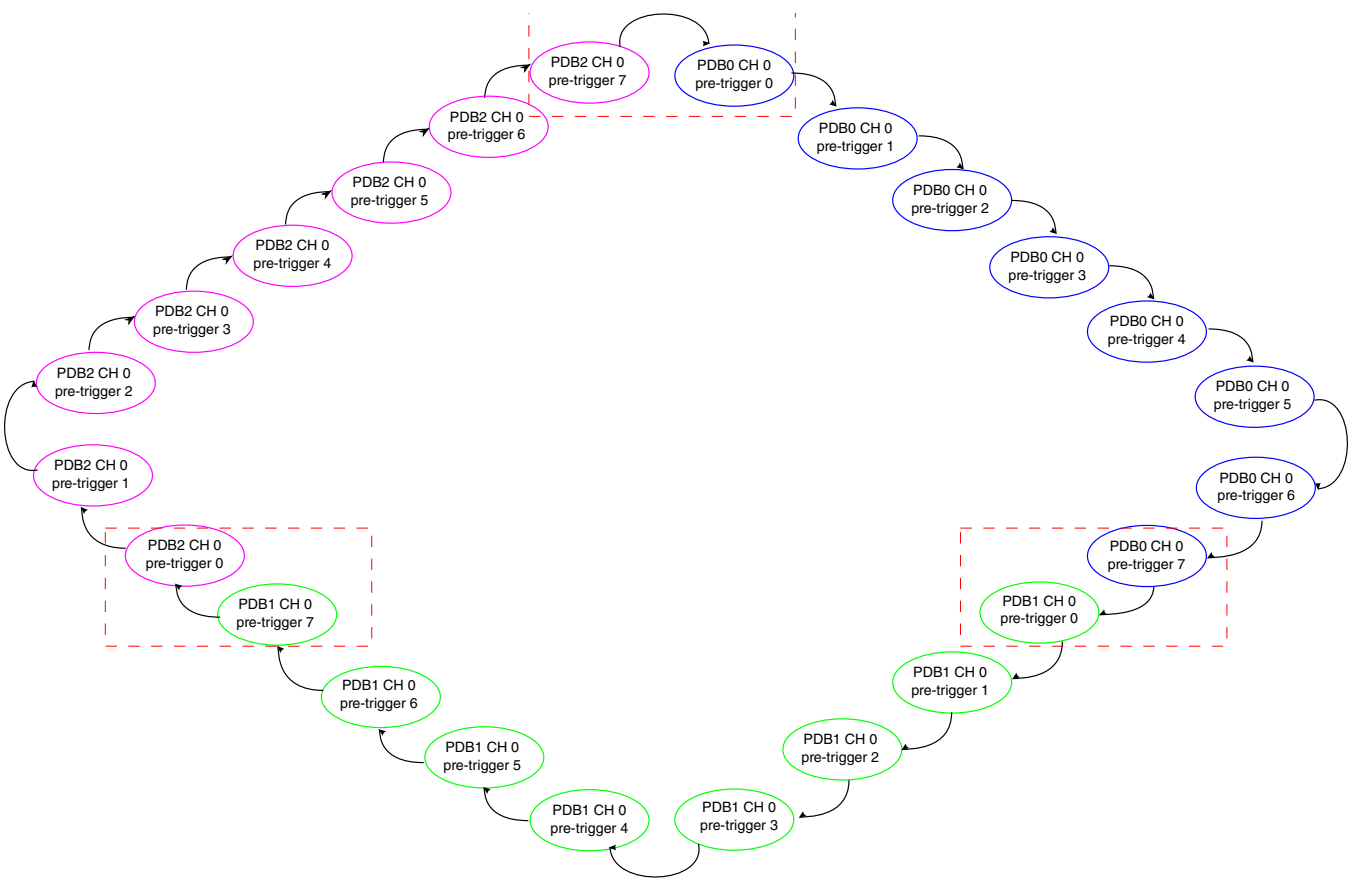


Figure 40-4. PDB back-to-back chain

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

40.1.3.2 PDB Interval Trigger Connections to DAC

In this MCU, each PDB unit supports interval DAC hardware trigger. The PDB interval trigger connections to DAC are implemented through TRGMUX, the user could select the DAC interval hardware trigger from any of the PDB unit. The TRGMUX provides the user a very flexible way for DAC interval hardware trigger.

40.1.3.3 DAC External Trigger Input Connections

The DAC interval trigger from PDB could be controlled by PDB interval counter or by external hardware trigger input (which is controlled by the EXT bit inside PDBx_DACINTCn register). In this MCU, the following DAC external trigger inputs are implemented:

- PDB0 DAC external trigger: ADC0SC1A_COCO
- PDB1 DAC external trigger: ADC1SC1A_COCO
- PDB2 DAC external trigger: ADC2SC1A_COCO

40.1.3.4 Pulse-Out Connection

Individual PDB Pulse-Out signals are connected to TRGMUX, which then could be routed to each CMP block and used for sample windows. Please refer to TRGMUX section for details.

40.1.3.5 Pulse-Out Enable Register Implementation

The following table shows the comparison of pulse-out enable register at the module and chip level. In this device, each PDB unit has one Pulse-Out channel. The Pulse-Out is connecting to TRGMUX, which is then flexible to work as sample window for any CMP module.

Table 40-1. PDB pulse-out enable register

Register	Module implementation	Chip implementation
PDBx_POEN	7:0 - POEN 31:8 - Reserved	0 - POEN[0] for CMP 31:1 - Reserved

40.2 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.

40.2.1 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.

40.2.2 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.

40.2.3 Back-to-back acknowledgment connections

PDB back-to-back operation acknowledgment connections are chip-specific. For implementation, see the chip configuration information.

40.2.4 DAC External Trigger Input Connections

The implementation of DAC external trigger inputs is chip-specific. See the chip configuration information for details.

40.2.5 Block diagram

This diagram illustrates the major components of the PDB.

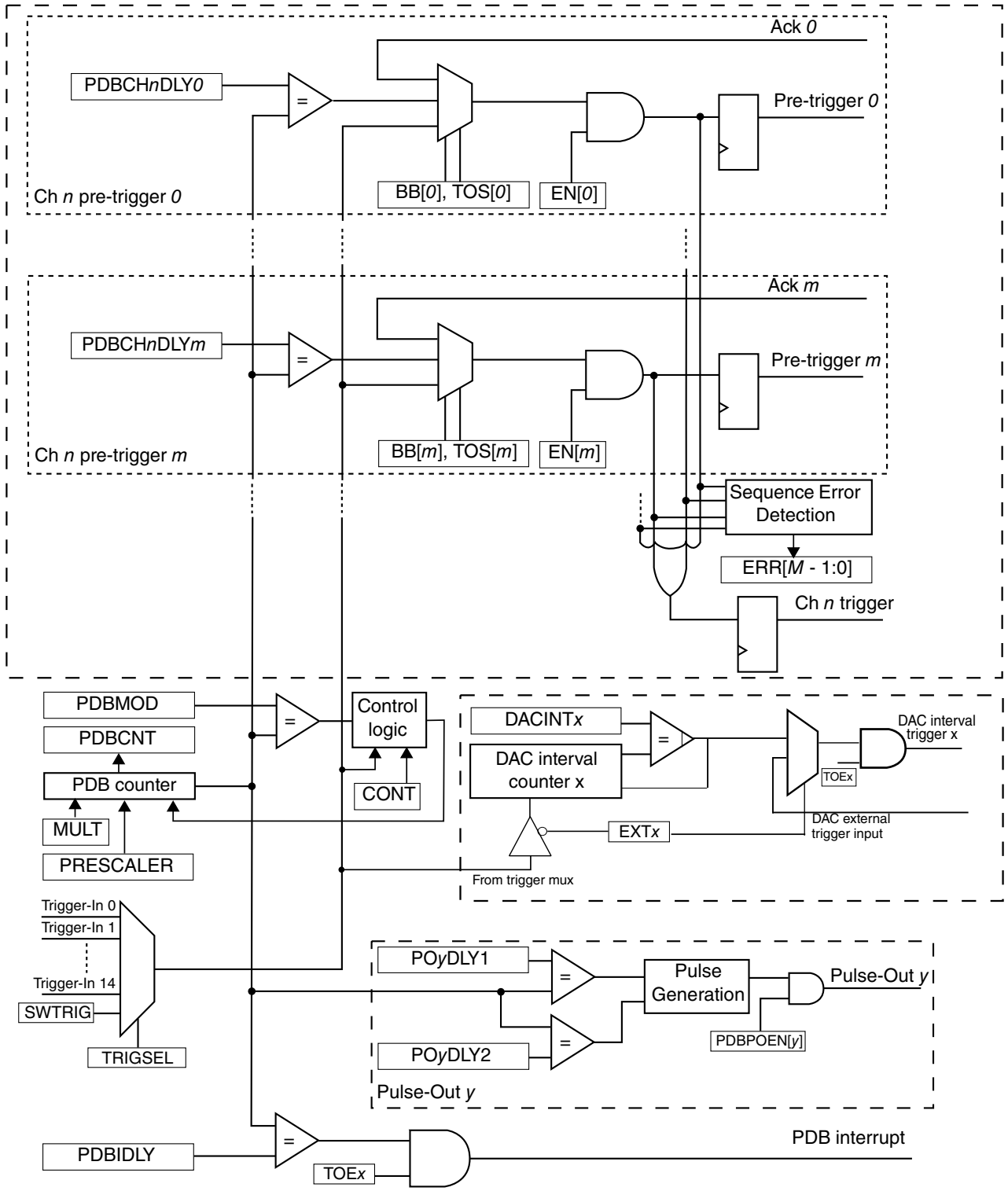


Figure 40-5. PDB block diagram

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.

40.2.6 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.

40.3 PDB signal descriptions

This table shows the detailed description of the external signal.

Table 40-2. 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

40.4 Memory map and register definition

PDB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_1000	Status and Control register (PDB1_SC)	32	R/W	0000_0000h	40.4.1/933
4003_1004	Modulus register (PDB1_MOD)	32	R/W	0000_FFFFh	40.4.2/936
4003_1008	Counter register (PDB1_CNT)	32	R	0000_0000h	40.4.3/936
4003_100C	Interrupt Delay register (PDB1_IDLY)	32	R/W	0000_FFFFh	40.4.4/937
4003_1010	Channel n Control register 1 (PDB1_CH0C1)	32	R/W	0000_0000h	40.4.5/937
4003_1014	Channel n Status register (PDB1_CH0S)	32	R/W	0000_0000h	40.4.6/938
4003_1018	Channel n Delay 0 register (PDB1_CH0DLY0)	32	R/W	0000_0000h	40.4.7/939
4003_101C	Channel n Delay 1 register (PDB1_CH0DLY1)	32	R/W	0000_0000h	40.4.8/940
4003_1020	Channel n Delay 2 register (PDB1_CH0DLY2)	32	R/W	0000_0000h	40.4.9/940
4003_1024	Channel n Delay 3 register (PDB1_CH0DLY3)	32	R/W	0000_0000h	40.4.10/ 941
4003_1028	Channel n Delay 4 register (PDB1_CH0DLY4)	32	R/W	0000_0000h	40.4.11/ 942
4003_102C	Channel n Delay 5 register (PDB1_CH0DLY5)	32	R/W	0000_0000h	40.4.12/ 942
4003_1030	Channel n Delay 6 register (PDB1_CH0DLY6)	32	R/W	0000_0000h	40.4.13/ 943
4003_1034	Channel n Delay 7 register (PDB1_CH0DLY7)	32	R/W	0000_0000h	40.4.14/ 944
4003_1150	DAC Interval Trigger n Control register (PDB1_DACINTC0)	32	R/W	0000_0000h	40.4.15/ 944
4003_1154	DAC Interval n register (PDB1_DACINT0)	32	R/W	0000_0000h	40.4.16/ 945
4003_1190	Pulse-Out n Enable register (PDB1_POEN)	32	R/W	0000_0000h	40.4.17/ 946
4003_1194	Pulse-Out n Delay register (PDB1_PO0DLY)	32	R/W	0000_0000h	40.4.18/ 946
4003_3000	Status and Control register (PDB2_SC)	32	R/W	0000_0000h	40.4.1/933
4003_3004	Modulus register (PDB2_MOD)	32	R/W	0000_FFFFh	40.4.2/936
4003_3008	Counter register (PDB2_CNT)	32	R	0000_0000h	40.4.3/936
4003_300C	Interrupt Delay register (PDB2_IDLY)	32	R/W	0000_FFFFh	40.4.4/937
4003_3010	Channel n Control register 1 (PDB2_CH0C1)	32	R/W	0000_0000h	40.4.5/937
4003_3014	Channel n Status register (PDB2_CH0S)	32	R/W	0000_0000h	40.4.6/938
4003_3018	Channel n Delay 0 register (PDB2_CH0DLY0)	32	R/W	0000_0000h	40.4.7/939
4003_301C	Channel n Delay 1 register (PDB2_CH0DLY1)	32	R/W	0000_0000h	40.4.8/940
4003_3020	Channel n Delay 2 register (PDB2_CH0DLY2)	32	R/W	0000_0000h	40.4.9/940
4003_3024	Channel n Delay 3 register (PDB2_CH0DLY3)	32	R/W	0000_0000h	40.4.10/ 941
4003_3028	Channel n Delay 4 register (PDB2_CH0DLY4)	32	R/W	0000_0000h	40.4.11/ 942

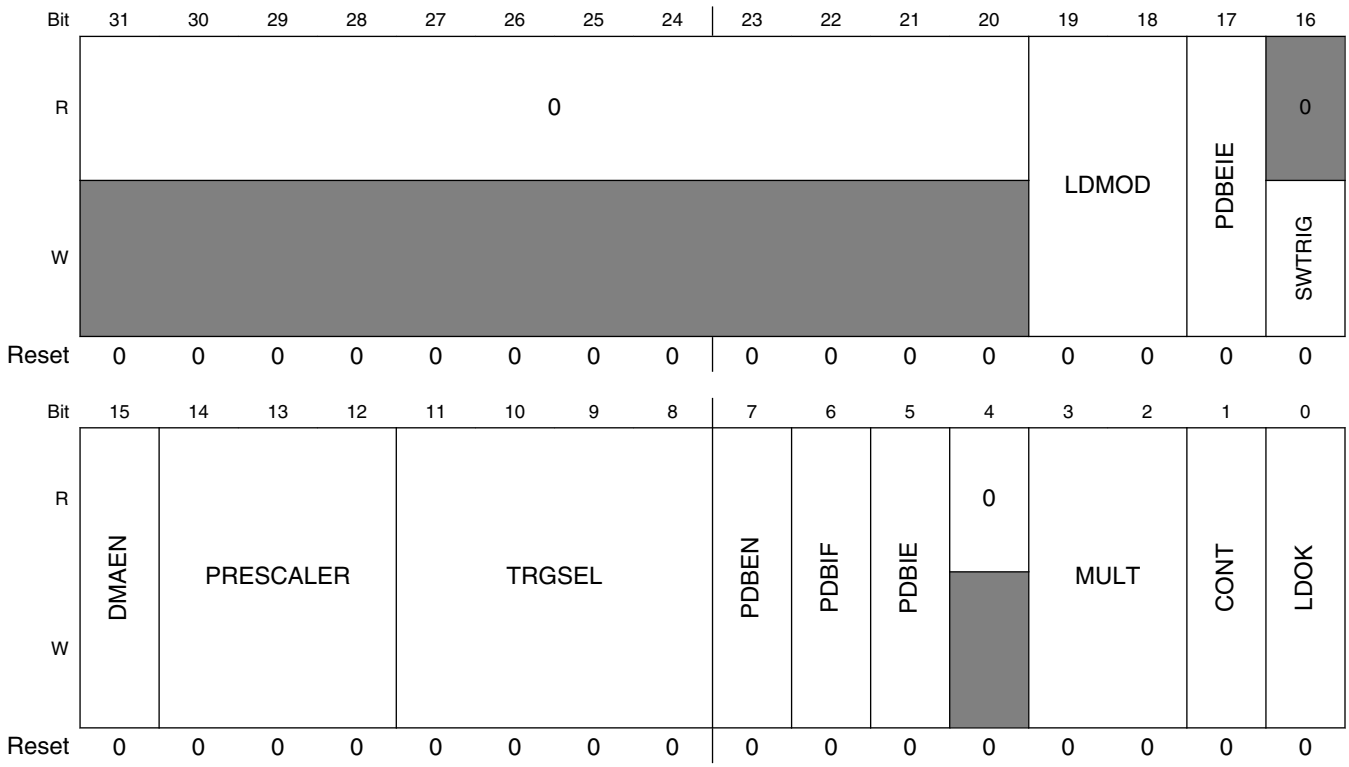
Table continues on the next page...

PDB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_302C	Channel n Delay 5 register (PDB2_CH0DLY5)	32	R/W	0000_0000h	40.4.12/ 942
4003_3030	Channel n Delay 6 register (PDB2_CH0DLY6)	32	R/W	0000_0000h	40.4.13/ 943
4003_3034	Channel n Delay 7 register (PDB2_CH0DLY7)	32	R/W	0000_0000h	40.4.14/ 944
4003_3150	DAC Interval Trigger n Control register (PDB2_DACINTC0)	32	R/W	0000_0000h	40.4.15/ 944
4003_3154	DAC Interval n register (PDB2_DACINT0)	32	R/W	0000_0000h	40.4.16/ 945
4003_3190	Pulse-Out n Enable register (PDB2_POEN)	32	R/W	0000_0000h	40.4.17/ 946
4003_3194	Pulse-Out n Delay register (PDB2_PO0DLY)	32	R/W	0000_0000h	40.4.18/ 946
4003_6000	Status and Control register (PDB0_SC)	32	R/W	0000_0000h	40.4.1/933
4003_6004	Modulus register (PDB0_MOD)	32	R/W	0000_FFFFh	40.4.2/936
4003_6008	Counter register (PDB0_CNT)	32	R	0000_0000h	40.4.3/936
4003_600C	Interrupt Delay register (PDB0_IDLY)	32	R/W	0000_FFFFh	40.4.4/937
4003_6010	Channel n Control register 1 (PDB0_CH0C1)	32	R/W	0000_0000h	40.4.5/937
4003_6014	Channel n Status register (PDB0_CH0S)	32	R/W	0000_0000h	40.4.6/938
4003_6018	Channel n Delay 0 register (PDB0_CH0DLY0)	32	R/W	0000_0000h	40.4.7/939
4003_601C	Channel n Delay 1 register (PDB0_CH0DLY1)	32	R/W	0000_0000h	40.4.8/940
4003_6020	Channel n Delay 2 register (PDB0_CH0DLY2)	32	R/W	0000_0000h	40.4.9/940
4003_6024	Channel n Delay 3 register (PDB0_CH0DLY3)	32	R/W	0000_0000h	40.4.10/ 941
4003_6028	Channel n Delay 4 register (PDB0_CH0DLY4)	32	R/W	0000_0000h	40.4.11/ 942
4003_602C	Channel n Delay 5 register (PDB0_CH0DLY5)	32	R/W	0000_0000h	40.4.12/ 942
4003_6030	Channel n Delay 6 register (PDB0_CH0DLY6)	32	R/W	0000_0000h	40.4.13/ 943
4003_6034	Channel n Delay 7 register (PDB0_CH0DLY7)	32	R/W	0000_0000h	40.4.14/ 944
4003_6150	DAC Interval Trigger n Control register (PDB0_DACINTC0)	32	R/W	0000_0000h	40.4.15/ 944
4003_6154	DAC Interval n register (PDB0_DACINT0)	32	R/W	0000_0000h	40.4.16/ 945
4003_6190	Pulse-Out n Enable register (PDB0_POEN)	32	R/W	0000_0000h	40.4.17/ 946
4003_6194	Pulse-Out n Delay register (PDB0_PO0DLY)	32	R/W	0000_0000h	40.4.18/ 946

40.4.1 Status and Control register (PDB*x*_SC)

Address: Base address + 0h offset



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 (CNT) = MOD + 1 CNT delay elapsed, 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 (CNT) = MOD + 1 CNT delay elapsed, 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 PDBEIE is set, any of the PDB channel sequence error flags generates a PDB sequence error interrupt.

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	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 SWTRIG resets and restarts the counter. Writing 0 to SWTRIG has no effect. Reading SWTRIG yields 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 Counting uses the peripheral clock divided by the product of a factor (selected by MULT field) and an integer factor (set by PRESCALAR field), or in other words, (peripheral clock)/(MULT x PRESCALAR). 000 Counting uses the peripheral clock divided by MULT (the multiplication factor). 001 Counting uses the peripheral clock divided by 2 x MULT (the multiplication factor). 010 Counting uses the peripheral clock divided by 4 x MULT (the multiplication factor). 011 Counting uses the peripheral clock divided by 8 x MULT (the multiplication factor). 100 Counting uses the peripheral clock divided by 16 x MULT (the multiplication factor). 101 Counting uses the peripheral clock divided by 32 x MULT (the multiplication factor). 110 Counting uses the peripheral clock divided by 64 x MULT (the multiplication factor). 111 Counting uses the peripheral clock divided by 128 x MULT (the multiplication factor).
11–8 TRGSEL	Trigger Input Source Select 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

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	0 PDB disabled. Counter is off. 1 PDB enabled.
6 PDBIF	PDB Interrupt Flag PDBIF is set when the counter value is equal to the IDLY register + 1. • Write zero to clear PDBIF. • Writing 1 to PDBIF has no effect.
5 PDBIE	PDB Interrupt Enable Enables the PDB interrupt. When PDBIE 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
0 LDOK	Load OK Writing 1 to LDOK bit updates the MOD, IDLY, CHnDLYm, DACINTx, and POyDLY registers with the values previously written to their internal buffers (and stored there). The new values of MOD, IDLY, CHnDLYm, DACINTx, and POyDLY registers will take effect according to the setting of the LDMOD field (Load Mode Select). Before 1 is written to the LDOK field, the values in the internal buffers of these registers are not effective, and new values cannot be written to the internal buffers until the existing values in the internal buffers are loaded into their corresponding registers. • LDOK can be written only when PDBEN is set, or LDOK can be written at the same time when PDBEN is written to 1. • LDOK is automatically cleared when the values in the internal buffers are loaded into the registers or when PDBEN bit (PDB Enable) is cleared. • Writing 0 to LDOK has no effect.

40.4.2 Modulus register (PDBx_MOD)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	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.

40.4.3 Counter register (PDBx_CNT)

NOTE

Writing to this read-only register will generate a transfer error (and possibly a hard fault).

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	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.

Table continues on the next page...

PDBx_CNT field descriptions (continued)

Field	Description
CNT	PDB Counter Contains the current value of the counter.

40.4.4 Interrupt Delay register (PDBx_IDLY)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + Ch offset

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

PDBx_IDLY field descriptions

Field	Description
31–16 Reserved	This 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.

40.4.5 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: Base address + 10h 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								BB								TOS								EN							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDBx_CHnC1 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 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 SWTRIG 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.

40.4.6 Channel n Status register (PDBx_CHnS)

Address: Base address + 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 (PDB_CNT) matches the value CHnDLY*m* + 1. Write 0 to clear CF.

Table continues on the next page...

PDBx_CHnS field descriptions (continued)

Field	Description
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 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.

40.4.7 Channel n Delay 0 register (PDBx_CHnDLY0)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 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.

40.4.8 Channel n Delay 1 register (PDB*x*_CH*n*DLY1)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 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

PDB*x*_CH*n*DLY1 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.

40.4.9 Channel n Delay 2 register (PDB*x*_CH*n*DLY2)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 20h 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_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.

40.4.10 Channel n Delay 3 register (PDBx_CHnDLY3)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 24h 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_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.

40.4.11 Channel n Delay 4 register (PDBx_CHnDLY4)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 28h 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_CHnDLY4 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.

40.4.12 Channel n Delay 5 register (PDBx_CHnDLY5)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 2Ch 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_CHnDLY5 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.

40.4.13 Channel n Delay 6 register (PDBx_CHnDLY6)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 30h 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_CHnDLY6 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.

40.4.14 Channel n Delay 7 register (PDBx_CHnDLY7)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 34h 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_CHnDLY7 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.

40.4.15 DAC Interval Trigger n Control register (PDBx_DACINTCn)

Address: Base address + 150h offset + (8d × 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															
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														EXT	TOE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDBx_DACINTCn field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 EXT	DAC External Trigger Input Enable

Table continues on the next page...

PDBx_DACINTC*n* field descriptions (continued)

Field	Description
	This bit enables the external trigger for DAC interval counter. 0 DAC external trigger input disabled. DAC interval counter is reset and started counting when a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 DAC external trigger input enabled. DAC interval counter is bypassed and DAC external trigger input triggers the DAC interval trigger.
0 TOE	DAC Interval Trigger Enable Enables the DAC interval trigger. 0 DAC interval trigger disabled. 1 DAC interval trigger enabled.

40.4.16 DAC Interval n register (PDBx_DACINT*n*)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 154h offset + (8d × 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																INT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDBx_DACINT*n* field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
INT	DAC Interval These bits specify the interval value for DAC interval trigger. DAC interval trigger triggers DAC[1:0] update when the DAC interval counter is equal to the DACINT. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

40.4.17 Pulse-Out n Enable register (PDB*x*_POEN)

Address: Base address + 190h offset

Bit	31	30	29	28	27	26	25	24	23	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 8 bits are implemented in this MCU. 0 PDB Pulse-Out disabled 1 PDB Pulse-Out enabled

40.4.18 Pulse-Out n Delay register (PDB*x*_PO*n*DLY)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 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 W	DLY1																DLY2															
Reset	0	0	0	0	0	0	0	0	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.

PDBx_PO*n*DLY field descriptions (continued)

Field	Description

40.5 Functional description

40.5.1 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 the Software Trigger bit (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 channel delay registers (CH*n*DLY*m*), and the pre-triggers can be enabled or disabled using the PDB Channel Pre-Trigger Enables (CH*n*C1[EN[*m*]]).

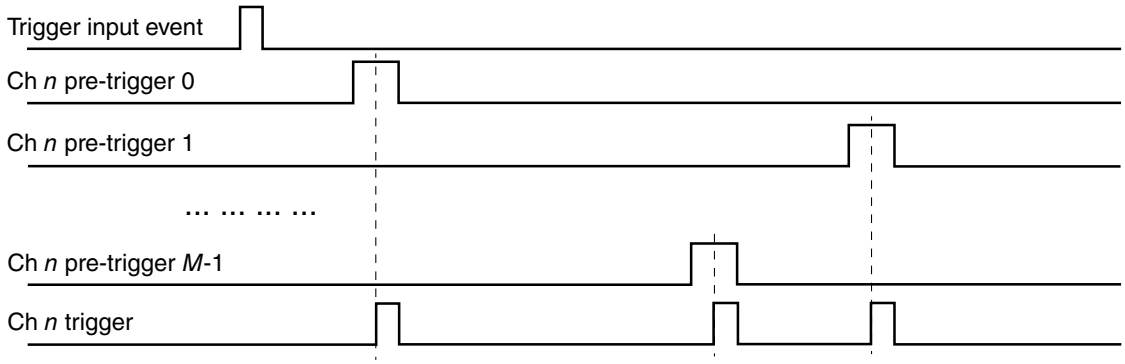


Figure 40-6. Pre-trigger and trigger outputs

The delay in the channel delay register (CH*n*DLY*m*) can be optionally bypassed, if the PDB Channel Pre-Trigger Output Select (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 the PDB Channel Pre-Trigger Back-to-Back Operation Enable (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 PDB Channel Sequence Error Flag (CH*n*S[ERR[*m*]], associated with the pre-trigger *m*) is set. If the PDB Sequence Error Interrupt Enable (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.
- If the pre-trigger delay is 0 cycles, then both channels will flag a PDB channel sequence error and ADC will not perform a conversion.

The PDB reports PDB channel sequence errors only for pre-triggers in same PDB channel. For situations when PDB triggering is done through different PDB channels, software must ensure sufficient delays in between the pre-triggers.

When the PDB counter reaches the value (CNT + 1), the PDB Interrupt Flag (SC[PDBIF]) is set. A PDB interrupt can be generated if the PDB Sequence Error Interrupt Enable (SC[PDBEIE]) is set and the DMA Enable (SC[DMAEN]) is cleared. If the DMA Enable (SC[DMAEN]) is set, then the PDB requests a DMA transfer when the PDB Interrupt Flag (SC[PDBIF]) is set.

The modulus value in the Modulus register (MOD) is used to reset the counter back to zero at the end of the count. If the Continuous Mode Enable (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.

40.5.2 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].

For the trigger input sources implemented in this MCU, see chip configuration information.

40.5.3 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.

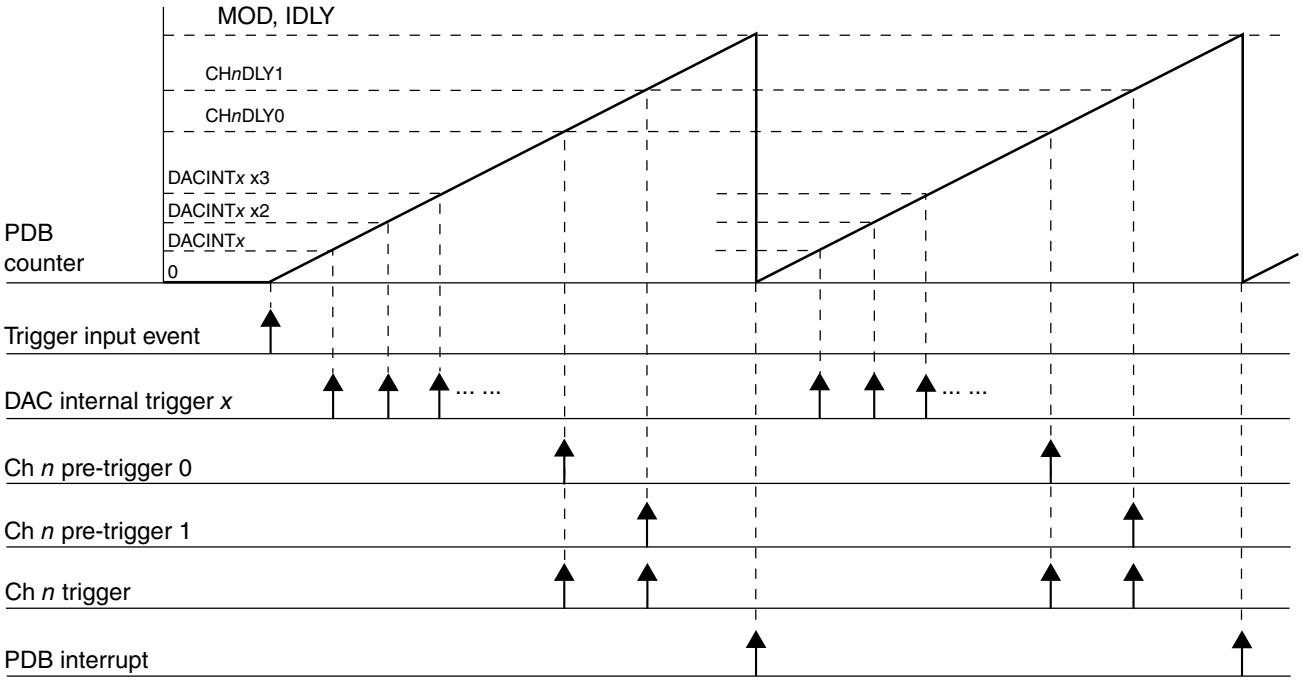


Figure 40-7. PDB ADC triggers and DAC interval triggers use case

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.

40.5.4 Pulse-Out's

PDB can generate pulse outputs of configurable width.

- When the PDB counter reaches the value set in PO*y*DLY[DLY1], then the Pulse-Out goes high.
- When the PDB counter reaches PO*y*DLY[DLY2], then 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.

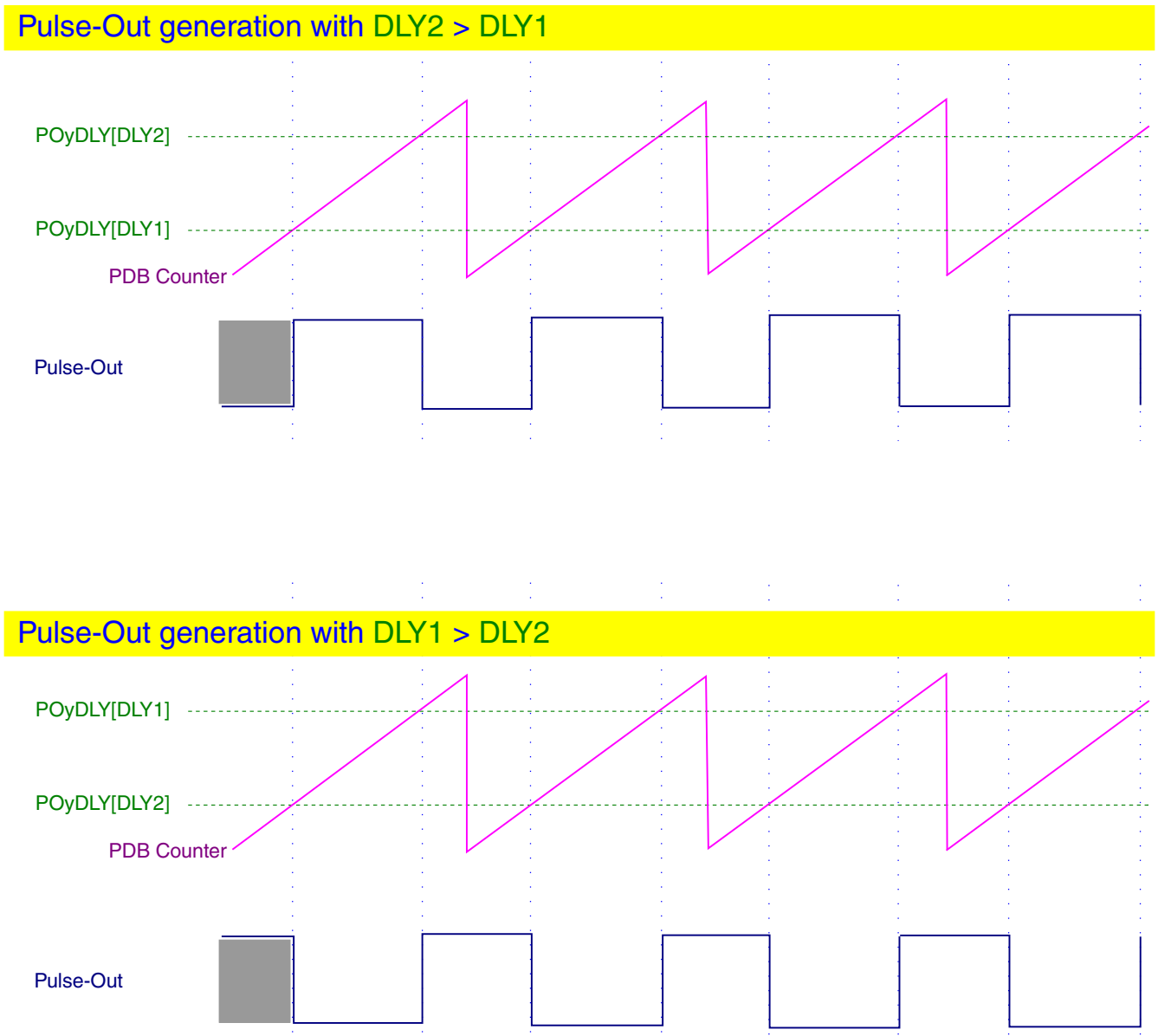


Figure 40-8. How Pulse Out is generated

40.5.5 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 in the next table.

Table 40-3. When delay registers are updated

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 PDB_MOD[MOD] + 1 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 PDB_MOD[MOD] + 1 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.

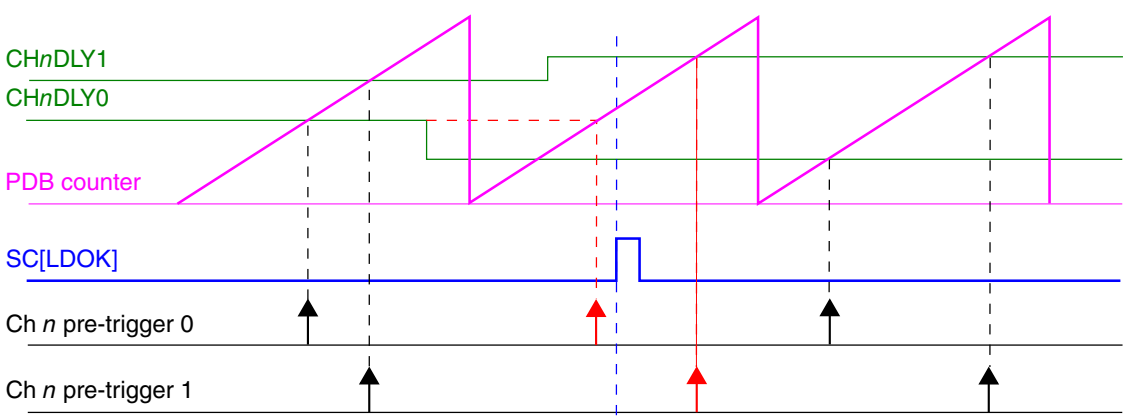


Figure 40-9. Registers update with SC[LDMOD] = 00

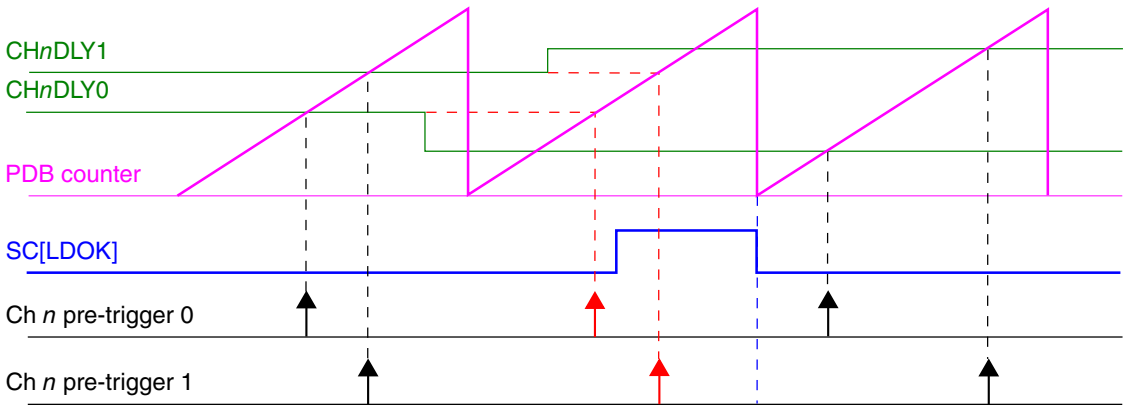


Figure 40-10. Registers update with SC[LDMOD] = x1

40.5.6 Interrupts

PDB can generate two interrupts: PDB interrupt and PDB sequence error interrupt. The following table summarizes the interrupts.

Table 40-4. 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

40.5.7 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.

40.6 Application information

40.6.1 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.

40.7 Usage Guide

40.7.1 Using PDB to precisely control ADC conversion

For detailed information, see the ADC trigger sections in the ADC chapter.

Chapter 41
FlexTimer Module (FTM)

41.1 Chip-specific information for this module

41.1.1 Instantiation Information

This device contains four FlexTimer modules.

The following table shows how these modules are configured.

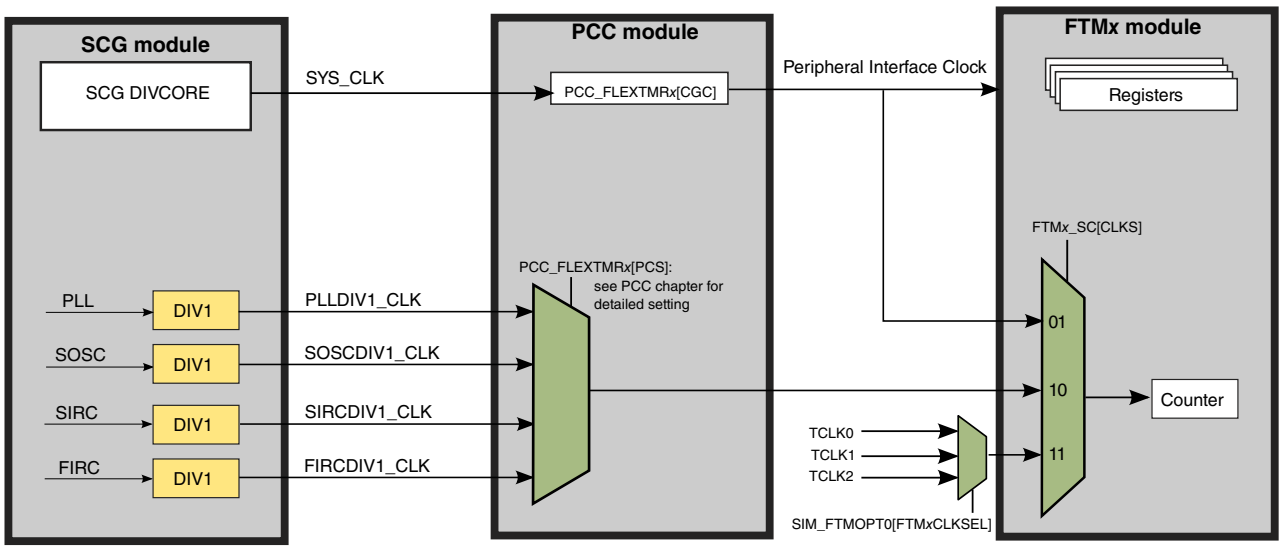
Table 41-1. FTM Instantiations

FTM instance	Number of channels	Features/usage
FTM0	8	• FTM enhanced features • GTB_EN
FTM1	8	• FTM enhanced features • GTB_EN • Quadrature Decoder
FTM2	8	• FTM enhanced features • GTB_EN • Quadrature Decoder
FTM3	8	• FTM enhanced features • GTB_EN

41.1.2 FTM Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - FTM



NOTE

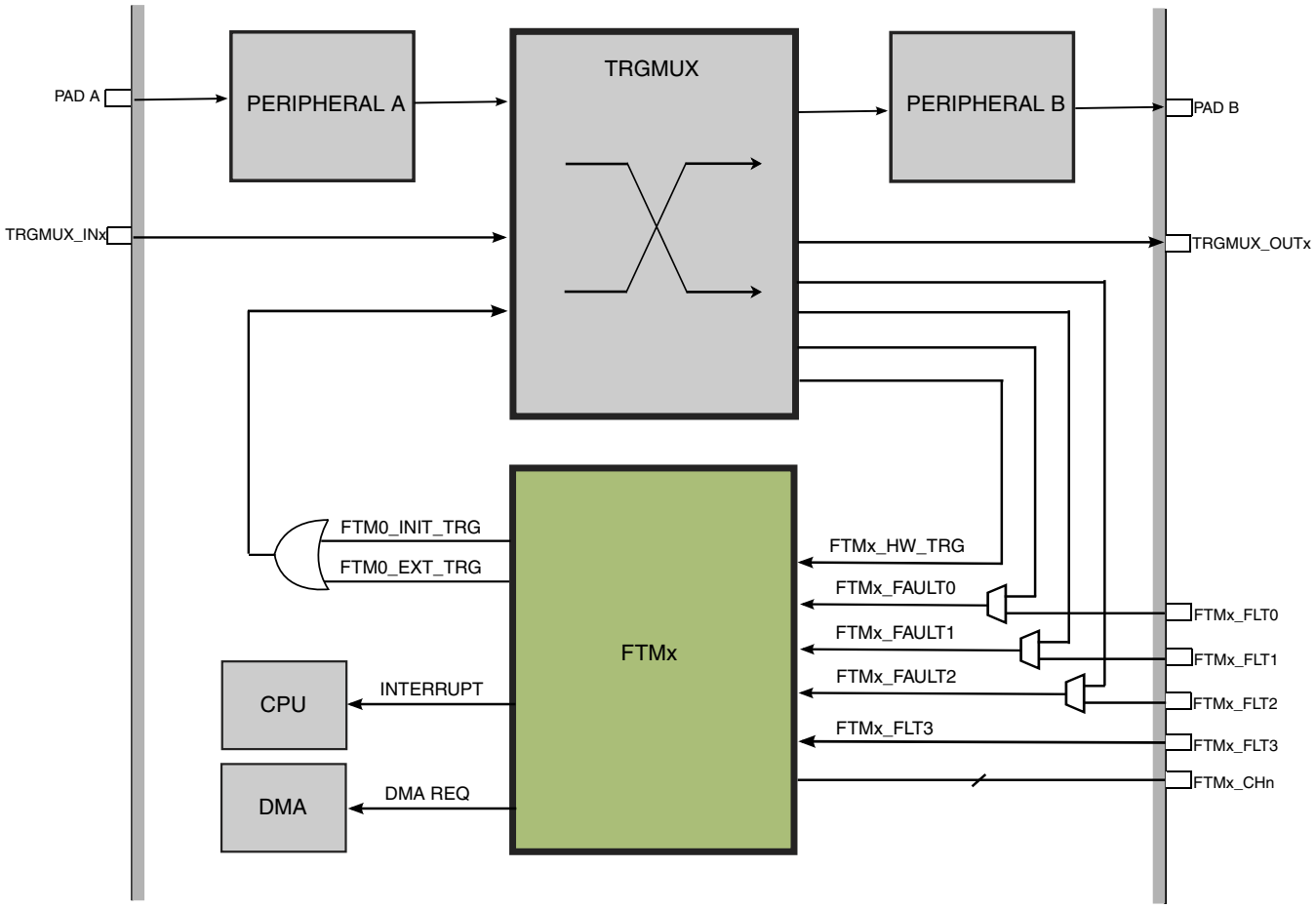
Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM system clock frequency (SYS_CLK).

NOTE

The external clock are synchronized by FTM system clock (SYS_CLK). Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

41.1.3 Inter-connectivity Information

The FTM inter-connectivity is shown in the following diagram.



NOTE

The diagram only shows some possible fault input sources. For the actual connections of each FTM, see FTM Fault Detection Inputs for details.

41.1.3.1 FTM Fault Detection Inputs

The following fault detection input options for the FTM modules are selected via the SIM_FTMOPT0 register. The external pin option is selected by default.

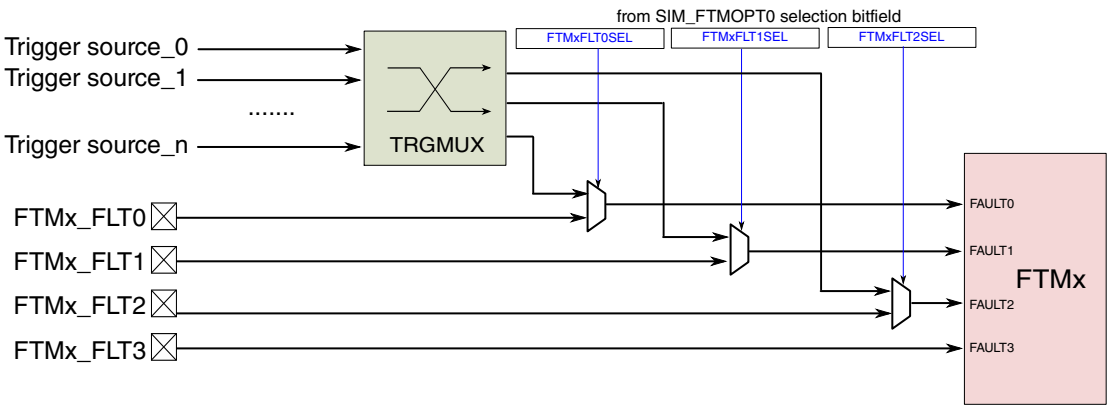
- FTM0 FAULT0 = FTM0_FLT0 pin or TRGMUX output
- FTM0 FAULT1 = FTM0_FLT1 pin or TRGMUX output
- FTM0 FAULT2 = FTM0_FLT2 pin or TRGMUX output
- FTM0 FAULT3 = FTM0_FLT3 pin

- FTM1 FAULT0 = FTM1_FLT0 pin or TRGMUX output
- FTM1 FAULT1 = FTM1_FLT1 pin or TRGMUX output

- FTM1 FAULT2 = FTM1_FLT2 pin or TRGMUX output
- FTM1 FAULT3 = FTM1_FLT3 pin

- FTM2 FAULT0 = FTM2_FLT0 pin or TRGMUX output
- FTM2 FAULT1 = FTM2_FLT1 pin or TRGMUX output
- FTM2 FAULT2 = FTM2_FLT2 pin or TRGMUX output
- FTM2 FAULT3 = FTM2_FLT3 pin

- FTM3 FAULT0 = FTM3_FLT0 pin or TRGMUX output
- FTM3 FAULT1 = FTM3_FLT1 pin or TRGMUX output
- FTM3 FAULT2 = FTM3_FLT2 pin or TRGMUX output
- FTM3 FAULT3 = FTM3_FLT3 pin



41.1.3.2 FTM Hardware Triggers and Synchronization

The FlexTimer support external hardware trigger input which can be used for timer dynamic synchronization between multiple FlexTimers or counter reset. The FlexTimer hardware trigger are implemented as following.

FTM0:

- FTM0 hardware trigger 0 = TRGMUX trigger output
- FTM0 hardware trigger 1 = SIM_FTMOPT1[FTM0SYNCBIT]
- FTM0 hardware trigger 2 = FTM0_FLT0 pin

FTM1:

- FTM1 hardware trigger 0 = TRGMUX trigger output
- FTM1 hardware trigger 1 = SIM_FTMOPT1[FTM1SYNCBIT]
- FTM1 hardware trigger 2 = FTM1_FLT0 pin

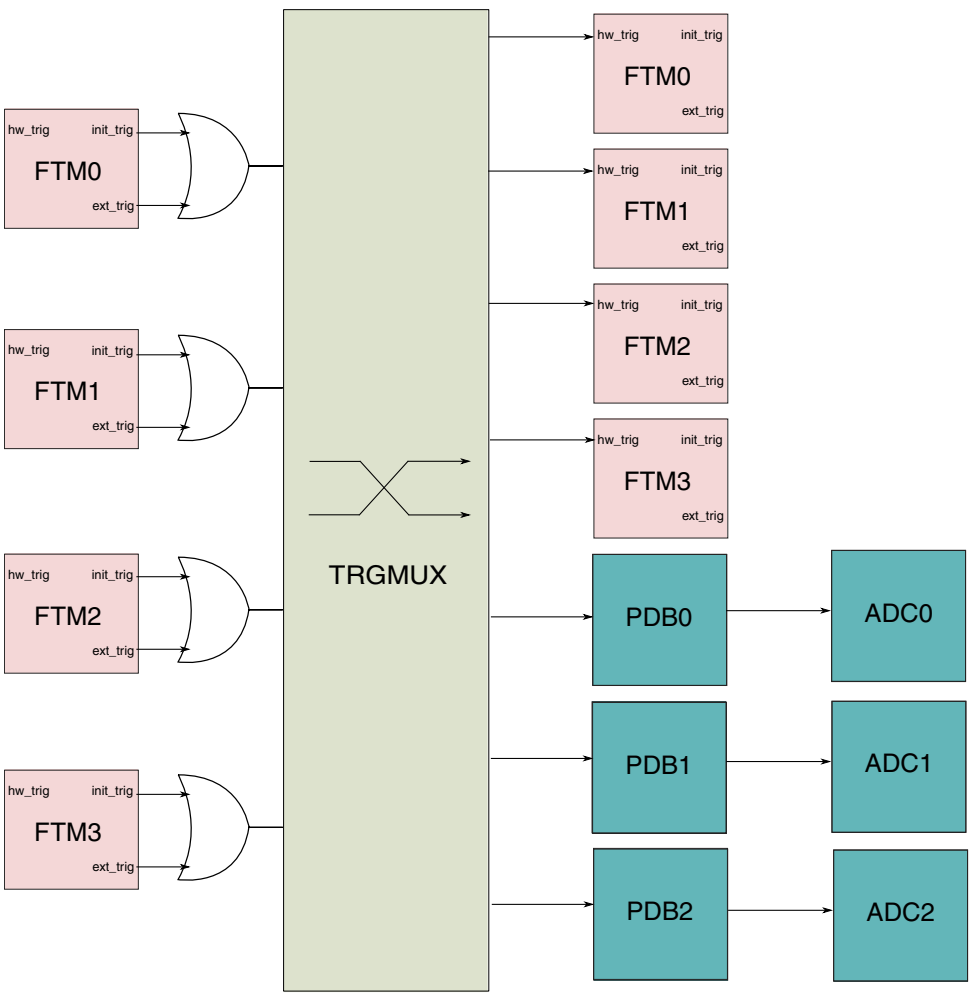
FTM2:

- FTM2 hardware trigger 0 = TRGMUX trigger output
- FTM2 hardware trigger 1 = SIM_FTMOPT1[FTM2SYNCBIT]
- FTM2 hardware trigger 2 = FTM2_FLT0 pin

FTM3:

- FTM3 hardware trigger 0 = TRGMUX trigger output
- FTM3 hardware trigger 1 = SIM_FTMOPT1[FTM3SYNCBIT]
- FTM3 hardware trigger 2 = FTM3_FLT0 pin

The hardware trigger source can be from many other modules via TRGMUX, like LPIT, Low Power Timer, CMP, etc. It also supports FlexTimer's self trigger outputs, ex: counter initlization trigger (init_trig) and channel match trigger (ext_trig), through the flexible TRGMUX module.



The FlexTimer trigger outputs are also usually used as trigger source by other modules, for example, the above diagram shows a case of triggering PDB and ADC. See "Chip-specific Information" in PDB chapter and ADC Trigger Sources in ADC chapter for details.

41.1.3.3 FTM Input Capture Options

The following channel 0 input capture source options are selected via SIM_FTMOPT1. The external pin option is selected by default.

- FTM1 channel 0 input capture = FTM1_CH0 pin or CMP0 output or CMP1 output or CMP2 output
- FTM2 channel 0 input capture = FTM2_CH0 pin or CMP0 output or CMP1 output or CMP2 output
- FTM2 channel 1 input capture = FTM2_CH1 pin or exclusive OR of FTM2_CH0, FTM2_CH1, and FTM1_CH1. See FTM Hall sensor support.

41.2 Introduction

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

The FlexTimer module (FTM) is a two-to-eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

41.2.1 FlexTimer philosophy

The FlexTimer is built upon a simple timer, the Timer PWM Module – TPM, used for many years on our HCS08 family of 8-bit microcontrollers. The FlexTimer extends the functionality to meet the demands of motor control, digital lighting solutions, and power conversion, while providing low cost and backwards compatibility with the TPM module.

Several key enhancements are made:

- Signed up counter
- Deadtime insertion hardware
- Fault control inputs

- Enhanced triggering functionality
- Initialization and polarity control

All of the features common with the TPM have fully backwards compatible register assignments. The FlexTimer can also use code on the same core platform without change to perform the same functions.

Motor control and power conversion features have been added through a dedicated set of registers and defaults turn off all new features. The new features, such as hardware deadtime insertion, polarity, fault control, and output forcing and masking, greatly reduce loading on the execution software and are usually each controlled by a group of registers.

FlexTimer input triggers can be from comparators, ADC, or other submodules to initiate timer functions automatically. These triggers can be linked in a variety of ways during integration of the sub modules so please note the options available for used FlexTimer configuration.

More than one FlexTimers may be synchronized to provide a larger timer with their counters incrementing in unison, assuming the initialization, the input clocks, the initial and final counting values are the same in each FlexTimer.

All main user access registers are buffered to ease the load on the executing software. A number of trigger options exist to determine which registers are updated with this user defined data.

41.2.2 Features

The FTM features include:

- FTM source clock is selectable
  - Source clock can be the FTM input clock, the fixed frequency clock, or an external clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the FTM input clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter

  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In Input Capture mode:
  - The capture can occur on rising edges, falling edges or both edges
  - An input filter can be selected for some channels. One unique prescaler is available for all filters
- In Output Compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode
- Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal
- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs
- The deadtime insertion is available for each complementary pair
- Generation of match triggers
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- The generation of an interrupt when a register reload point occurs
- Synchronized loading of write buffered FTM registers
- Half cycle and Full cycle register reload capacity
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input capture mode

- Direct access to input pin states
- Dual edge capture for pulse and period width measurement
- Quadrature decoder with prescaled input filters, relative position counting, and interrupt on position count or capture of position count on external event
- The FTM channels can be selected to generate a trigger pulse on channel output instead of a PWM
- Dithering capability to simulate fine edge control for both PWM period or PWM duty cycle

41.2.3 Modes of operation

When the chip is in an active Debug mode, the FTM temporarily suspends all counting until the chip returns to normal user operating mode. During Stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During Wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the chip from Wait mode, the power can then be saved by disabling FTM functions before entering Wait mode.

41.2.4 Block Diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

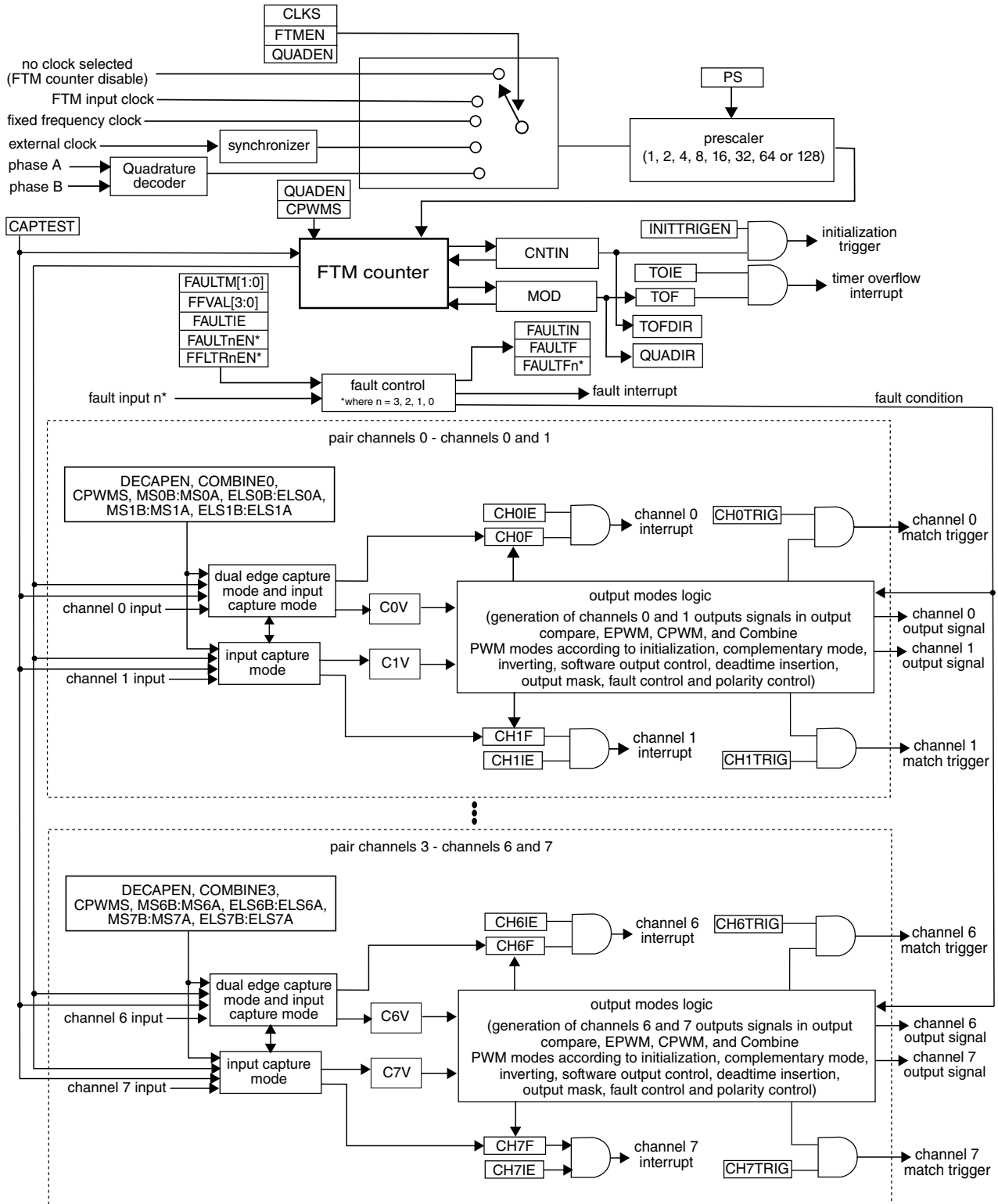


Figure 41-1. FTM Block Diagram

41.3 FTM signal descriptions

Table 41-2 shows the user-accessible signals for the FTM.

Table 41-2. FTM signal descriptions

Signal	Description	I/O	Function
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I	The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of FTM input clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.
CHn	FTM channel (n), where n can be 7-0	I/O	Each FTM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.
FAULTj	Fault input (j), where j can be 3-0	I	The fault input signals are used to control the CHn channel output state. If a fault is detected, the FAULTj signal is asserted and the channel output is put in a safe state. The behavior of the fault logic is defined by the FAULTM[1:0] control bits in the MODE register and FAULTEN bit in the COMBINE register. Note that each FAULTj input may affect all channels selectively since FAULTM[1:0] and FAULTEN control bits are defined for each pair of channels. Because there are several FAULTj inputs, maximum of 4 for the FTM module, each one of these inputs is activated by the FAULTjEN bit in the FLTCTRL register.
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I	The quadrature decoder phase A input is used as the Quadrature Decoder mode is selected. The phase A input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder mode.
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I	The quadrature decoder phase B input is used as the Quadrature Decoder mode is selected. The phase B input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder mode.

41.4 Memory map and register definition

41.4.1 Memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

The registers and bits of an unavailable function in the FTM remain in the memory map and in the reset value, but they have no active function.

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

Note

Do not write in the region from the CNTIN register through the PWMLOAD register when FTMEN = 0.

41.4.2 Register descriptions

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved. Double buffered register writes must be done using 32-bit operations.

FTM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_6000	Status And Control (FTM3_SC)	32	R/W	0000_0000h	41.4.3/973
4002_6004	Counter (FTM3_CNT)	32	R/W	0000_0000h	41.4.4/977
4002_6008	Modulo (FTM3_MOD)	32	R/W	0000_0000h	41.4.5/978
4002_600C	Channel (n) Status And Control (FTM3_C0SC)	32	R/W	0000_0000h	41.4.6/979
4002_6010	Channel (n) Value (FTM3_C0V)	32	R/W	0000_0000h	41.4.7/981
4002_6014	Channel (n) Status And Control (FTM3_C1SC)	32	R/W	0000_0000h	41.4.6/979
4002_6018	Channel (n) Value (FTM3_C1V)	32	R/W	0000_0000h	41.4.7/981
4002_601C	Channel (n) Status And Control (FTM3_C2SC)	32	R/W	0000_0000h	41.4.6/979
4002_6020	Channel (n) Value (FTM3_C2V)	32	R/W	0000_0000h	41.4.7/981
4002_6024	Channel (n) Status And Control (FTM3_C3SC)	32	R/W	0000_0000h	41.4.6/979
4002_6028	Channel (n) Value (FTM3_C3V)	32	R/W	0000_0000h	41.4.7/981
4002_602C	Channel (n) Status And Control (FTM3_C4SC)	32	R/W	0000_0000h	41.4.6/979
4002_6030	Channel (n) Value (FTM3_C4V)	32	R/W	0000_0000h	41.4.7/981
4002_6034	Channel (n) Status And Control (FTM3_C5SC)	32	R/W	0000_0000h	41.4.6/979
4002_6038	Channel (n) Value (FTM3_C5V)	32	R/W	0000_0000h	41.4.7/981
4002_603C	Channel (n) Status And Control (FTM3_C6SC)	32	R/W	0000_0000h	41.4.6/979
4002_6040	Channel (n) Value (FTM3_C6V)	32	R/W	0000_0000h	41.4.7/981
4002_6044	Channel (n) Status And Control (FTM3_C7SC)	32	R/W	0000_0000h	41.4.6/979
4002_6048	Channel (n) Value (FTM3_C7V)	32	R/W	0000_0000h	41.4.7/981

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_604C	Counter Initial Value (FTM3_CNTIN)	32	R/W	0000_0000h	41.4.8/981
4002_6050	Capture And Compare Status (FTM3_STATUS)	32	R/W	0000_0000h	41.4.9/982
4002_6054	Features Mode Selection (FTM3_MODE)	32	R/W	0000_0004h	41.4.10/ 984
4002_6058	Synchronization (FTM3_SYNC)	32	R/W	0000_0000h	41.4.11/ 986
4002_605C	Initial State For Channels Output (FTM3_OUTINIT)	32	R/W	0000_0000h	41.4.12/ 988
4002_6060	Output Mask (FTM3_OUTMASK)	32	R/W	0000_0000h	41.4.13/ 990
4002_6064	Function For Linked Channels (FTM3_COMBINE)	32	R/W	0000_0000h	41.4.14/ 992
4002_6068	Deadtime Configuration (FTM3_DEADTIME)	32	R/W	0000_0000h	41.4.15/ 996
4002_606C	FTM External Trigger (FTM3_EXTTRIG)	32	R/W	0000_0000h	41.4.16/ 997
4002_6070	Channels Polarity (FTM3_POL)	32	R/W	0000_0000h	41.4.17/ 999
4002_6074	Fault Mode Status (FTM3_FMS)	32	R/W	0000_0000h	41.4.18/ 1002
4002_6078	Input Capture Filter Control (FTM3_FILTER)	32	R/W	0000_0000h	41.4.19/ 1004
4002_607C	Fault Control (FTM3_FLTCTRL)	32	R/W	0000_0000h	41.4.20/ 1005
4002_6080	Quadrature Decoder Control And Status (FTM3_QDCTRL)	32	R/W	0000_0000h	41.4.21/ 1008
4002_6084	Configuration (FTM3_CONF)	32	R/W	0000_0000h	41.4.22/ 1010
4002_6088	FTM Fault Input Polarity (FTM3_FLTPOL)	32	R/W	0000_0000h	41.4.23/ 1011
4002_608C	Synchronization Configuration (FTM3_SYNCONF)	32	R/W	0000_0000h	41.4.24/ 1012
4002_6090	FTM Inverting Control (FTM3_INVCTRL)	32	R/W	0000_0000h	41.4.25/ 1014
4002_6094	FTM Software Output Control (FTM3_SWOCTRL)	32	R/W	0000_0000h	41.4.26/ 1015
4002_6098	FTM PWM Load (FTM3_PWMLOAD)	32	R/W	0000_0000h	41.4.27/ 1018
4002_609C	Half Cycle Register (FTM3_HCR)	32	R/W	0000_0000h	41.4.28/ 1020
4002_6200	Mirror of Modulo Value (FTM3_MOD_MIRROR)	32	R/W	0000_0000h	41.4.29/ 1020
4002_6204	Mirror of Channel (n) Match Value (FTM3_C0V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_6208	Mirror of Channel (n) Match Value (FTM3_C1V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_620C	Mirror of Channel (n) Match Value (FTM3_C2V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_6210	Mirror of Channel (n) Match Value (FTM3_C3V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_6214	Mirror of Channel (n) Match Value (FTM3_C4V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_6218	Mirror of Channel (n) Match Value (FTM3_C5V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_621C	Mirror of Channel (n) Match Value (FTM3_C6V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4002_6220	Mirror of Channel (n) Match Value (FTM3_C7V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_8000	Status And Control (FTM0_SC)	32	R/W	0000_0000h	41.4.3/973
4003_8004	Counter (FTM0_CNT)	32	R/W	0000_0000h	41.4.4/977
4003_8008	Modulo (FTM0_MOD)	32	R/W	0000_0000h	41.4.5/978
4003_800C	Channel (n) Status And Control (FTM0_C0SC)	32	R/W	0000_0000h	41.4.6/979
4003_8010	Channel (n) Value (FTM0_C0V)	32	R/W	0000_0000h	41.4.7/981
4003_8014	Channel (n) Status And Control (FTM0_C1SC)	32	R/W	0000_0000h	41.4.6/979
4003_8018	Channel (n) Value (FTM0_C1V)	32	R/W	0000_0000h	41.4.7/981
4003_801C	Channel (n) Status And Control (FTM0_C2SC)	32	R/W	0000_0000h	41.4.6/979
4003_8020	Channel (n) Value (FTM0_C2V)	32	R/W	0000_0000h	41.4.7/981
4003_8024	Channel (n) Status And Control (FTM0_C3SC)	32	R/W	0000_0000h	41.4.6/979
4003_8028	Channel (n) Value (FTM0_C3V)	32	R/W	0000_0000h	41.4.7/981
4003_802C	Channel (n) Status And Control (FTM0_C4SC)	32	R/W	0000_0000h	41.4.6/979
4003_8030	Channel (n) Value (FTM0_C4V)	32	R/W	0000_0000h	41.4.7/981
4003_8034	Channel (n) Status And Control (FTM0_C5SC)	32	R/W	0000_0000h	41.4.6/979
4003_8038	Channel (n) Value (FTM0_C5V)	32	R/W	0000_0000h	41.4.7/981
4003_803C	Channel (n) Status And Control (FTM0_C6SC)	32	R/W	0000_0000h	41.4.6/979
4003_8040	Channel (n) Value (FTM0_C6V)	32	R/W	0000_0000h	41.4.7/981
4003_8044	Channel (n) Status And Control (FTM0_C7SC)	32	R/W	0000_0000h	41.4.6/979
4003_8048	Channel (n) Value (FTM0_C7V)	32	R/W	0000_0000h	41.4.7/981
4003_804C	Counter Initial Value (FTM0_CNTIN)	32	R/W	0000_0000h	41.4.8/981
4003_8050	Capture And Compare Status (FTM0_STATUS)	32	R/W	0000_0000h	41.4.9/982
4003_8054	Features Mode Selection (FTM0_MODE)	32	R/W	0000_0004h	41.4.10/ 984
4003_8058	Synchronization (FTM0_SYNC)	32	R/W	0000_0000h	41.4.11/ 986
4003_805C	Initial State For Channels Output (FTM0_OUTINIT)	32	R/W	0000_0000h	41.4.12/ 988

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8060	Output Mask (FTM0_OUTMASK)	32	R/W	0000_0000h	41.4.13/ 990
4003_8064	Function For Linked Channels (FTM0_COMBINE)	32	R/W	0000_0000h	41.4.14/ 992
4003_8068	Deadtime Configuration (FTM0_DEADTIME)	32	R/W	0000_0000h	41.4.15/ 996
4003_806C	FTM External Trigger (FTM0_EXTTRIG)	32	R/W	0000_0000h	41.4.16/ 997
4003_8070	Channels Polarity (FTM0_POL)	32	R/W	0000_0000h	41.4.17/ 999
4003_8074	Fault Mode Status (FTM0_FMS)	32	R/W	0000_0000h	41.4.18/ 1002
4003_8078	Input Capture Filter Control (FTM0_FILTER)	32	R/W	0000_0000h	41.4.19/ 1004
4003_807C	Fault Control (FTM0_FLTCTRL)	32	R/W	0000_0000h	41.4.20/ 1005
4003_8080	Quadrature Decoder Control And Status (FTM0_QDCTRL)	32	R/W	0000_0000h	41.4.21/ 1008
4003_8084	Configuration (FTM0_CONF)	32	R/W	0000_0000h	41.4.22/ 1010
4003_8088	FTM Fault Input Polarity (FTM0_FLTPOL)	32	R/W	0000_0000h	41.4.23/ 1011
4003_808C	Synchronization Configuration (FTM0_SYNCONF)	32	R/W	0000_0000h	41.4.24/ 1012
4003_8090	FTM Inverting Control (FTM0_INVCTRL)	32	R/W	0000_0000h	41.4.25/ 1014
4003_8094	FTM Software Output Control (FTM0_SWOCTRL)	32	R/W	0000_0000h	41.4.26/ 1015
4003_8098	FTM PWM Load (FTM0_PWMLOAD)	32	R/W	0000_0000h	41.4.27/ 1018
4003_809C	Half Cycle Register (FTM0_HCR)	32	R/W	0000_0000h	41.4.28/ 1020
4003_8200	Mirror of Modulo Value (FTM0_MOD_MIRROR)	32	R/W	0000_0000h	41.4.29/ 1020
4003_8204	Mirror of Channel (n) Match Value (FTM0_C0V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_8208	Mirror of Channel (n) Match Value (FTM0_C1V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_820C	Mirror of Channel (n) Match Value (FTM0_C2V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_8210	Mirror of Channel (n) Match Value (FTM0_C3V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_8214	Mirror of Channel (n) Match Value (FTM0_C4V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8218	Mirror of Channel (n) Match Value (FTM0_C5V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_821C	Mirror of Channel (n) Match Value (FTM0_C6V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_8220	Mirror of Channel (n) Match Value (FTM0_C7V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9000	Status And Control (FTM1_SC)	32	R/W	0000_0000h	41.4.3/973
4003_9004	Counter (FTM1_CNT)	32	R/W	0000_0000h	41.4.4/977
4003_9008	Modulo (FTM1_MOD)	32	R/W	0000_0000h	41.4.5/978
4003_900C	Channel (n) Status And Control (FTM1_C0SC)	32	R/W	0000_0000h	41.4.6/979
4003_9010	Channel (n) Value (FTM1_C0V)	32	R/W	0000_0000h	41.4.7/981
4003_9014	Channel (n) Status And Control (FTM1_C1SC)	32	R/W	0000_0000h	41.4.6/979
4003_9018	Channel (n) Value (FTM1_C1V)	32	R/W	0000_0000h	41.4.7/981
4003_901C	Channel (n) Status And Control (FTM1_C2SC)	32	R/W	0000_0000h	41.4.6/979
4003_9020	Channel (n) Value (FTM1_C2V)	32	R/W	0000_0000h	41.4.7/981
4003_9024	Channel (n) Status And Control (FTM1_C3SC)	32	R/W	0000_0000h	41.4.6/979
4003_9028	Channel (n) Value (FTM1_C3V)	32	R/W	0000_0000h	41.4.7/981
4003_902C	Channel (n) Status And Control (FTM1_C4SC)	32	R/W	0000_0000h	41.4.6/979
4003_9030	Channel (n) Value (FTM1_C4V)	32	R/W	0000_0000h	41.4.7/981
4003_9034	Channel (n) Status And Control (FTM1_C5SC)	32	R/W	0000_0000h	41.4.6/979
4003_9038	Channel (n) Value (FTM1_C5V)	32	R/W	0000_0000h	41.4.7/981
4003_903C	Channel (n) Status And Control (FTM1_C6SC)	32	R/W	0000_0000h	41.4.6/979
4003_9040	Channel (n) Value (FTM1_C6V)	32	R/W	0000_0000h	41.4.7/981
4003_9044	Channel (n) Status And Control (FTM1_C7SC)	32	R/W	0000_0000h	41.4.6/979
4003_9048	Channel (n) Value (FTM1_C7V)	32	R/W	0000_0000h	41.4.7/981
4003_904C	Counter Initial Value (FTM1_CNTIN)	32	R/W	0000_0000h	41.4.8/981
4003_9050	Capture And Compare Status (FTM1_STATUS)	32	R/W	0000_0000h	41.4.9/982
4003_9054	Features Mode Selection (FTM1_MODE)	32	R/W	0000_0004h	41.4.10/ 984
4003_9058	Synchronization (FTM1_SYNC)	32	R/W	0000_0000h	41.4.11/ 986
4003_905C	Initial State For Channels Output (FTM1_OUTINIT)	32	R/W	0000_0000h	41.4.12/ 988
4003_9060	Output Mask (FTM1_OUTMASK)	32	R/W	0000_0000h	41.4.13/ 990
4003_9064	Function For Linked Channels (FTM1_COMBINE)	32	R/W	0000_0000h	41.4.14/ 992
4003_9068	Deadtime Configuration (FTM1_DEADTIME)	32	R/W	0000_0000h	41.4.15/ 996
4003_906C	FTM External Trigger (FTM1_EXTTRIG)	32	R/W	0000_0000h	41.4.16/ 997

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_9070	Channels Polarity (FTM1_POL)	32	R/W	0000_0000h	41.4.17/ 999
4003_9074	Fault Mode Status (FTM1_FMS)	32	R/W	0000_0000h	41.4.18/ 1002
4003_9078	Input Capture Filter Control (FTM1_FILTER)	32	R/W	0000_0000h	41.4.19/ 1004
4003_907C	Fault Control (FTM1_FLTCTRL)	32	R/W	0000_0000h	41.4.20/ 1005
4003_9080	Quadrature Decoder Control And Status (FTM1_QDCTRL)	32	R/W	0000_0000h	41.4.21/ 1008
4003_9084	Configuration (FTM1_CONF)	32	R/W	0000_0000h	41.4.22/ 1010
4003_9088	FTM Fault Input Polarity (FTM1_FLTPOL)	32	R/W	0000_0000h	41.4.23/ 1011
4003_908C	Synchronization Configuration (FTM1_SYNCONF)	32	R/W	0000_0000h	41.4.24/ 1012
4003_9090	FTM Inverting Control (FTM1_INVCTRL)	32	R/W	0000_0000h	41.4.25/ 1014
4003_9094	FTM Software Output Control (FTM1_SWOCTRL)	32	R/W	0000_0000h	41.4.26/ 1015
4003_9098	FTM PWM Load (FTM1_PWMLOAD)	32	R/W	0000_0000h	41.4.27/ 1018
4003_909C	Half Cycle Register (FTM1_HCR)	32	R/W	0000_0000h	41.4.28/ 1020
4003_9200	Mirror of Modulo Value (FTM1_MOD_MIRROR)	32	R/W	0000_0000h	41.4.29/ 1020
4003_9204	Mirror of Channel (n) Match Value (FTM1_C0V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9208	Mirror of Channel (n) Match Value (FTM1_C1V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_920C	Mirror of Channel (n) Match Value (FTM1_C2V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9210	Mirror of Channel (n) Match Value (FTM1_C3V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9214	Mirror of Channel (n) Match Value (FTM1_C4V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9218	Mirror of Channel (n) Match Value (FTM1_C5V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_921C	Mirror of Channel (n) Match Value (FTM1_C6V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_9220	Mirror of Channel (n) Match Value (FTM1_C7V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A000	Status And Control (FTM2_SC)	32	R/W	0000_0000h	41.4.3/973
4003_A004	Counter (FTM2_CNT)	32	R/W	0000_0000h	41.4.4/977

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_A008	Modulo (FTM2_MOD)	32	R/W	0000_0000h	41.4.5/978
4003_A00C	Channel (n) Status And Control (FTM2_C0SC)	32	R/W	0000_0000h	41.4.6/979
4003_A010	Channel (n) Value (FTM2_C0V)	32	R/W	0000_0000h	41.4.7/981
4003_A014	Channel (n) Status And Control (FTM2_C1SC)	32	R/W	0000_0000h	41.4.6/979
4003_A018	Channel (n) Value (FTM2_C1V)	32	R/W	0000_0000h	41.4.7/981
4003_A01C	Channel (n) Status And Control (FTM2_C2SC)	32	R/W	0000_0000h	41.4.6/979
4003_A020	Channel (n) Value (FTM2_C2V)	32	R/W	0000_0000h	41.4.7/981
4003_A024	Channel (n) Status And Control (FTM2_C3SC)	32	R/W	0000_0000h	41.4.6/979
4003_A028	Channel (n) Value (FTM2_C3V)	32	R/W	0000_0000h	41.4.7/981
4003_A02C	Channel (n) Status And Control (FTM2_C4SC)	32	R/W	0000_0000h	41.4.6/979
4003_A030	Channel (n) Value (FTM2_C4V)	32	R/W	0000_0000h	41.4.7/981
4003_A034	Channel (n) Status And Control (FTM2_C5SC)	32	R/W	0000_0000h	41.4.6/979
4003_A038	Channel (n) Value (FTM2_C5V)	32	R/W	0000_0000h	41.4.7/981
4003_A03C	Channel (n) Status And Control (FTM2_C6SC)	32	R/W	0000_0000h	41.4.6/979
4003_A040	Channel (n) Value (FTM2_C6V)	32	R/W	0000_0000h	41.4.7/981
4003_A044	Channel (n) Status And Control (FTM2_C7SC)	32	R/W	0000_0000h	41.4.6/979
4003_A048	Channel (n) Value (FTM2_C7V)	32	R/W	0000_0000h	41.4.7/981
4003_A04C	Counter Initial Value (FTM2_CNTIN)	32	R/W	0000_0000h	41.4.8/981
4003_A050	Capture And Compare Status (FTM2_STATUS)	32	R/W	0000_0000h	41.4.9/982
4003_A054	Features Mode Selection (FTM2_MODE)	32	R/W	0000_0004h	41.4.10/ 984
4003_A058	Synchronization (FTM2_SYNC)	32	R/W	0000_0000h	41.4.11/ 986
4003_A05C	Initial State For Channels Output (FTM2_OUTINIT)	32	R/W	0000_0000h	41.4.12/ 988
4003_A060	Output Mask (FTM2_OUTMASK)	32	R/W	0000_0000h	41.4.13/ 990
4003_A064	Function For Linked Channels (FTM2_COMBINE)	32	R/W	0000_0000h	41.4.14/ 992
4003_A068	Deadtime Configuration (FTM2_DEADTIME)	32	R/W	0000_0000h	41.4.15/ 996
4003_A06C	FTM External Trigger (FTM2_EXTTRIG)	32	R/W	0000_0000h	41.4.16/ 997
4003_A070	Channels Polarity (FTM2_POL)	32	R/W	0000_0000h	41.4.17/ 999
4003_A074	Fault Mode Status (FTM2_FMS)	32	R/W	0000_0000h	41.4.18/ 1002
4003_A078	Input Capture Filter Control (FTM2_FILTER)	32	R/W	0000_0000h	41.4.19/ 1004
4003_A07C	Fault Control (FTM2_FLTCTRL)	32	R/W	0000_0000h	41.4.20/ 1005

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_A080	Quadrature Decoder Control And Status (FTM2_QDCTRL)	32	R/W	0000_0000h	41.4.21/ 1008
4003_A084	Configuration (FTM2_CONF)	32	R/W	0000_0000h	41.4.22/ 1010
4003_A088	FTM Fault Input Polarity (FTM2_FLTPOL)	32	R/W	0000_0000h	41.4.23/ 1011
4003_A08C	Synchronization Configuration (FTM2_SYNCONF)	32	R/W	0000_0000h	41.4.24/ 1012
4003_A090	FTM Inverting Control (FTM2_INVCTRL)	32	R/W	0000_0000h	41.4.25/ 1014
4003_A094	FTM Software Output Control (FTM2_SWOCTRL)	32	R/W	0000_0000h	41.4.26/ 1015
4003_A098	FTM PWM Load (FTM2_PWMLOAD)	32	R/W	0000_0000h	41.4.27/ 1018
4003_A09C	Half Cycle Register (FTM2_HCR)	32	R/W	0000_0000h	41.4.28/ 1020
4003_A200	Mirror of Modulo Value (FTM2_MOD_MIRROR)	32	R/W	0000_0000h	41.4.29/ 1020
4003_A204	Mirror of Channel (n) Match Value (FTM2_C0V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A208	Mirror of Channel (n) Match Value (FTM2_C1V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A20C	Mirror of Channel (n) Match Value (FTM2_C2V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A210	Mirror of Channel (n) Match Value (FTM2_C3V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A214	Mirror of Channel (n) Match Value (FTM2_C4V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A218	Mirror of Channel (n) Match Value (FTM2_C5V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A21C	Mirror of Channel (n) Match Value (FTM2_C6V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021
4003_A220	Mirror of Channel (n) Match Value (FTM2_C7V_MIRROR)	32	R/W	0000_0000h	41.4.30/ 1021

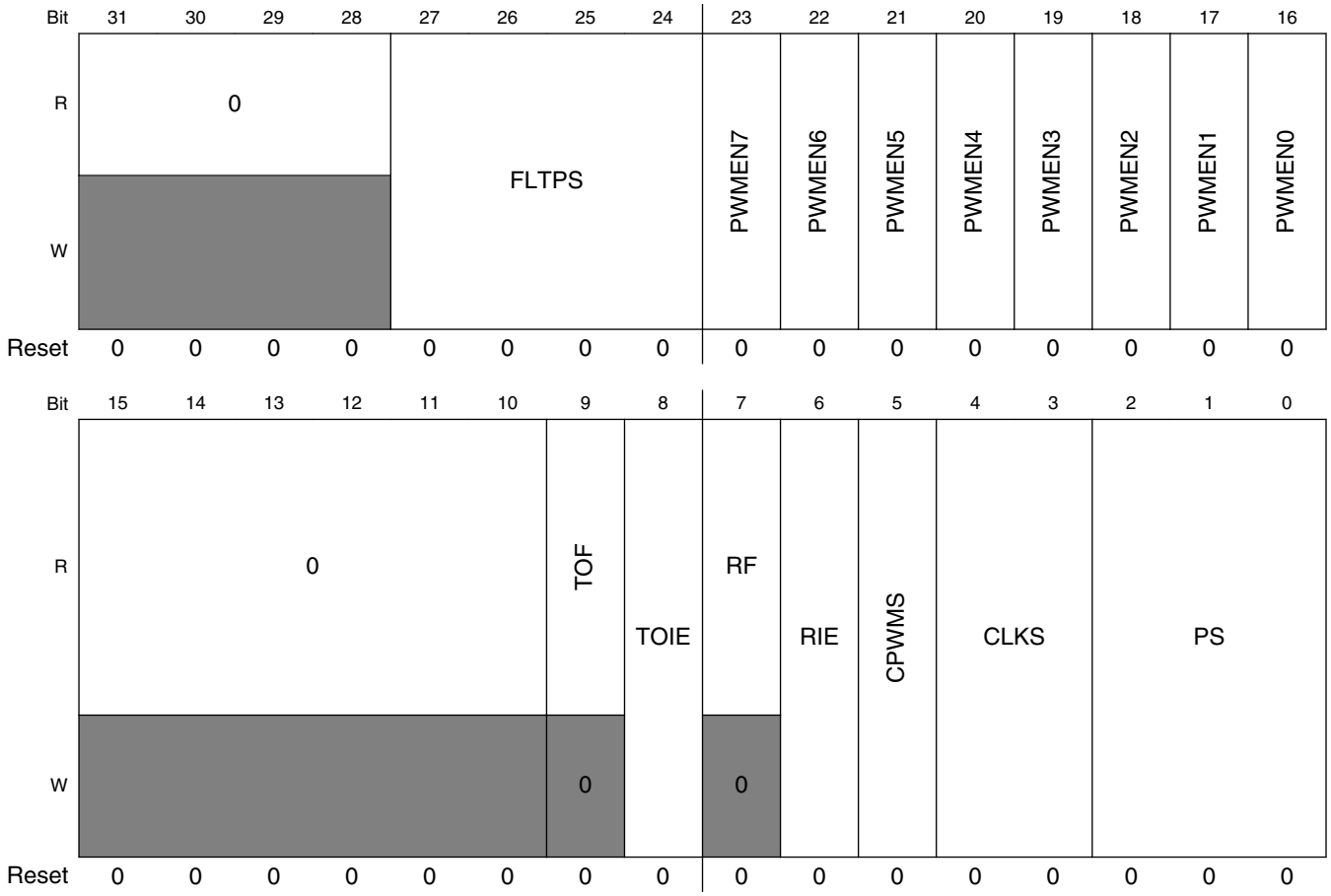
41.4.3 Status And Control (FTMx_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, filter prescaler, and prescaler factor.

This register also contains the output enable control bits and the reload opportunity flag control.

These controls relate to all channels within this module.

Address: Base address + 0h offset



FTMx_SC 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 FLTPS	Filter Prescaler Selects one of 16 division factors for the clock used at input filters. The new prescaler factor has effect on the next FTM input clock cycle after the new value is updated into the register bits. The filter prescaler affects the channel input filters, quadrature input filters and fault control input filters. 0000 Divide by 1 0001 Divide by 2 0010 Divide by 3 0011 Divide by 4 0100 Divide by 5 0101 Divide by 6 0110 Divide by 7 0111 Divide by 8 1000 Divide by 9 1001 Divide by 10 1010 Divide by 11 1011 Divide by 12

Table continues on the next page...

FTMx_SC field descriptions (continued)

Field	Description
	1100 Divide by 13 1101 Divide by 14 1110 Divide by 15 1111 Divide by 16
23 PWMEN7	Channel 7 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
22 PWMEN6	Channel 6 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
21 PWMEN5	Channel 5 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
20 PWMEN4	Channel 4 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
19 PWMEN3	Channel 3 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
18 PWMEN2	Channel 2 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
17 PWMEN1	Channel 1 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled

Table continues on the next page...

FTMx_SC field descriptions (continued)

Field	Description
16 PWMEN0	Channel 0 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0 Channel output port is disabled 1 Channel output port is enabled
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect. If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case, a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0 FTM counter has not overflowed. 1 FTM counter has overflowed.
8 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts. 0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.
7 RF	Reload Flag Set by hardware when FTM counter matches the value of a reload point configured by FTMxPWMLOAD register. The RF bit is cleared by reading the SC register while RF is set and then writing a 0 to RF bit. Writing 1 to RF has no effect. If another reload point is reached between the read and write operations, the write operation has no effect; therefore, RF remains set. 0 FTM counter did not reach a reload point. 1 FTM counter reached a reload point.
6 RIE	Reload Interrupt Enable Enables the reload oportunity interrupt. 0 Reload interrupt is disabled. 1 Reload interrupt is enabled.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in Up-Down Counting mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 FTM counter operates in Up Counting mode. 1 FTM counter operates in Up-Down Counting mode.
4–3 CLKS	Clock Source Selection Selects one of the three FTM counter clock sources.

Table continues on the next page...

FTMx_SC field descriptions (continued)

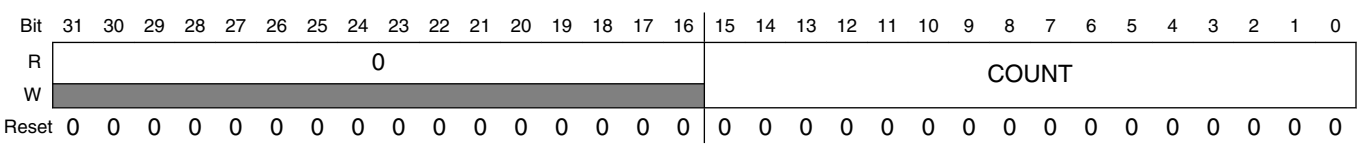
Field	Description
	This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 No clock selected. This in effect disables the FTM counter. 01 FTM input clock 10 Fixed frequency clock 11 External clock
PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next FTM input clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

41.4.4 Counter (FTMx_CNT)

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

Address: Base address + 4h offset



FTMx_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Counter Value

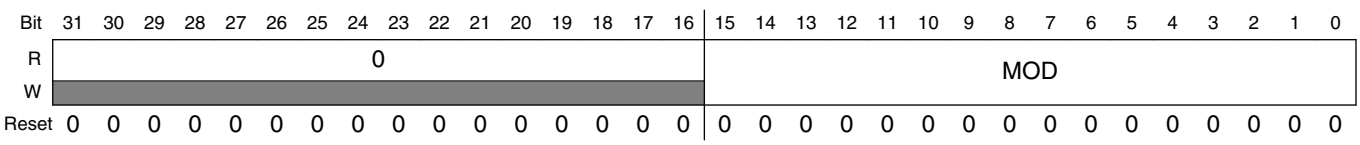
41.4.5 Modulo (FTMx_MOD)

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock cycle, and the next value of FTM counter depends on the selected counting method; see Counter.

Writes to the MOD register are done on its write buffer. The MOD register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.

Initialize the FTM counter, by writing to CNT, before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Address: Base address + 8h offset



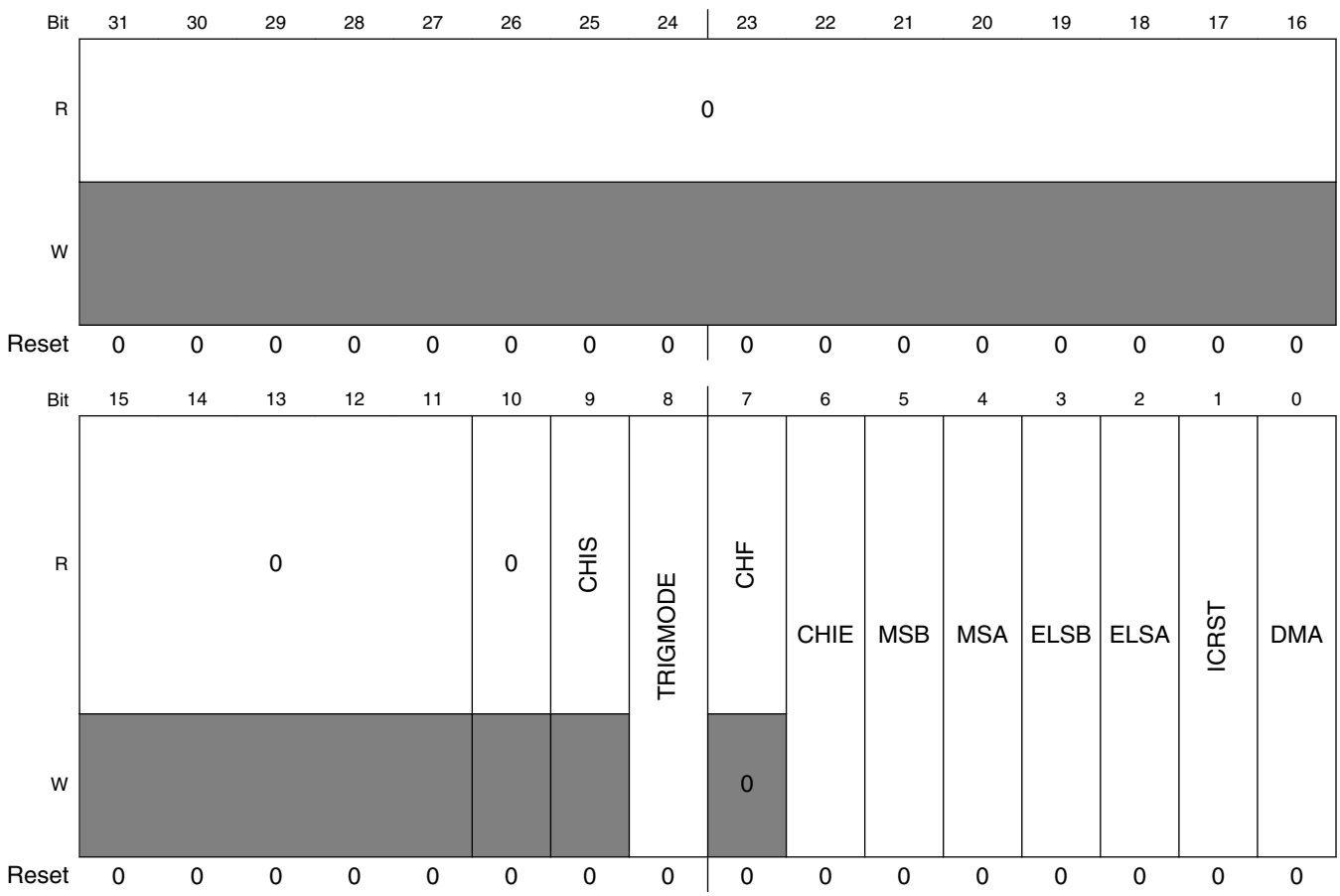
FTMx_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MOD	Modulo Value

41.4.6 Channel (n) Status And Control (FTM*x*_C*n*SC)

CnSC contains channel (n) status bits and control bits that select the channel (n) mode and its functionality.

Address: Base address + Ch offset + (8d × i), where i=0d to 7d



FTM*x*_C*n*SC field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 CHIS	Channel (n) Input State The CHIS bit has the value of the channel (n) input after the double-sampling or the filtering (if the channel (n) filter is enabled) both them are inside the FTM. **NOTE:** The CHIS bit should be ignored when the channel (n) is not in an input mode.

Table continues on the next page...

FTMx_CnSC field descriptions (continued)

Field	Description
	NOTE: When the pair channels is on dual edge mode, the channel (n+1) CHIS bit is the channel (n+1) input value and not the channel (n) input value (this signal is the input signal used by the dual edge mode). 0 The channel (n) input is zero. 1 The channel (n) input is one.
8 TRIGMODE	Trigger mode control This bit controls the trigger generation on FTM channel outputs. This mode is allowed only if when FTM channel is configured to EPWM (up counting) or CPWM (up-down counting) modes. If a match in the channel occurs, a trigger pulse with one FTM clock cycle width will be generated in the channel output. See Channel trigger output for more details about trigger mode feature. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channel outputs will generate the normal PWM outputs without generating a pulse. 1 If a match in the channel occurs, a trigger generation on channel output will happen. The trigger pulse width has one FTM clock cycle.
7 CHF	Channel (n) Flag Set by hardware when an event occurs on the channel (n). CHF is cleared by reading the CnSC register while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF. 0 No channel (n) event has occurred. 1 A channel (n) event has occurred.
6 CHIE	Channel (n) Interrupt Enable Enables channel (n) interrupt. 0 Disable channel (n) interrupt. Use software polling. 1 Enable channel (n) interrupt.
5 MSB	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
4 MSA	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
3 ELSB	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
2 ELSA	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_C*n*SC field descriptions (continued)

Field	Description
1 ICRST	FTM counter reset by the selected input capture event. FTM counter reset is driven by the selected event of the channel (n) in the Input Capture mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 FTM counter is not reset when the selected channel (n) input event is detected. 1 FTM counter is reset when the selected channel (n) input event is detected.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0 Disable DMA transfers. 1 Enable DMA transfers.

41.4.7 Channel (n) Value (FTMx_C*n*V)

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In Input Capture, Capture Test, and Dual Edge Capture modes, any write to a CnV register is ignored.

In output modes, writes to the CnV register are done on its write buffer. The CnV register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.

Address: Base address + 10h offset + (8d × 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	0																VAL															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_C*n*V field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

41.4.8 Counter Initial Value (FTMx_CNTIN)

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to Registers updated from write buffers.

When the FTM clock is initially selected, by writing a non-zero value to the CLKS bits, the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the the CNTIN register and then initialize the FTM counter by writing any value to the CNT register.

Address: Base address + 4Ch offset

Bit	31	30	29	28	27	26	25	24	23	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																INIT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_CNTIN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
INIT	Initial Value Of The FTM Counter

41.4.9 Capture And Compare Status (FTMx_STATUS)

The STATUS register contains a copy of the status flag CHF bit in CnSC for each FTM channel for software convenience.

Each CHF bit in STATUS is a mirror of CHF bit in CnSC. All CHF bits can be checked using only one read of STATUS. All CHF bits can be cleared by reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHF is cleared by reading STATUS while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect.

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case, a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

Address: Base address + 50h 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								CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_STATUS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7F	Channel 7 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
6 CH6F	Channel 6 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
5 CH5F	Channel 5 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
4 CH4F	Channel 4 Flag See the register description.

Table continues on the next page...

FTMx_STATUS field descriptions (continued)

Field	Description
	0 No channel event has occurred. 1 A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.

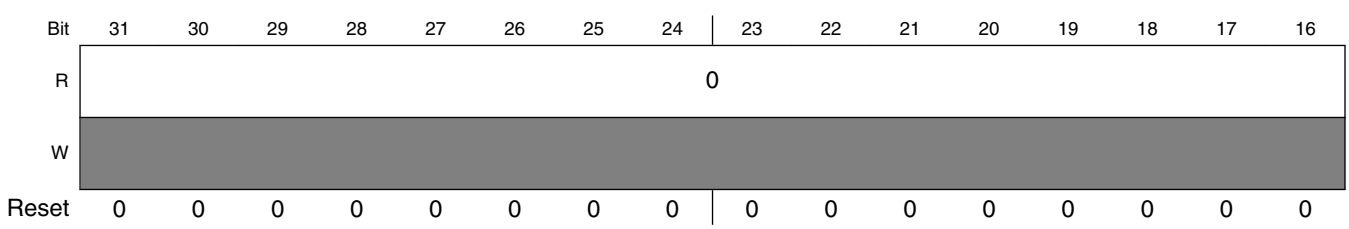
41.4.10 Features Mode Selection (FTMx_MODE)

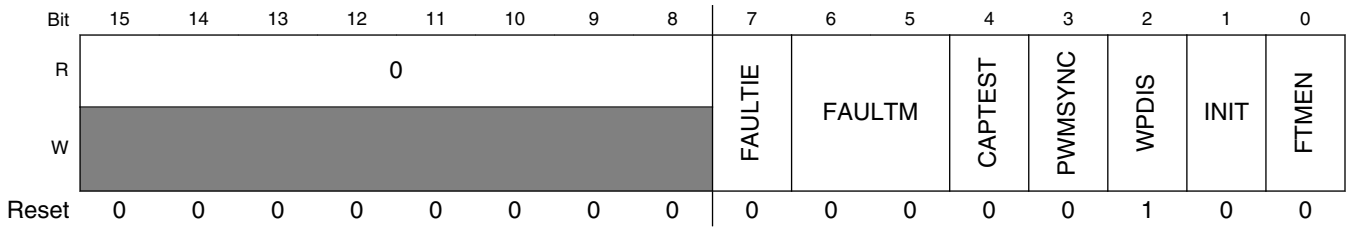
This register contains the global enable bit for FTM-specific features and the control bits used to configure:

- Fault control mode and interrupt
- Capture Test mode
- PWM synchronization
- Write protection
- Channel output initialization

These controls relate to all channels within this module.

Address: Base address + 54h offset





FTMx_MODE field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 FAULTIE	Fault Interrupt Enable Enables the generation of an interrupt when a fault is detected by FTM and the FTM fault control is enabled. 0 Fault control interrupt is disabled. 1 Fault control interrupt is enabled.
6–5 FAULTM	Fault Control Mode Defines the FTM fault control mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 Fault control is disabled for all channels. 01 Fault control is enabled for even channels only (channels 0, 2, 4, and 6), and the selected mode is the manual fault clearing. 10 Fault control is enabled for all channels, and the selected mode is the manual fault clearing. 11 Fault control is enabled for all channels, and the selected mode is the automatic fault clearing.
4 CAPTEST	Capture Test Mode Enable Enables the capture test mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Capture test mode is disabled. 1 Capture test mode is enabled.
3 PWMSYNC	PWM Synchronization Mode Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. See PWM synchronization. The PWMSYNC bit configures the synchronization when SYNCMODE is 0. 0 No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. 1 Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.
2 WPDIS	Write Protection Disable When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect.

Table continues on the next page...

FTMx_MODE field descriptions (continued)

Field	Description
	0 Write protection is enabled. 1 Write protection is disabled.
1 INIT	Initialize The Channels Output When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect. The INIT bit is always read as 0.
0 FTMEN	FTM Enable This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 TPM compatibility. Free running counter and synchronization compatible with TPM. 1 Free running counter and synchronization are different from TPM behavior.

41.4.11 Synchronization (FTMx_SYNC)

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

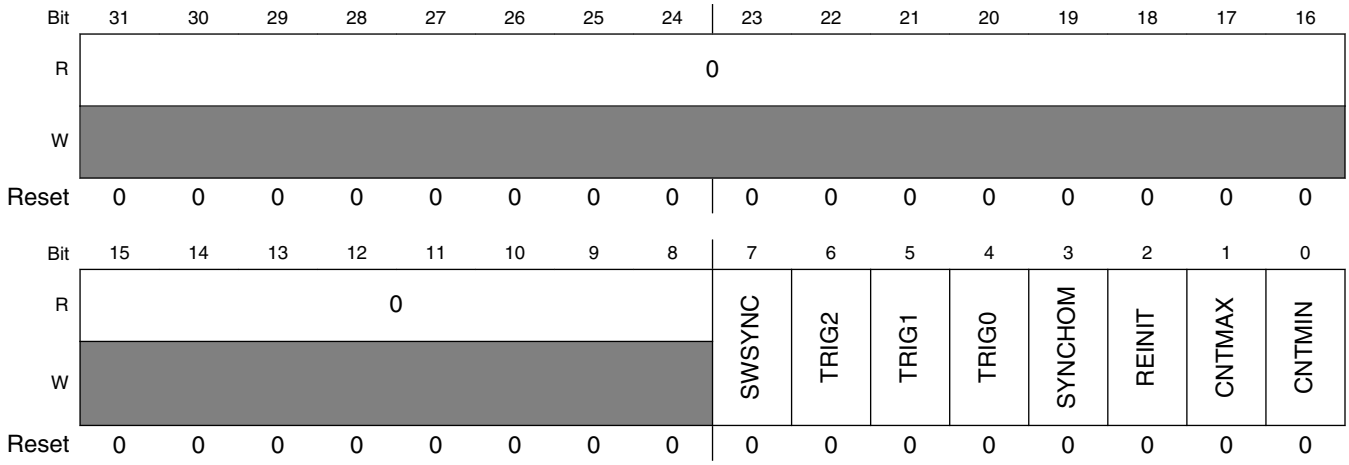
NOTE

The software trigger, SWSYNC bit, and hardware triggers TRIG0, TRIG1, and TRIG2 bits have a potential conflict if used together when SYNCMODE = 0. Use only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.

The selection of the loading point, CNTMAX and CNTMIN bits, is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection, TRIG0, TRIG1, or TRIG2 bits, is likely to result in unpredictable behavior.

The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See PWM synchronization.

Address: Base address + 58h offset



FTMx_SYNC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 SWSYNC	PWM Synchronization Software Trigger Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit. 0 Software trigger is not selected. 1 Software trigger is selected.
6 TRIG2	PWM Synchronization Hardware Trigger 2 Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
5 TRIG1	PWM Synchronization Hardware Trigger 1 Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
4 TRIG0	PWM Synchronization Hardware Trigger 0 Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 occurs when a rising edge is detected at the trigger 0 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
3 SYNCHOM	Output Mask Synchronization Selects when the OUTMASK register is updated with the value of its buffer.

Table continues on the next page...

FTMx_SYNC field descriptions (continued)

Field	Description
	0 OUTMASK register is updated with the value of its buffer in all rising edges of the FTM input clock. 1 OUTMASK register is updated with the value of its buffer only by the PWM synchronization.
2 REINIT	FTM Counter Reinitialization By Synchronization (FTM counter synchronization) Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected. The REINIT bit configures the synchronization when SYNCMODE is zero. 0 FTM counter continues to count normally. 1 FTM counter is updated with its initial value when the selected trigger is detected.
1 CNTMAX	Maximum Loading Point Enable Selects the maximum loading point to PWM synchronization. See Synchronization Points. If CNTMAX is 1, the selected loading point is when the FTM counter reaches its maximum value (MOD register). 0 The maximum loading point is disabled. 1 The maximum loading point is enabled.
0 CNTMIN	Minimum Loading Point Enable Selects the minimum loading point to PWM synchronization. See Synchronization Points. If CNTMIN is one, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register). 0 The minimum loading point is disabled. 1 The minimum loading point is enabled.

41.4.12 Initial State For Channels Output (FTMx_OUTINIT)

Address: Base address + 5Ch 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								CH7OI	CH6OI	CH5OI	CH4OI	CH3OI	CH2OI	CH1OI	CH0OI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_OUTINIT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

FTMx_OUTINIT field descriptions (continued)

Field	Description
7 CH7OI	Channel 7 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
6 CH6OI	Channel 6 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
5 CH5OI	Channel 5 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
4 CH4OI	Channel 4 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
3 CH3OI	Channel 3 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
2 CH2OI	Channel 2 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
1 CH1OI	Channel 1 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
0 CH0OI	Channel 0 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.

41.4.13 Output Mask (FTMx_OUTMASK)

This register provides a mask for each FTM channel. The mask of a channel determines if its output responds, that is, it is masked or not, when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to PWM synchronization.

Output Mask bits must not be set for trigger mode.

Address: Base address + 60h 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								CH7OM	CH6OM	CH5OM	CH4OM	CH3OM	CH2OM	CH1OM	CH0OM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_OUTMASK field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7OM	Channel 7 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
6 CH6OM	Channel 6 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
5 CH5OM	Channel 5 Output Mask Defines if the channel output is masked or unmasked.

Table continues on the next page...

FTMx_OUTMASK field descriptions (continued)

Field	Description
	0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
4 CH4OM	Channel 4 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
3 CH3OM	Channel 3 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
2 CH2OM	Channel 2 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
1 CH1OM	Channel 1 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
0 CH0OM	Channel 0 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.

41.4.14 Function For Linked Channels (FTMx_COMBINE)

This register contains the configuration bits for each pair of channels.

Address: Base address + 64h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	FAULTEN3	SYNCEN3	DTEN3	DECAP3	DECAPEN3	COMP3	COMBINE3	0	FAULTEN2	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FAULTEN1	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	0	FAULTEN0	SYNCEN0	DTEN0	DECAP0	DECAPEN0	COMP0	COMBINE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_COMBINE field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 FAULTEN3	Fault Control Enable For n = 6 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
29 SYNCEN3	Synchronization Enable For n = 6 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
28 DTEN3	Deadtime Enable For n = 6 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
27 DECAP3	Dual Edge Capture Mode Captures For n = 6

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
26 DECAPEN3	Dual Edge Capture Mode Enable For n = 6 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
25 COMP3	Complement Of Channel (n) for n = 6 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
24 COMBINE3	Combine Channels For n = 6 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 FAULTEN2	Fault Control Enable For n = 4 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
21 SYNCEN2	Synchronization Enable For n = 4 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
20 DTEN2	Deadtime Enable For n = 4 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
19 DECAP2	Dual Edge Capture Mode Captures For n = 4 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
18 DECAPEN2	Dual Edge Capture Mode Enable For n = 4 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
17 COMP2	Complement Of Channel (n) For n = 4 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
16 COMBINE2	Combine Channels For n = 4 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 FAULTEN1	Fault Control Enable For n = 2 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
13 SYNCEN1	Synchronization Enable For n = 2 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
12 DTEN1	Deadtime Enable For n = 2 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
11 DECAP1	Dual Edge Capture Mode Captures For n = 2 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if Dual Edge Capture – One-Shot mode is selected and when the capture of channel (n+1) event is made.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	0 The dual edge captures are inactive. 1 The dual edge captures are active.
10 DECAPEN1	Dual Edge Capture Mode Enable For n = 2 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
9 COMP1	Complement Of Channel (n) For n = 2 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
8 COMBINE1	Combine Channels For n = 2 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FAULTEN0	Fault Control Enable For n = 0 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
5 SYNCEN0	Synchronization Enable For n = 0 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
4 DTEN0	Deadtime Enable For n = 0 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
3 DECAP0	Dual Edge Capture Mode Captures For n = 0 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
2 DECAPEN0	Dual Edge Capture Mode Enable For n = 0 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 COMP0	Complement Of Channel (n) For n = 0 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
0 COMBINE0	Combine Channels For n = 0 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.

41.4.15 Deadtime Configuration (FTMx_DEADTIME)

This register selects the deadtime prescaler and value for all pair of channels.

Address: Base address + 68h offset

Bit	31	30	29	28	27	26	25	24	23	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												DTVALEX				0								DTPS		DTVAL					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_DEADTIME field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DTVALEX	Extended Deadtime Value This field is a bit extension of the DTVAL field. It defines the 4 most significant bits of the deadtime value. The maximum deadtime value is extended to 1023 using the concatenation {DTVALEX, DTVAL}. Deadtime insert value = (DTPS × {DTVALEX, DTVAL}). This field is write protected. It can be written only when MODE[WPDIS] = 1. **NOTE:** If full compatibility is needed with previous software versions, write 0 to DTVALEX bits.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 DTPS	Deadtime Prescaler Value Selects the division factor of the FTM input clock. This prescaled clock is used by the deadtime counter.

Table continues on the next page...

FTMx_DEADTIME field descriptions (continued)

Field	Description
	This field is write protected. It can be written only when MODE[WPDIS] = 1. 0x Divide the FTM input clock by 1. 10 Divide the FTM input clock by 4. 11 Divide the FTM input clock by 16.
DTVAL	Deadtime Value Selects the deadtime value. This field is write protected. It can be written only when MODE[WPDIS] = 1.

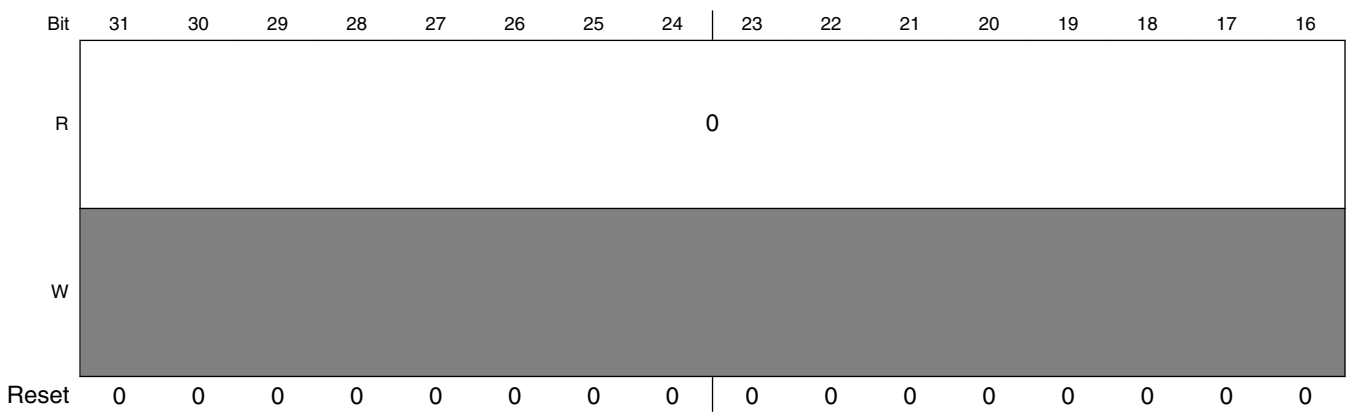
41.4.16 FTM External Trigger (FTMx_EXTTRIG)

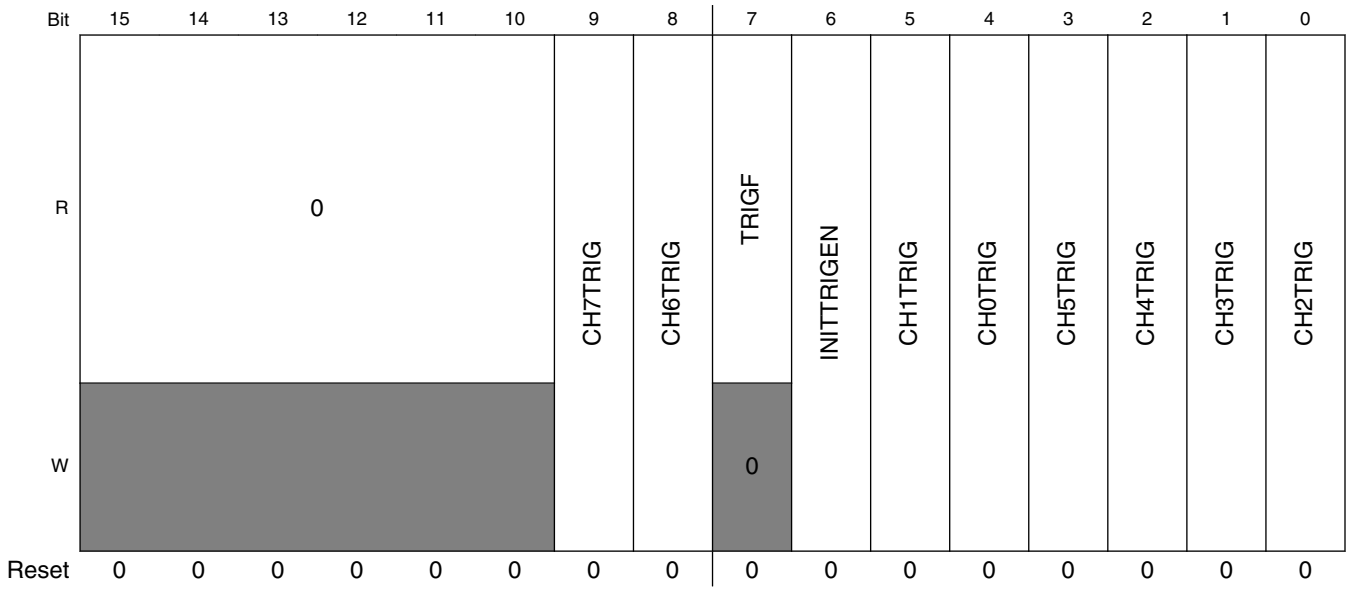
This register:

- Indicates when a channel trigger was generated
- Enables the generation of a trigger when the FTM counter is equal to its initial value
- Selects which channels are used in the generation of the channel triggers

Several channels can be selected to generate multiple triggers in one PWM period. See External Trigger and Initialization trigger

Address: Base address + 6Ch offset





FTMx_EXTTRIG field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 CH7TRIG	Channel 7 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
8 CH6TRIG	Channel 6 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
7 TRIGF	Channel Trigger Flag Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect. If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF. 0 No channel trigger was generated. 1 A channel trigger was generated.
6 INITTRIGEN	Initialization Trigger Enable Enables the generation of the trigger when the FTM counter is equal to the CNTIN register. 0 The generation of initialization trigger is disabled. 1 The generation of initialization trigger is enabled.
5 CH1TRIG	Channel 1 Trigger Enable

Table continues on the next page...

FTMx_EXTTRIG field descriptions (continued)

Field	Description
	Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
4 CH0TRIG	Channel 0 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
3 CH5TRIG	Channel 5 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
2 CH4TRIG	Channel 4 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
1 CH3TRIG	Channel 3 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
0 CH2TRIG	Channel 2 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.

41.4.17 Channels Polarity (FTMx_POL)

This register defines the output polarity of the FTM channels.

NOTE

The safe value that is driven in a channel output when the fault control is enabled and a fault condition is detected is the inactive state of the channel. That is, the safe value of a channel is the value of its POL bit.

Address: Base address + 70h 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								POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_POL field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 POL7	Channel 7 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
6 POL6	Channel 6 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
5 POL5	Channel 5 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
4 POL4	Channel 4 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
3 POL3	Channel 3 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
2 POL2	Channel 2 Polarity

Table continues on the next page...

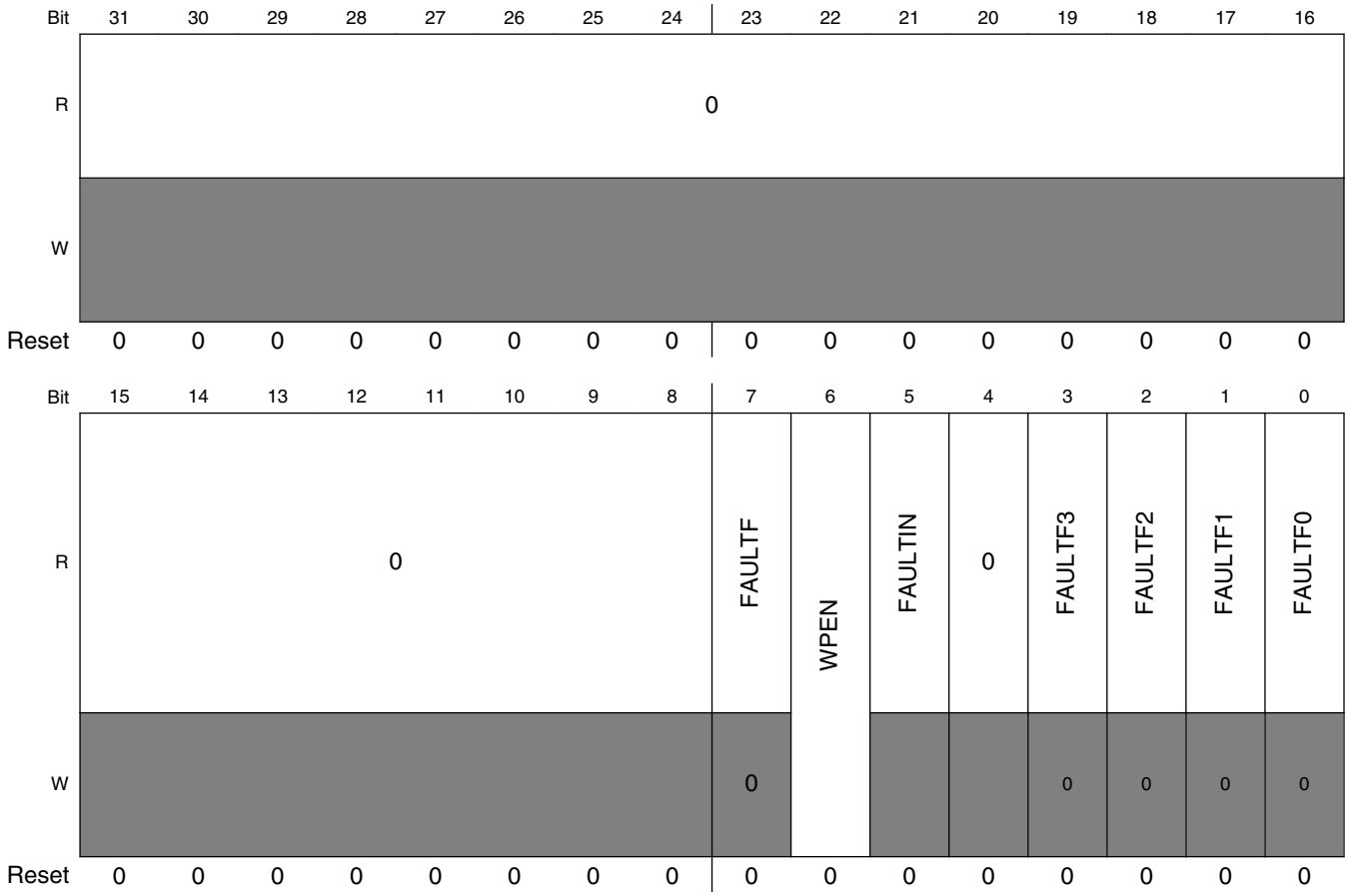
FTMx_POL field descriptions (continued)

Field	Description
	Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
1 POL1	Channel 1 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
0 POL0	Channel 0 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.

41.4.18 Fault Mode Status (FTMx_FMS)

This register contains the fault detection flags, write protection enable bit, and the logic OR of the enabled fault inputs.

Address: Base address + 74h offset



FTMx_FMS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 FAULTF	Fault Detection Flag Represents the logic OR of the individual FAULTFj bits where j = 3, 2, 1, 0. Clear FAULTF by reading the FMS register while FAULTF is set and then writing a 0 to FAULTF while there is no existing fault condition at the enabled fault inputs. Writing a 1 to FAULTF has no effect. If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTF remains set after the clearing sequence is completed for the earlier fault condition. FAULTF is also cleared when FAULTFj bits are cleared individually. 0 No fault condition was detected. 1 A fault condition was detected.

Table continues on the next page...

FTMx_FMS field descriptions (continued)

Field	Description
6 WPEN	Write Protection Enable The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect. 0 Write protection is disabled. Write protected bits can be written. 1 Write protection is enabled. Write protected bits cannot be written.
5 FAULTIN	Fault Inputs Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled. 0 The logic OR of the enabled fault inputs is 0. 1 The logic OR of the enabled fault inputs is 1.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FAULTF3	Fault Detection Flag 3 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF3 by reading the FMS register while FAULTF3 is set and then writing a 0 to FAULTF3 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF3 has no effect. FAULTF3 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF3 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
2 FAULTF2	Fault Detection Flag 2 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF2 by reading the FMS register while FAULTF2 is set and then writing a 0 to FAULTF2 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF2 has no effect. FAULTF2 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF2 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
1 FAULTF1	Fault Detection Flag 1 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF1 by reading the FMS register while FAULTF1 is set and then writing a 0 to FAULTF1 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF1 has no effect. FAULTF1 bit is also cleared when FAULTF bit is cleared.

Table continues on the next page...

FTMx_FMS field descriptions (continued)

Field	Description
	If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF1 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
0 FAULTF0	Fault Detection Flag 0 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF0 by reading the FMS register while FAULTF0 is set and then writing a 0 to FAULTF0 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF0 has no effect. FAULTF0 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF0 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.

41.4.19 Input Capture Filter Control (FTMx_FILTER)

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

NOTE

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

Address: Base address + 78h offset

Bit	31	30	29	28	27	26	25	24	23	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																CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FILTER field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 CH3FVAL	Channel 3 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

Table continues on the next page...

FTMx_FILTER field descriptions (continued)

Field	Description
11–8 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
7–4 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

41.4.20 Fault Control (FTMx_FLTCTRL)

This register selects the filter value for the fault inputs, enables the fault inputs and the fault inputs filter. This register also controls the output state when a fault event happens.

Address: Base address + 7Ch 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	FSTATE	0			FFVAL				FFLTR3EN	FFLTR2EN	FFLTR1EN	FFLTR0EN	FAULT3EN	FAULT2EN	FAULT1EN	FAULT0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FLTCTRL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 FSTATE	Fault output state This configuration allows to put the FTM outputs tri-stated when a fault event is ongoing. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_FLTCTRL field descriptions (continued)

Field	Description
	0 FTM outputs will be placed into safe values when fault events in ongoing (defined by POL bits). 1 FTM outputs will be tri-stated when fault event is ongoing
14–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 FFVAL	Fault Input Filter Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero. **NOTE:** Writing to this field has immediate effect and must be done only when the fault control or all fault inputs are disabled. Failure to do this could result in a missing fault detection.
7 FFLTR3EN	Fault Input 3 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
6 FFLTR2EN	Fault Input 2 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
5 FFLTR1EN	Fault Input 1 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
4 FFLTR0EN	Fault Input 0 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
3 FAULT3EN	Fault Input 3 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
2 FAULT2EN	Fault Input 2 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

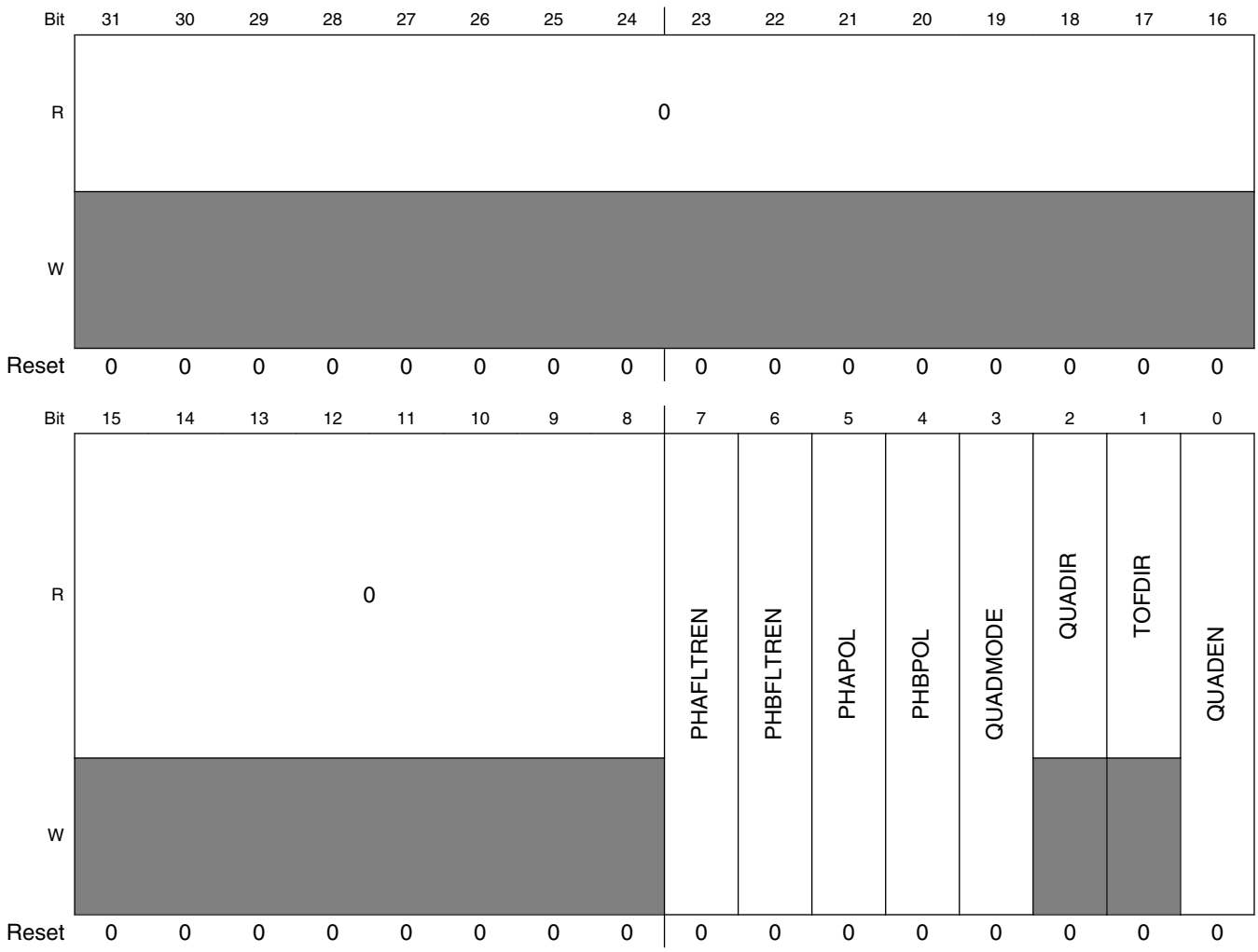
FTMx_FLTCTRL field descriptions (continued)

Field	Description
	0 Fault input is disabled. 1 Fault input is enabled.
1 FAULT1EN	Fault Input 1 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
0 FAULT0EN	Fault Input 0 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.

41.4.21 Quadrature Decoder Control And Status (FTMx_QDCTRL)

This register has the control and status bits for the Quadrature Decoder mode.

Address: Base address + 80h offset



FTMx_QDCTRL field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 PHAFLTREN	Phase A Input Filter Enable Enables the filter for the quadrature decoder phase A input. The filter value for the phase A input is defined by the CH0FVAL field of FILTER. The phase A filter is also disabled when CH0FVAL is zero. 0 Phase A input filter is disabled. 1 Phase A input filter is enabled.

Table continues on the next page...

FTMx_QDCTRL field descriptions (continued)

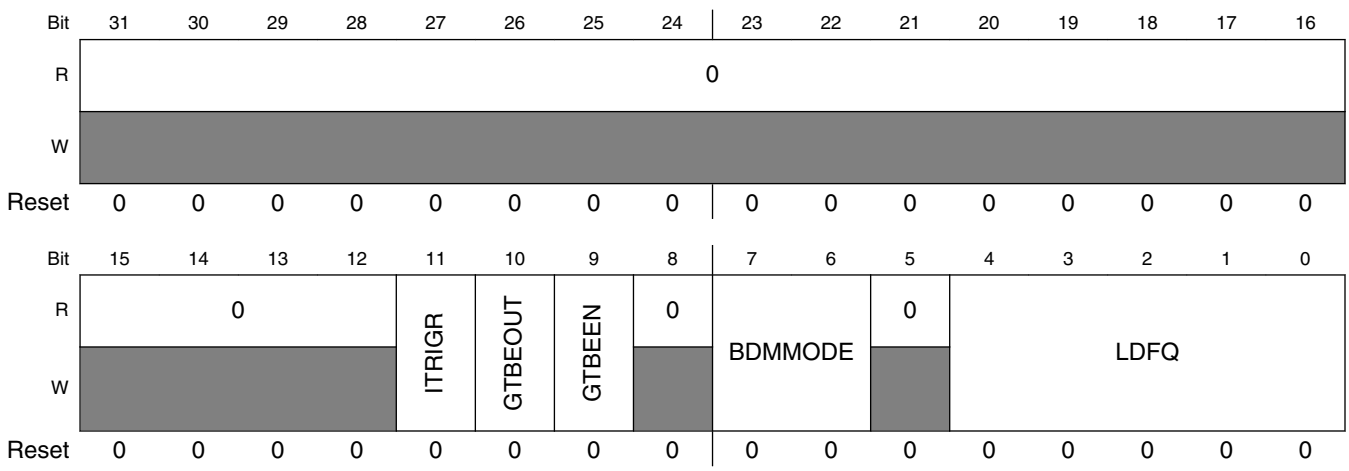
Field	Description
6 PHBFLTREN	Phase B Input Filter Enable Enables the filter for the quadrature decoder phase B input. The filter value for the phase B input is defined by the CH1FVAL field of FILTER. The phase B filter is also disabled when CH1FVAL is zero. 0 Phase B input filter is disabled. 1 Phase B input filter is enabled.
5 PHAPOL	Phase A Input Polarity Selects the polarity for the quadrature decoder phase A input. 0 Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal.
4 PHBPOL	Phase B Input Polarity Selects the polarity for the quadrature decoder phase B input. 0 Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal.
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the Quadrature Decoder mode. 0 Phase A and phase B encoding mode. 1 Count and direction encoding mode.
2 QUADIR	FTM Counter Direction In Quadrature Decoder Mode Indicates the counting direction. 0 Counting direction is decreasing (FTM counter decrement). 1 Counting direction is increasing (FTM counter increment).
1 TOFDIR	Timer Overflow Direction In Quadrature Decoder Mode Indicates if the TOF bit was set on the top or the bottom of counting. 0 TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (CNTIN register) to its maximum value (MOD register). 1 TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (CNTIN register).
0 QUADEN	Quadrature Decoder Mode Enable Enables the Quadrature Decoder mode. In this mode, the phase A and B input signals control the FTM counter direction. The Quadrature Decoder mode has precedence over the other modes. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Quadrature Decoder mode is disabled. 1 Quadrature Decoder mode is enabled.

41.4.22 Configuration (FTMx_CONF)

This register selects the number of times that a reload opportunity should occur before the RF bit is set, the FTM behavior in Debug modes, the use of an external global time base, and the global time base signal generation.

This register also controls if initialization trigger should be generated when a reload point is reached.

Address: Base address + 84h offset



FTMx_CONF field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 ITRIGR	Initialization trigger on Reload Point This bit controls whether an initialization trigger is generated when a reload point configured by PWMLOAD register is reached considering the FTM_CONF[LDFQ] settings. 0 Initialization trigger is generated on counter wrap events. 1 Initialization trigger is generated when a reload point is reached.
10 GTBEOUT	Global Time Base Output Enables the global time base signal generation to other FTMs. 0 A global time base signal generation is disabled. 1 A global time base signal generation is enabled.
9 GTBEEN	Global Time Base Enable Configures the FTM to use an external global time base signal that is generated by another FTM. 0 Use of an external global time base is disabled. 1 Use of an external global time base is enabled.

Table continues on the next page...

FTMx_CONF field descriptions (continued)

Field	Description
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 BDMMODE	Debug Mode Selects the FTM behavior in Debug mode. See Debug mode.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LDFQ	Load Frequency Selects PWM reload frequency. LDFQ = 0: RF bit is set every reload oportunity. LDFQ = 1: RF bit is set every 2 reload oportunities. LDFQ = 2: RF bit is set every 3 reload oportunities. LDFQ = 3: RF bit is set every 4 reload oportunities. This pattern continues up to a maximum of 32.

41.4.23 FTM Fault Input Polarity (FTMx_FLTPOL)

This register defines the fault inputs polarity.

Address: Base address + 88h 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												FLT3POL	FLT2POL	FLT1POL	FLT0POL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FLTPOL field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FLT3POL	Fault Input 3 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_FLTPOL field descriptions (continued)

Field	Description
	0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
2 FLT2POL	Fault Input 2 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
1 FLT1POL	Fault Input 1 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
0 FLT0POL	Fault Input 0 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.

41.4.24 Synchronization Configuration (FTMx_SYNCONF)

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit, where j = 0, 1, 2, when the hardware trigger j is detected.

Address: Base address + 8Ch offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SWSOC	SWINVC	SWOM	SWWRBUF	SWRSTCNT	SYNCMODE	0	SWOC	INVC	0	CNTINC	0	HWTRIGMODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SYNCONF field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 HWSOC	Software output control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the SWOCTRL register synchronization. 1 A hardware trigger activates the SWOCTRL register synchronization.
19 HWINVC	Inverting control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the INVCTRL register synchronization. 1 A hardware trigger activates the INVCTRL register synchronization.
18 HWOM	Output mask synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the OUTMASK register synchronization. 1 A hardware trigger activates the OUTMASK register synchronization.
17 HWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization. 1 A hardware trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
16 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the FTM counter synchronization. 1 A hardware trigger activates the FTM counter synchronization.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 SWSOC	Software output control synchronization is activated by the software trigger. 0 The software trigger does not activate the SWOCTRL register synchronization. 1 The software trigger activates the SWOCTRL register synchronization.
11 SWINVC	Inverting control synchronization is activated by the software trigger. 0 The software trigger does not activate the INVCTRL register synchronization. 1 The software trigger activates the INVCTRL register synchronization.
10 SWOM	Output mask synchronization is activated by the software trigger. 0 The software trigger does not activate the OUTMASK register synchronization. 1 The software trigger activates the OUTMASK register synchronization.
9 SWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by the software trigger. 0 The software trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization. 1 The software trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
8 SWRSTCNT	FTM counter synchronization is activated by the software trigger. 0 The software trigger does not activate the FTM counter synchronization. 1 The software trigger activates the FTM counter synchronization.
7 SYNCMODE	Synchronization Mode Selects the PWM Synchronization mode.

Table continues on the next page...

FTMx_SYNCONF field descriptions (continued)

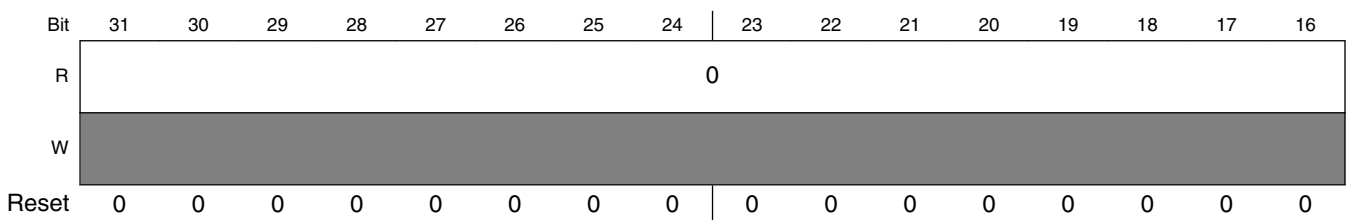
Field	Description
	0 Legacy PWM synchronization is selected. 1 Enhanced PWM synchronization is selected.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SWOC	SWOCTRL Register Synchronization 0 SWOCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1 SWOCTRL register is updated with its buffer value by the PWM synchronization.
4 INVC	INVCTRL Register Synchronization 0 INVCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1 INVCTRL register is updated with its buffer value by the PWM synchronization.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CNTINC	CNTIN Register Synchronization 0 CNTIN register is updated with its buffer value at all rising edges of FTM input clock. 1 CNTIN register is updated with its buffer value by the PWM synchronization.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HWTRIGMODE	Hardware Trigger Mode 0 FTM clears the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2. 1 FTM does not clear the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.

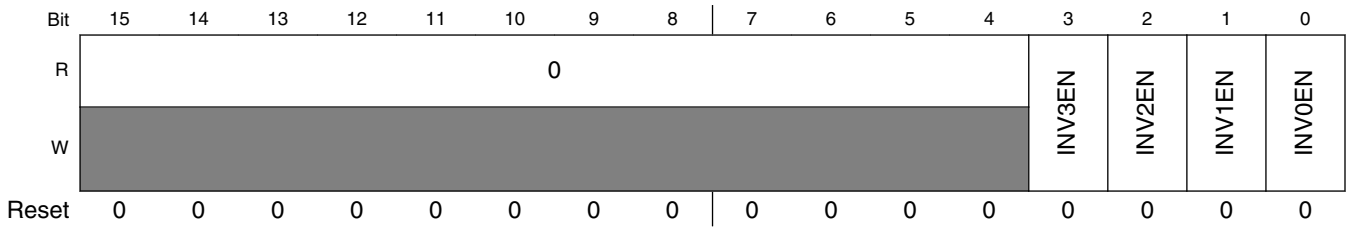
41.4.25 FTM Inverting Control (FTMx_INVCTRL)

This register controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

Address: Base address + 90h offset





FTMx_INVCTRL field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 INV3EN	Pair Channels 3 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
2 INV2EN	Pair Channels 2 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
1 INV1EN	Pair Channels 1 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
0 INV0EN	Pair Channels 0 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

41.4.26 FTM Software Output Control (FTMx_SWOCTRL)

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CH(n)OC bits enable the control of the corresponding channel (n) output by software.
- The CH(n)OCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Address: Base address + 94h 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	CH7OCV	CH6OCV	CH5OCV	CH4OCV	CH3OCV	CH2OCV	CH1OCV	CH0OCV	CH7OC	CH6OC	CH5OC	CH4OC	CH3OC	CH2OC	CH1OC	CH0OC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SWOCTRL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 CH7OCV	Channel 7 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
14 CH6OCV	Channel 6 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
13 CH5OCV	Channel 5 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
12 CH4OCV	Channel 4 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
11 CH3OCV	Channel 3 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
10 CH2OCV	Channel 2 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
9 CH1OCV	Channel 1 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
8 CH0OCV	Channel 0 Software Output Control Value

Table continues on the next page...

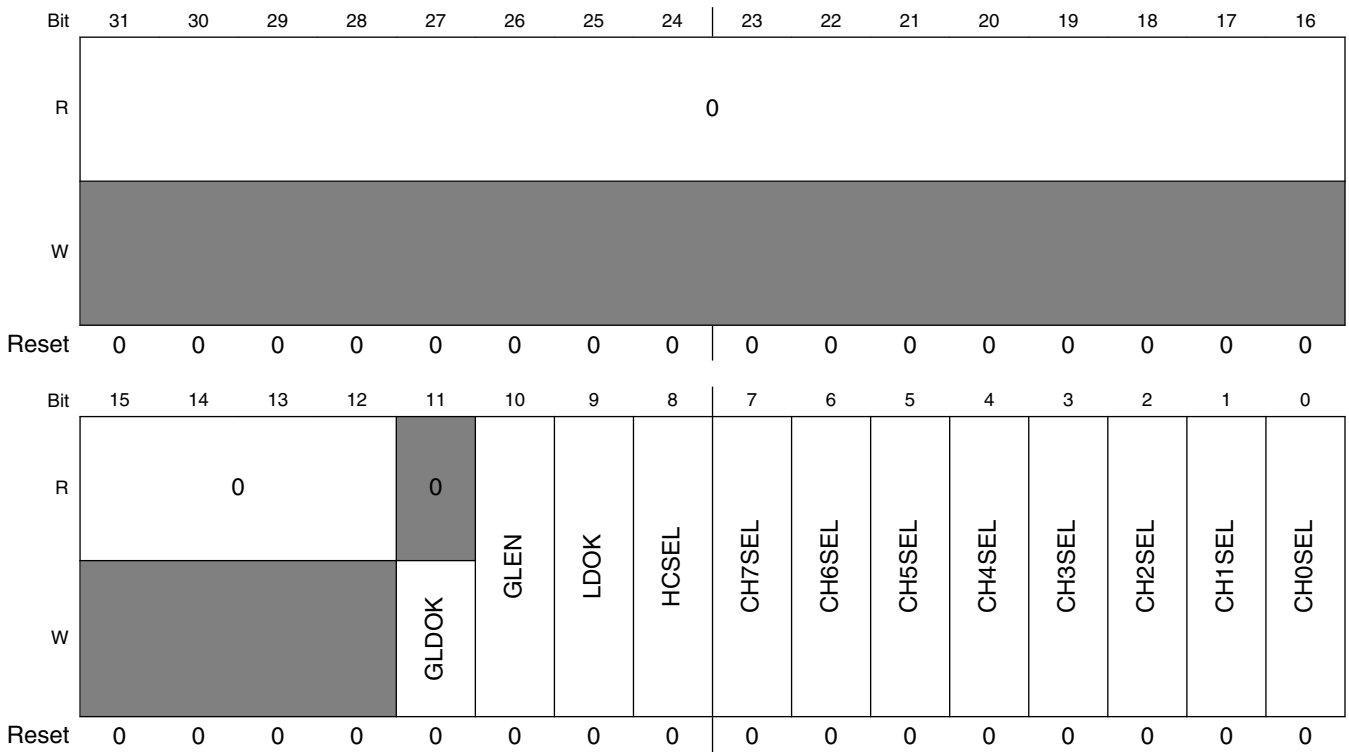
FTMx_SWOCTRL field descriptions (continued)

Field	Description
	0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
7 CH7OC	Channel 7 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
6 CH6OC	Channel 6 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
5 CH5OC	Channel 5 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
4 CH4OC	Channel 4 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
3 CH3OC	Channel 3 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
2 CH2OC	Channel 2 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
1 CH1OC	Channel 1 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
0 CH0OC	Channel 0 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.

41.4.27 FTM PWM Load (FTMx_PWMLOAD)

Enables the reload of the MOD, HCR, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for channel (j) when FTM counter = C(j)V. A reload can also occurs when FTM counter = HCR register at a half cycle match. This register also controls the local and global load mechanisms.

Address: Base address + 98h offset



FTMx_PWMLOAD field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 GLDOK	Global Load OK This bit controls the global load mechanism. It generates a pulse at FTM module global load output with one FTM clock cycle width, which is used to set LDOK bits of FTM and other modules (including other FTMs). This bit is self-cleared and read value is always zero. The global load mechanism depends on SoC specific information. Refer to FTM SoC specific information to more details. 0 No action. 1 LDOK bit is set.

Table continues on the next page...

FTMx_PWMLOAD field descriptions (continued)

Field	Description
10 GLEN	Global Load Enable This bit enables the global load mechanism implemented by GLDOK. If GLEN bit is set, then an external event on the FTM global load input sets the LDOK bit. The clear of the LDOK bit is done by CPU writes '0' to the bit. 0 Global Load Ok disabled. 1 Global Load OK enabled. A pulse event on the module global load input sets the LDOK bit.
9 LDOK	Load Enable Enables the loading of the MOD, CNTIN, HCR and CV registers with the values of their buffers. The LDOK bit can also be set by the Global Load mechanism if GLEN bit is enabled. 0 Loading updated values is disabled. 1 Loading updated values is enabled.
8 HCSEL	Half Cycle Select This bit enables the half cycle match as a reload oportunity. A half cycle is defined by when the FTM counter matches the HCR register. 0 Half cycle reload is disabled and it is not considered as a reload opportunity. 1 Half cycle reload is enabled and it is considered as a reload opportunity.
7 CH7SEL	Channel 7 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
6 CH6SEL	Channel 6 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
5 CH5SEL	Channel 5 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
4 CH4SEL	Channel 4 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
3 CH3SEL	Channel 3 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
2 CH2SEL	Channel 2 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.
1 CH1SEL	Channel 1 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.

Table continues on the next page...

FTMx_PWMLOAD field descriptions (continued)

Field	Description
0 CH0SEL	Channel 0 Select 0 Channel match is not included as a reload opportunity. 1 Channel match is included as a reload opportunity.

41.4.28 Half Cycle Register (FTMx_HCR)

The Half Cycle Register contains the match value for FTM half cycle reload feature. After FTM counter reaches this value, a reload opportunity is generated if FTM_PWMLOAD[HCSEL] is enabled.

Writing to the HCR register latches the value into a buffer. The HCR register is updated with the value of its write buffer according to Registers updated from write buffers.

Address: Base address + 9Ch offset

Bit	31	30	29	28	27	26	25	24	23	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																HCVAL															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_HCR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
HCVAL	Half Cycle Value

41.4.29 Mirror of Modulo Value (FTMx_MOD_MIRROR)

This register contains the integer and fractional modulo value for the FTM counter.

Address: Base address + 200h offset

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

FTMx_MOD_MIRROR field descriptions

Field	Description
31–16 MOD	Mirror of the Modulo Integer Value See the field MOD of the register MOD.
15–11 FRACMOD	Modulo Fractional Value The modulo fractional value is used in the PWM period dithering. This value is added to an internal accumulator at the end of each PWM period. Writes to the field FRACMOD are done on its write buffer. The FRACMOD is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

41.4.30 Mirror of Channel (n) Match Value (FTMx_CnV_MIRROR)

This register contains the integer and fractional value of the channel (n) match.

Address: Base address + 204h 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	VAL																FRACVAL					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

FTMx_CnV_MIRROR field descriptions

Field	Description
31–16 VAL	Mirror of the Channel (n) Match Integer Value See the field VAL of the register CnV.
15–11 FRACVAL	Channel (n) Match Fractional Value The channel (n) match fractional value is used in the PWM edge dithering. This value is added to the channel (n) internal accumulator at the end of each PWM period. Writes to the field FRACVAL are done on its write buffer. The FRACVAL is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

41.5 Functional description

The notation used in this document to represent the counters and the generation of the signals is shown in the following figure.

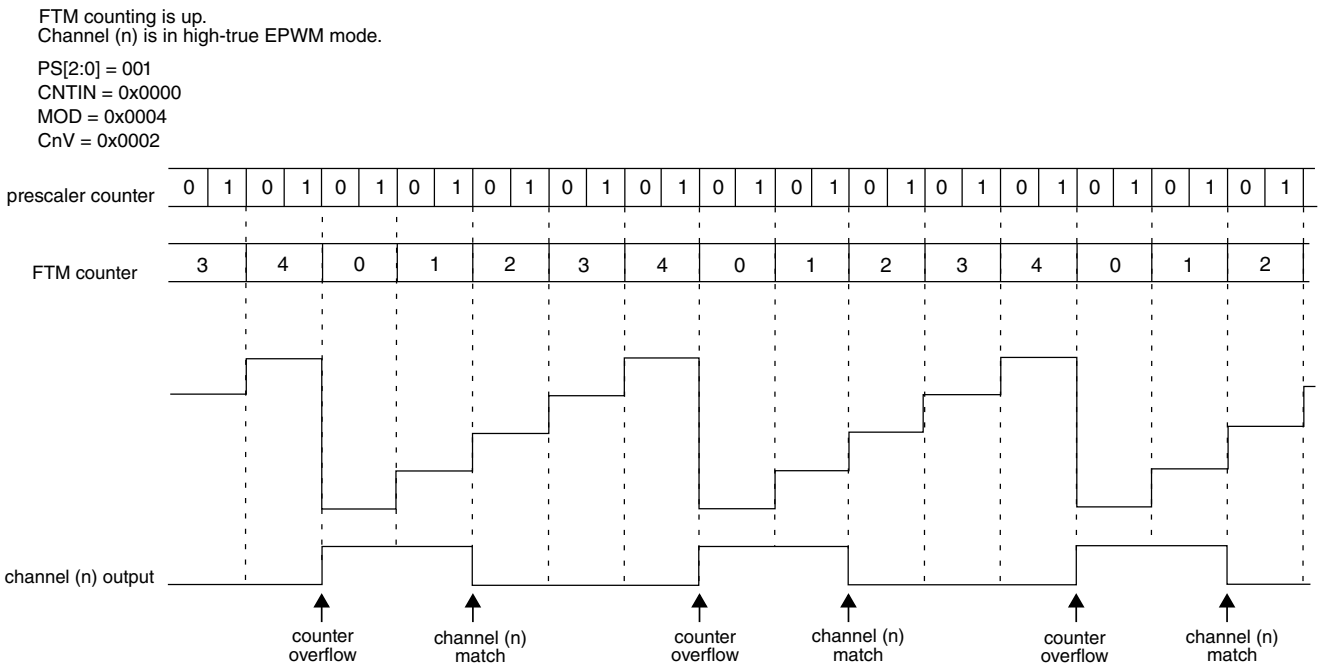


Figure 41-2. Notation used

41.5.1 Clock source

The FTM has only one clock domain: the FTM input clock.

41.5.1.1 Counter clock source

The CLKS[1:0] bits in the SC register select one of three possible clock sources for the FTM counter or disable the FTM counter. After any chip reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the FTM input clock or an external clock. This clock input is defined by chip integration. Refer to the chip specific documentation for further information. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM input clock frequency.

The external clock passes through a synchronizer clocked by the FTM input clock to assure that counter transitions are properly aligned to FTM input clock transitions.Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the FTM input clock frequency.

41.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and FTM counter.

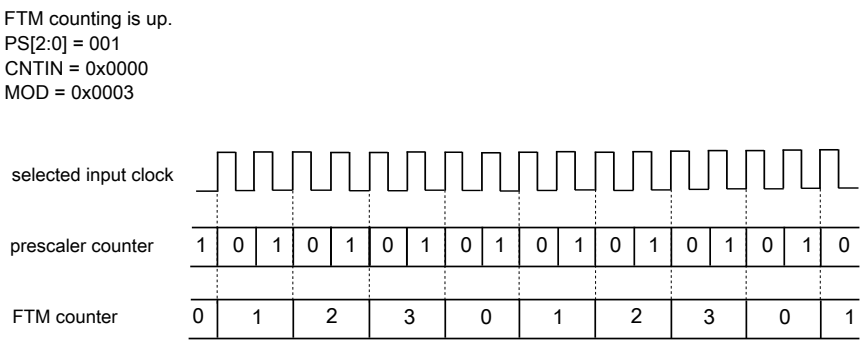


Figure 41-3. Example of the prescaler counter

41.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- Up counting
- Up-down counting
- Quadrature Decoder mode

41.5.3.1 Up counting

Up counting is selected when:

- QUADEN = 0, and
- CPWMS = 0

CNTIN defines the starting value of the count and MOD defines the final value of the count, see the following figure. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is (MOD – CNTIN + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN. See Counter events for more details.

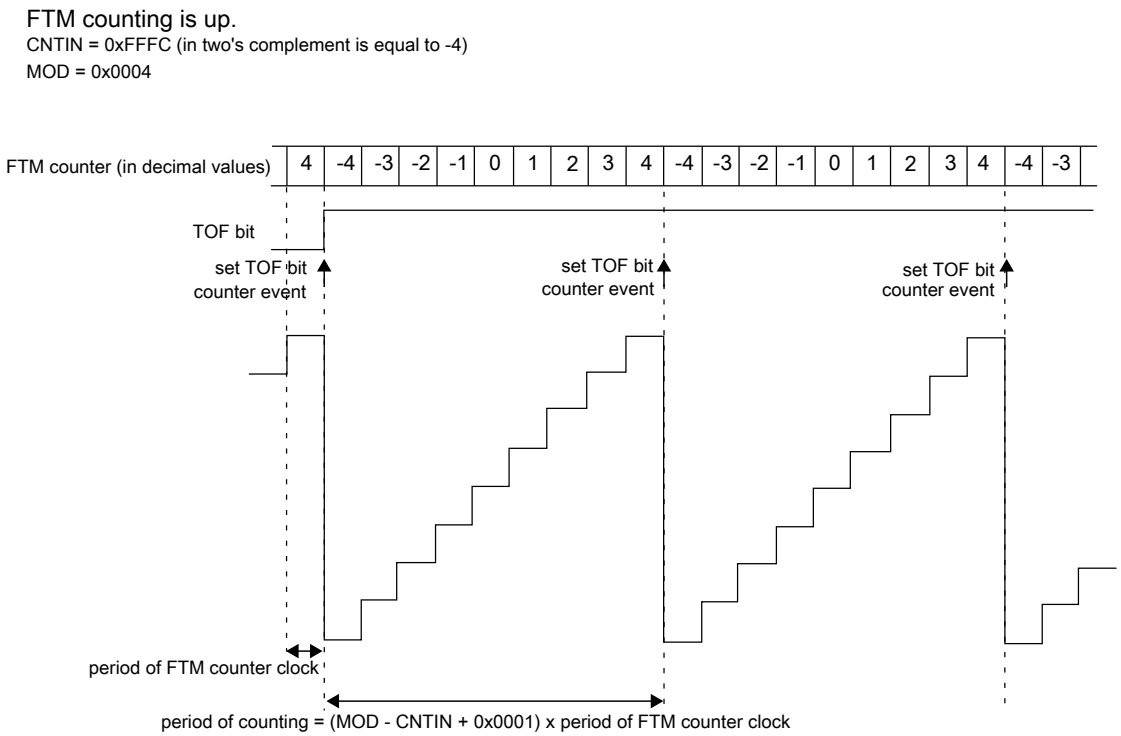


Figure 41-4. Example of FTM up and signed counting

Table 41-3. FTM counting based on CNTIN value

When	Then
CNTIN = 0x0000	The FTM counting is equivalent to TPM up counting, that is, up and unsigned counting. See the following figure.
CNTIN[15] = 1	The initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed.
CNTIN[15] = 0 and CNTIN ≠ 0x0000	The initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

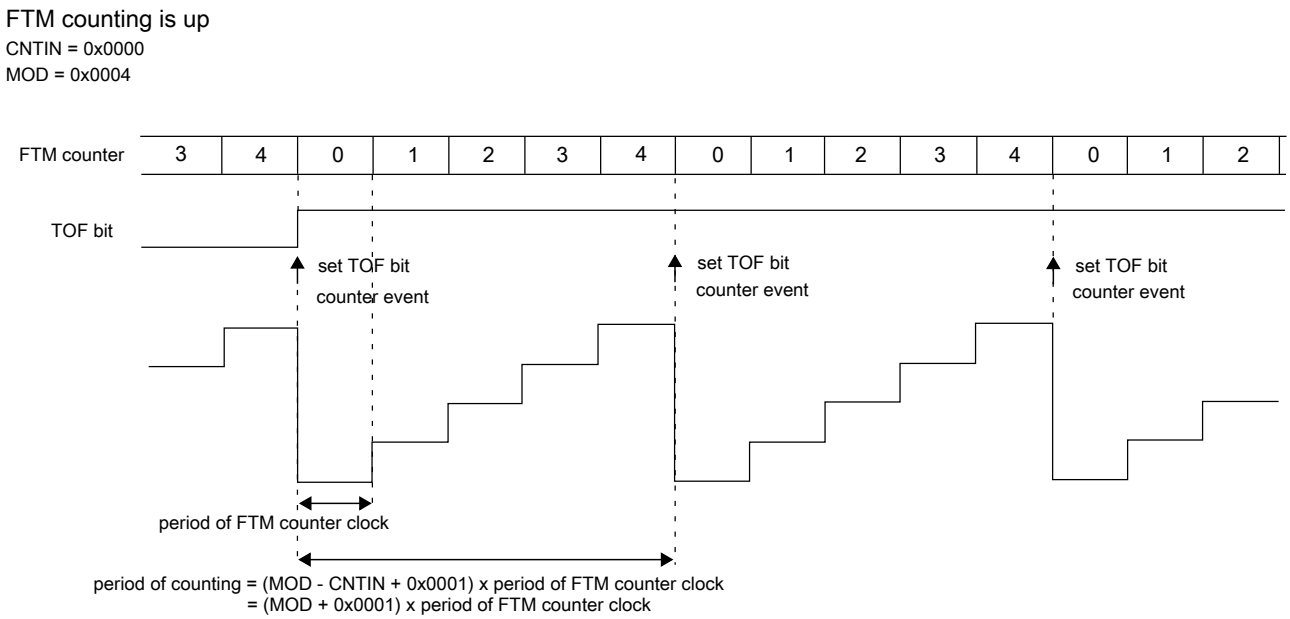


Figure 41-5. Example of FTM up counting with CNTIN = 0x0000

Note

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register, either in the unsigned counting or signed counting. It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000, for example after reset, and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

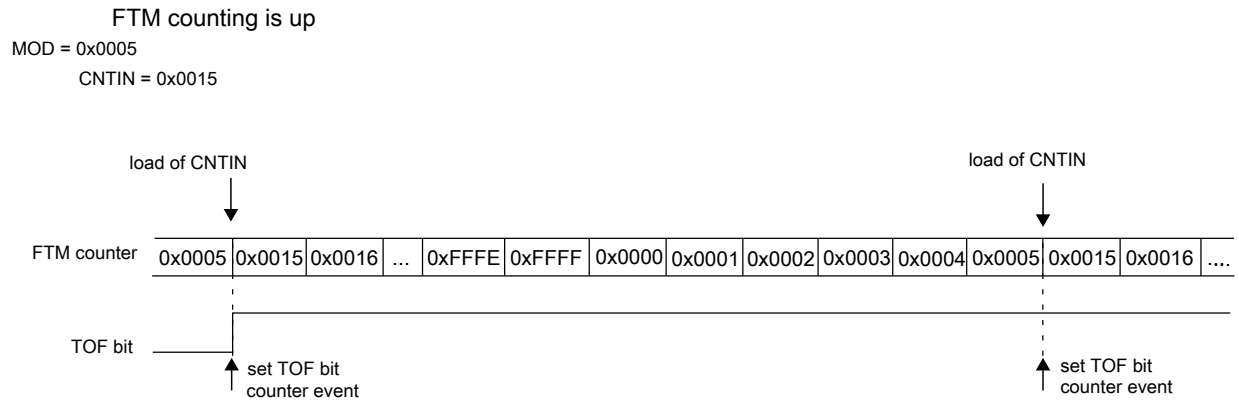


Figure 41-6. Example of up counting when the value of CNTIN is greater than the value of MOD

41.5.3.2 Up-down counting

Up-down counting is selected when:

- QUADEN = 0, and
- CPWMS = 1

CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MOD – CNTIN) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting, that is, up-down and unsigned counting. See the following figure.

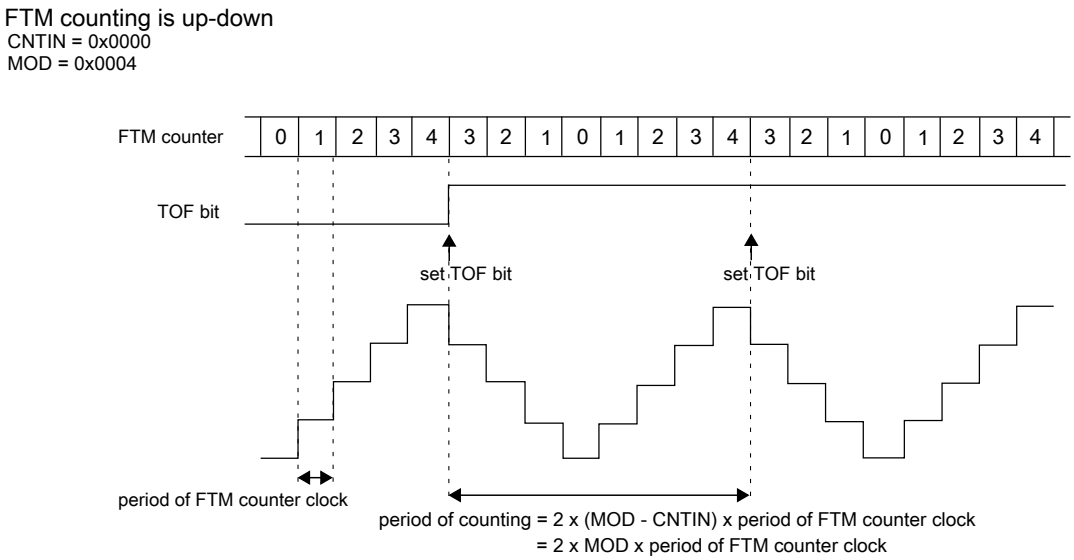


Figure 41-7. Example of up-down counting when CNTIN = 0x0000

Note

When CNTIN is different from zero in the up-down counting, a valid CPWM signal is generated:

- if CnV > CNTIN, or
- if CnV = 0 or if CnV[15] = 1. In this case, 0% CPWM is generated.

The figure below shows the possible counter events when in up-down counting mode. See Counter events for more details.

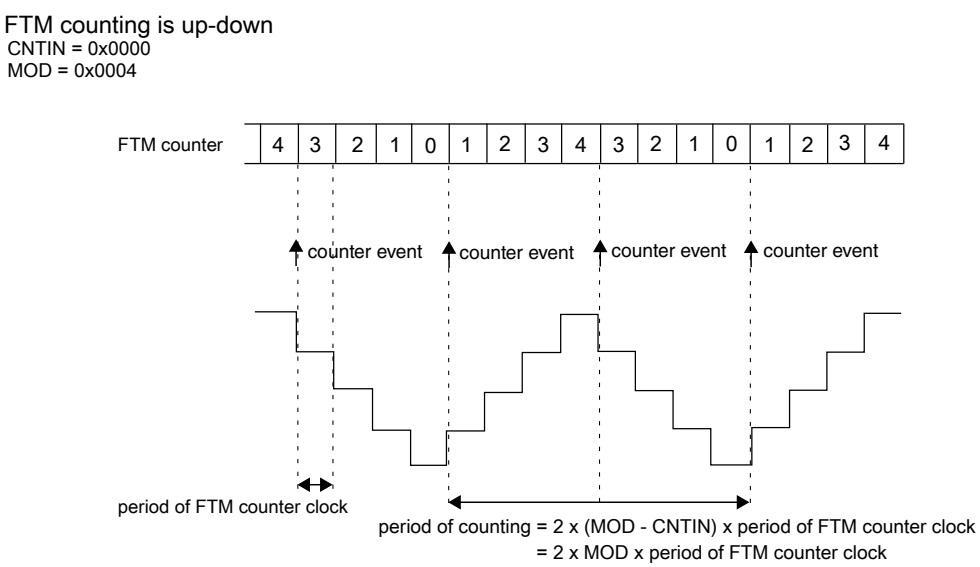


Figure 41-8. Example of counter events in up-down counting mode when CNTIN = 0x0000

41.5.3.3 Free running counter

If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000. See the following figure.

A counter event occurs at the same time of TOF bit set when the FTM counter changes from 0xFFFF to 0x0000. See Counter events for more details.

Figure 41-9. Example when the FTM counter is free running

The FTM counter is also a free running counter when:

- FTMEN = 1
- QUADEN = 0
- CPWMS = 0
- CNTIN = 0x0000, and
- MOD = 0xFFFF

41.5.3.4 Counter reset

Any one of the following cases resets the FTM counter to the value in the CNTIN register and the channels output to its initial value, except for channels in Output Compare mode.

- Any write to CNT.
- FTM counter synchronization.
- A channel in Input Capture mode with ICRST = 1 (FTM Counter Reset in Input Capture Mode).

Note that reseting the counter also generates a counter event. See Counter events for more details.

41.5.3.5 Counter events

Counter events can be used as reload opportunities to FTM register sychronization mechanism. See Reload Points for more details. There are some possible counter events depending on the counter mode. Please see the table below for more details.

Table 41-4. FTM counter events

When	Then
FTM counter is in up counting mode or freerunning	• A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN (counter wrap). Figure at Up counting shows the counter event generation. • When in freerunning, there is a counter event when FTM counter changes from 0xFFFF to 0x0000. Figure at Free running counter shows the counter event generation.
FTM counter is in up-down counting mode	• In up-down counting mode, there are two possible counter events when FTM counter turns from down to up counting and when counter turns from up to down counting. User can select which point will be used to generate the counter event. Figure at Up-down counting shows the possible counter events.
FTM counter is reseted (see Counter reset) or a value different from zero is written at CLKS field	• In up-counting mode, all counter reset events or a write in the CLKS with a value different from zero generates a counter event. • In up-down counting mode, counter reset events only generates a counter event if the minimum load point when FTM counter turns from down to up counting is configured. A write in the CLKS with a value different from zero always generates a counter event in up-down counting mode.

41.5.4 Channel Modes

The following table shows the channel modes selection.

Table 41-5. Channel Modes Selection

DECAPEN	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
X	X	X	XX	00	Pin not used for FTM—revert the channel pin to general purpose I/O or other peripheral control	
0	0	0	00	01	Input Capture	Capture on Rising Edge Only
				10		Capture on Falling Edge Only

Table continues on the next page...

Table 41-5. Channel Modes Selection (continued)

DECAPEN	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
				11		Capture on Rising or Falling Edge
			01	01	Output Compare	Toggle Output on match
				10		Clear Output on match
				11		Set Output on match
			1X	10	Edge-Aligned PWM	High-true pulses (clear Output on match)
				X1		Low-true pulses (set Output on match)
		1	XX	10	Center-Aligned PWM	High-true pulses (clear Output on match-up)
				X1		Low-true pulses (set Output on match-up)
	1	0	XX	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
				X1		Low-true pulses (clear on channel (n) match, and set on channel (n +1) match)
1	0	0	X0	See Table 41-6.	Dual Edge Capture	One-Shot Capture mode
			X1			Continuous Capture mode

Table 41-6. Dual Edge Capture Mode — Edge Polarity Selection

ELSB	ELSA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

41.5.5 Input Capture mode

The Input Capture mode is selected when:

- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0
- MSB:MSA = 0:0, and
- ELSB:ELSA ≠ 0:0

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnV register, at the same time the CHF bit is set and the channel interrupt is generated if enabled by CHIE = 1. See the following figure.

When a channel is configured for input capture, the FTMxCHn pin is an edge-sensitive input. ELSB:ELSA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is FTM input clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register is ignored in Input Capture mode.

While in Debug mode, the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of Debug, is captured into the CnV register and the CHF bit is set.

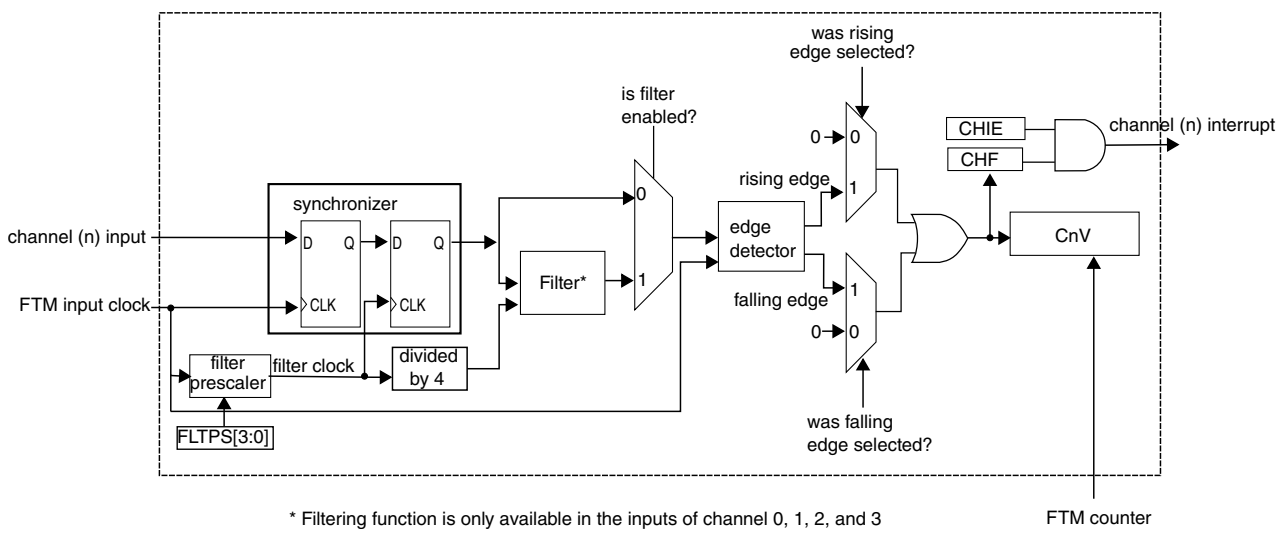


Figure 41-10. Input Capture mode

If the channel input does not have a filter enabled, then the input signal is always delayed 3 rising edges of the FTM input clock, that is, two rising edges to the synchronizer plus one more rising edge to the edge detector. In other words, the CHF bit is set on the third rising edge of the FTM input clock after a valid edge occurs on the channel input.

41.5.5.1 Filter for Input Capture mode

The filter function is only available on channels 0, 1, 2, and 3.

First, the input signal is synchronized by the FTM input clock. Following synchronization, the input signal enters the filter block. See the following figure.

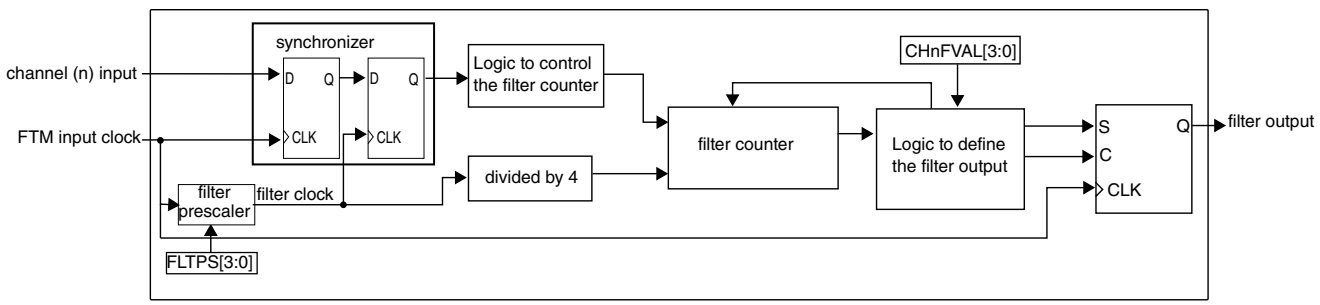


Figure 41-11. Channel input filter

NOTE

The Channel Input Filter internal counter clock is further divided by 4 in order to reject high frequency glitches.

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to CHnFVAL[3:0], the state change of the input signal is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by (CHnFVAL[3:0] × 4 filter clock cycles) is regarded as a glitch and is not passed on to the edge detector. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed 3 rising edges (1 FTM input clock + 2 filter clocks). If (CHnFVAL[3:0] ≠ 0000), then the delay through the filter logic is given by the expression: 2 FTM input

clocks + (2 + 4 x CHnFVAL) filter clocks. In other words, CHF is set (2 FTM input clocks + 2 filter clocks + 4 filter clocks × CHnFVAL[3:0]) clock periods after a valid edge occurs on the channel input.

The clock for the counter in the channel input filter is the filter clock divided by 4.

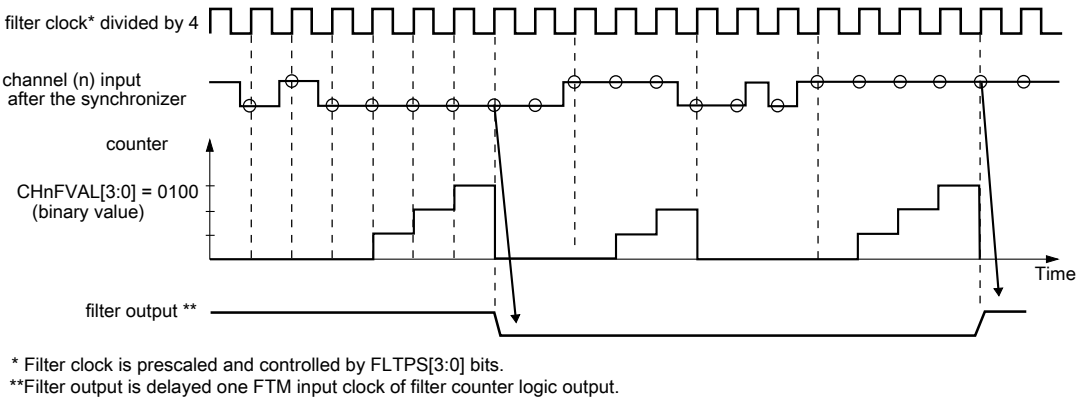


Figure 41-12. Channel input filter example

The figure below shows the delay through the input filter logic considering each internal filter element. In this example the filter prescaler is set to zero. Note that the input signal is delayed only by the synchronizer and edge dector logic if the filter is disabled.

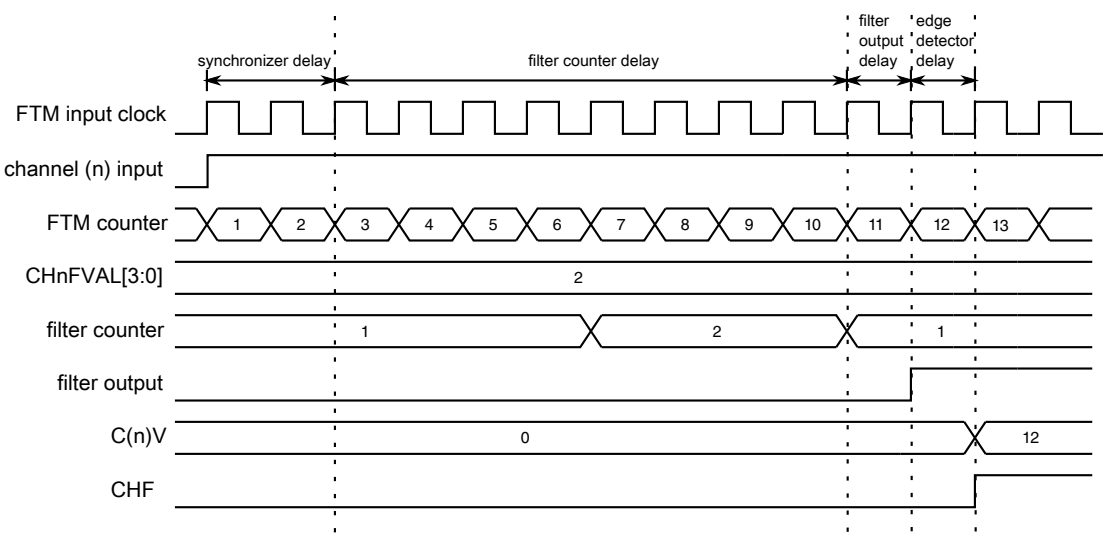


Figure 41-13. Input capture example

41.5.5.2 FTM Counter Reset in Input Capture Mode

If the channel (n) is in input capture mode and CnSC[ICRST = 1], then when the selected input capture event occurs in the channel (n) input signal, the current value of the FTM counter is captured into the CnV register, the CHF bit is set, the channel (n) interrupt is generated (if CHIE = 1) and the FTM counter is reset to the CNTIN register value.

This allows the FTM to measure a period/pulse being applied to the channel (n) input (number of the FTM input clocks) without having to implement a subtraction calculation in software subsequent to the event occurring.

The figure below shows the FTM counter reset when the selected input capture event is detected in a channel in input capture mode with ICRST = 1.

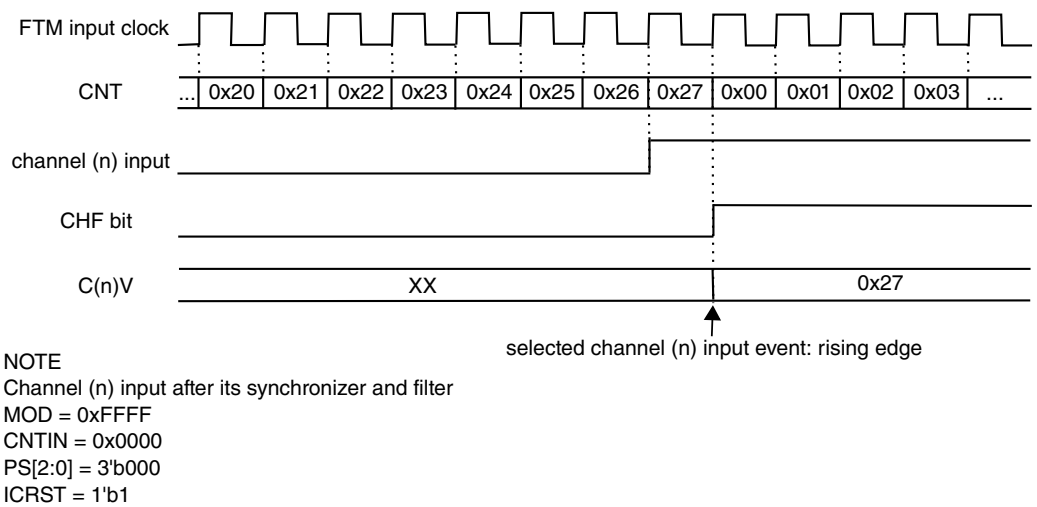


Figure 41-14. Example of the Input Capture mode with ICRST = 1

NOTE

- It is expected that the ICRST bit be set only when the channel is in input capture mode.
- If the FTM counter is reset because the channel is in input capture mode with ICRST = 1, then the prescaler counter (Prescaler) is also reset.

41.5.6 Output Compare mode

The Output Compare mode is selected when:

- DECAPEN = 0
- COMBINE = 0

- CPWMS = 0, and
- MSB:MSA = 0:1

In Output Compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM counter = CnV).

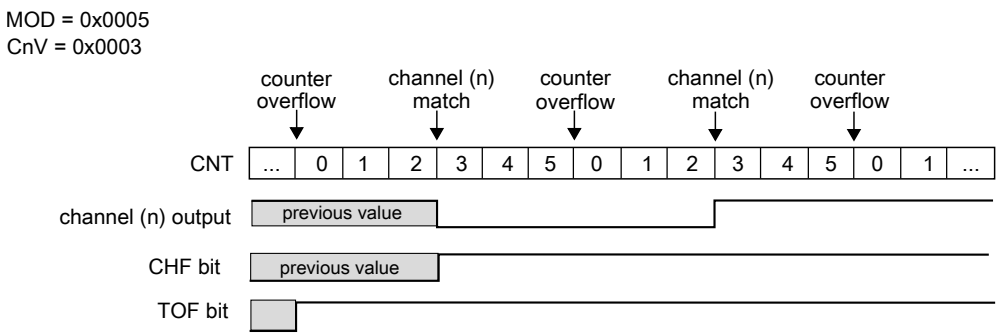


Figure 41-15. Example of the Output Compare mode when the match toggles the channel output

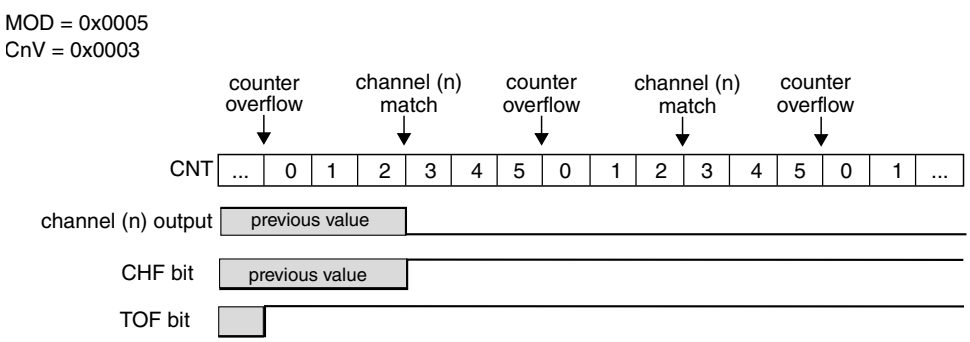


Figure 41-16. Example of the Output Compare mode when the match clears the channel output

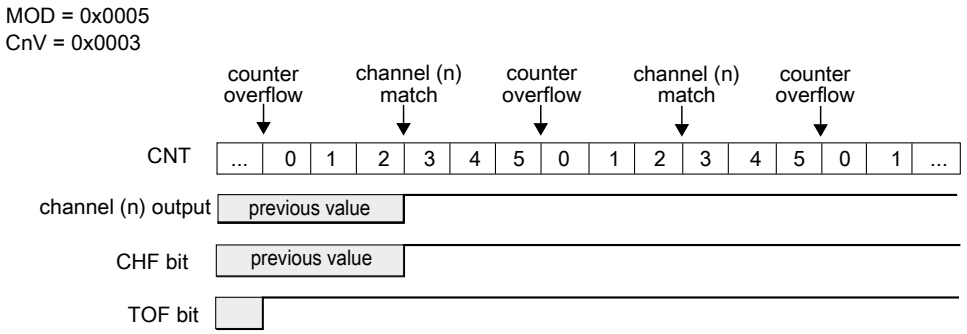


Figure 41-17. Example of the Output Compare mode when the match sets the channel output

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not modified and controlled by FTM.

41.5.7 Edge-Aligned PWM (EPWM) mode

The Edge-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0, and
- MSB = 1

The EPWM period is determined by (MOD − CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV − CNTIN).

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.

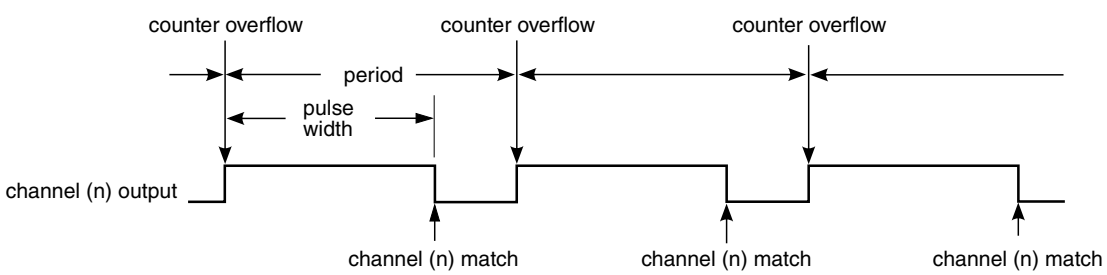


Figure 41-18. EPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced low at the channel (n) match (FTM counter = CnV). See the following figure.

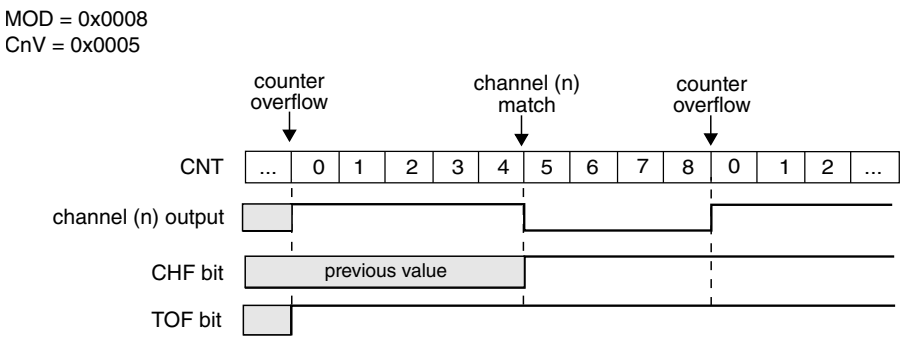


Figure 41-19. EPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced high at the channel (n) match (FTM counter = CnV). See the following figure.

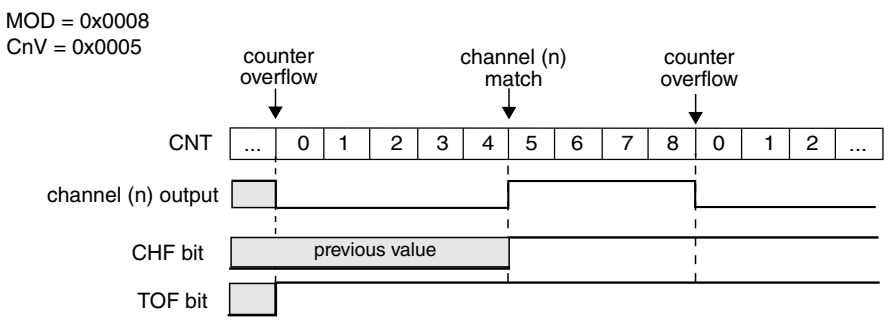


Figure 41-20. EPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHF bit is not set even when there is the channel (n) match.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHF bit is not set. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

When CNTIN is different from zero the following EPWM signals can be generated:

- 0% EPWM signal if CnV = CNTIN,

- EPWM signal between 0% and 100% if CNTIN < CnV <= MOD,
- 100% EPWM signal when CNTIN > CnV or CnV > MOD.

41.5.8 Center-Aligned PWM (CPWM) mode

The Center-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 0, and
- CPWMS = 1

The CPWM pulse width (duty cycle) is determined by 2 × (CnV − CNTIN) and the period is determined by 2 × (MOD − CNTIN). See the following figure. MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

The CHF bit is set and channel (n) interrupt is generated (if CHIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of CNTIN.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

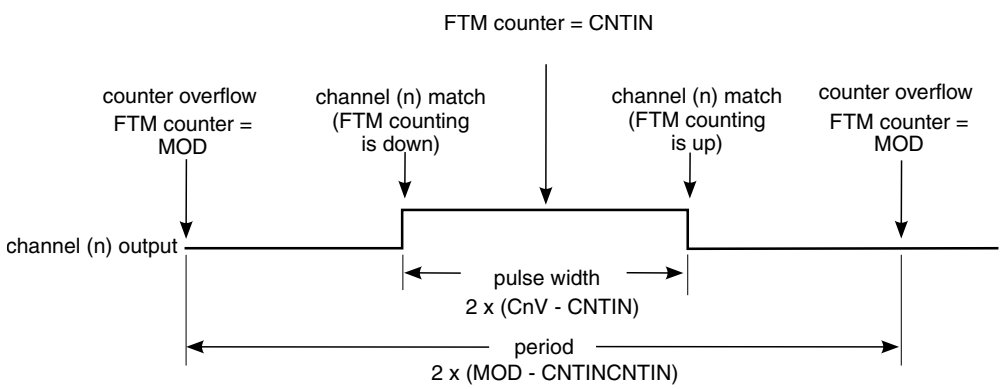


Figure 41-21. CPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the FTM counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated (if CHIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. See the following figure.

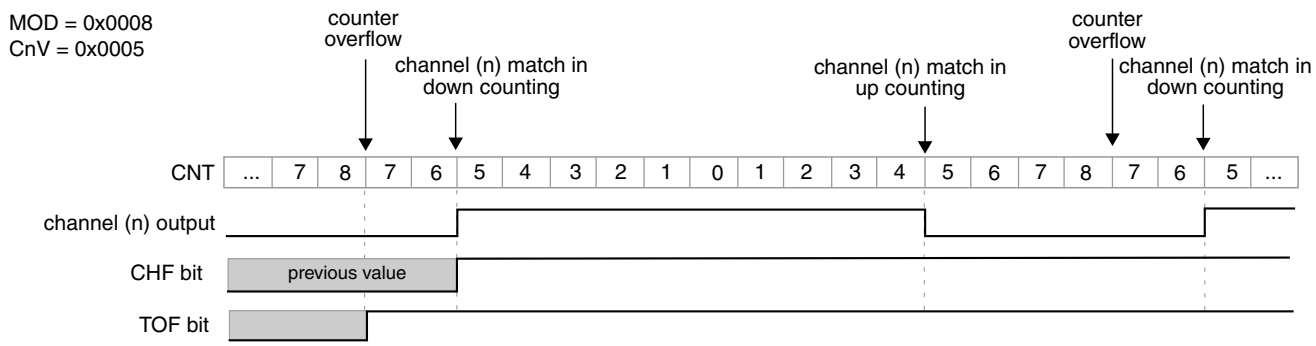


Figure 41-22. CPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. See the following figure.

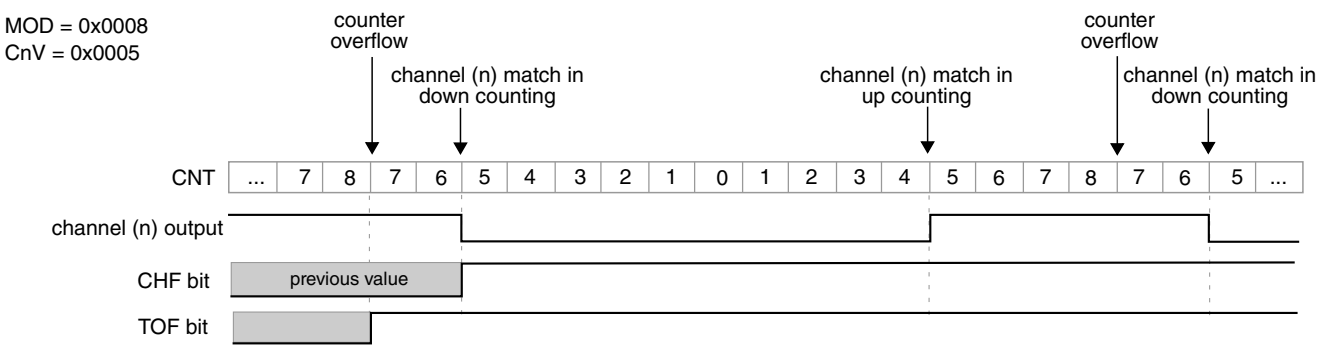


Figure 41-23. CPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000) or CnV is a negative value, that is (CnV[15] = 1), then the channel (n) output is a 0% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match.

If CnV is a positive value, that is (CnV[15] = 0), (CnV ≥ MOD), and (MOD ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE, 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

41.5.9 Combine mode

The Combine mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 1, and
- CPWMS = 0

In Combine mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by (MOD − CNTIN + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The CHF bit is set and the channel (n) interrupt is generated (if CHIE = 1) at the channel (n) match (FTM counter = C(n)V). The channel (n+1) CHF bit is set and the channel (n +1) interrupt is generated, if channel (n+1) CHIE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If (ELSB:ELSA = 1:0), then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced high at the channel (n) match (FTM counter = C(n)V). See the following figure.

If (ELSB:ELSA = X:1), then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced low at the channel (n) match (FTM counter = C(n)V). See the following figure.

In Combine mode, the channel (n) ELSB and channel (n) ELSA bits are not used in the generation of the channels (n) and (n+1) output. However, if (ELSB:ELSA = 0:0) then the channel (n) output is not controlled by FTM, and if (channel (n+1) ELSB:ELSA = 0:0) then the channel (n+1) output is not controlled by FTM.

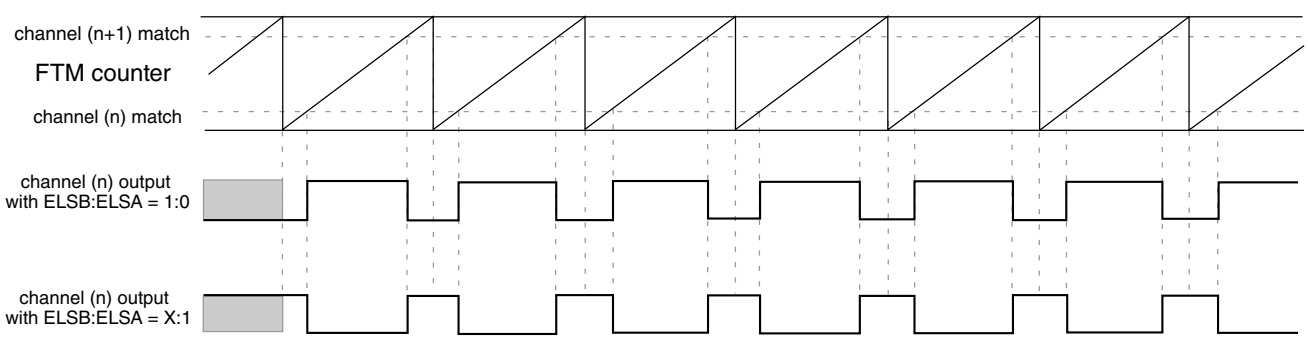


Figure 41-24. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

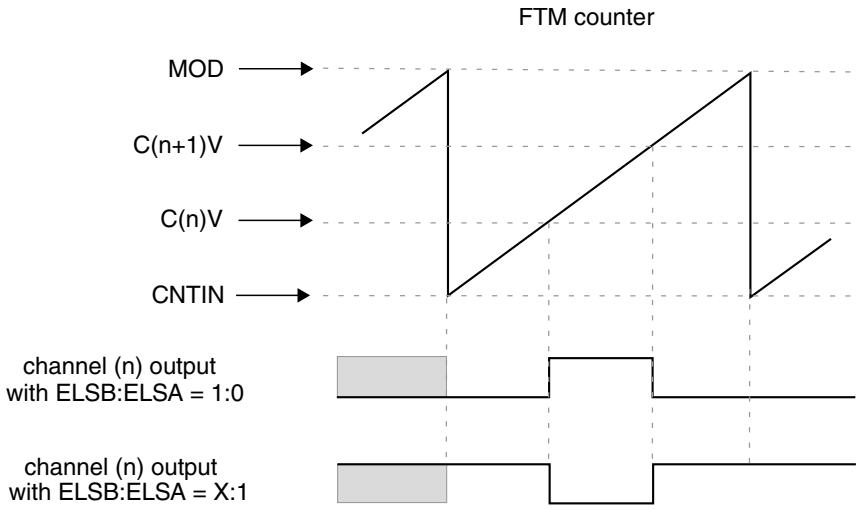


Figure 41-25. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V < C(n+1)V)

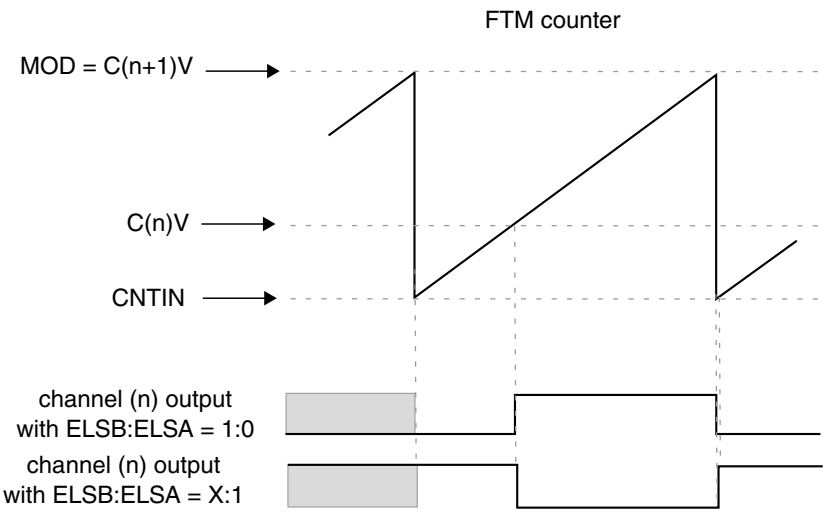


Figure 41-26. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n+1)V = MOD)

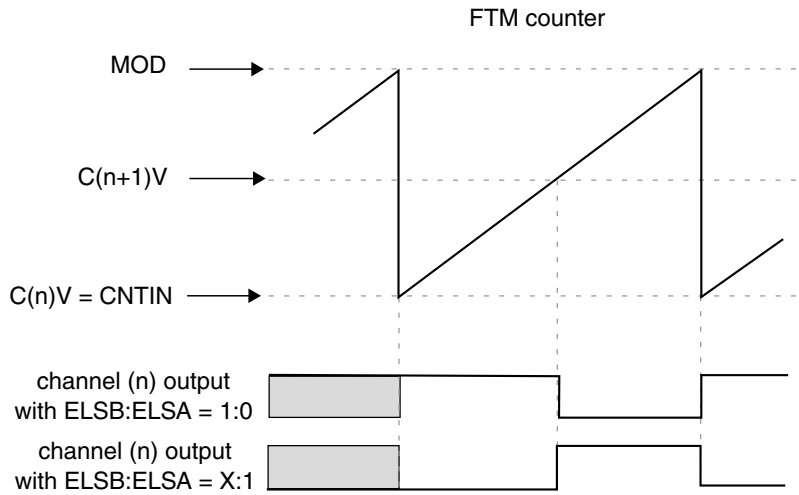


Figure 41-27. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD)

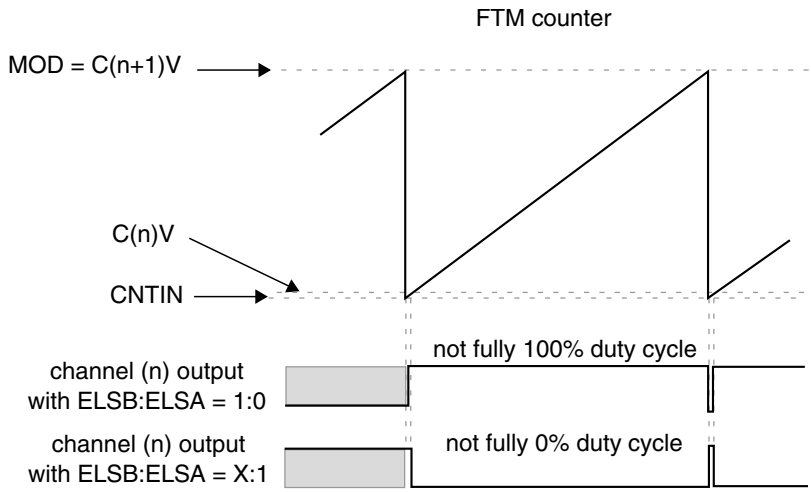


Figure 41-28. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n)V is Almost Equal to CNTIN) and (C(n+1)V = MOD)

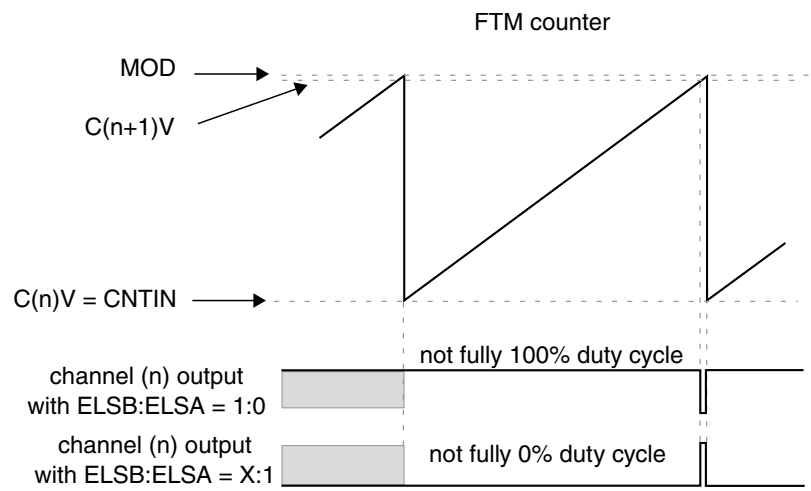


Figure 41-29. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD) and (C(n+1)V is Almost Equal to MOD)

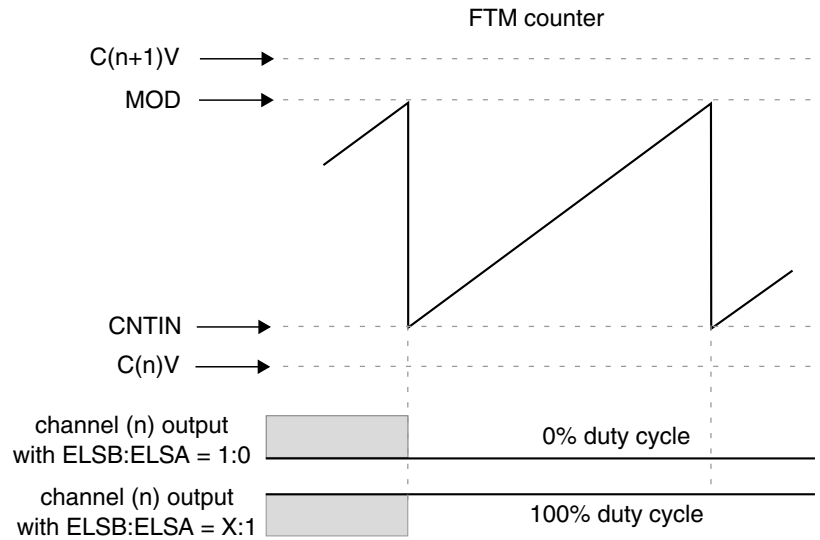


Figure 41-30. Channel (n) output if C(n)V and C(n+1)V are not between CNTIN and MOD

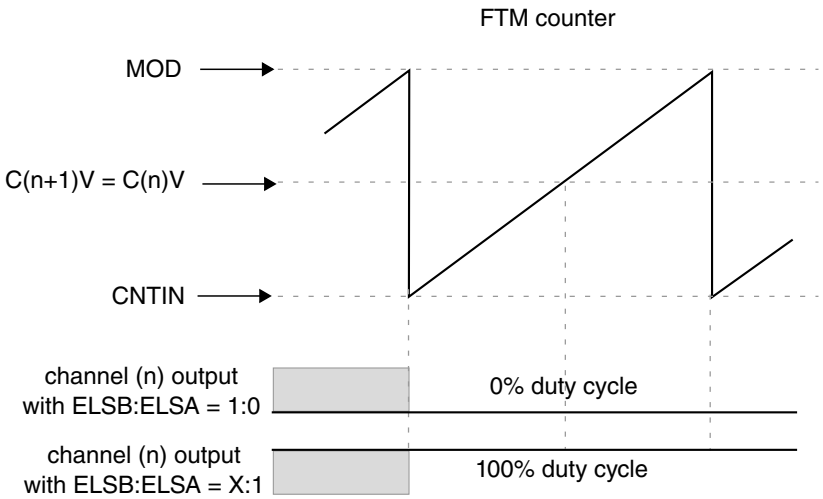


Figure 41-31. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V = C(n+1)V)

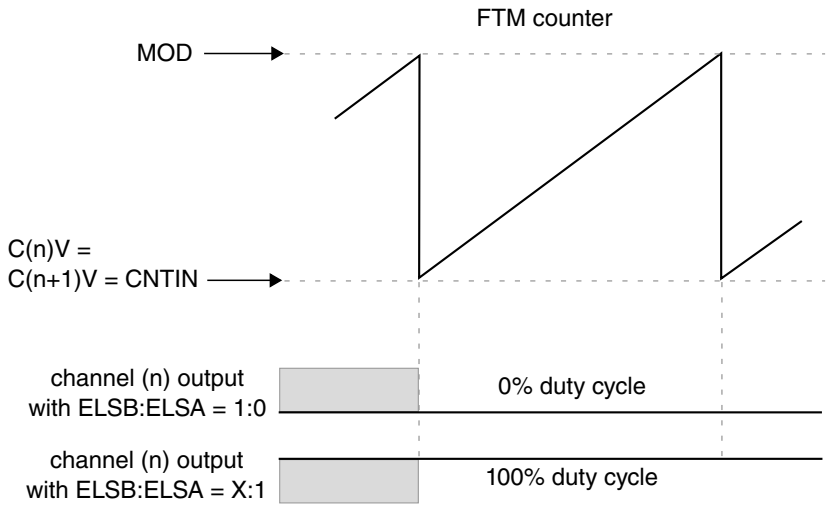


Figure 41-32. Channel (n) output if (C(n)V = C(n+1)V = CNTIN)

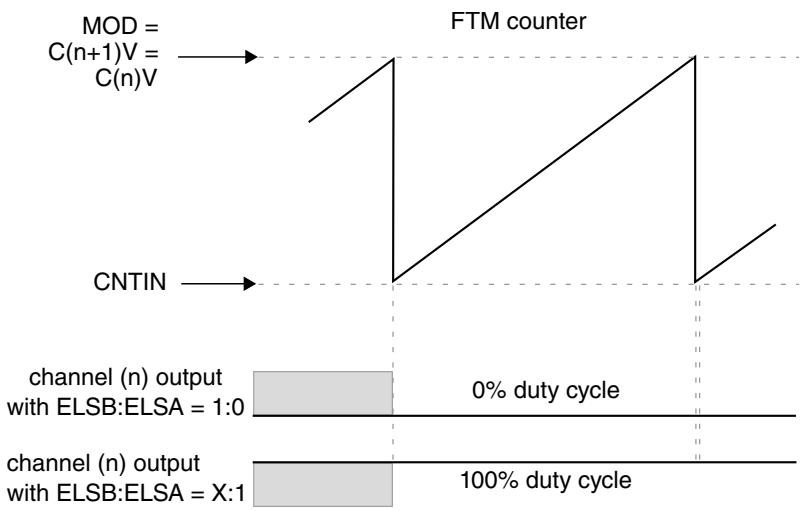


Figure 41-33. Channel (n) output if (C(n)V = C(n+1)V = MOD)

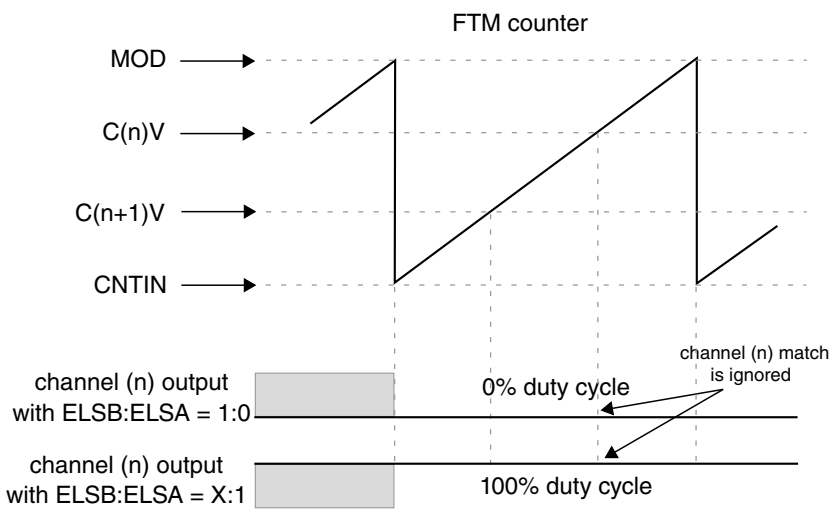


Figure 41-34. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V > C(n+1)V)

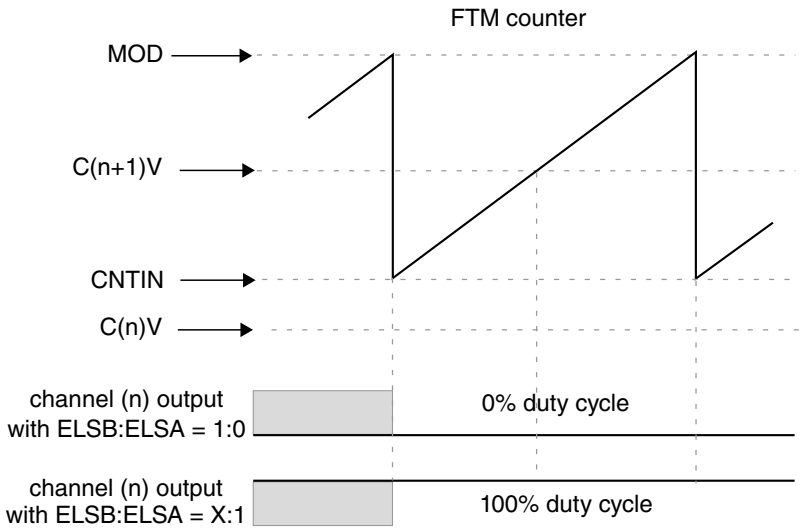


Figure 41-35. Channel (n) output if (C(n)V < CNTIN) and (CNTIN < C(n+1)V < MOD)

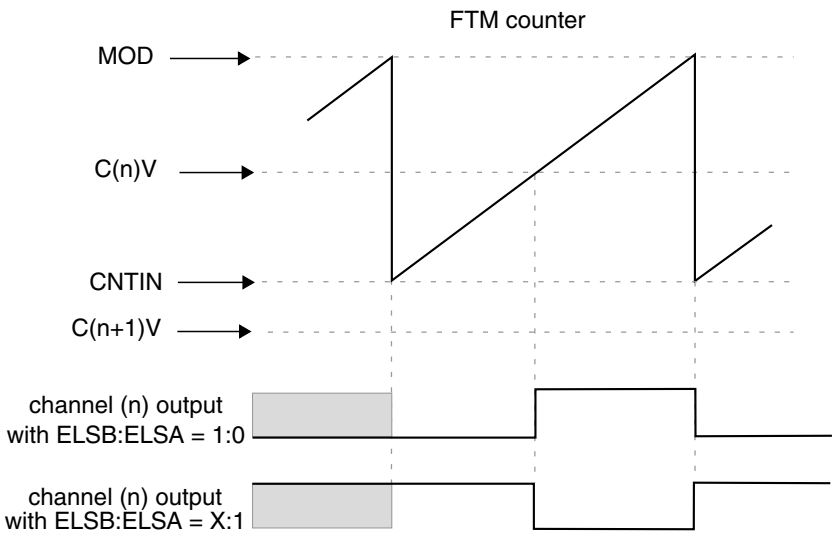


Figure 41-36. Channel (n) output if (C(n+1)V < CNTIN) and (CNTIN < C(n)V < MOD)

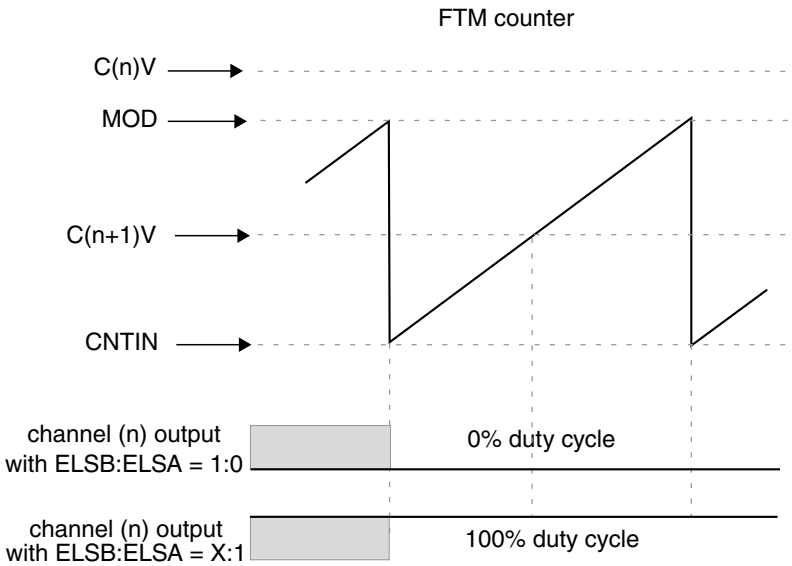


Figure 41-37. Channel (n) output if (C(n)V > MOD) and (CNTIN < C(n+1)V < MOD)

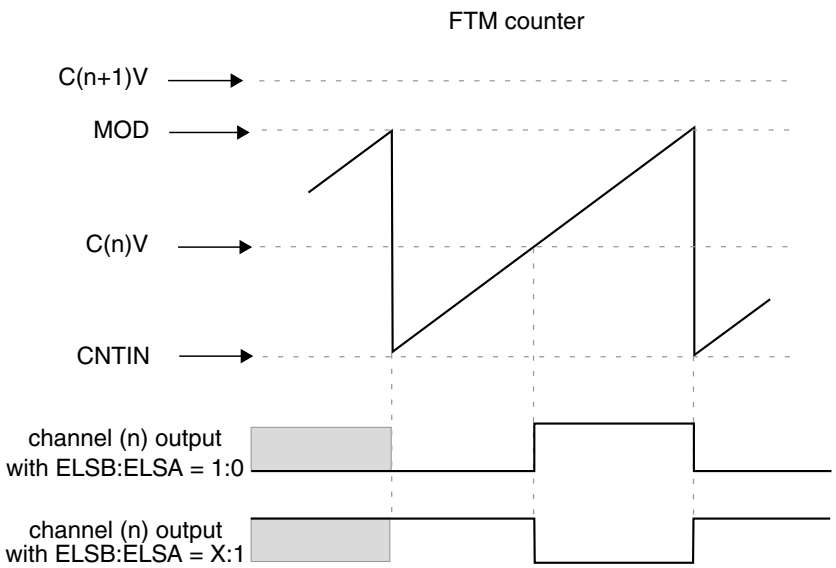


Figure 41-38. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V < MOD)

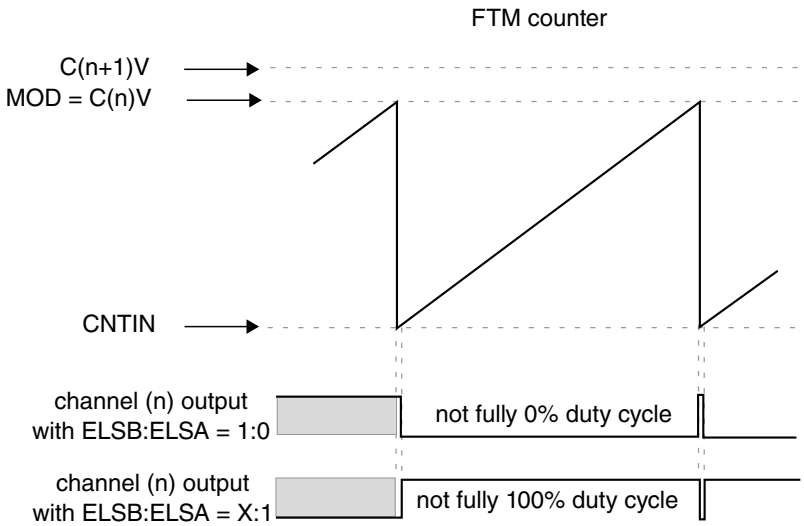


Figure 41-39. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V = MOD)

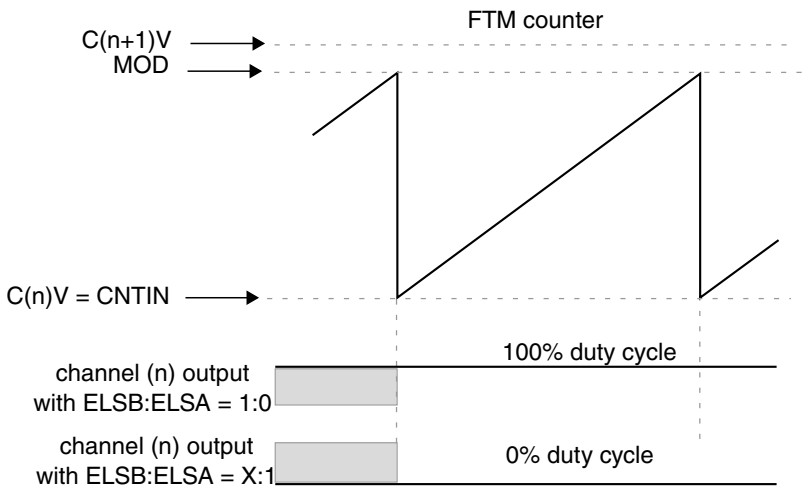


Figure 41-40. Channel (n) output if (C(n)V = CNTIN) and (C(n+1)V > MOD)

41.5.9.1 Asymmetrical PWM

In Combine mode, the PWM first edge (channel (n) match: FTM counter = C(n)V) is independent of the PWM second edge (channel (n+1) match: FTM counter = C(n+1)V).

41.5.10 Complementary Mode

The Complementary mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1

In Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

So, the channel (n+1) output is the same as the channel (n) output when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 0

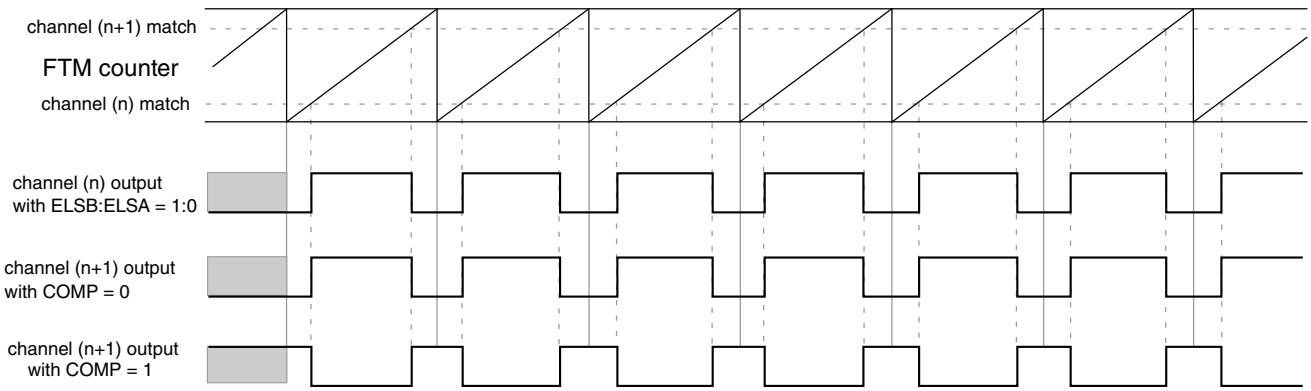


Figure 41-41. Channel (n+1) output in Complementary mode with (ELSB:ELSA = 1:0)

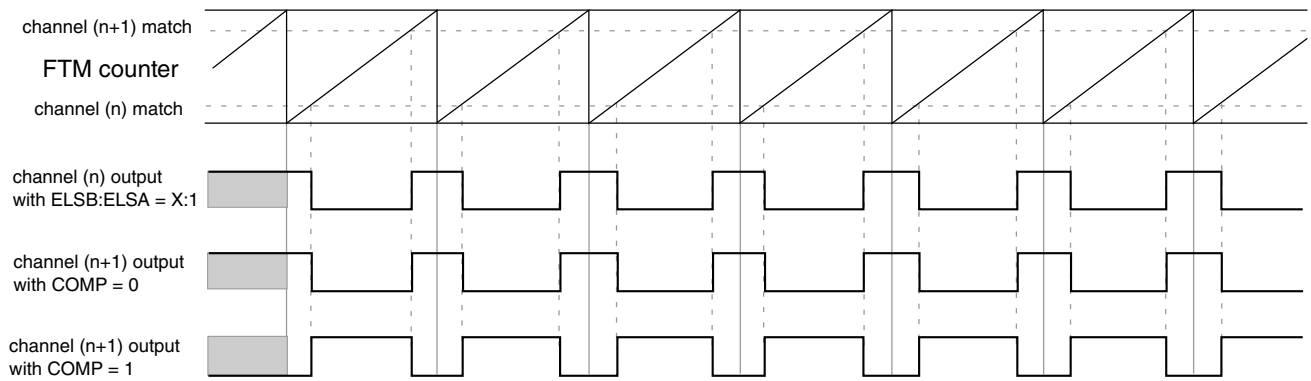


Figure 41-42. Channel (n+1) output in Complementary mode with (ELSB:ELSA = X:1)

NOTE

The Complementary Mode is not available on Output Compare mode.

41.5.11 Registers updated from write buffers

FTM has many ways to synchronize PWM registers. Current implementation allows to bypass the buffers, use legacy and PWM synchronization (hardware and software trigger) and it is also possible to use a half or full cycle reload strategy.

41.5.11.1 CNTIN register update

The following table describes when CNTIN register is updated:

Table 41-7. CNTIN register update

When	Then CNTIN register is updated
CLKS[1:0] = 0:0	When CNTIN register is written, independent of FTMEN bit.
• FTMEN = 0, or • CNTINC = 0	At the next FTM input clock after CNTIN was written.
• FTMEN = 1, • SYNCMODE = 1, and • CNTINC = 1	By the CNTIN register synchronization.
• CNTINC = 1, and • LDOK = 1	By the Reload Points.

41.5.11.2 MOD and HCR registers update

The following table describes when MOD or HCR registers are updated:

Table 41-8. MOD and HCR updates

When	Then MOD or HCR is updated
CLKS[1:0] = 0:0	When MOD (or HCR) is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the CPWMS bit, that is: • If the selected mode is not CPWM then MOD (or HCR) is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM then MOD (or HCR) register is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	By the MOD register synchronization. HCR follows the same procedure of MOD register in this case.
• LDOK = 1	By the Reload Points.

41.5.11.3 CnV register update

The following table describes when CnV register is updated:

Table 41-9. CnV register update

When	Then CnV register is updated
CLKS[1:0] = 0:0	When CnV register is written, independent of FTMEN bit.

Table continues on the next page...

Table 41-9. CnV register update (continued)

When	Then CnV register is updated
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the selected mode, that is: • If the selected mode is Output Compare, then CnV register is updated on the next FTM counter change, end of the prescaler counting, after CnV register was written. • If the selected mode is EPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	According to the selected mode, that is: • If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter, the end of the prescaler counting. If (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization. • If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization.
• SYNCEN = 1, and • LDOK = 1	By the Reload Points.

41.5.12 PWM synchronization

The PWM synchronization provides an opportunity to update the MOD, HCR, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

Note

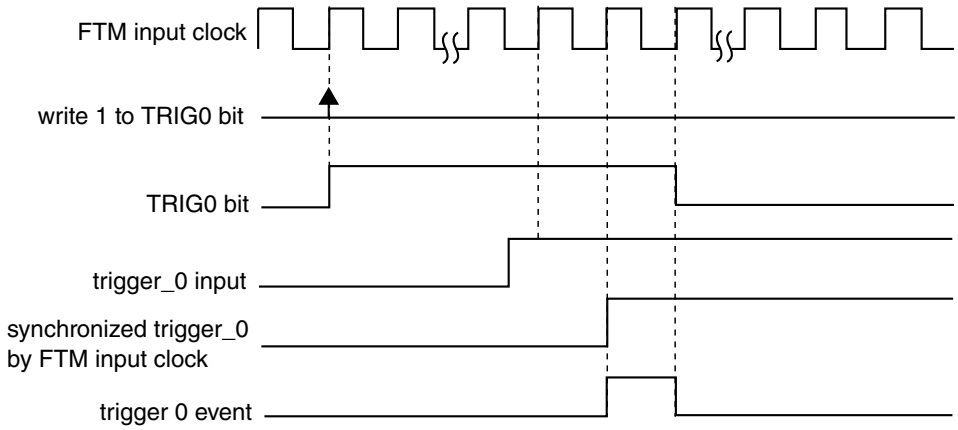
The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, only the enhanced PWM synchronization must be used.

41.5.12.1 Hardware trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the FTM input clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGn bit, then the synchronization is initiated, but TRIGn bit remains set due to the write operation.



Note
All hardware trigger inputs have the same behavior.

Figure 41-43. Hardware trigger event with HWTRIGMODE = 0

If HWTRIGMODE = 1, then the TRIGn bit is only cleared when 0 is written to it.

NOTE

The HWTRIGMODE bit must be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

41.5.12.2 Software trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point after that the

software trigger event occurred; see Synchronization Points and the following figure. If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see the following figure. If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.

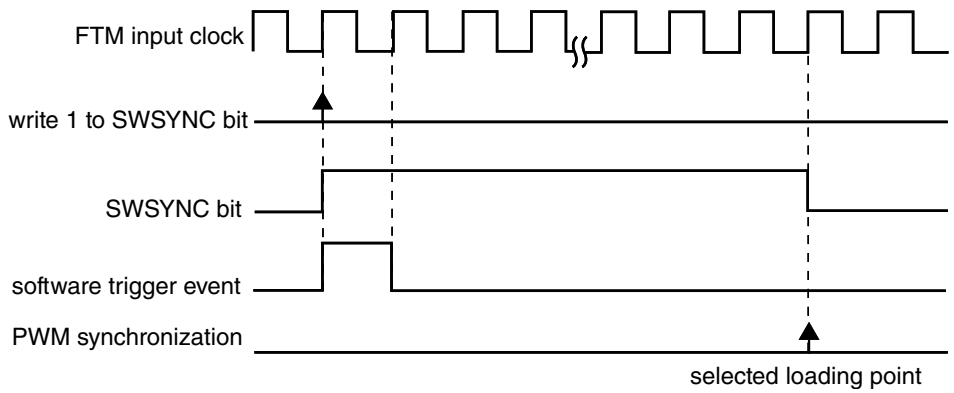


Figure 41-44. Software trigger event

41.5.12.3 Synchronization Points

The synchronization points are points where the registers can be updated with their write buffer by PWM synchronization. These synchronization points are safe points because guarantee smooth transitions in the generated PWM signals.

In Up counting mode, the synchronization points are when the FTM counter changes from MOD to CNTIN. In this case, the synchronization points are enabled if (CNTMIN = 1) or (CNTMAX = 1).

In Up-down counting mode, the synchronization points are:

- if (CNTMAX = 1), when the FTM counter changes from (MOD - 1) to MOD;
- if (CNTMIN = 1), when the FTM counter changes from (CNTIN + 1) to CNTIN.

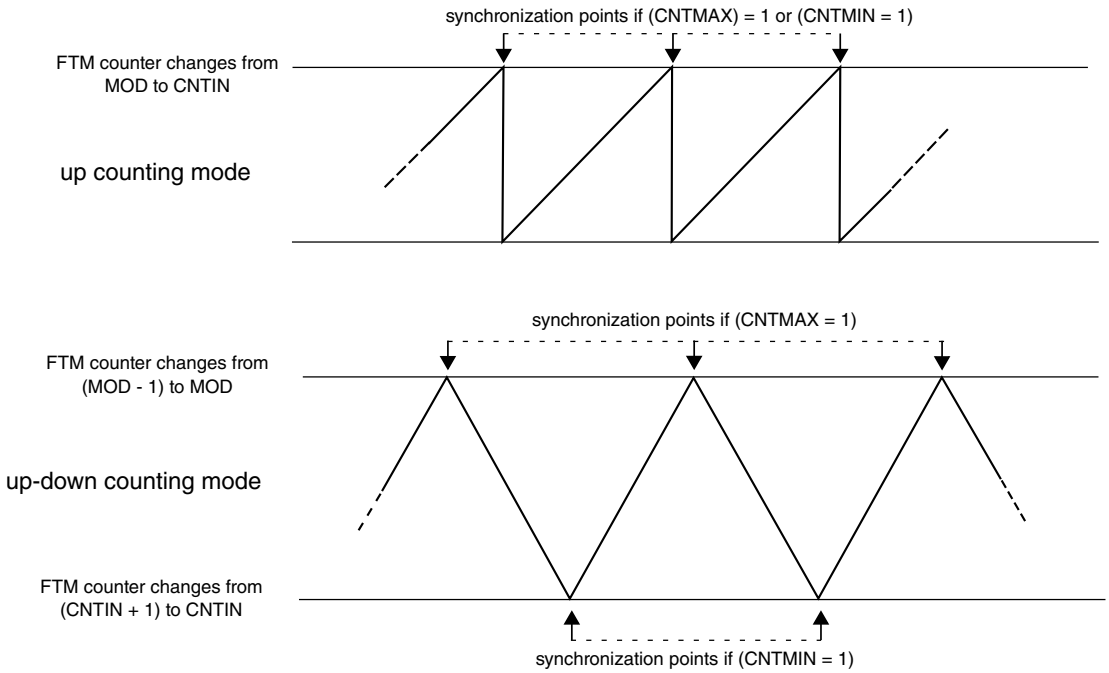


Figure 41-45. Synchronization Points

41.5.12.4 MOD register synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF, and HWRSTCNT bits according to this flowchart:

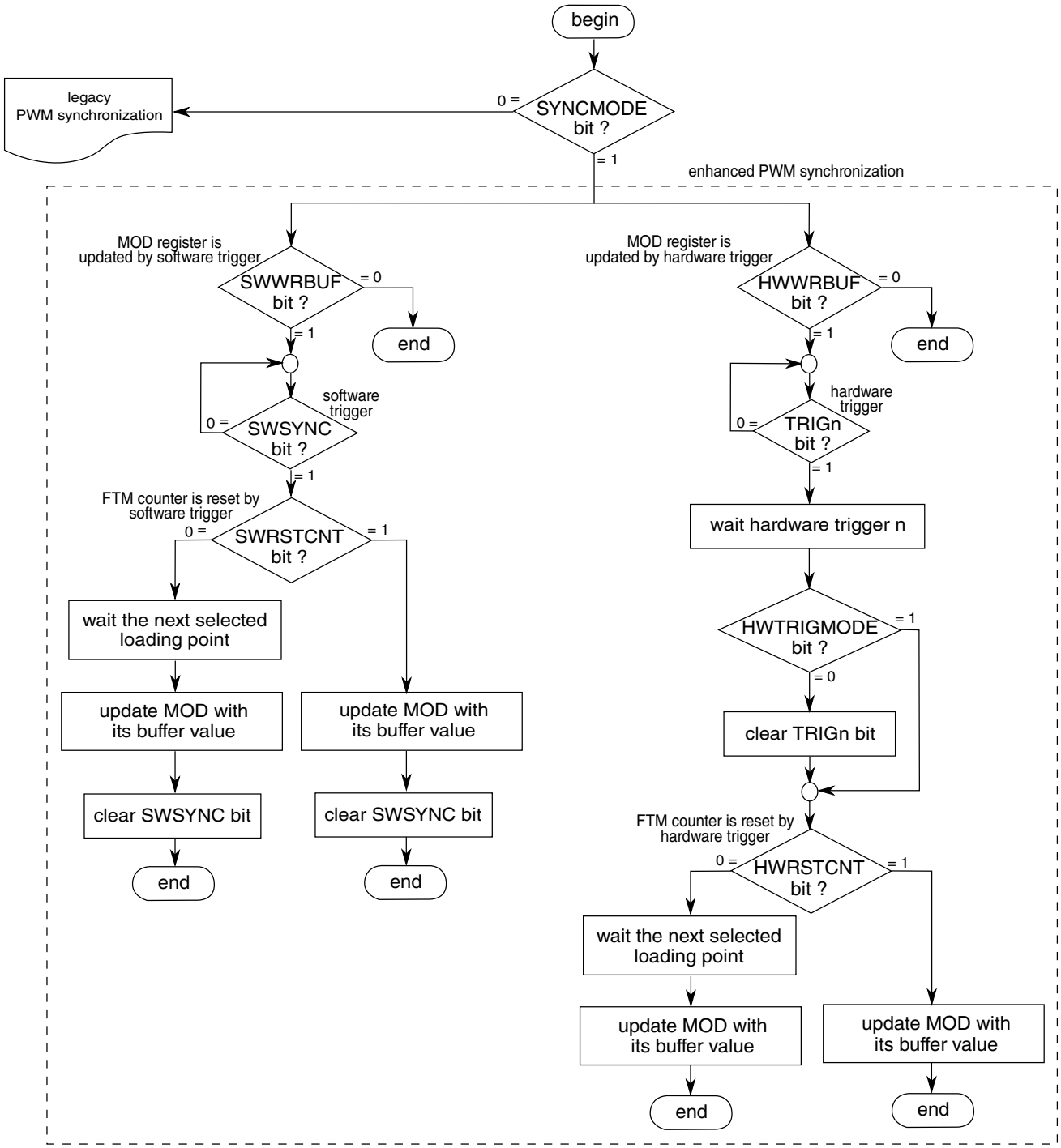


Figure 41-46. MOD register synchronization flowchart

In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 0), then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected

loading point. If the trigger event was a hardware trigger, then the trigger enable bit (TRIGn) is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

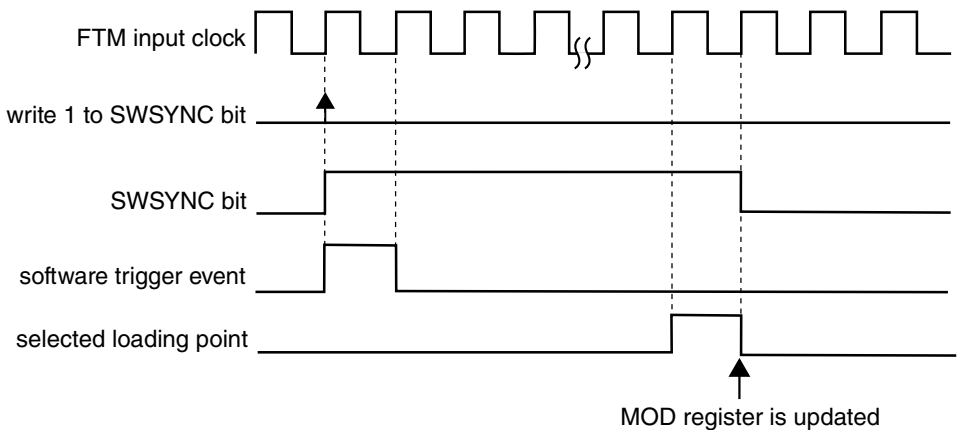


Figure 41-47. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and software trigger was used

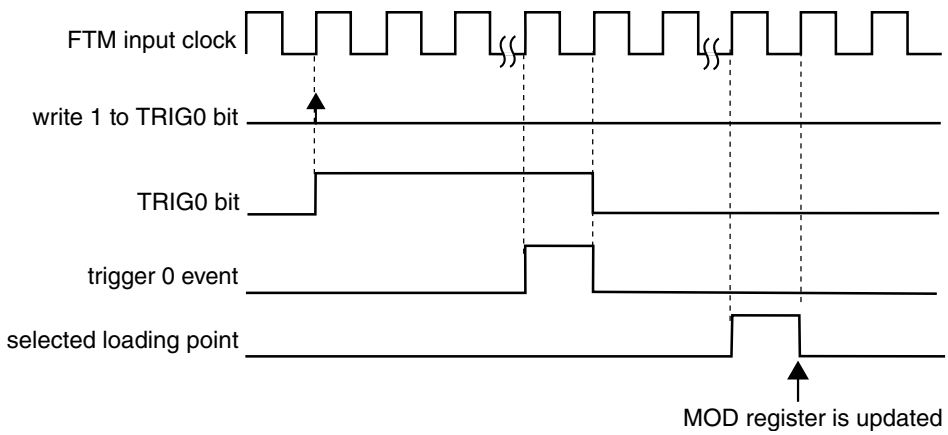


Figure 41-48. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and a hardware trigger was used

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 1), then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

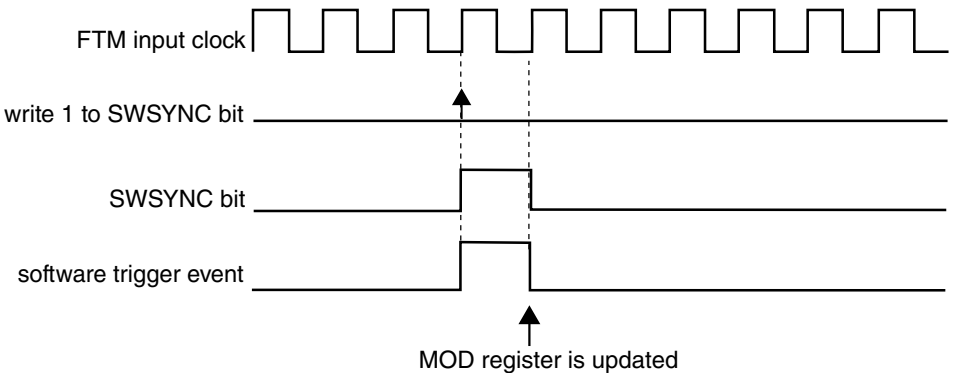


Figure 41-49. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and software trigger was used

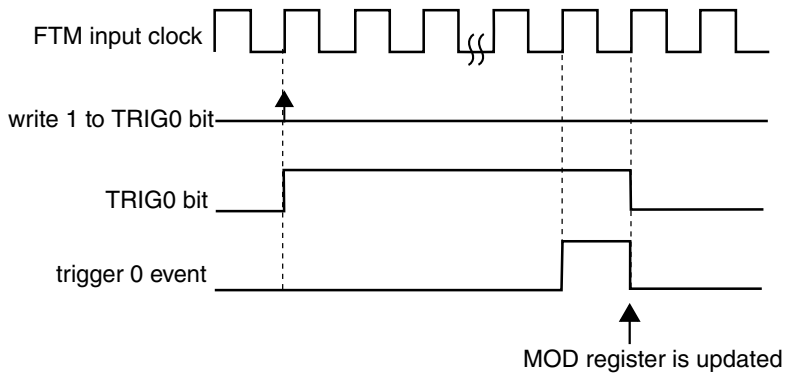


Figure 41-50. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and a hardware trigger was used

If (SYNCMODE = 0) and (PWMSYNC = 1), then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:

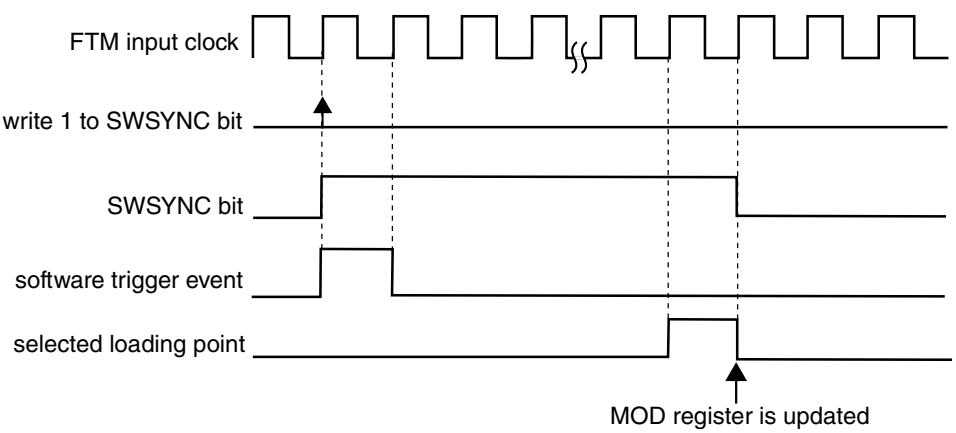


Figure 41-51. MOD synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)

41.5.12.5 CNTIN register synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1), and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization; see MOD register synchronization.

41.5.12.6 C(n)V and C(n+1)V register synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the MOD register synchronization. However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

41.5.12.7 OUTMASK register synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of FTM input clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

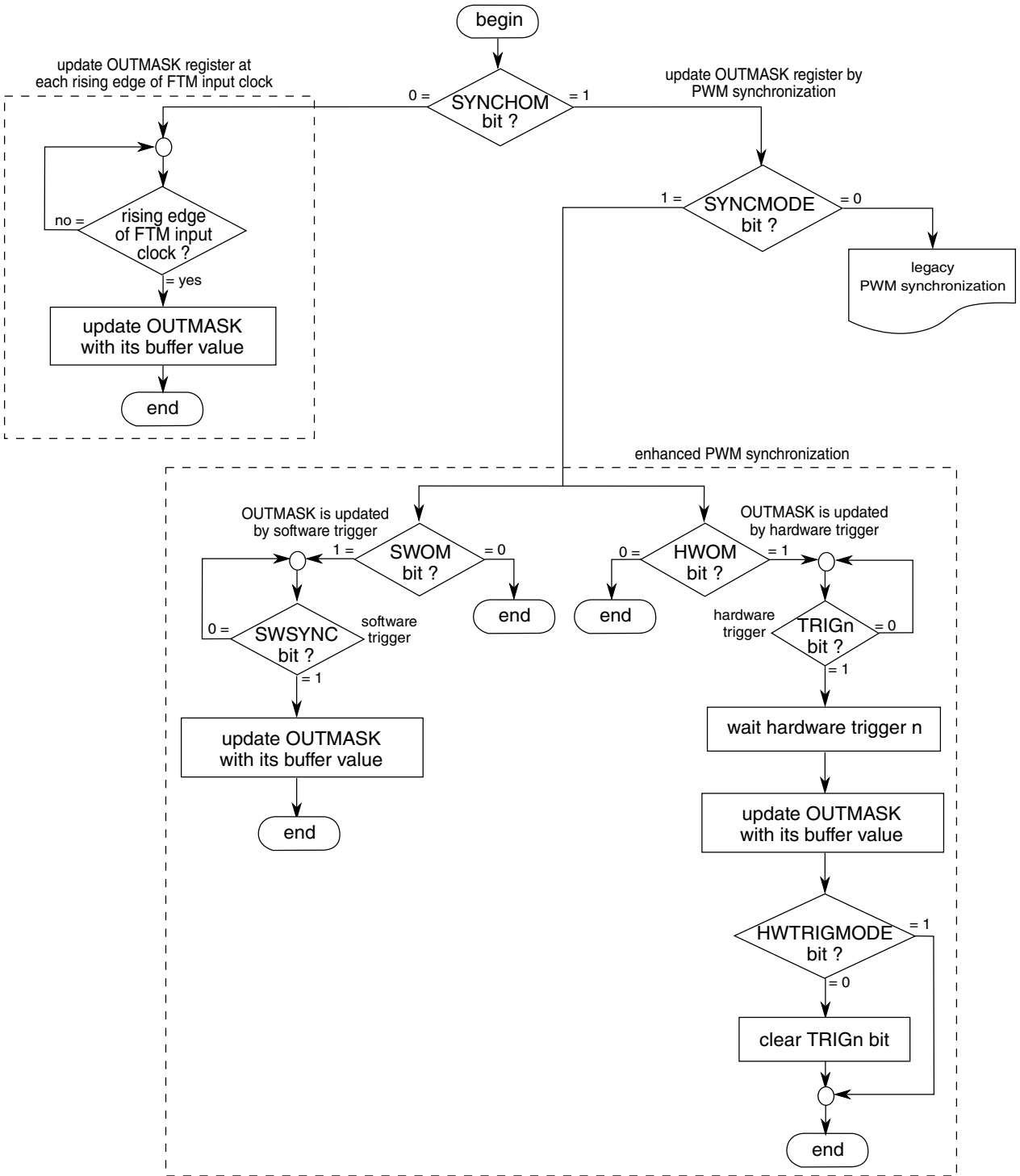


Figure 41-52. OUTMASK register synchronization flowchart

In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 0), then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

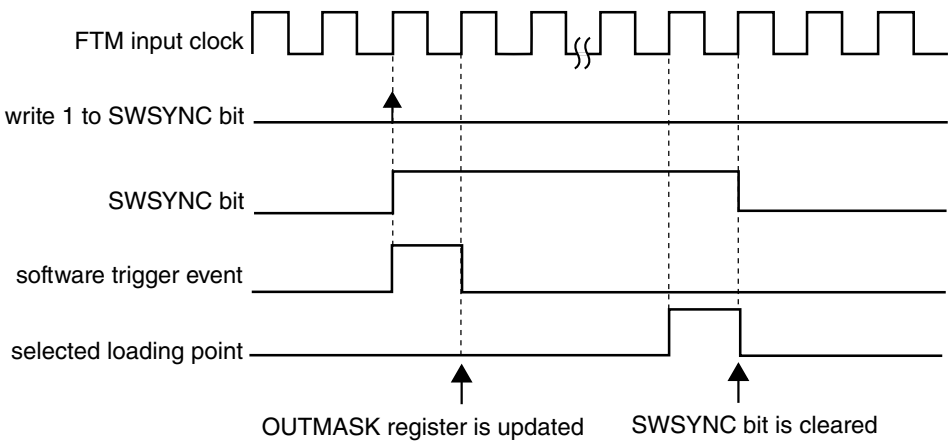


Figure 41-53. OUTMASK synchronization with (SYNCMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0) and software trigger was used

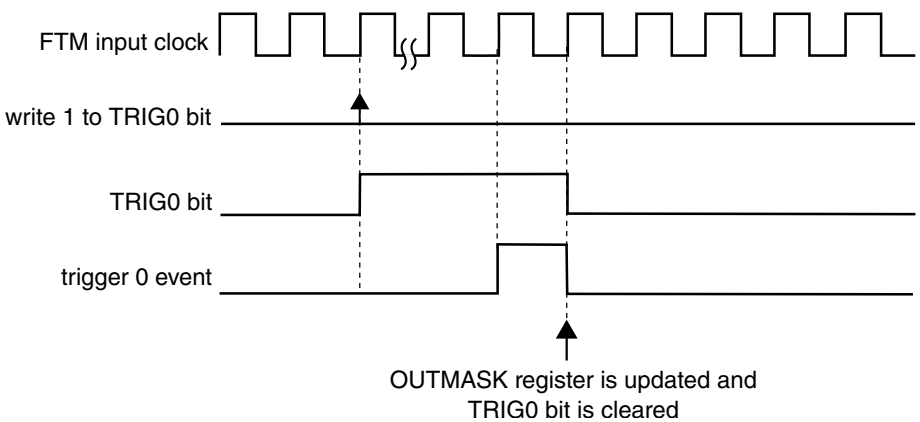


Figure 41-54. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 1), then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger. An example with a hardware trigger follows.

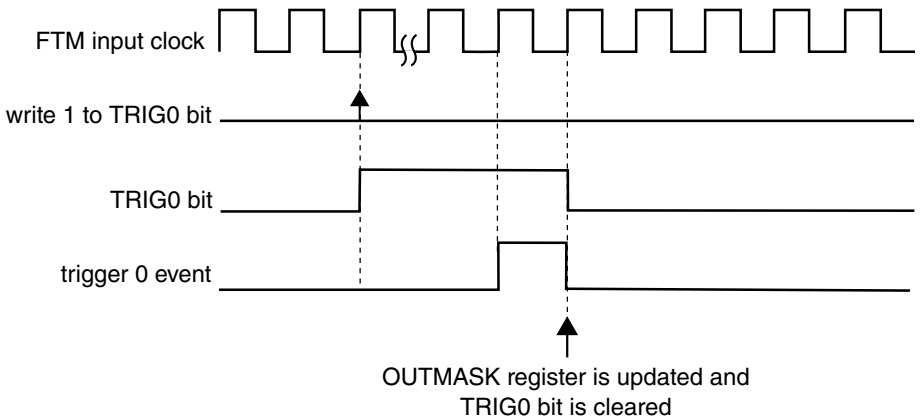


Figure 41-55. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and a hardware trigger was used

41.5.12.8 INVCTRL register synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of FTM input clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

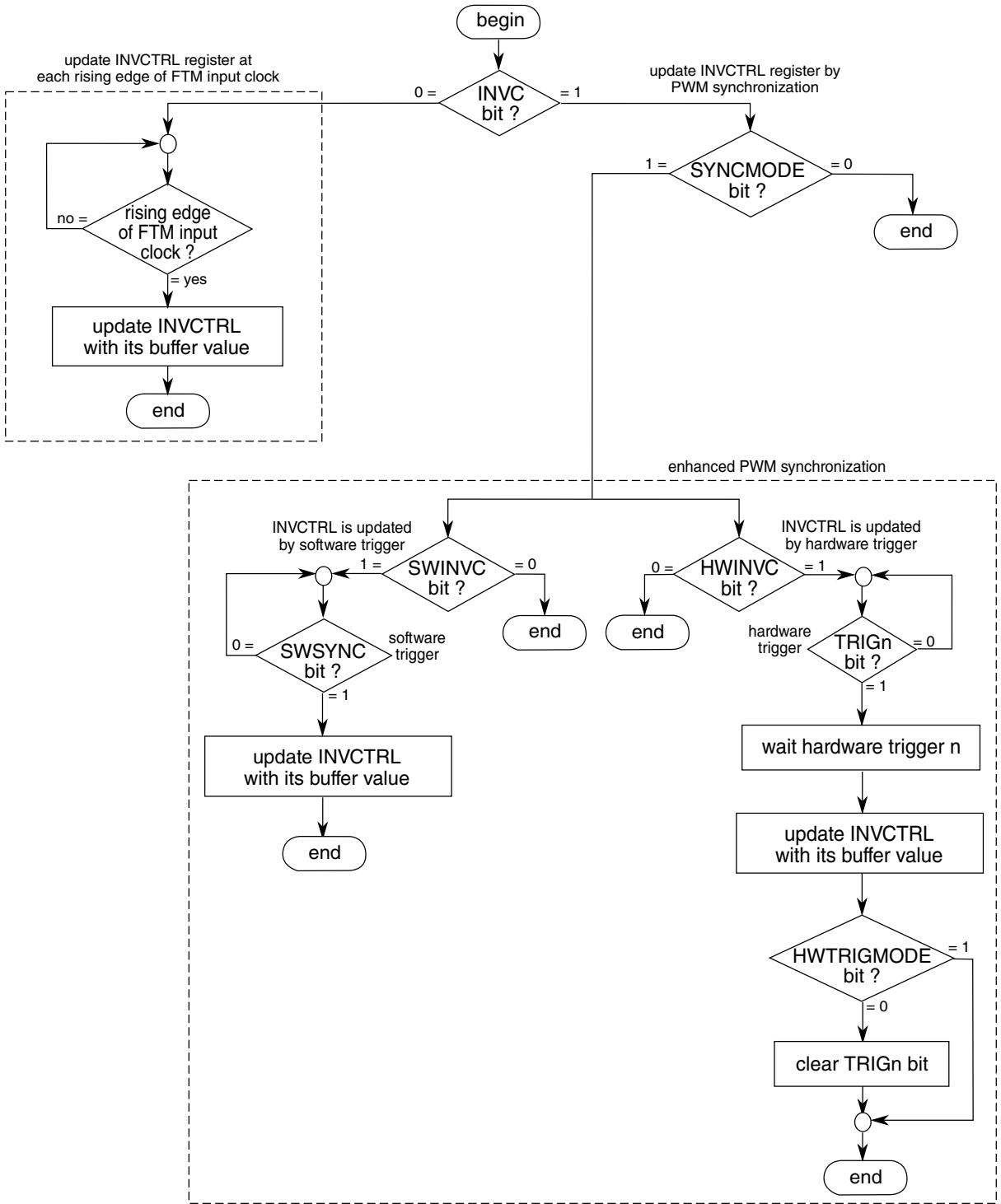


Figure 41-56. INVCTRL register synchronization flowchart

41.5.12.9 SWOCTRL register synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of FTM input clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

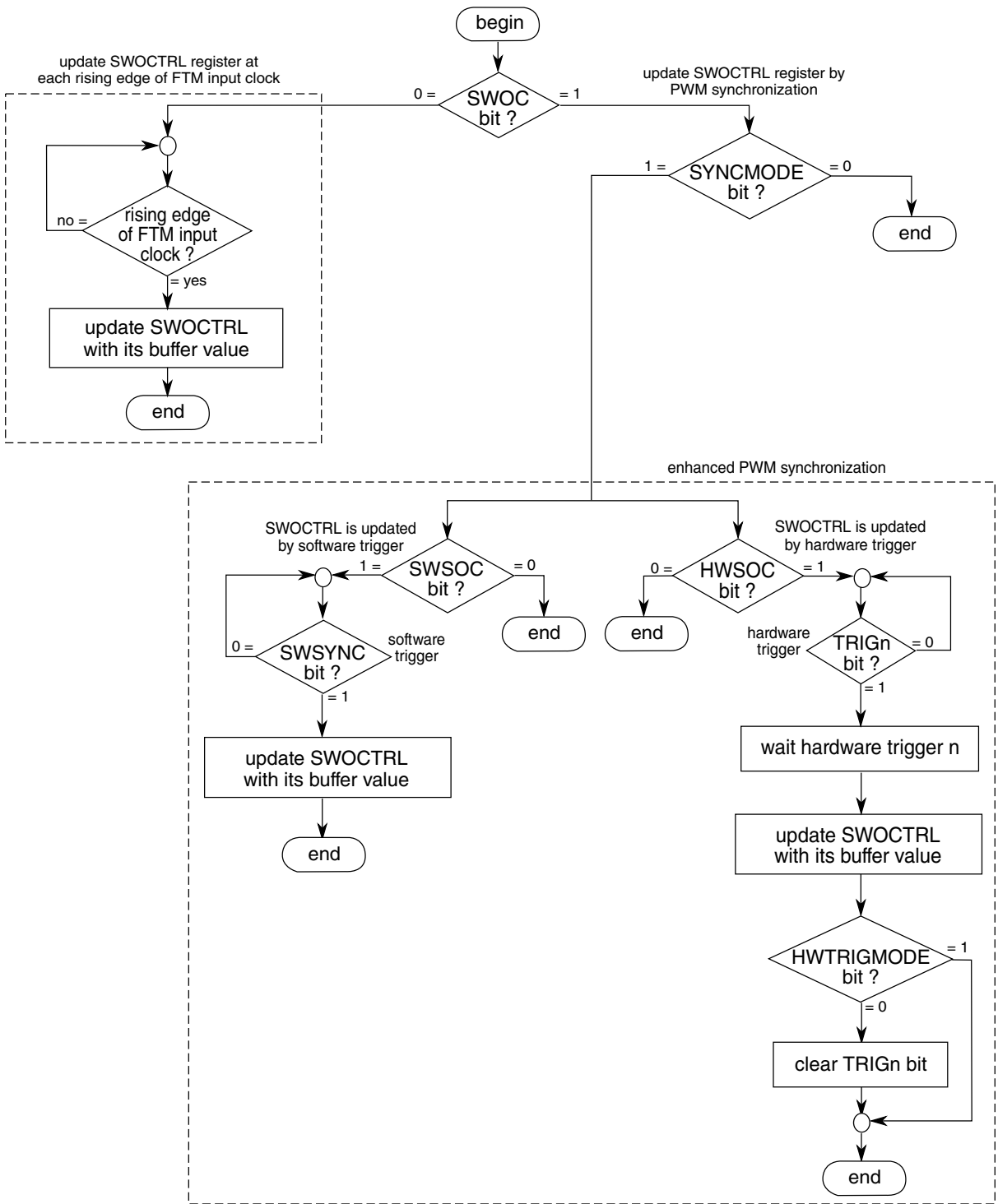


Figure 41-57. SWOCTRL register synchronization flowchart

41.5.12.10 FTM counter synchronization

The FTM counter synchronization is a mechanism that allows the FTM to restart the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value, except for channels in Output Compare mode, and the FTM counter is forced to its initial counting value defined by CNTIN register.

The following figure shows the FTM counter synchronization. Note that after the synchronization event occurs, the channel (n) is set to its initial value and the channel (n +1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected, then the channel (n+1) transitions to logical value 1 immediately after the synchronization event occurs.

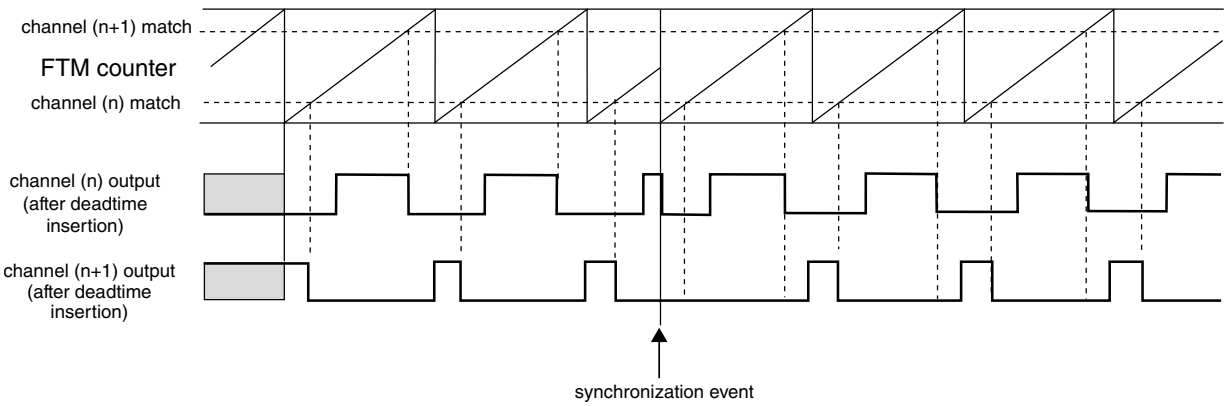


Figure 41-58. FTM counter synchronization

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, the FTM counter must be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

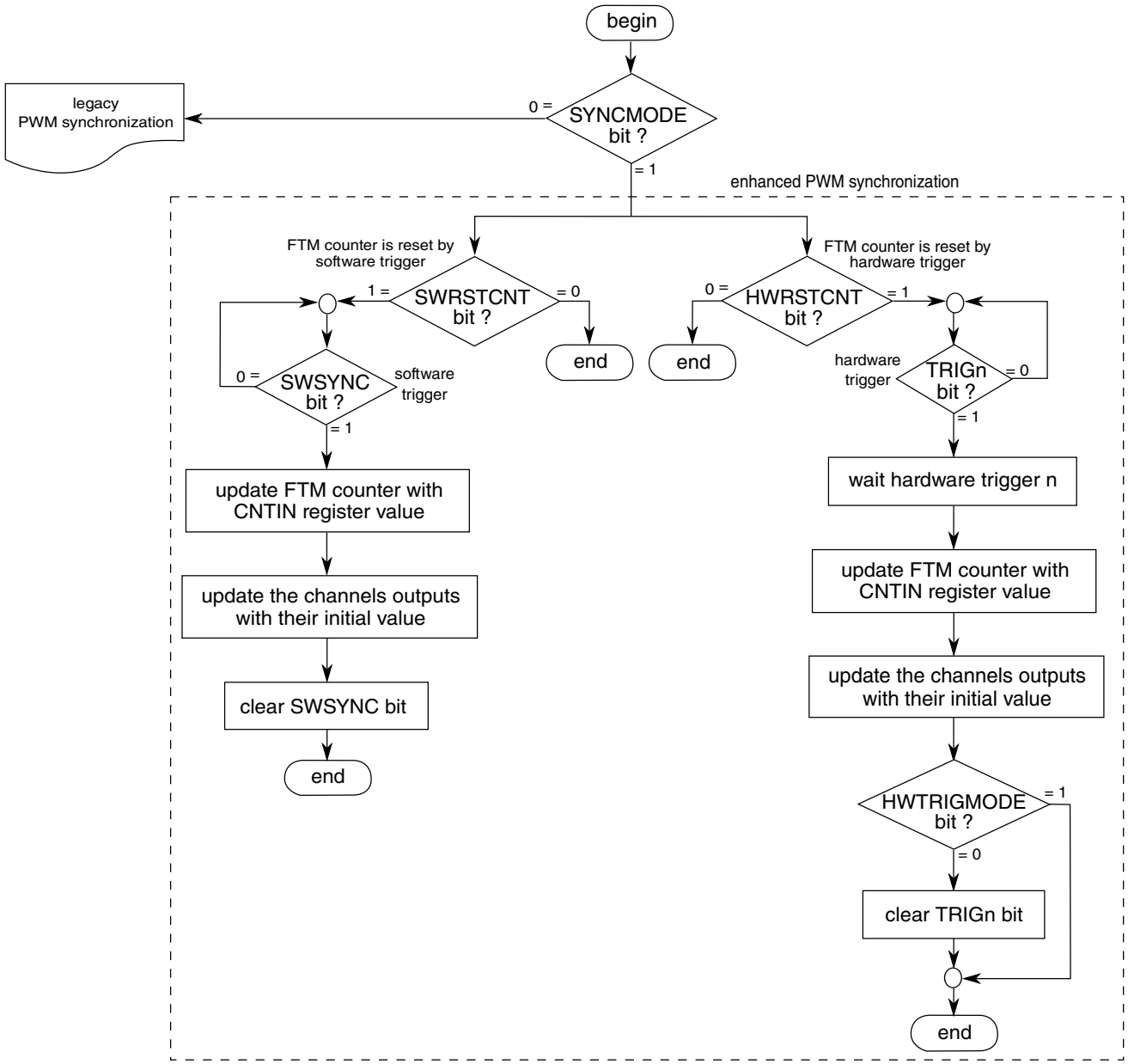


Figure 41-59. FTM counter synchronization flowchart

In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

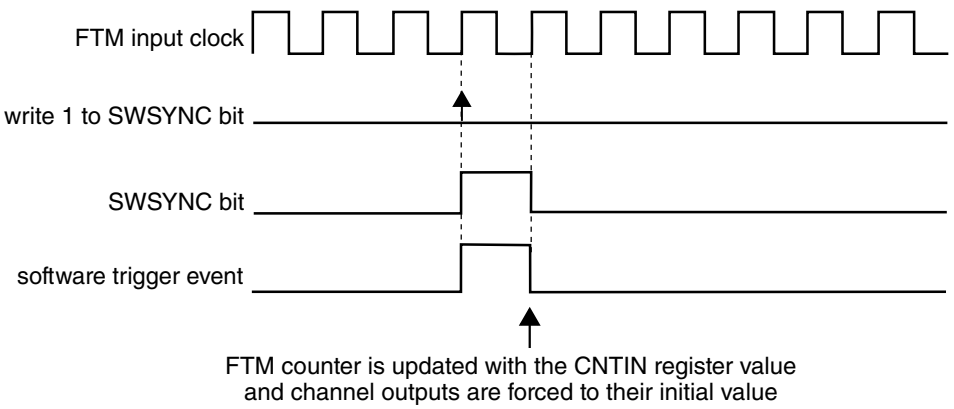


Figure 41-60. FTM counter synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and software trigger was used

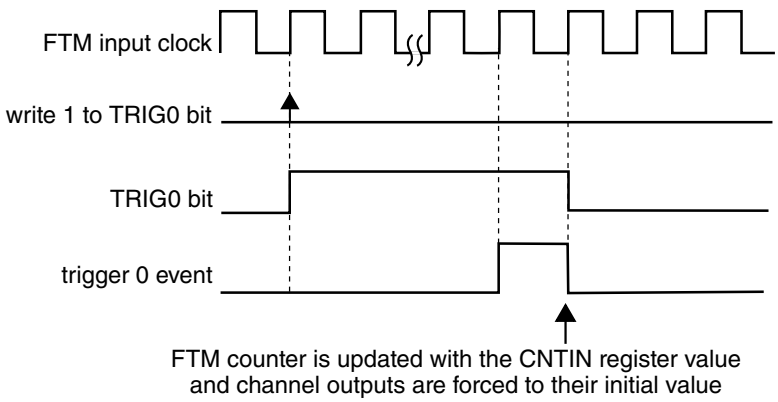


Figure 41-61. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger.

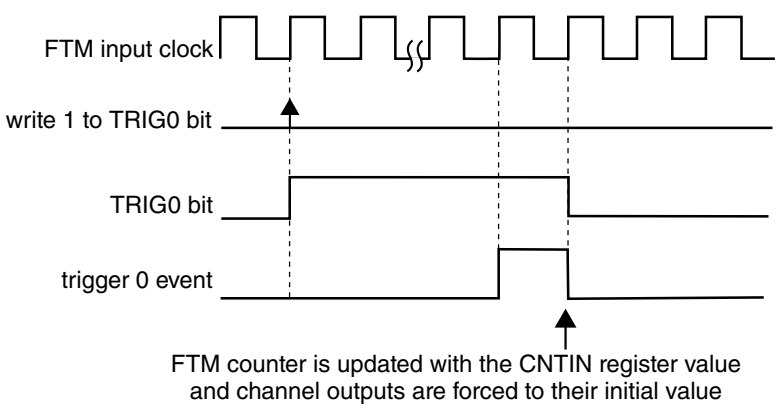


Figure 41-62. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and a hardware trigger was used

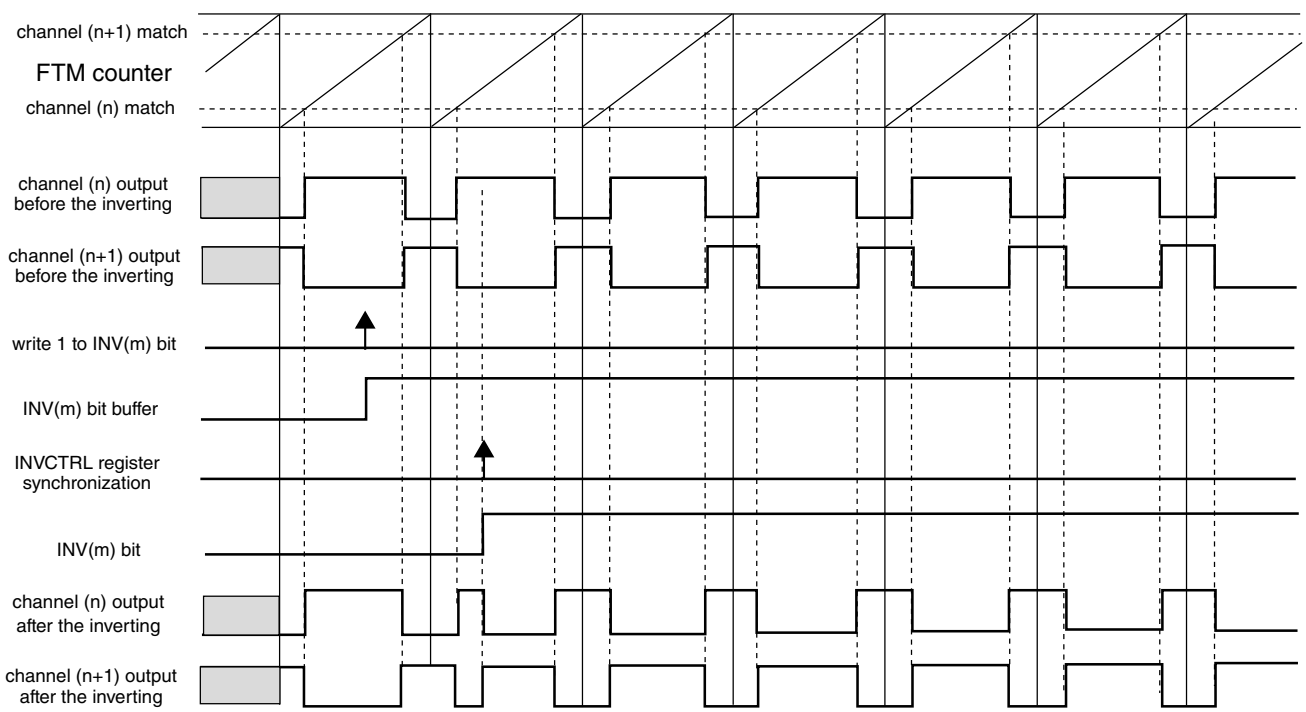
41.5.13 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1, and
- INVm = 1 (where m represents a channel pair)

The INVm bit in INVCTRL register is updated with its buffer value according to INVCTRL register synchronization

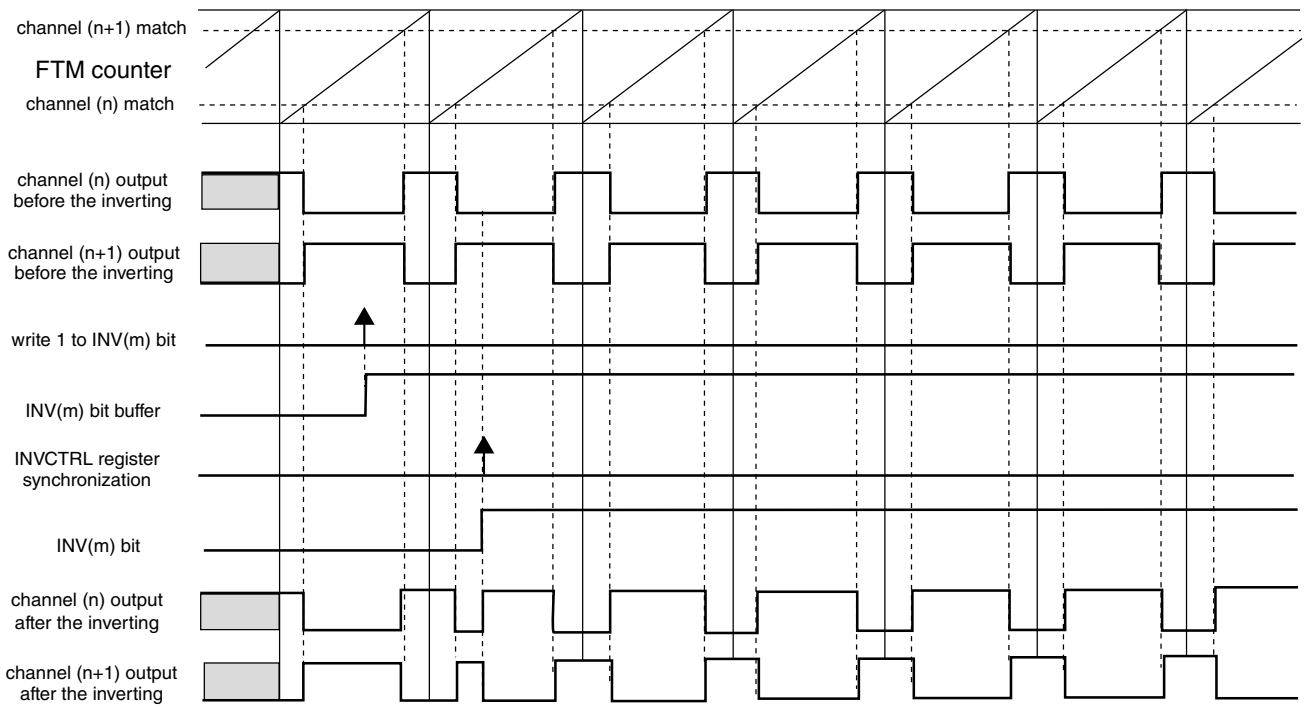
In High-True (ELSB:ELSA = 1:0) Combine mode, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.



NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 41-63. Channels (n) and (n+1) outputs after the inverting in High-True (ELSB:ELSA = 1:0) Combine mode

Note that the ELSB:ELSA bits value should be considered because they define the active state of the channels outputs. In Low-True (ELSB:ELSA = X:1) Combine mode, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. When inverting is selected, the channels (n) and (n+1) present waveforms as shown in the following figure.



NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 41-64. Channels (n) and (n+1) outputs after the inverting in Low-True (ELSB:ELSA = X:1) Combine mode

NOTE

The Inverting is not available in Output Compare mode.

41.5.14 Software Output Control Mode

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

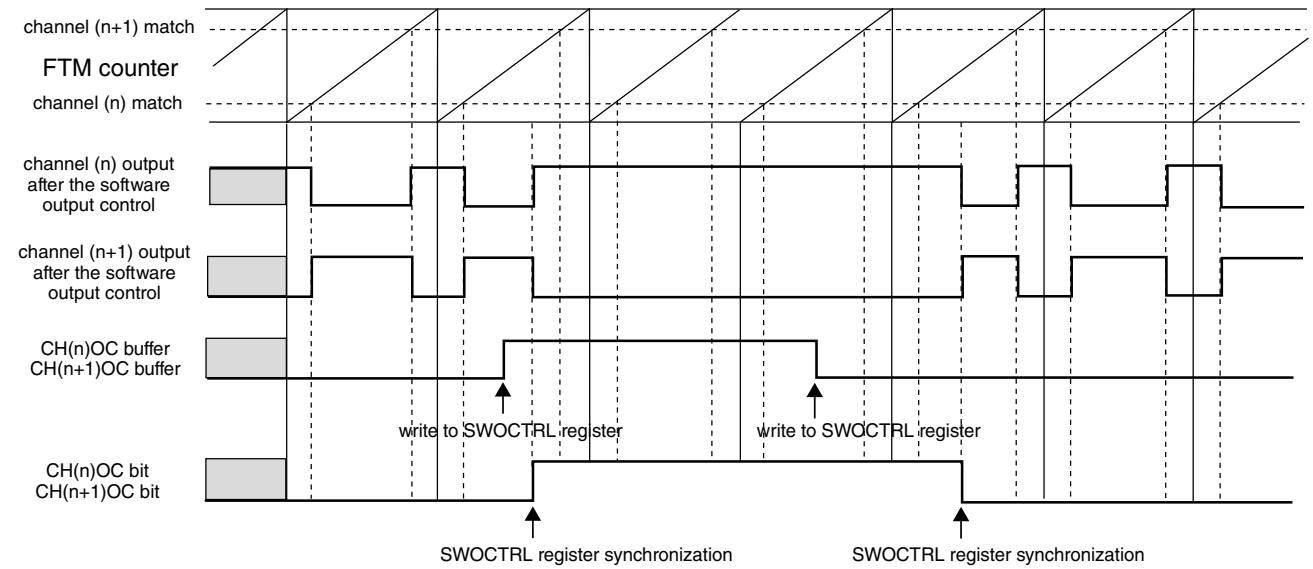
The software output control is selected when:

- QUADEN = 0
- DECAPEN = 0, and
- CH(n)OC = 1

The CH(n)OC bit enables the software output control for a specific channel output and the CH(n)OCV selects the value that is forced to this channel output.

Both CH(n)OC and CH(n)OCV bits in SWOCTRL register are buffered and updated with their buffer value according to SWOCTRL register synchronization.

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to Combine and Complementary mode.



NOTE
CH(n)OCV = 1 and CH(n+1)OCV = 0.

Figure 41-65. Example of software output control in Combine and Complementary mode

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

Table 41-10. Software ouput control behavior when (COMP = 0)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

Table 41-11. Software ouput control behavior when (COMP = 1)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

Note

- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit must not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n +1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS field description in the Status and Control register).

41.5.15 Deadtime insertion

The deadtime insertion is enabled when DTEN is set and the concatenation {DTVALEX[3:0],DTVAL[5:0]} is non- zero.

DEADTIME register defines the deadtime delay that can be used for all FTM channels. The clock for the DEADTIME delay is the FTM input clock divided by DTPS bits and the {DTVALEX[3:0],DTVAL[5:0]} bits define the deadtime modulo, that is, the number of the deadtime prescaler clocks.

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

If POL(n) = 0, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If POL(n) = 1, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly,

when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n +1) output is cleared.

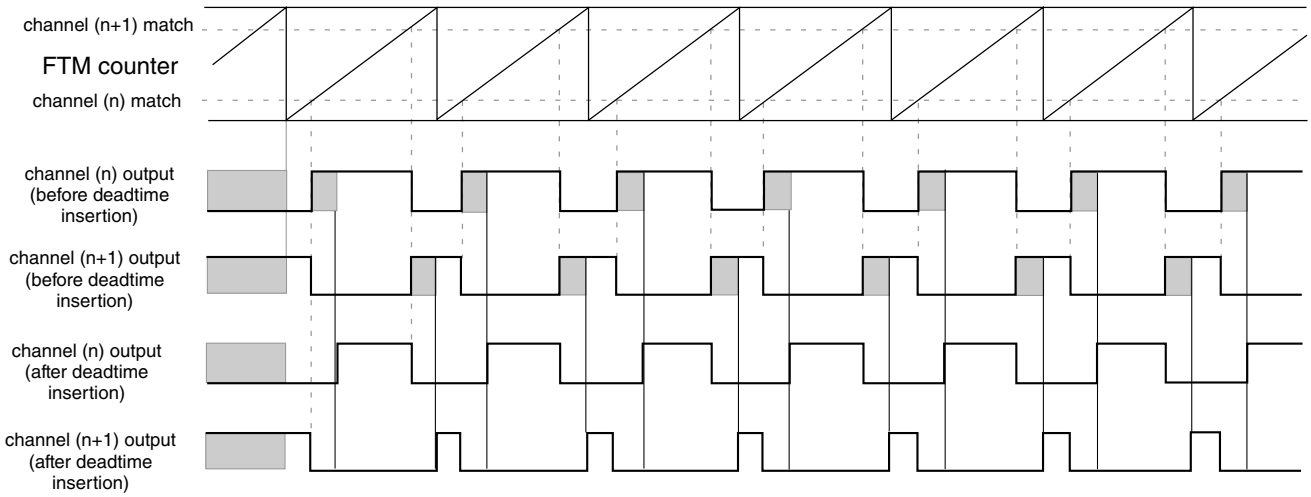


Figure 41-66. Deadtime insertion with ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0

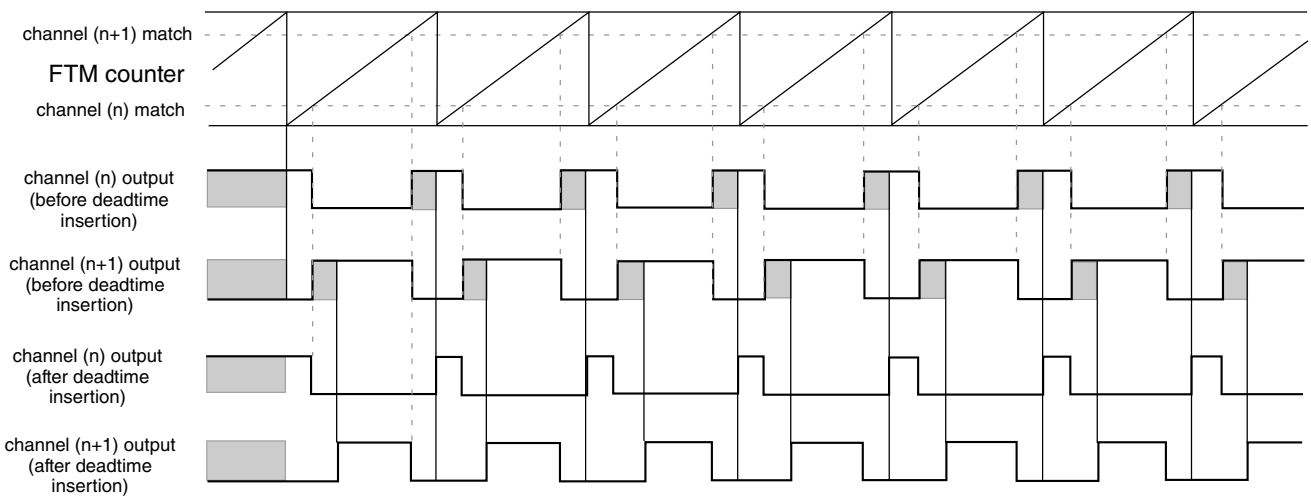


Figure 41-67. Deadtime insertion with ELSB:ELSA = X:1, POL(n) = 0, and POL(n+1) = 0

NOTE

- The deadtime feature must be used only in Complementary mode.
- The deadtime feature is not available in Output Compare mode.

41.5.15.1 Deadtime insertion corner cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ((C(n +1)V – C(n)V) × FTM input clock), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ((MOD – CNTIN + 1 – (C(n+1)V – C(n)V)) × FTM input clock), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.

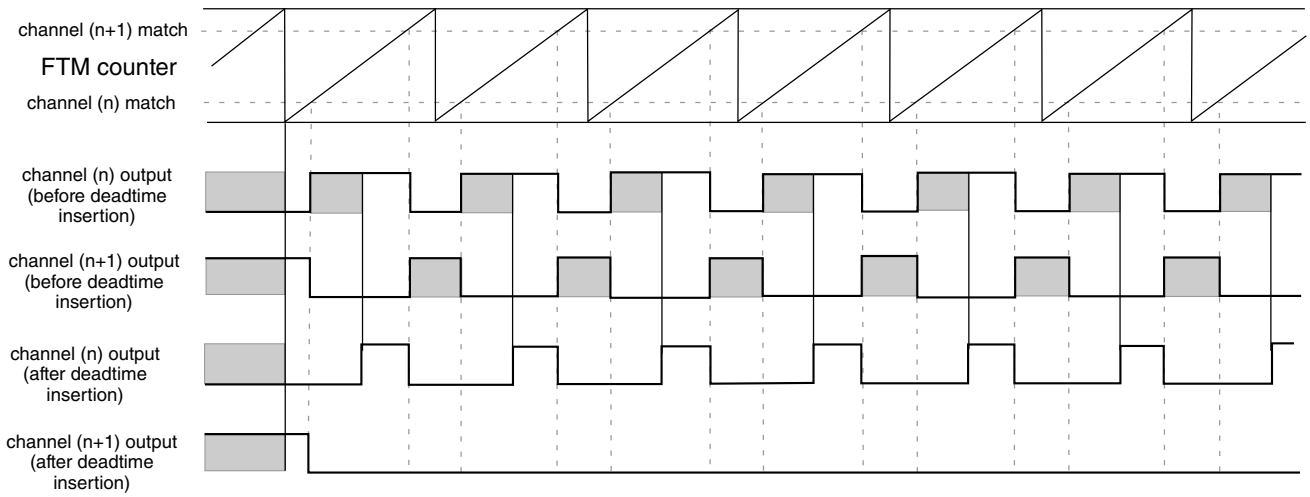


Figure 41-68. Example of the deadtime insertion (ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channel (n+1) duty cycle

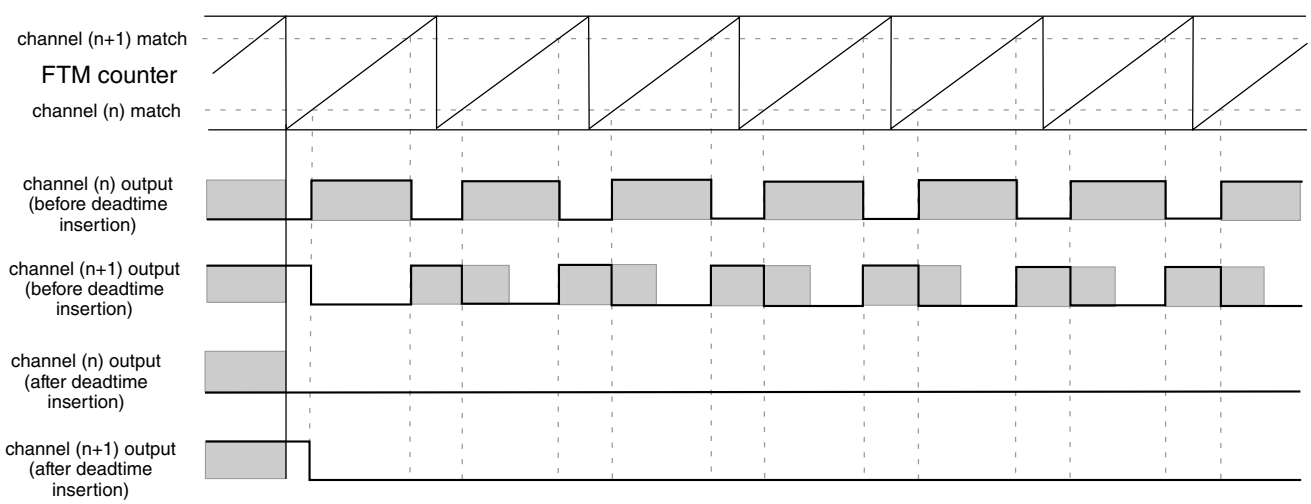


Figure 41-69. Example of the deadtime insertion (ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channels (n) and (n+1) duty cycle

41.5.16 Output mask

The output mask can be used to force channels output to their inactive state through software. For example: to control a BLDC motor.

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization; see OUTMASK register synchronization.

If CH(n)OM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CH(n)OM = 0, then the channel (n) output is unaffected by the output mask. See the following figure.

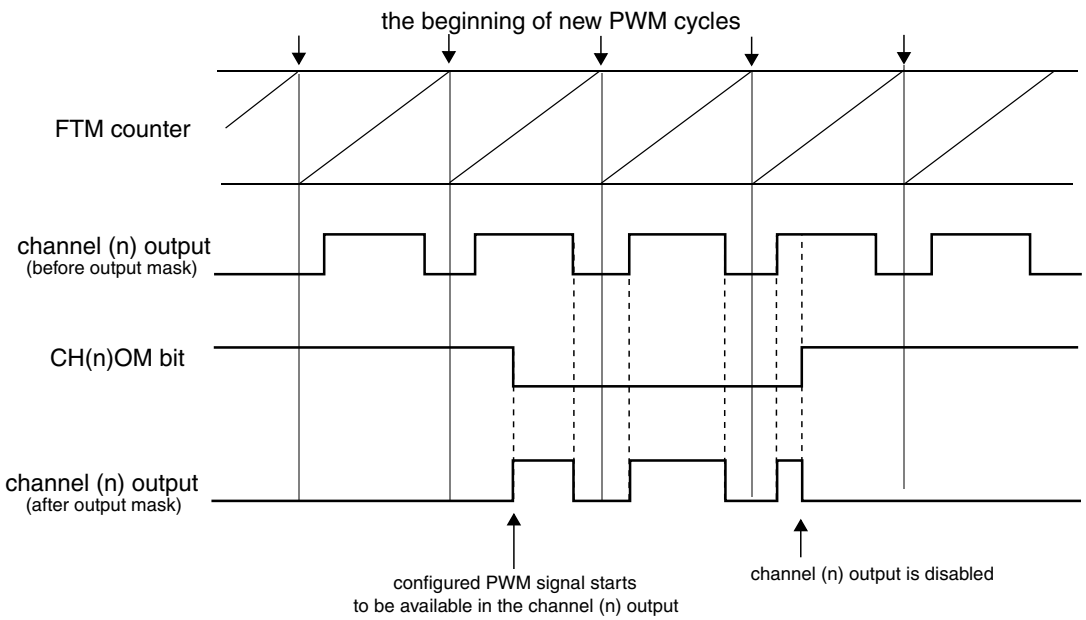


Figure 41-70. Output mask with POLn = 0

The following table shows the output mask result before the polarity control.

Table 41-12. Output mask result for channel (n) before the polarity control

CH(n)OM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

41.5.17 Fault control

The fault control is enabled if (FAULTM[1:0] ≠ 0:0).

FTM can have up to four fault inputs. FAULTnEN bit (where n = 0, 1, 2, 3) enables the fault input n and FFLTRnEN bit enables the fault input n filter. FFVAL[3:0] bits select the value of the enabled filter in each enabled fault input.

First, each fault input signal is synchronized by the FTM input clock; see the synchronizer block in the following figure. Following synchronization, the fault input n signal enters the filter block. The filter clock is prescaled and controlled by FLTPS[3:0] bits. When there is a state change in the fault input n signal, the 5-bit counter is reset and starts counting up. As long as the new state is stable on the fault input n, the counter continues to increment. If the 5-bit counter overflows, that is, the counter exceeds the value of the FFVAL[3:0] bits, the new fault input n value is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the fault input n signal before validation (counter overflow), the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by FFVAL[3:0] bits (× filter clock) is regarded as a glitch and is not passed on to the edge detector.

The fault input n filter is disabled when the FFVAL[3:0] bits are zero or when FAULTnEN = 0. In this case, the fault input n signal is delayed 2 rising edges of the FTM input clock and the FAULTFn bit is set on 3th rising edge of the FTM input clock after a rising edge occurs on the fault input n.

If FFVAL[3:0] ≠ 0000 and FAULTnEN = 1, then the delay is (2 FTM input clocks + (2 + FFVAL) filter clk), that is, the FAULTFn bit is set (2 FTM input clocks + 2 filter clocks + FFVAL[3:0]) clock edges after a rising edge occurs on the fault input n.

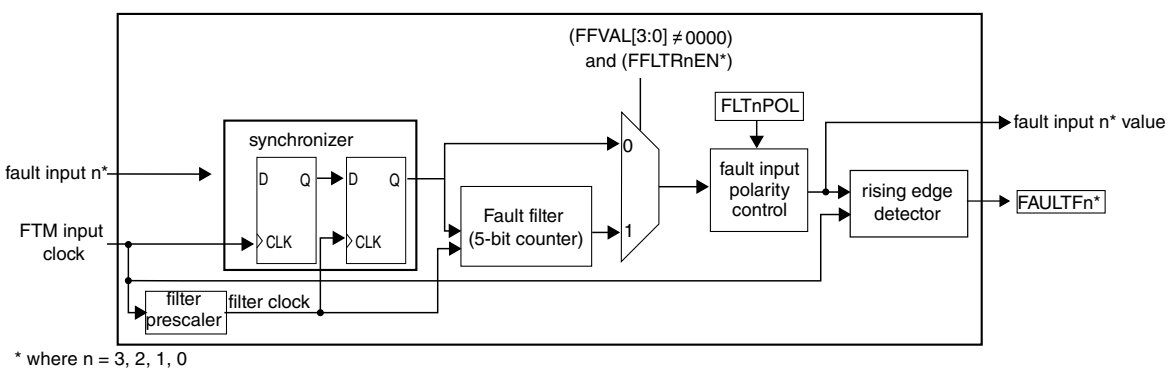


Figure 41-71. Fault input n control block diagram

If the fault control and fault input n are enabled and a rising edge at the fault input n signal is detected, a fault condition has occurred and the FAULTFn bit is set. The FAULTF bit is the logic OR of FAULTFn[3:0] bits. See the following figure.

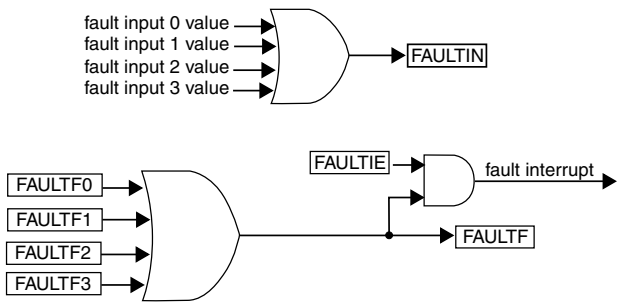


Figure 41-72. FAULTF and FAULTIN bits and fault interrupt

If the fault control is enabled (FAULTM[1:0] ≠ 0:0), a fault condition has occurred and (FAULTEN = 1), then outputs are forced to their safe values:

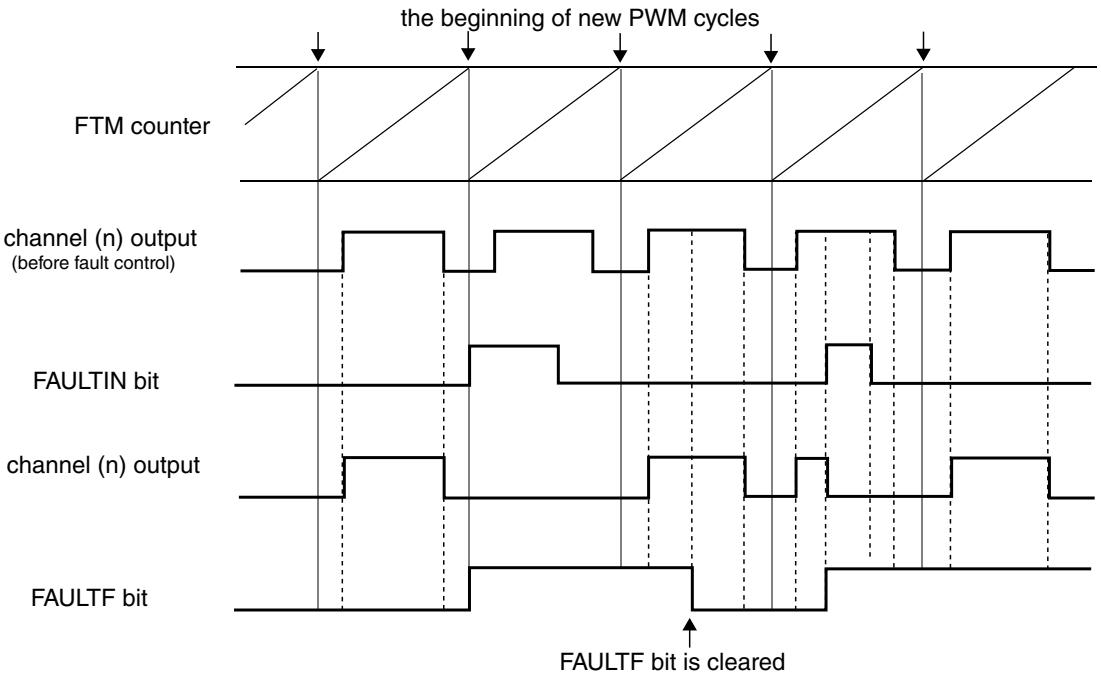
- Channel (n) output takes the value of POL(n)
- Channel (n+1) takes the value of POL(n+1)

The fault interrupt is generated when (FAULTF = 1) and (FAULTIE = 1). This interrupt request remains set until:

- Software clears the FAULTF bit by reading FAULTF bit as 1 and writing 0 to it
- Software clears the FAULTIE bit
- A reset occurs

41.5.17.1 Automatic fault clearing

If the automatic fault clearing is selected (FAULTM[1:0] = 1:1), then the channels output disabled by fault control is again enabled when the fault input signal (FAULTIN) returns to zero and a new PWM cycle begins. See the following figure.

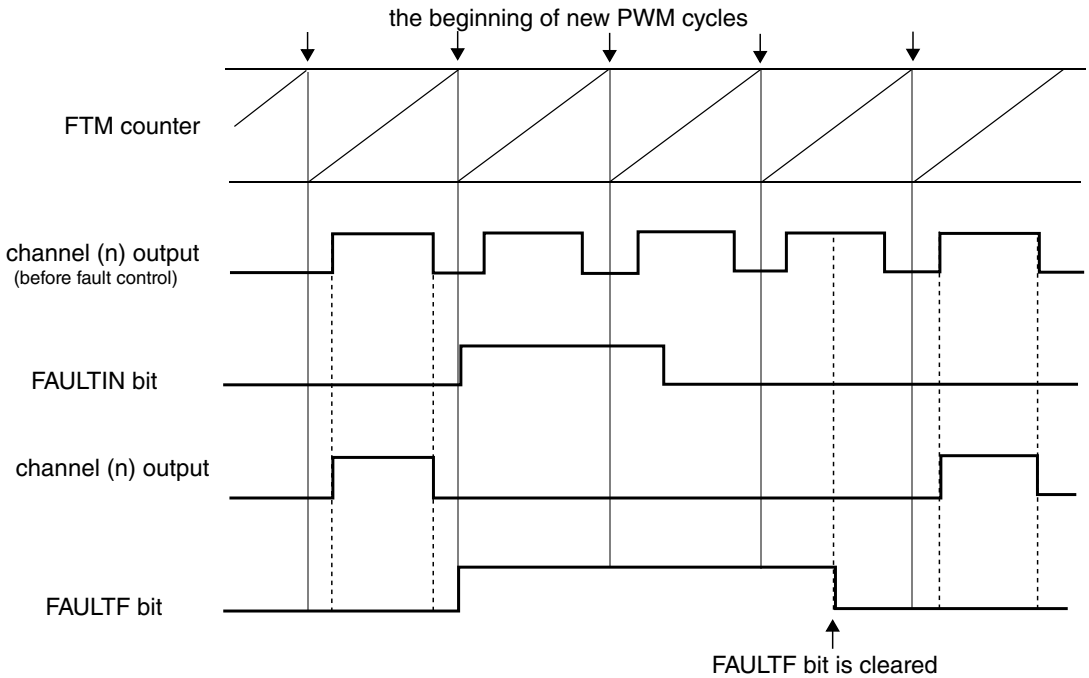


NOTE
The channel (n) output is after the fault control with automatic fault clearing and POLn = 0.

Figure 41-73. Fault control with automatic fault clearing

41.5.17.2 Manual fault clearing

If the manual fault clearing is selected (FAULTM[1:0] = 0:1 or 1:0), then the channels output disabled by fault control is again enabled when the FAULTF bit is cleared and a new PWM cycle begins. See the following figure.



NOTE
The channel (n) output is after the fault control with manual fault clearing and POLn = 0.

Figure 41-74. Fault control with manual fault clearing

41.5.17.3 Fault inputs polarity control

The FLTjPOL bit selects the fault input j polarity, where j = 0, 1, 2, 3:

- If FLTjPOL = 0, the fault j input polarity is high, so the logical one at the fault input j indicates a fault.
- If FLTjPOL = 1, the fault j input polarity is low, so the logical zero at the fault input j indicates a fault.

41.5.18 Polarity Control

The POLn bit selects the channel (n) output polarity:

- If POLn = 0, the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If POLn = 1, the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

41.5.19 Initialization

The initialization forces the CH(n)OI bit value to the channel (n) output when a one is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

Table 41-13. Initialization behavior when (COMP = 0 and DTEN = 0)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

Table 41-14. Initialization behavior when (COMP = 1 or DTEN = 1)

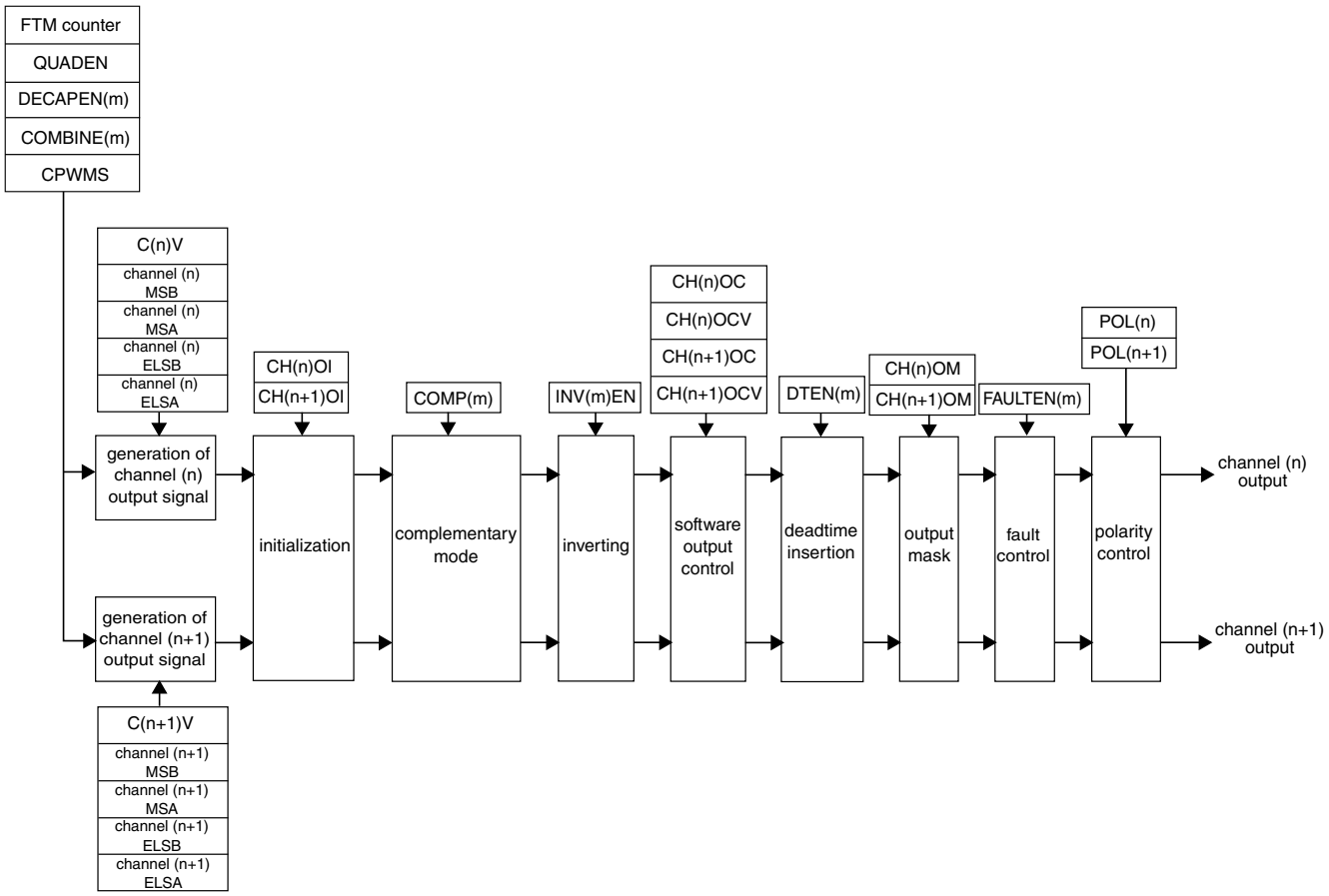
CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

Note

The initialization feature must be used only with disabled FTM counter. See the description of the CLKS field in the Status and Control register.

41.5.20 Features priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.



NOTE
The channels (n) and (n+1) are in Output Compare, EPWM, CPWM or Combine modes.

Figure 41-75. Priority of the features used at the generation of channels (n) and (n+1) output

NOTE

The Initialization must not be used with Inverting and Software Output Control Mode.

41.5.21 External Trigger

If the CH(j)TRIG bit of the External Trigger (FTM_EXTTRIG) register is set, where j = 0, 1, 2, 3, 4, 5, 6 or 7, then the FTM generates a trigger when the channel (j) match occurs (FTM counter = C(j)V).

The external trigger feature provides a trigger signal which has one FTM clock period width and is used for on-chip modules.

The FTM is able to generate multiple triggers in one PWM period. Because each trigger is generated for a specific channel, several channels are required to implement this functionality. This behavior is described in Figure 41-76.

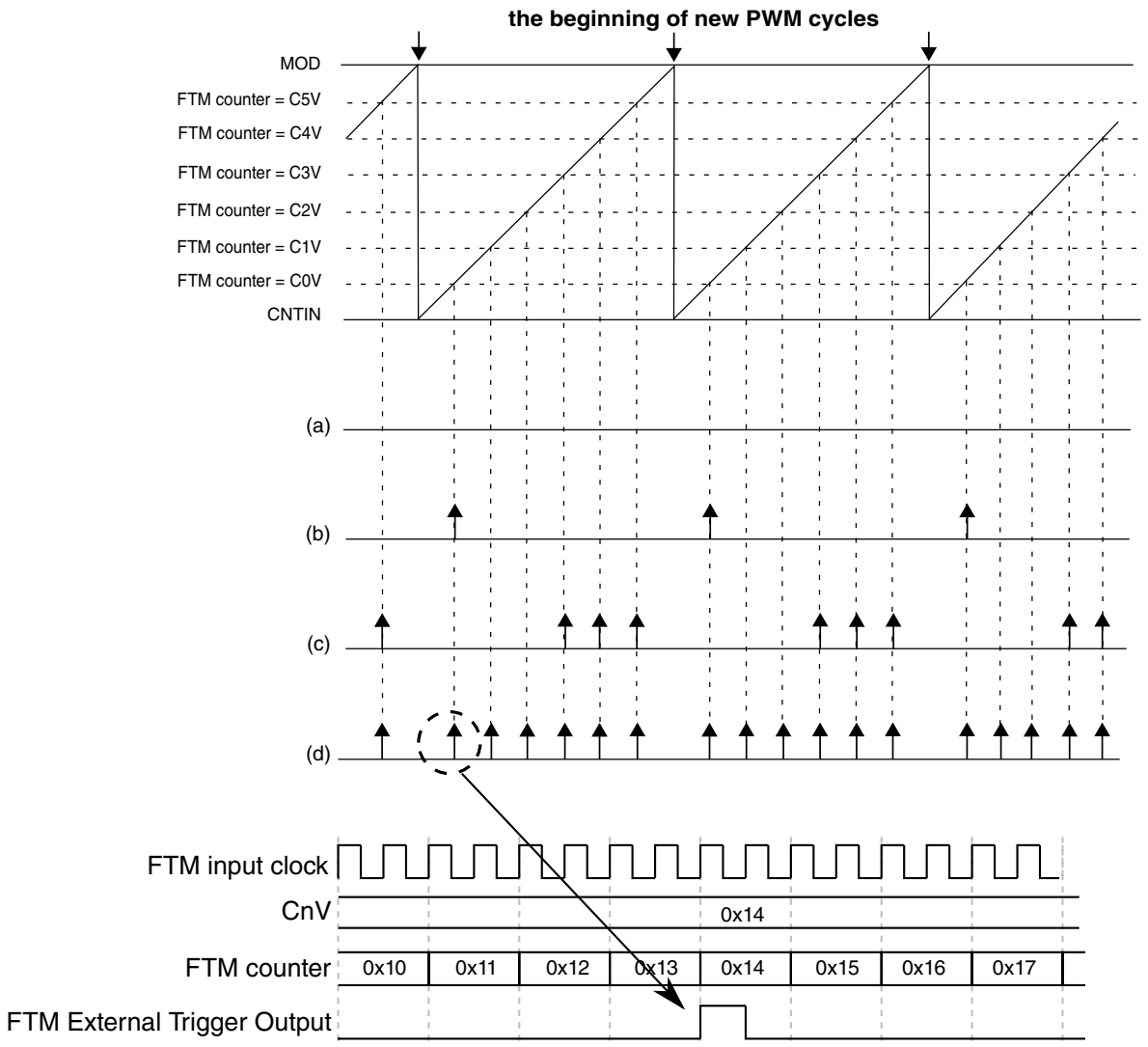


Figure 41-76. External Trigger

41.5.22 Channel trigger output

The channel trigger output provides a trigger signal which has one FTM clock period width in the channel output signal.

If the TRIGMODE bit of the CnSC register is set (TRIGMODE=1), a trigger pulse with one FTM clock cycle width is generated in the channel output when a match occurs. It is only allowed to use trigger mode when channel is in EPWM (up counting) or CPWM (up-down counting).

The figures below show some cases of trigger generation in a channel output.

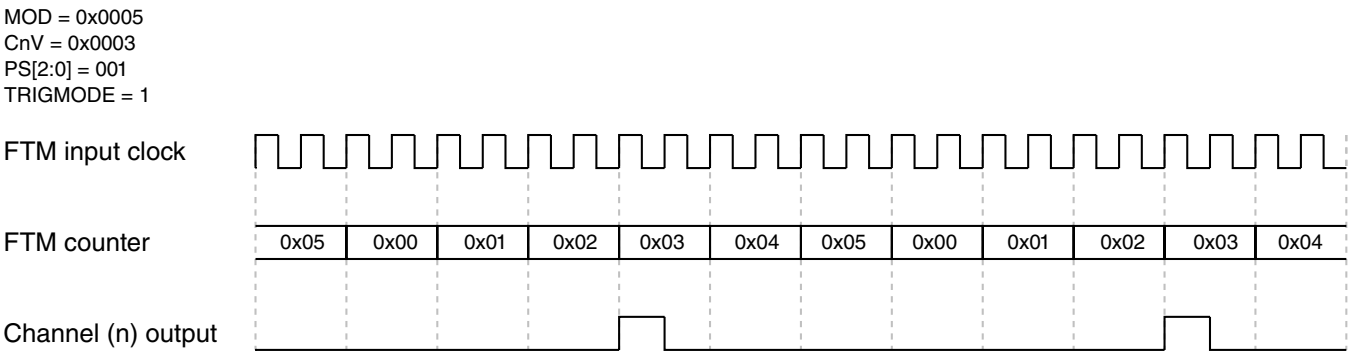


Figure 41-77. Example of trigger generation in the output channel for up counting mode

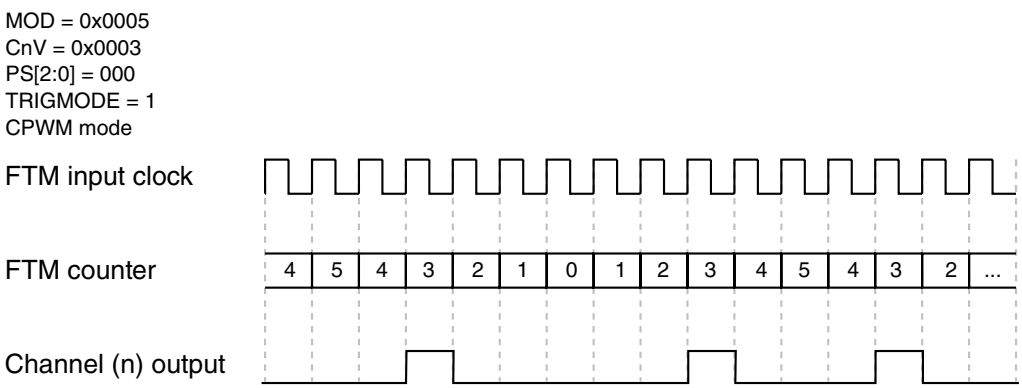


Figure 41-78. Example of trigger generation in the output channel for up-down counting mode

41.5.23 Initialization trigger

Initialization trigger allows FTM to generate an external trigger in some specific points of FTM counter cycle. This feature is controlled by two bits. INITTRIGEN enables the trigger generation and the ITRIGR selects in which events the initialization trigger should be generated. If INITTRIGEN = 1 and ITRIGR = 1, then the initialization trigger is generated when FTM counter reaches a reload point considering the Load Frequency configuration. See the Reload Points for more details about reload points. If INITTRIGEN = 1 and ITRIGR = 0, then FTM generates a trigger when the FTM counter is updated with the CNTIN register value in the following cases:

- In all cycles that FTM counter is automatically updated with CNTIN register value.

- When there is a write to CNT register.
- When there is the FTM counter synchronization.
- If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits.
- If the channel (n) is in Input Capture mode, (ICRST = 1) and the selected input capture event occurs in the channel (n) input.

The following figures show these cases.

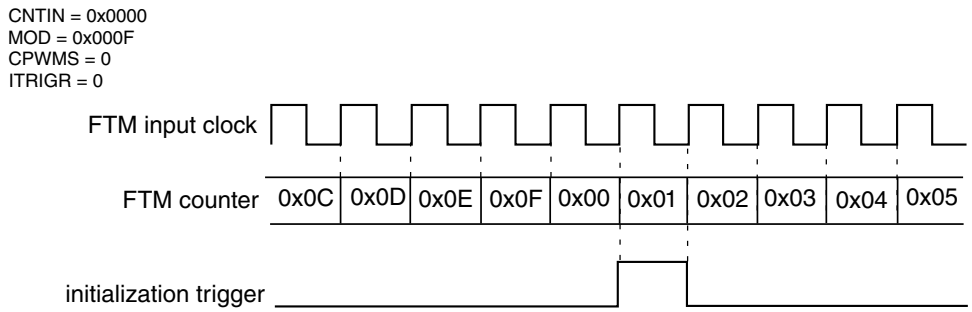


Figure 41-79. Initialization trigger is generated when the FTM counting achieves the CNTIN register value and ITRIGR = 0

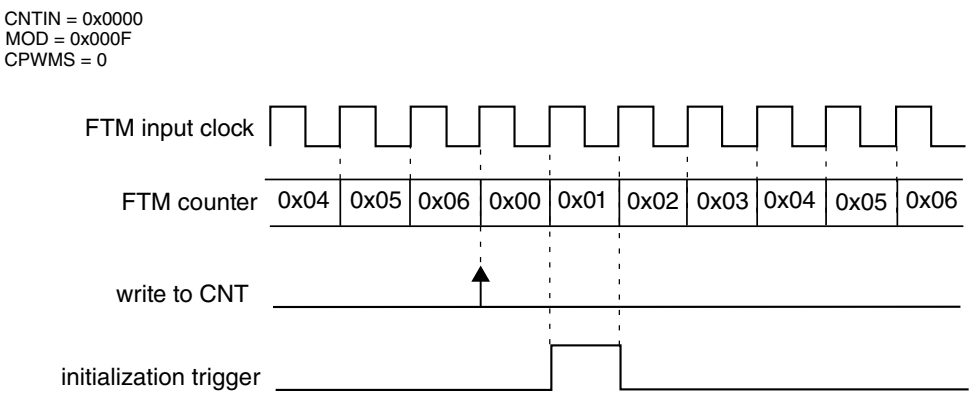


Figure 41-80. Initialization trigger is generated when there is a write to CNT register

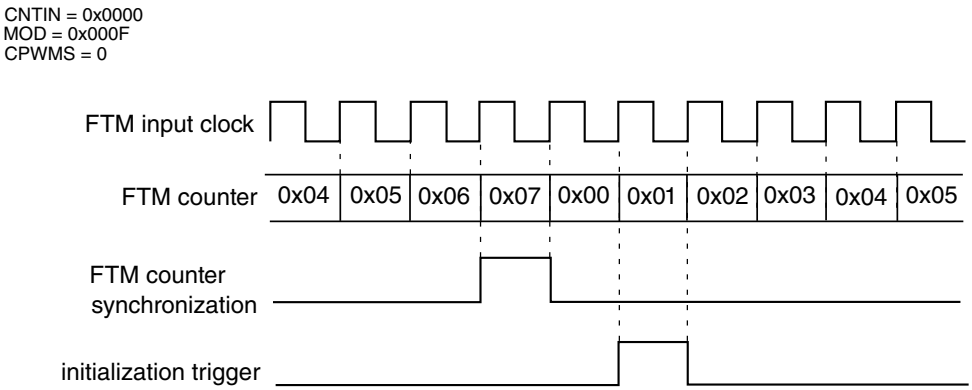


Figure 41-81. Initialization trigger is generated when there is the FTM counter synchronization

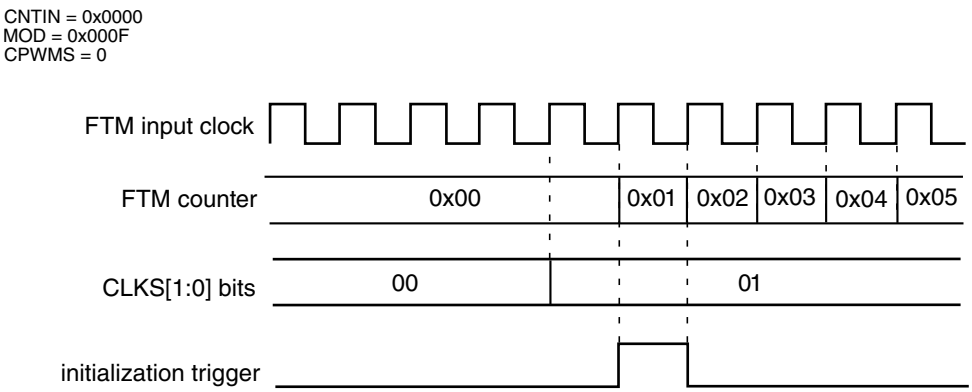
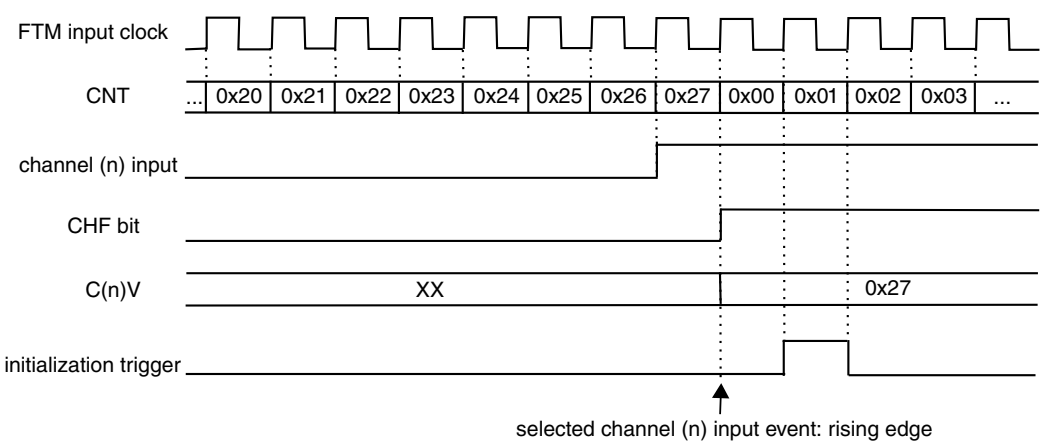


Figure 41-82. Initialization trigger is generated if (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits



NOTE
Channel (n) input after its synchronizer and filter
MOD = 0xFFFF
CNTIN = 0x0000
PS[2:0] = 3'b000
ICRST = 1'b1

Figure 41-83. Initialization trigger is generated if the channel (n) is in Input Capture mode, ICRST = 1 and the selected input capture event occurs in the channel (n) input

The initialization trigger output provides a trigger signal that is used for on-chip modules.

Note

- When FTM is in up-down count mode (CPWMS = 1), the initialization trigger can be generated according to loadpoints CNTMAX and CNTMIN at SYNC register if ITRIGR=1.

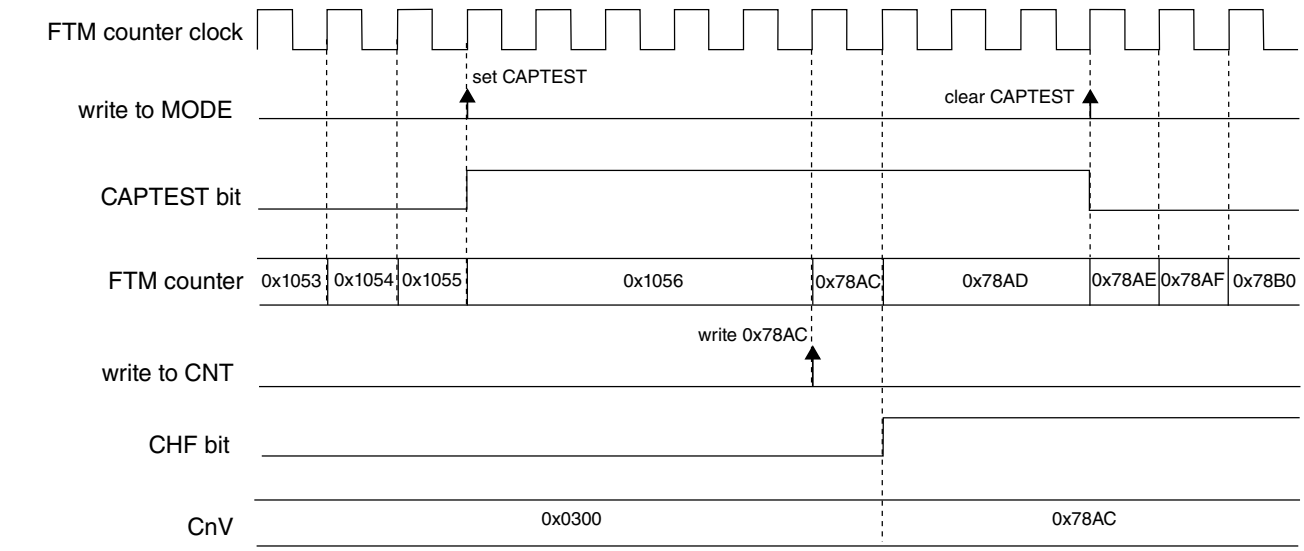
41.5.24 Capture Test Mode

The Capture Test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for Input Capture mode and FTM counter must be configured to the Up counting.

When the Capture Test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter; see the following figure. After it was written, all CnV registers are updated with the written value to CNT register and CHF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration. Its next value depends on CNTIN, MOD, and the written value to FTM counter.

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.



NOTE
- FTM counter is free running and (FTMEN = 1);
- FTM channel (n) is in Input Capture Mode.

Figure 41-84. Capture Test Mode

41.5.25 DMA

The channel generates a DMA transfer request according to DMA and CHIE bits. See the following table.

Table 41-15. Channel DMA transfer request

DMA	CHIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
1	1	The channel DMA transfer request is generated if (CHF = 1).	The channel interrupt is not generated.

If DMA = 1, the CHF bit is cleared either by channel DMA transfer done or reading CnSC while CHF is set and then writing a zero to CHF bit according to CHIE bit. See the following table.

Table 41-16. Clear CHF bit when DMA = 1

CHIE	How CHF Bit Can Be Cleared
0	CHF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHF is set and then writing a 0 to CHF bit.
1	CHF bit is cleared when the channel DMA transfer is done.

41.5.26 Dual Edge Capture mode

The Dual Edge Capture mode is selected if DECAPEN = 1. This mode allows to measure a pulse width or period of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode when n is 0 or 2.

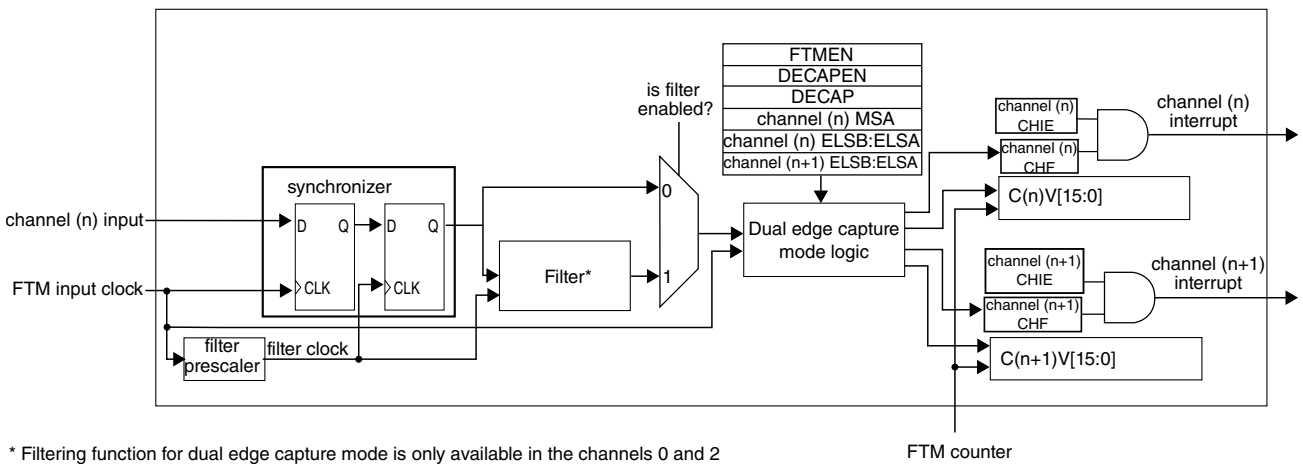


Figure 41-85. Dual Edge Capture mode block diagram

The channel (n) MSA bit defines if the Dual Edge Capture mode is one-shot or continuous.

The channel (n) ELSB:ELSA bits select the edge that is captured by channel (n), and channel (n+1) ELSB:ELSA bits select the edge that is captured by channel (n+1). If both channel (n) ELSB:ELSA and channel (n+1) ELSB:ELSA bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the Dual Edge Capture mode, only channel (n) input is used and channel (n+1) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then channel (n) CHF bit is set and the channel (n) interrupt is generated (if channel (n) CHIE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input and (channel (n) CHF = 1), then channel (n+1) CHF bit is set and the channel (n+1) interrupt is generated (if channel (n+1) CHIE = 1).

The C(n)V register stores the value of FTM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of FTM counter when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, a coherency mechanism ensures coherent data when the C(n)V and C(n +1)V registers are read. The only requirement is that C(n)V must be read before C(n +1)V.

Note

- The channel (n)n CHF, channel (n) CHIE, channel (n) MSA, channel (n) ELSB, and channel (n) ELSA bits are channel (n) bits.
- The channel (n+1) CHF, channel (n+1) CHIE, channel (n +1) MSA, channel (n+1) ELSB, and channel (n+1) ELSA bits are channel (n+1) bits.
- The Dual Edge Capture mode must be used with channel (n) ELSB:ELSA = 0:1 or 1:0, channel (n+1) ELSB:ELSA = 0:1 or 1:0 and the FTM counter in Free running counter.

41.5.26.1 One-Shot Capture mode

The One-Shot Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The channel (n) ELSB:ELSA bits select the first edge to be captured, and channel (n+1) ELSB:ELSA bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in One-Shot Capture mode, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

41.5.26.2 Continuous Capture mode

The Continuous Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The channel (n) ELSB:ELSA bits select the initial edge to be captured, and channel (n+1) ELSB:ELSA bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the channel (n+1) CHF bit. Therefore, when the channel (n+1) CHF bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

For a new sequence of the measurements in the Dual Edge Capture – Continuous mode, clear the channel (n) CHF and channel (n+1) CHF bits to start new measurements.

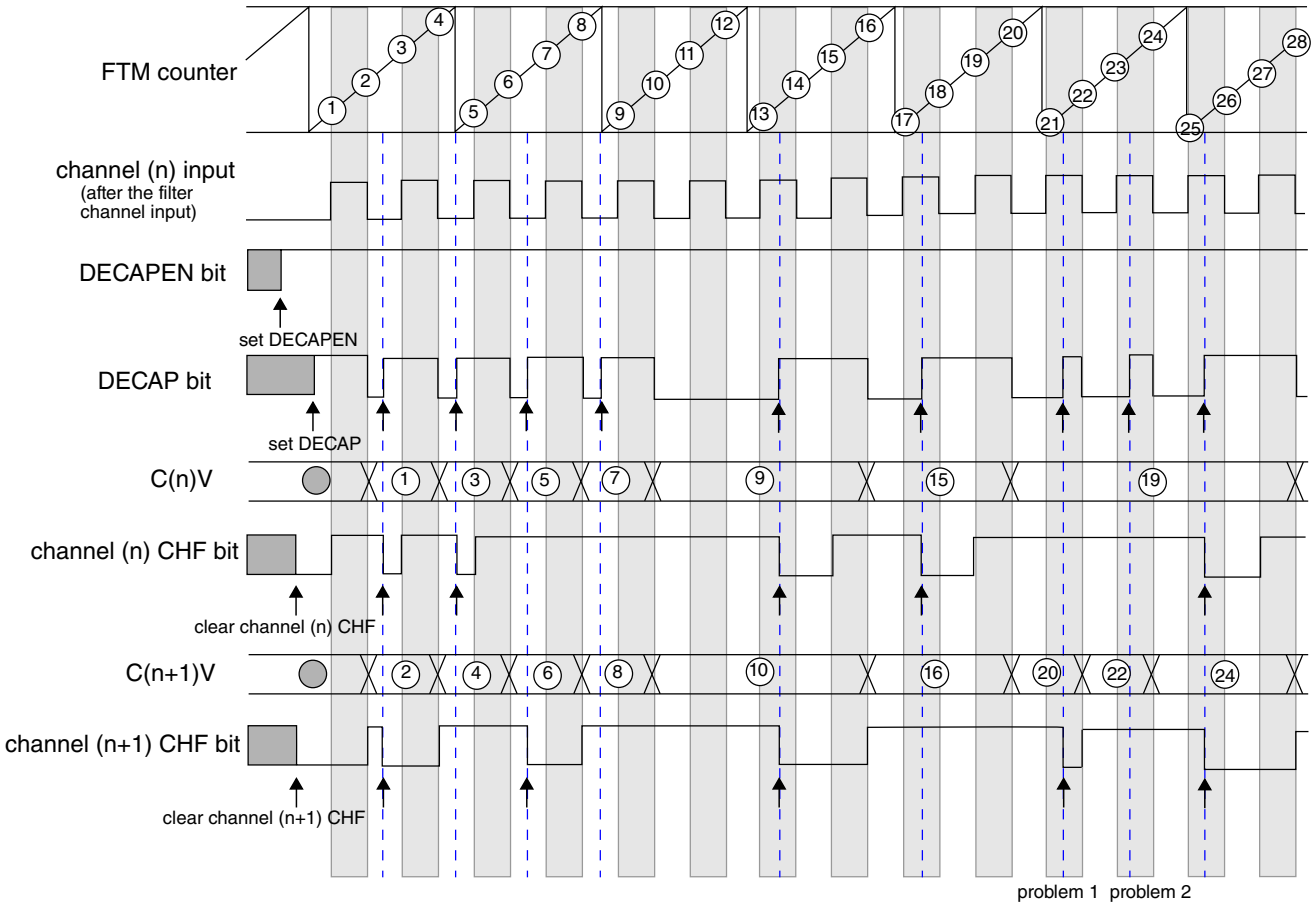
41.5.26.3 Pulse width measurement

If the channel (n) is configured to capture rising edges (channel (n) ELSB:ELSA = 0:1) and the channel (n+1) to capture falling edges (channel (n+1) ELSB:ELSA = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (channel (n) ELSB:ELSA = 1:0) and the channel (n+1) to capture rising edges (channel (n+1) ELSB:ELSA = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in One-Shot Capture mode or Continuous Capture mode.

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next positive polarity pulse width. The channel (n) CHF bit is set when the first edge of this pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second edge of this pulse is

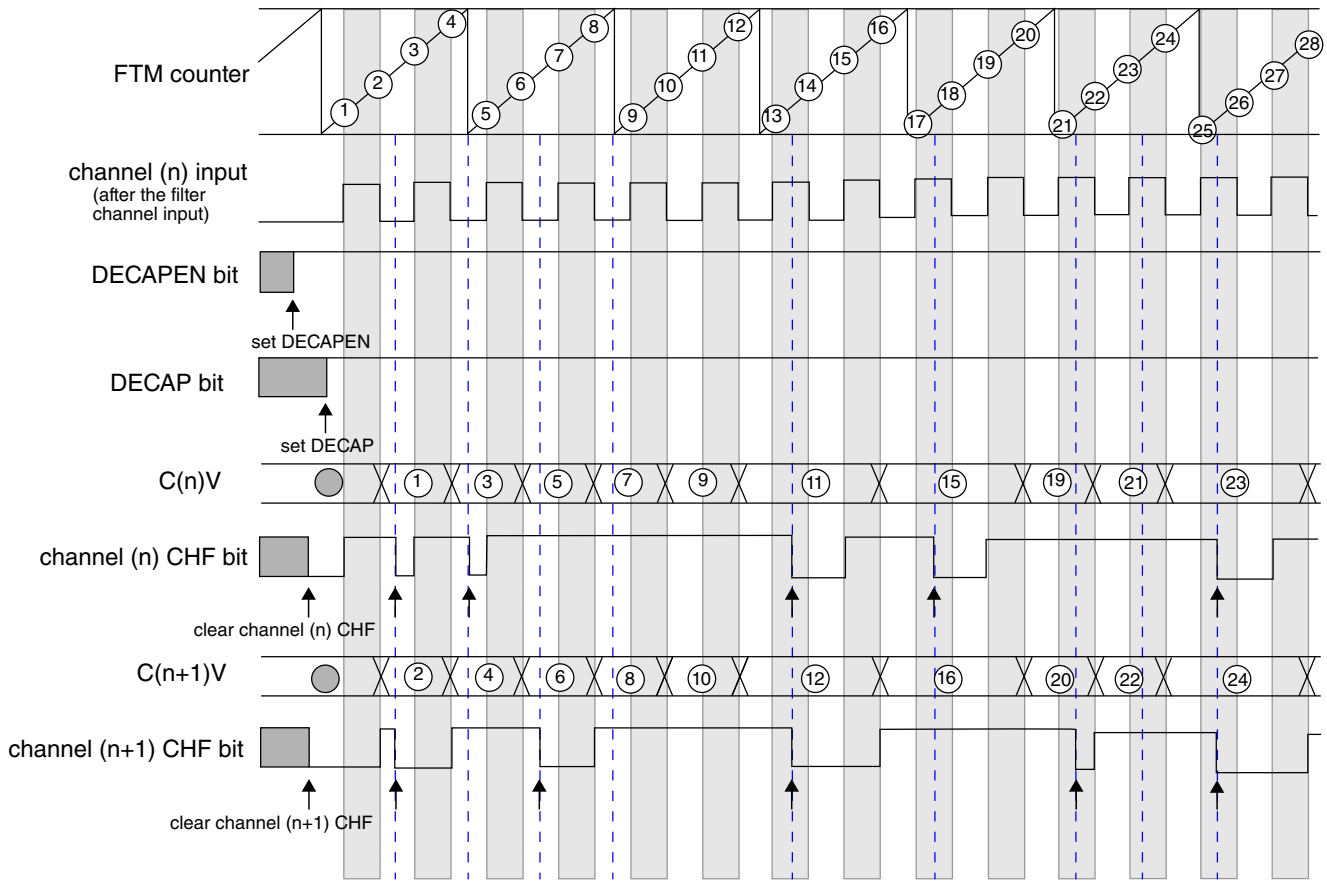
detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.
- Problem 1: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and clear channel (n+1) CHF.
- Problem 2: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.

Figure 41-86. Dual Edge Capture – One-Shot mode for positive polarity pulse width measurement

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second edge of this pulse is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.

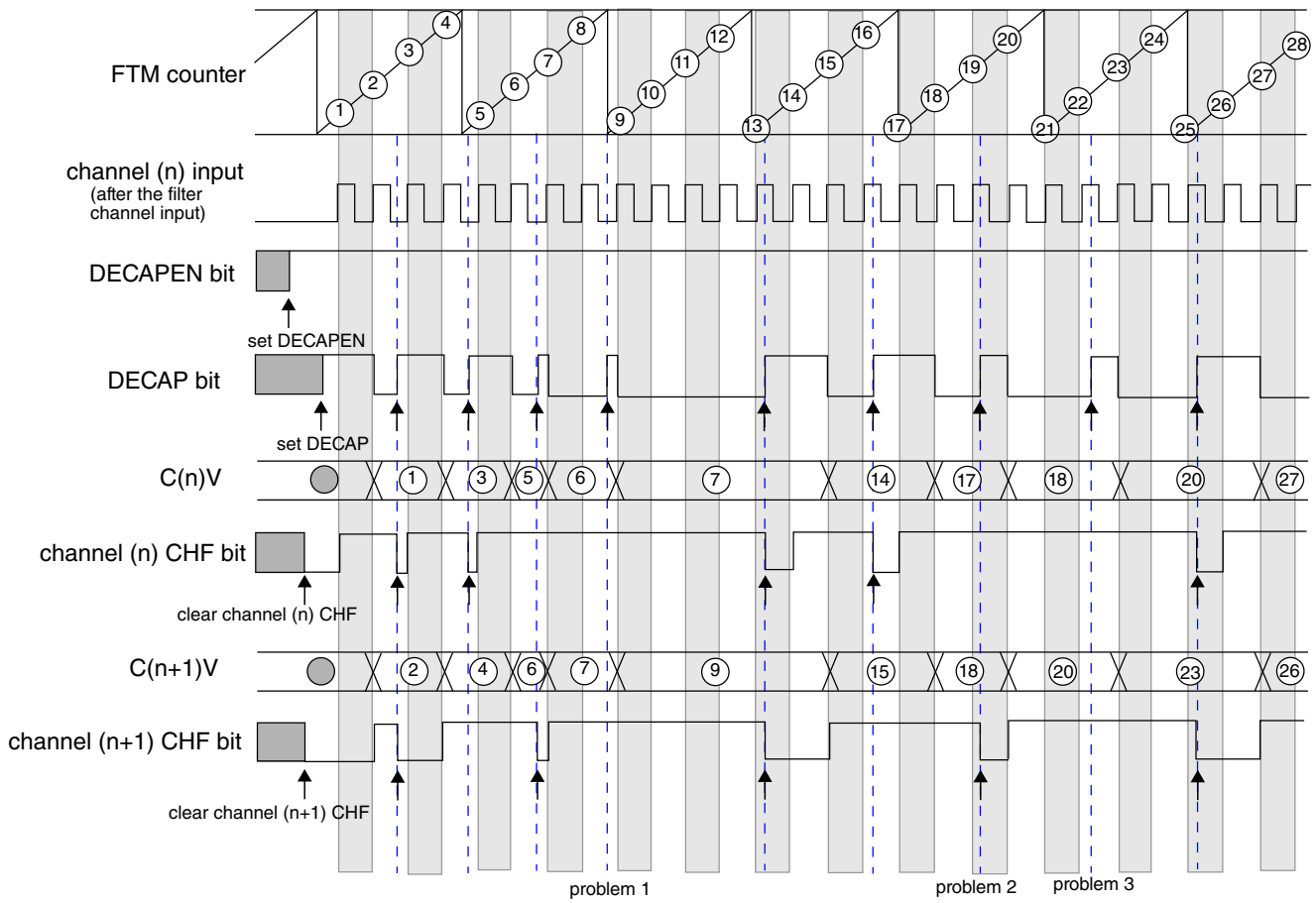
Figure 41-87. Dual Edge Capture – Continuous mode for positive polarity pulse width measurement

41.5.26.4 Period measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (channel (n) ELSB:ELSA = 0:1 and channel (n+1) ELSB:ELSA = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (channel (n) ELSB:ELSA = 1:0 and channel (n+1) ELSB:ELSA = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in One-Shot Capture mode or Continuous Capture mode.

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next period. The channel (n) CHF bit is set when the first rising edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.

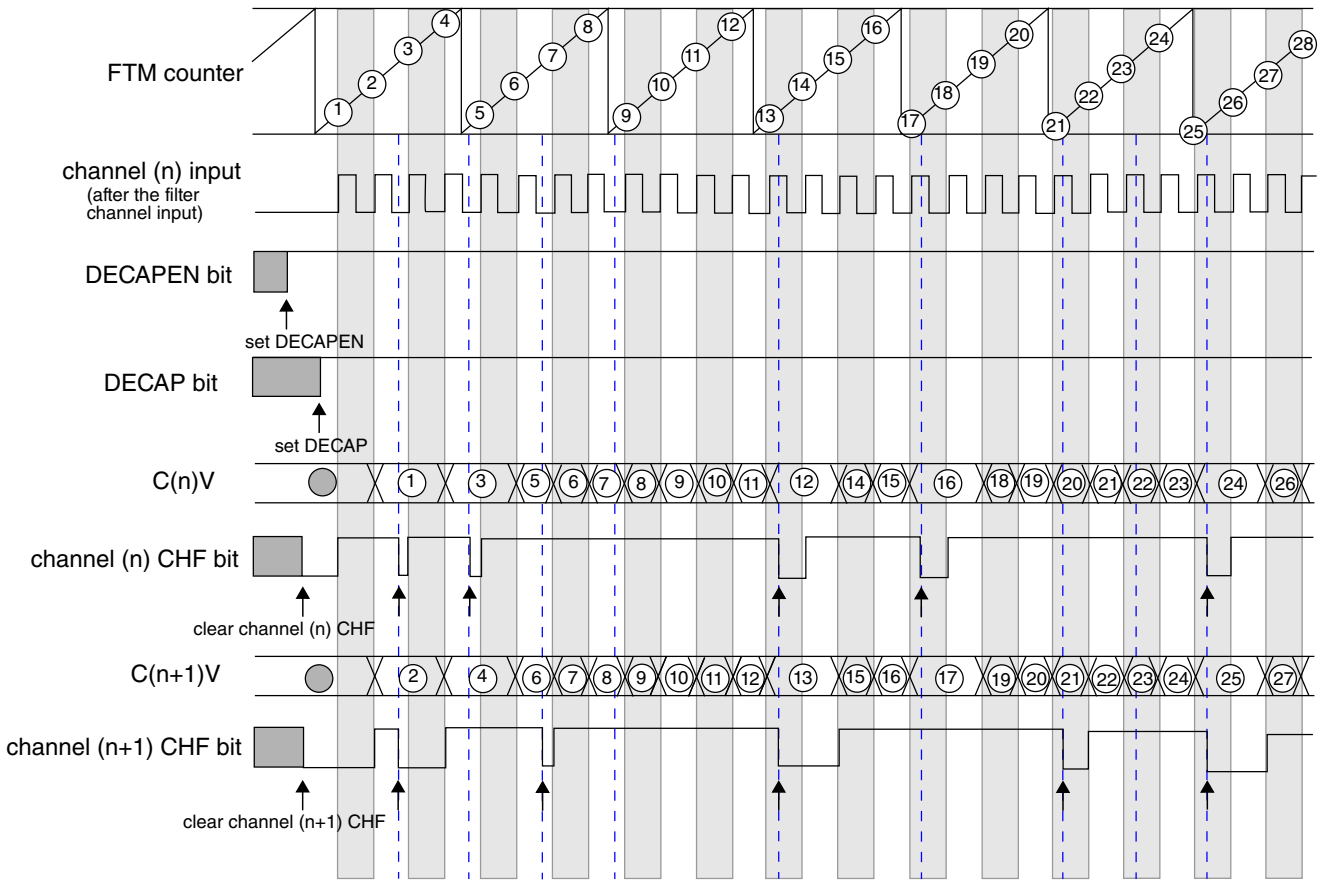


Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.
- Problem 2: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and clear channel (n+1) CHF.
- Problem 3: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.

Figure 41-88. Dual Edge Capture – One-Shot mode to measure of the period between two consecutive rising edges

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first rising

edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.

Figure 41-89. Dual Edge Capture – Continuous mode to measure of the period between two consecutive rising edges

41.5.26.5 Read coherency mechanism

The Dual Edge Capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n +1) are in Dual Edge Capture – Continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal. C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

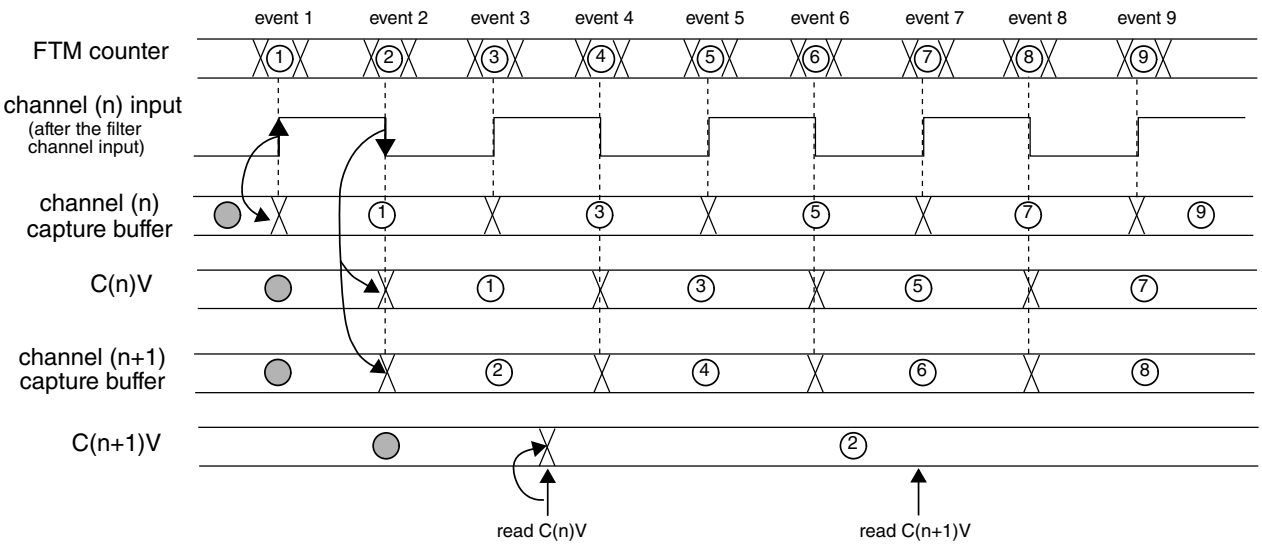


Figure 41-90. Dual Edge Capture mode read coherency mechanism

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

41.5.27 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

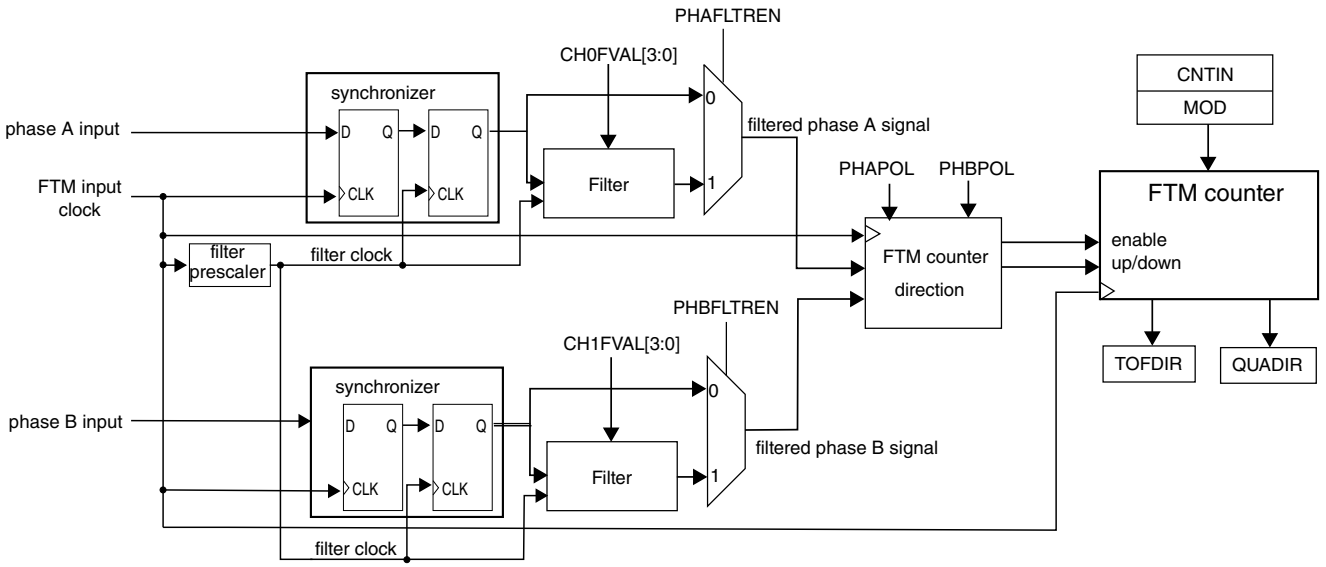


Figure 41-91. Quadrature Decoder block diagram

Each one of input signals phase A and B has a filter that is equivalent to the filter used in the channels input; Filter for Input Capture mode. The phase A input filter is enabled by PHAFLTREN bit and this filter's value is defined by CH0FVAL[3:0] bits (CH(n)FVAL[3:0] bits in FILTER0 register). The phase B input filter is enabled by PHBFLTREN bit and this filter's value is defined by CH1FVAL[3:0] bits (CH(n +1)FVAL[3:0] bits in FILTER0 register). The FLTPS[3:0] bits controls both filter prescalers.

Except for CH0FVAL[3:0] and CH1FVAL[3:0] bits, no channel logic is used in Quadrature Decoder mode.

Note

Notice that the FTM counter is clocked by the phase A and B input signals when quadrature decoder mode is selected. Therefore it is expected that the Quadrature Decoder be used only with the FTM channels in input capture or output compare modes.

Note

An edge at phase A must not occur together an edge at phase B and vice-versa.

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the phase B input value indicates the counting direction, and the phase A input defines the counting rate. The FTM counter is updated when there is a rising edge at phase A input signal.

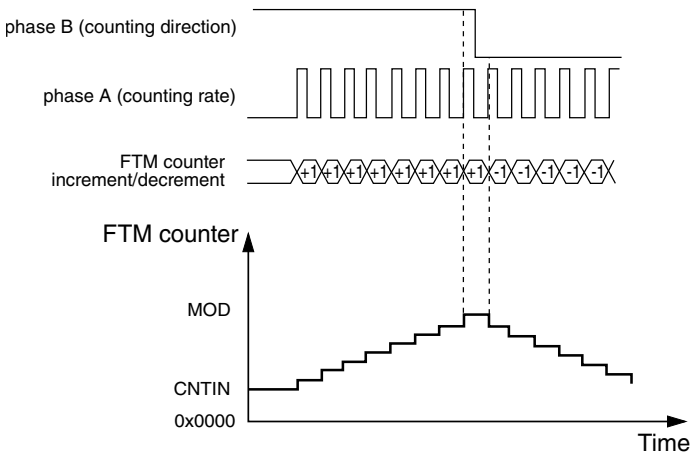


Figure 41-92. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the Phase A and Phase B Encoding mode is enabled; see the following figure. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate. The FTM counter is updated when there is an edge either at the phase A or phase B signals.

If PHAPOL = 0 and PHBPOL = 0, then the FTM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one;

and the FTM counter decrement happens when:

- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

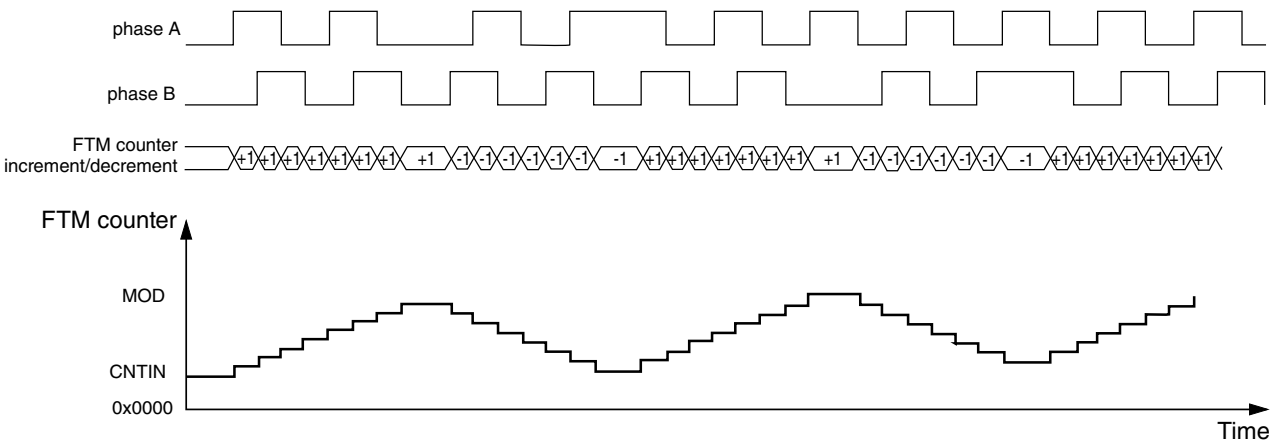


Figure 41-93. Quadrature Decoder – Phase A and Phase B Encoding mode

The following figure shows the FTM counter overflow in up counting. In this case, when the FTM counter changes from MOD to CNTIN, TOF and TOFDIR bits are set. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was up when the FTM counter overflow occurred.

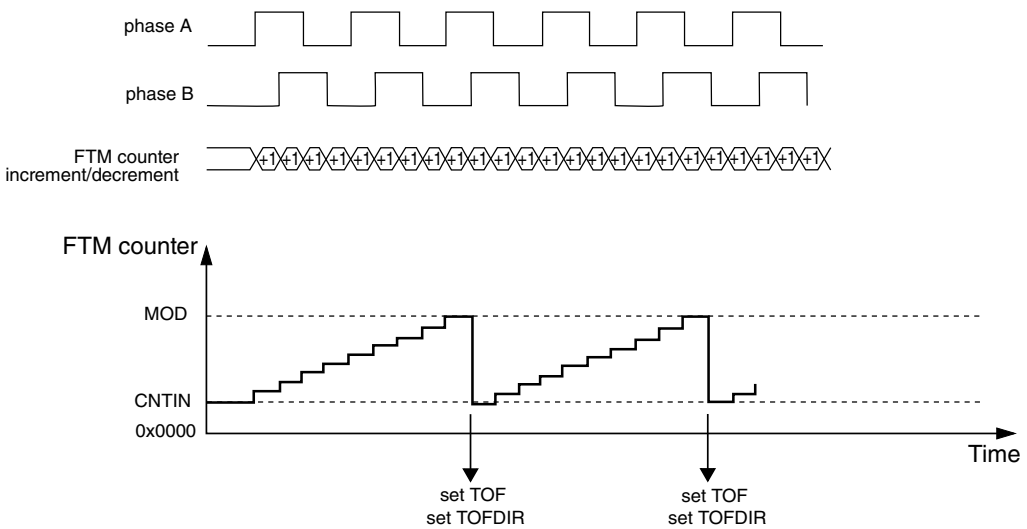


Figure 41-94. FTM Counter overflow in up counting for Quadrature Decoder mode

The following figure shows the FTM counter overflow in down counting. In this case, when the FTM counter changes from CNTIN to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was down when the FTM counter overflow occurred.

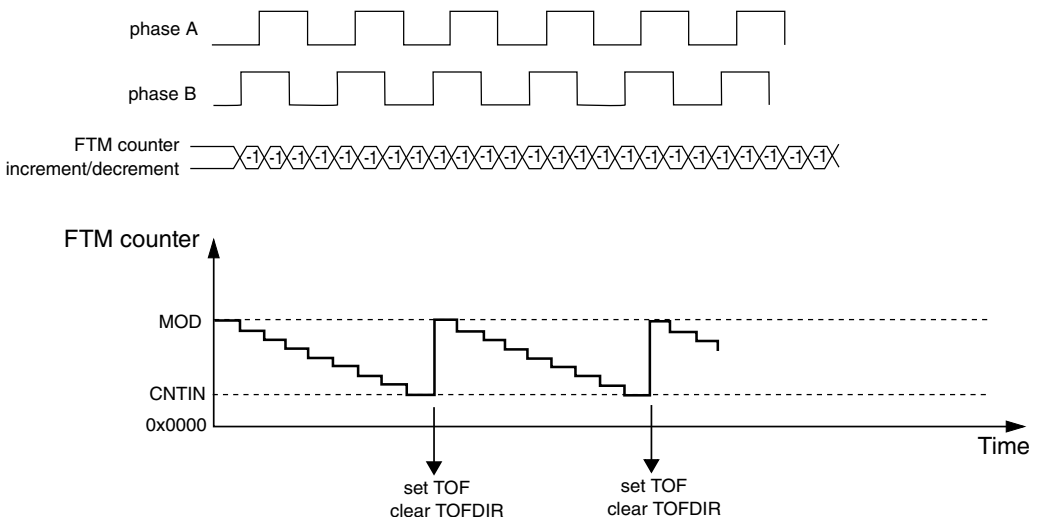


Figure 41-95. FTM counter overflow in down counting for Quadrature Decoder mode

41.5.27.1 Quadrature Decoder boundary conditions

The following figures show the FTM counter responding to motor jittering typical in motor position control applications.

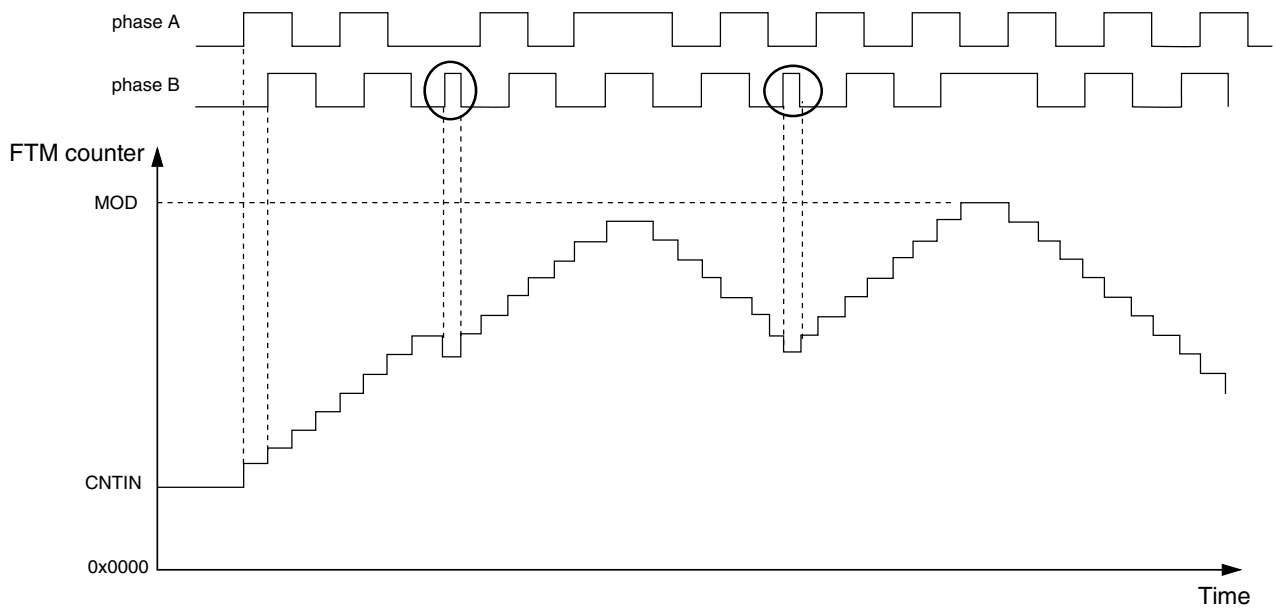


Figure 41-96. Motor position jittering in a mid count value

The following figure shows motor jittering produced by the phase B and A pulses respectively:

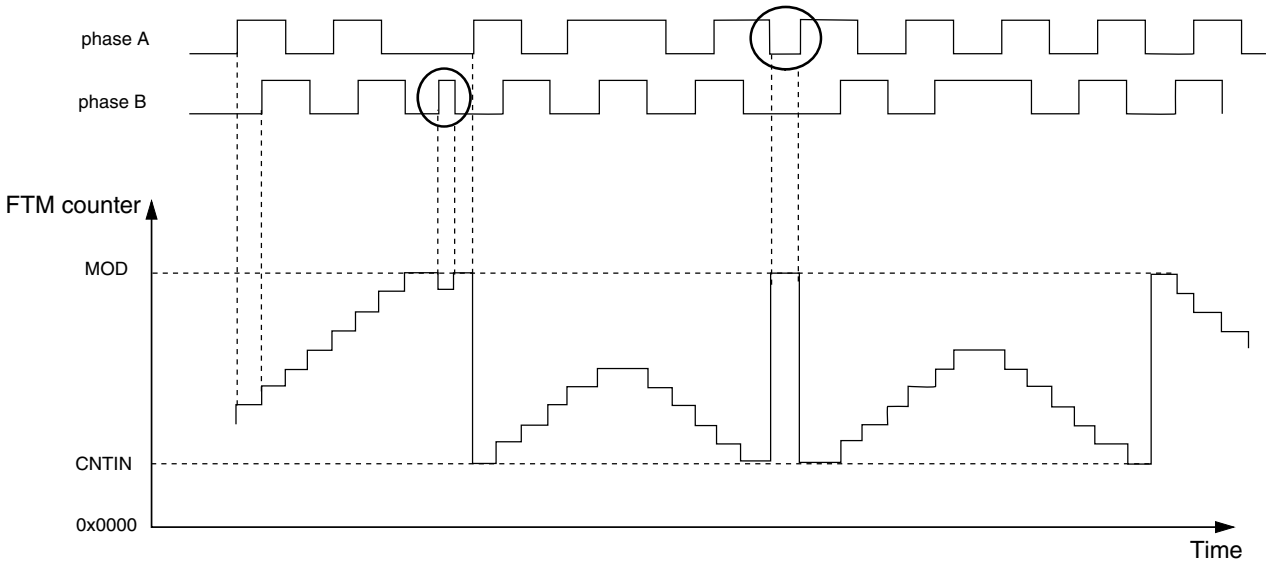


Figure 41-97. Motor position jittering near maximum and minimum count value

The first highlighted transition causes a jitter on the FTM counter value near the maximum count value (MOD). The second indicated transition occurs on phase A and causes the FTM counter transition between the maximum and minimum count values which are defined by MOD and CNTIN registers.

The appropriate settings of the phase A and phase B input filters are important to avoid glitches that may cause oscillation on the FTM counter value. The preceding figures show examples of oscillations that can be caused by poor input filter setup. Thus, it is important to guarantee a minimum pulse width to avoid these oscillations.

41.5.28 Debug mode

When the chip is in Debug mode, the BDMMODE[1:0] bits select the behavior of the FTM counter, the channel (n) CHF bit, the channels output, and the writes to the MOD, CNTIN, and C(n)V registers according to the following table.

Table 41-17. FTM behavior when the chip Is in Debug mode

BDMMODE	FTM Counter	channel (n) CHF bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
00	Stopped	can be set	Functional mode	Writes to these registers bypass the registers buffers
01	Stopped	is not set	The channels outputs are forced to their safe value according to POLn bit	Writes to these registers bypass the registers buffers
10	Stopped	is not set	The channels outputs are frozen when the chip enters in Debug mode	Writes to these registers bypass the registers buffers

Table continues on the next page...

Table 41-17. FTM behavior when the chip Is in Debug mode (continued)

BDMMODE	FTM Counter	channel (n) CHF bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
11	Functional mode	can be set	Functional mode	Functional mode

Note that if BDMMODE[1:0] = 2’b00 then the channels outputs remain at the value when the chip enters in Debug mode, because the FTM counter is stopped. However, the following situations modify the channels outputs in this Debug mode.

- Write any value to CNT register; see Counter reset. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for those channels set to Output Compare mode.
- FTM counter is reset by PWM Synchronization mode; see FTM counter synchronization. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for channels in Output Compare mode.
- In the channels outputs initialization, the channel (n) output is forced to the CH(n)OI bit value when the value 1 is written to INIT bit. See Initialization.

Note

The BDMMODE[1:0] = 2’b00 must not be used with the Fault control. Even if the fault control is enabled and a fault condition exists, the channels outputs values are updated as above.

Note

If CLKS[1:0] = 2'b00 in BDM, a non-zero value is written to CLKS in BDM, and CnV = CNTIN when the BDM is disabled, then the CHF bit is set (since if the channel is a 0% EPWM signal) when the BDM is disabled.

41.5.29 Reload Points

The reload points are points where the registers MOD, CNTIN, C(n)V and HCR can be updated with their write buffer value.

There are multiple reload opportunities. Each reload opportunity can turn into a reload point or not according to the load frequency. For example, if the load frequency is zero, then any reload opportunity is also a reload point. Note that when a reload point is

reached, a register reload will only occur if LDOK bit is enabled. The reload flag (RF) and initialization trigger generation are independent of LDOK bit. The table below shows which are the reload opportunities.

Table 41-18. When possible reload opportunities are enabled

Loading point	Enabled
When a counter event happens. See Counter events.	Always
At the channel (j) match (FTM counter = C(j)V)	When CHjSEL = 1
At the Half cycle event match (FTM counter = HCR)	When HCSEL = 1

The figure below shows a simplified representation of the reload logic. The Reload Flag (RF) can be used to generate an external interrupt when a load point is reached. It is also possible to generate an initialization trigger and a register reload when a load point is reached. Note that Load Frequency configuration can modify the RF generation.

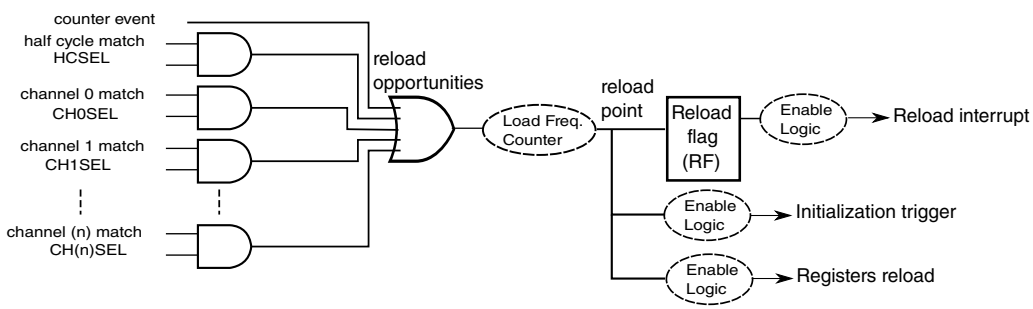


Figure 41-98. Registers reload logic

The following figure shows some examples of enabled reload opportunities when counter is in up counting mode. Note that the example below also uses a channel match as reload opportunity, but generally applications uses only the half cycle match if a non full cycle reload is needed.

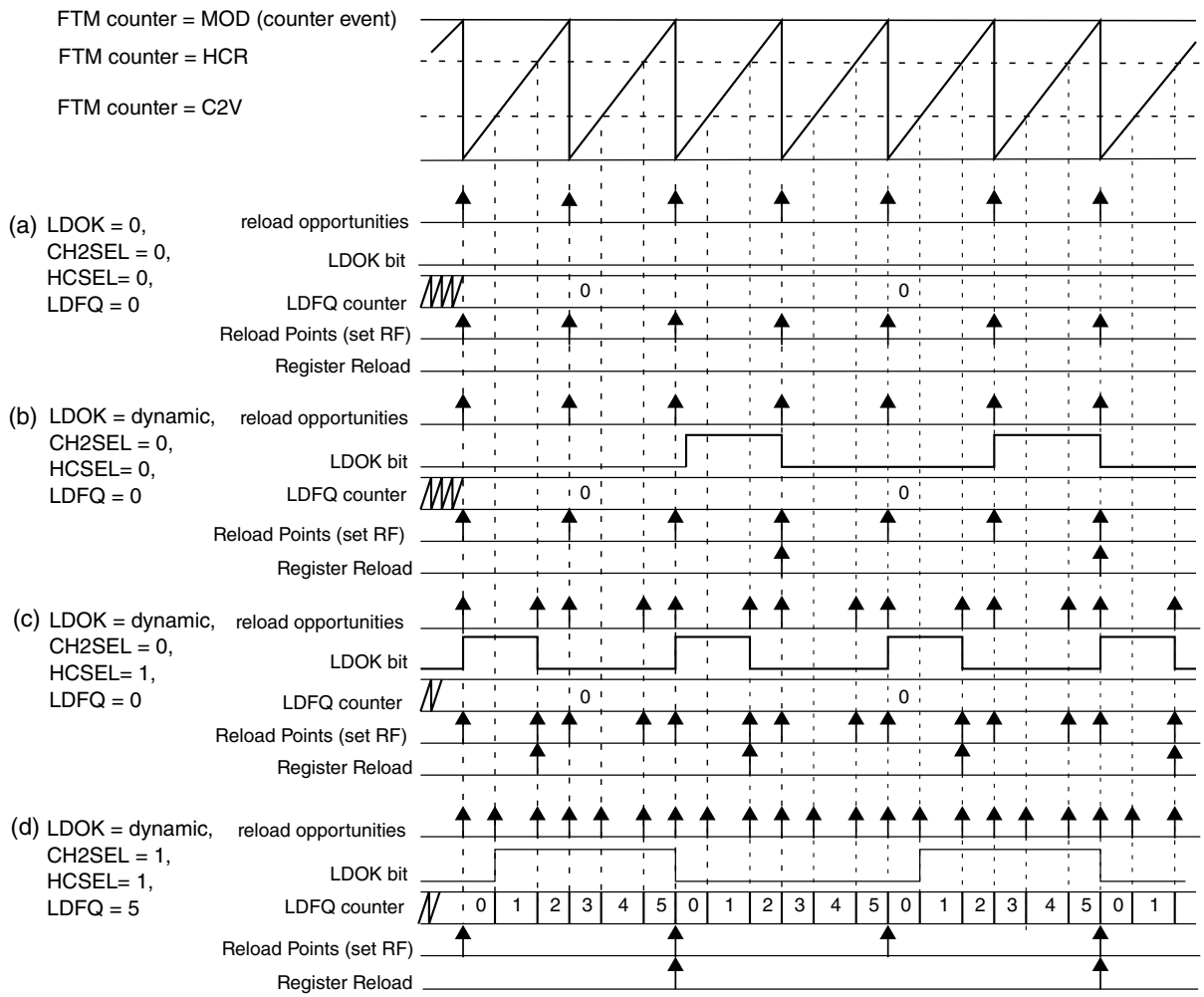


Figure 41-99. Reload opportunities to half and full cycle reload when up counting

The table below shows the possible counter events selection (reload opportunities) to up-down counting mode:

Table 41-19. Reload opportunities to up-down counting mode

FTM_SYNC bits	Reload opportunities selected
CNTMIN = 0 and CNTMAX = 0	When the counter turns from up to down (compatibility mode).
CNTMIN = 1 and CNTMAX = 0	When the counter turns from down to up.
CNTMIN = 0 and CNTMAX = 1	When the counter turns from up to down.
CNTMIN = 1 and CNTMAX = 1	When the counter turns from down to up and When the counter turns from up to down.

After enabling the reload opportunities, the LDOK bit must be set for the reload to occur. In this case, the reload occurs at the next enabled reload point considering the Load Frequency according to the following conditions:

Table 41-20. Conditions for reload occurring at the next enabled reload point

When a new value was written	Then
To the MOD register	The MOD register is updated with its write buffer value.
To the HCR register	The HCR register is updated with its write buffer value.
To the CNTIN register and CNTINC = 1	The CNTIN register is updated with its write buffer value.
To the C(n)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n)V register is updated with its write buffer value.
To the C(n+1)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n+1)V register is updated with its write buffer value.

NOTE

The reload points feature is independent of the PWM synchronization.

At these reload points neither the channels outputs nor the FTM counter are changed. Software must select these reload points at the safe points in time.

41.5.30 Global Load

The global load mechanism allows several modules to have their double buffered registers synchronously reloaded after a synchronization event if a write to one operation is performed in the global load OK (GLDOK) bit in the FTM_PWMLOAD register. Global load may be enabled or disabled configuring the global load enable (GLEN) bit in the FTM_PWMLOAD register. Writing one in the GLDOK bit with GLEN enabled has the same effect of writing one in the LDOK bit. Refer to SoC specific information about global load connections.

Global load mechanism allows MOD, HCR, CNTIN, and C(n)V registers to be updated with the content of the register buffer at configurable reload point. The figure below shows an example of connection between FTM global load inputs and outputs considering that GLDOK bit is implemented outside from FTM module.

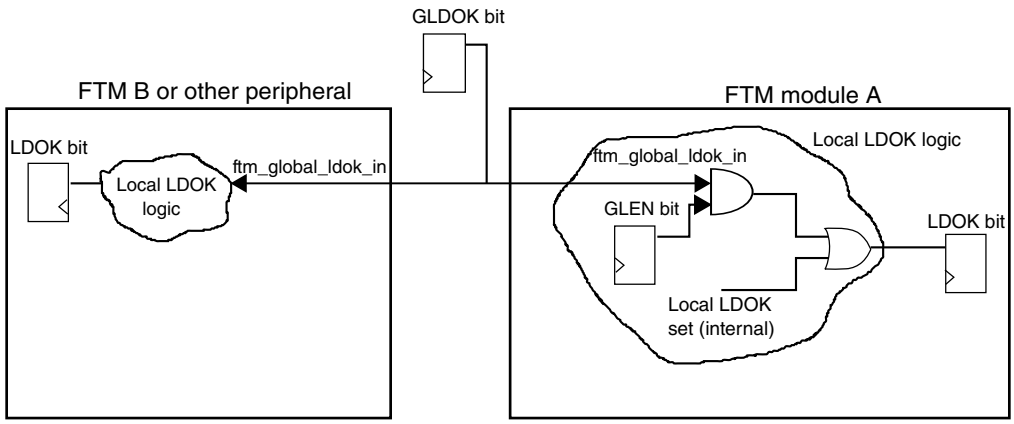


Figure 41-100. Global load logic

41.5.31 Global time base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.

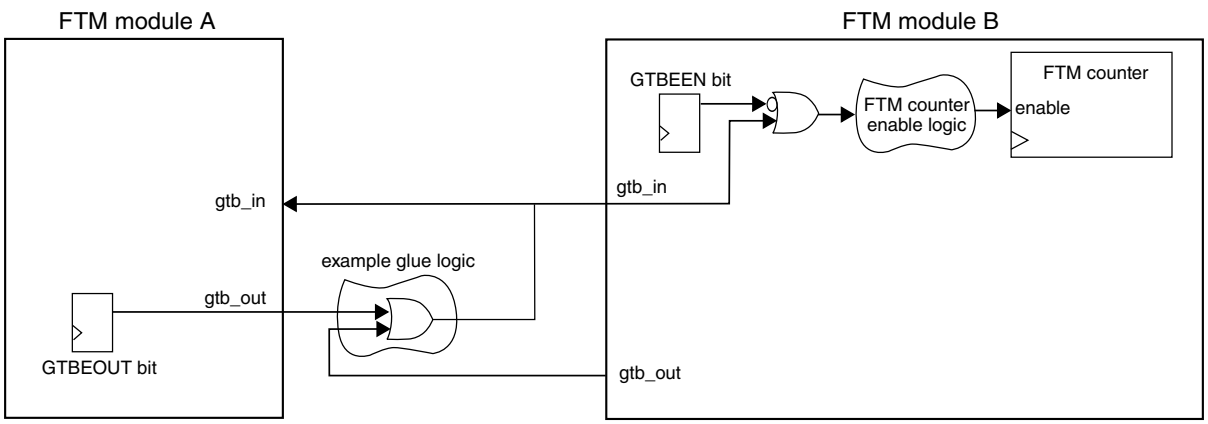


Figure 41-101. Global time base (GTB) block diagram

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the input signal *gtb_in*, and the output signal *gtb_out*. The GTBEEN bit enables gtb_in to control the FTM counter enable signal:

- If GTBEEN = 0, each one of FTM modules works independently according to their configured mode.
- If GTBEEN = 1, the FTM counter update is enabled only when gtb_in is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the gtb_out signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the gtb_in and

gtb_out signals, represented by the example glue logic shown in the figure. Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the chip-specific FTM information for the chip's specific implementation.

NOTE

- In order to use the GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation. The GTB feature only allows the FTM counters to *start* their operation synchronously.

41.5.31.1 Enabling the global time base (GTB)

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM to the intended configuration. The FTM counter mode needs to be consistent across all participating modules.
3. Write 1 to CONF[GTBEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. The clock source needs to be consistent across all participating modules.
5. Reset the FTM counter: Write any value to the CNT register.

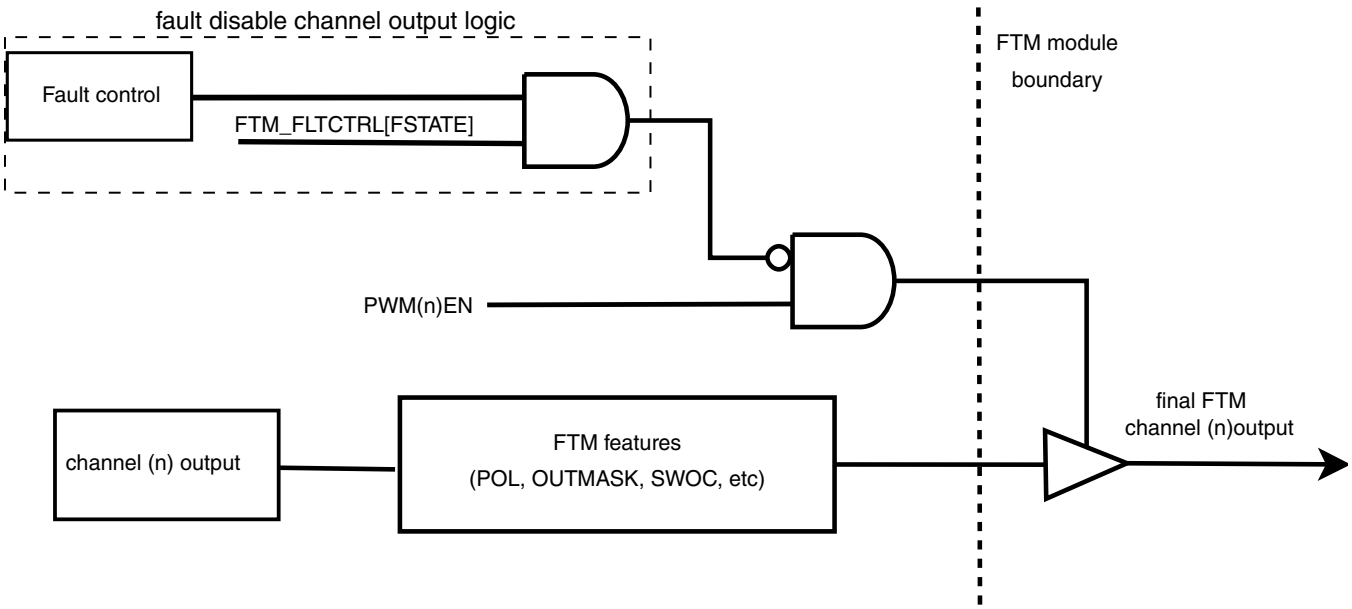
To initiate the GTB feature in the configuration described in the preceding figure, write 1 to CONF[GTBEOUT] in the FTM module used as the time base.

41.5.32 Output Logic

The following figure shows the output logic of each FTM channel output including how each PWM output has individual fault disabling and output enable. This allows flexibility regarding the external circuitry interface.

The output buffer logic depends on PWM_EN bit of FTM_SC register and fault disable channel output logic (see Fault control to more details) . Channel outputs will be enabled only if PWM_EN is enabled and there is no fault event ongoing configured to tri-state the outputs by FSTATE bit at FTM_FLTCTRL register. Note that Polarity logic will act before channel enable logic. Therefore, it is imperative that the user program the channel

polarities before enabling the output pins. A fault condition can result in the PWM output being tristated, forced to a logic 1, or forced to a logic 0 depending on the values programmed into the POL, and FSTATE fields.



41.5.33 Dithering

FTM implements a fractional delay to achieve fine resolution on the generated PWM signals using dithering. The dithering can be used by applications where more resolution than one unit of the FTM counter is needed.

Two kinds of dithering are available: PWM period dithering and edge dithering.

41.5.33.1 PWM Period Dithering

The PWM period dithering is enabled when a non-zero value is written to FRACMOD.

The internal accumulator used in the PWM period dithering is reset when:

- the field MOD of the register MOD_MIRROR is updated with the value of its write buffer,
- the FRACMOD is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

For the PWM period dithering, the register MOD_MIRROR should be used instead of the register MOD.

To avoid inconsistencies, the field FRACMOD is cleared when the field MOD of the register MOD is updated with the value of its write buffer.

The PWM period dithering is not available:

- when the FTM counter is a free running counter,
- when the FTM is in quadrature decoder mode.

41.5.33.1.1 Up Counting

When the FTM counter is an up counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period, and the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0xFFFE for PWM period dithering with unsigned counting and 0x7FFE for PWM period dithering with signed counting.

The figure belows an examples of PWM period dithering when the FTM counter is an up counter.

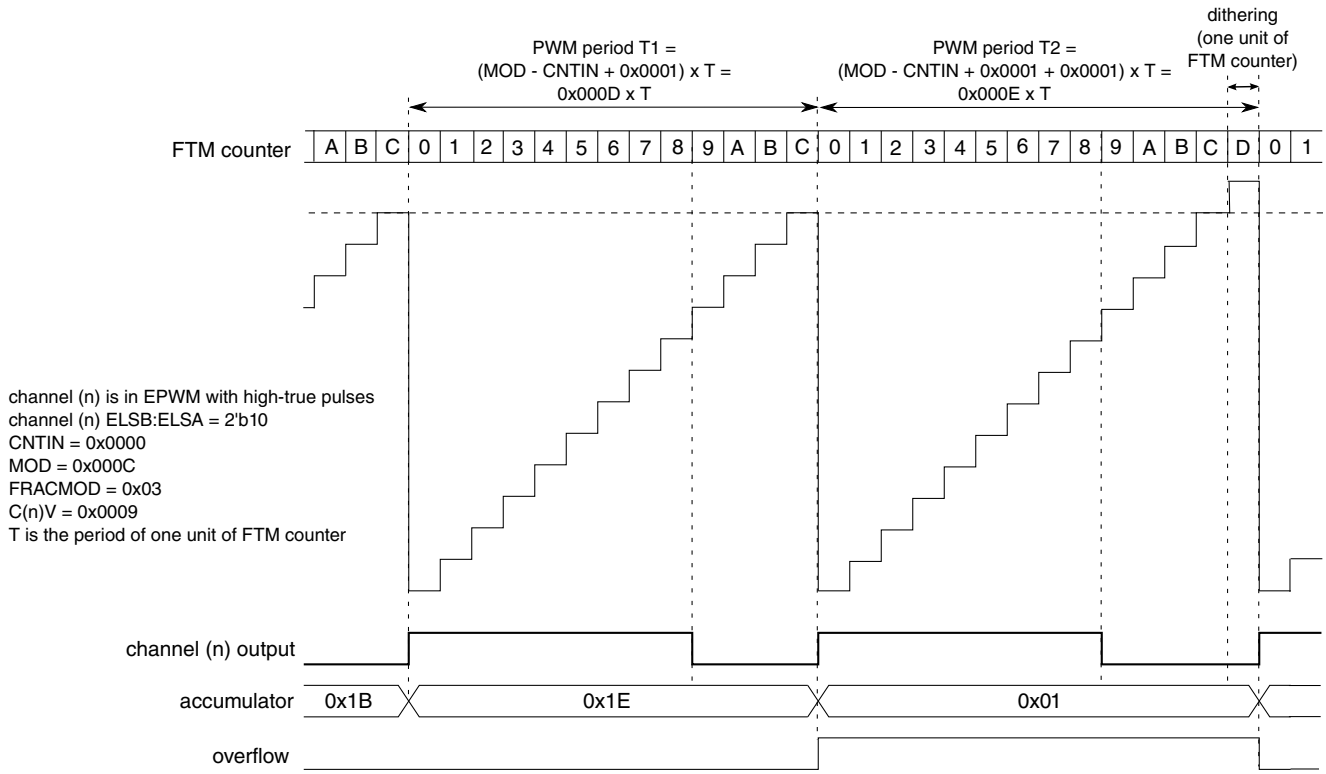


Figure 41-102. PWM Period Dithering with Up Counting

Assuming:

- the FTM counter is an up counter,
- T is one unit of FTM counter,
- the PWM period without period dithering is [(MOD - CNTIN + 1) x T],
- the number of PWM periods with period dithering is FRACMOD,
- the PWM period with period dithering is [(MOD - CNTIN + 1 + 1) x T],

thus, the average period (in decimal) is [(MOD - CNTIN + 1) + (FRACMOD/32)] x T, where the integer value is (MOD - CNTIN + 1) and the fractional value is (FRACMOD/32). See the example below.

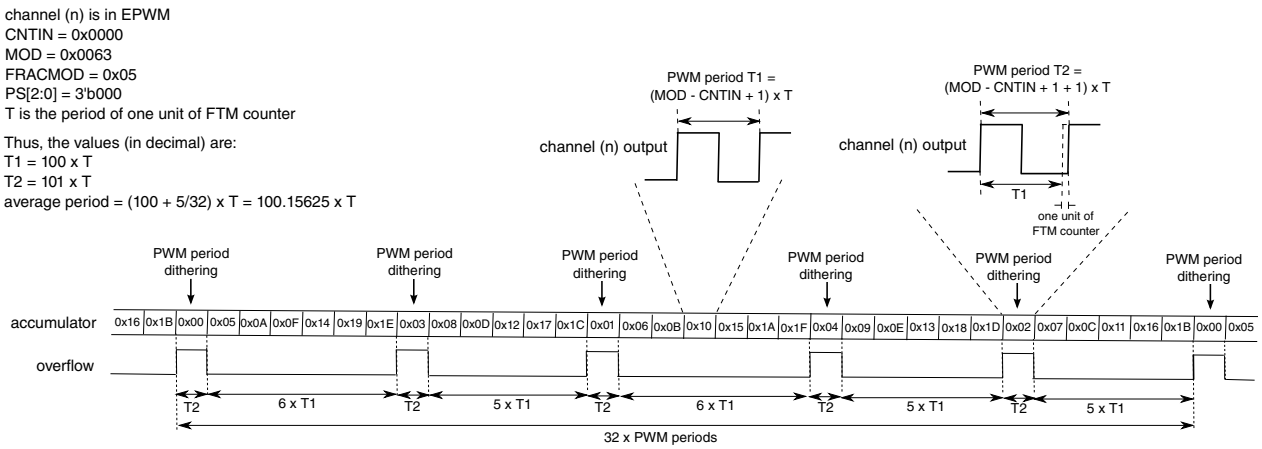


Figure 41-103. Example of Average Period when the PWM Period Dithering is used with the Up Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using EPWM mode and PWM Period Dithering, it is recommended to use (C(n) > MOD + 1).

For the generation of PWM signals in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Period Dithering, it is recommended to use:

- For 0% PWM signal: (C(n)V > MOD + 1) and (C(n+1)V > MOD + 1);
- For 100% PWM signal: (C(n)V = CNTIN) and (C(n+1)V > MOD + 1).

41.5.33.1.2 Up-Down Counting

When the FTM counter is an up-down counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period and other unit is added to the begin of the next PWM period (see the figure below). After the accumulator overflows, the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0x7FFE for PWM period dithering in up-down counting (CPWM mode).

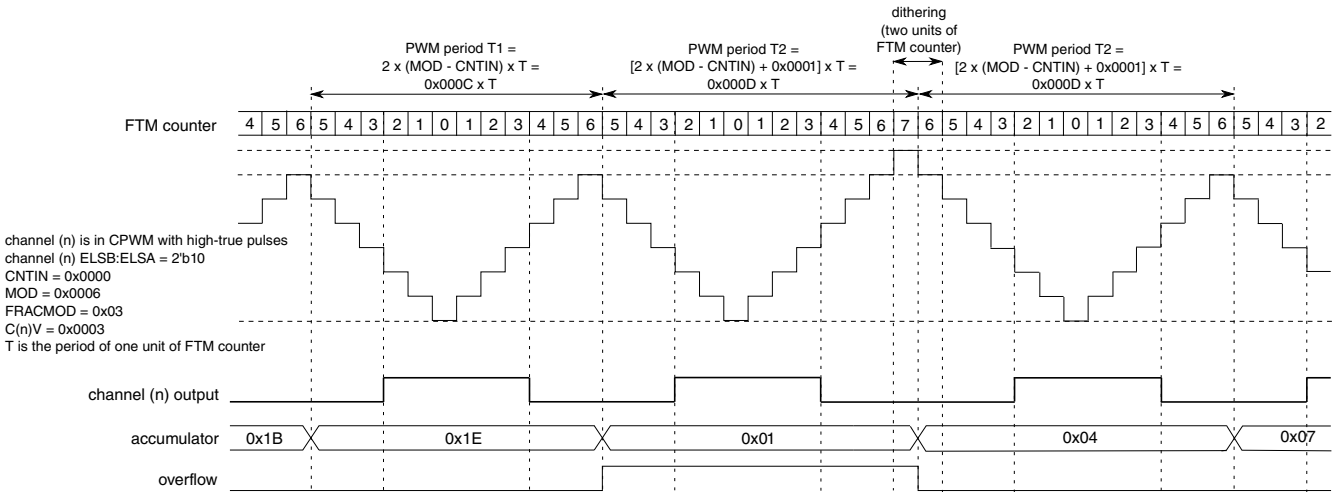


Figure 41-104. PWM Period Dithering with Up-Down Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using CPWM mode and PWM Period Dithering, it is recommended to use (C(n)V[15] = 0) and (C(n)V > MOD + 1) and (MOD ≠ 0x0000).

41.5.33.2 PWM Edge Dithering

The channel (n) internal accumulator used in the PWM edge dithering is reset when:

- the field VAL of the register C(n)V_MIRROR is updated with the value of its write buffer,
- the FRACVAL is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

For the PWM edge dithering, the register C(n)V_MIRROR should be used instead of the register C(n)V.

To avoid inconsistencies, the field FRACVAL is cleared when the field VAL of the register C(n)V is updated with the value of its write buffer.

The PWM edge dithering is not available:

- to the channel in input modes, and
- to the channel in output compare mode.

41.5.33.2.1 EPWM Mode

The PWM edge dithering for channel (n) in EPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in EPWM mode and the PWM edge dithering is enabled, at the end of each EPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, the initial edge of EPWM duty cycle happens when (FTM counter = CNTIN), its position is not modified by the PWM edge dithering. If there was not the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens on the channel (n) match (FTM counter = C(n)V), that is, its position is not modified by the edge dithering. However, if there was the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens when (FTM counter = C(n)V + 0x0001).

The figure below shows an example of PWM edge dithering when the channel (n) is in EPWM mode.

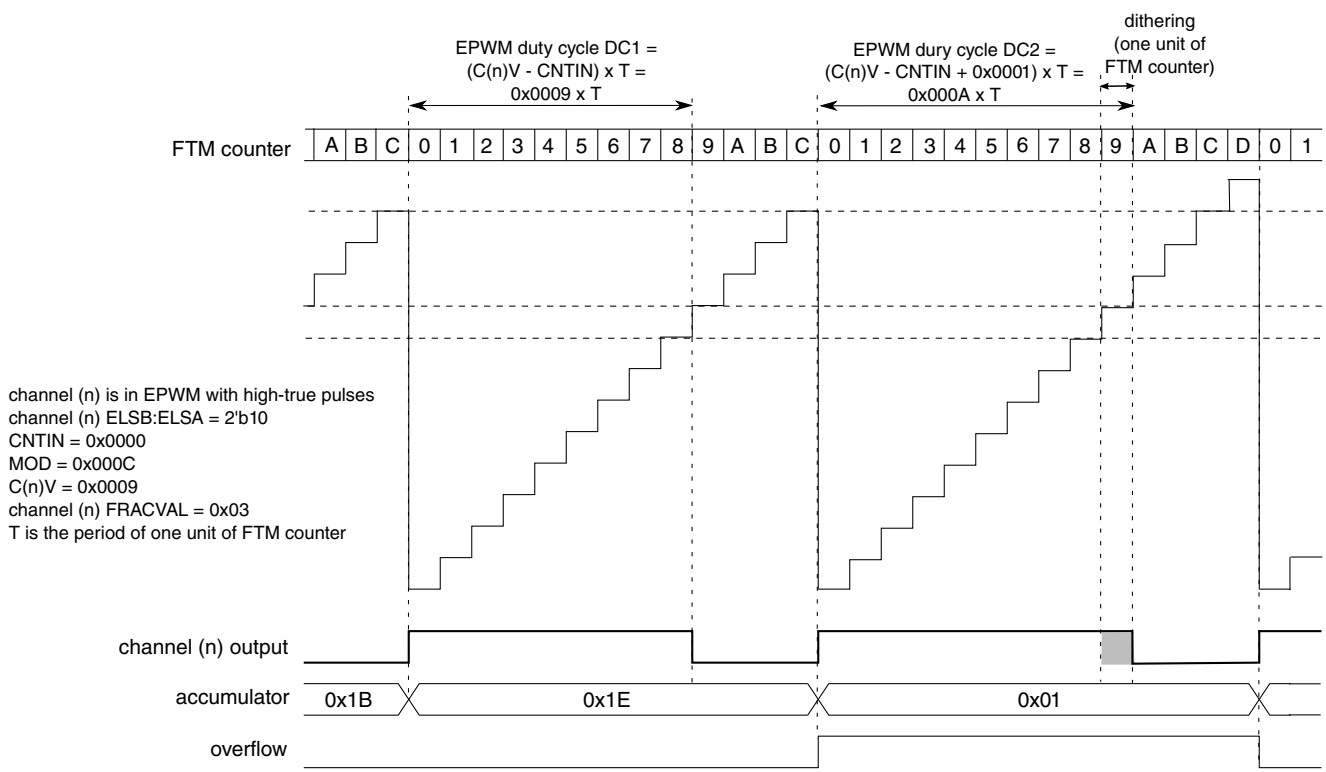


Figure 41-105. Channel (n) is in EPWM Mode with PWM Edge Dithering

Assuming:

- the channel (n) is in EPWM mode,
- T is one unit of FTM counter,
- the EPWM duty cycle without edge dithering is [(C(n)V - CNTIN) x T],
- the number of PWM periods which duty cycle that has edge dithering is FRACVAL,
- the EWM duty cycle with edge dithering is [(C(n)V - CNTIN + 1) x T],

thus, the average duty cycle (in decimal) is [(C(n)V - CNTIN) + (FRACVAL/32)] x T, where the integer value is (C(n)V - CNTIN) and the fractional value is (FRACVAL/32). See the example below.

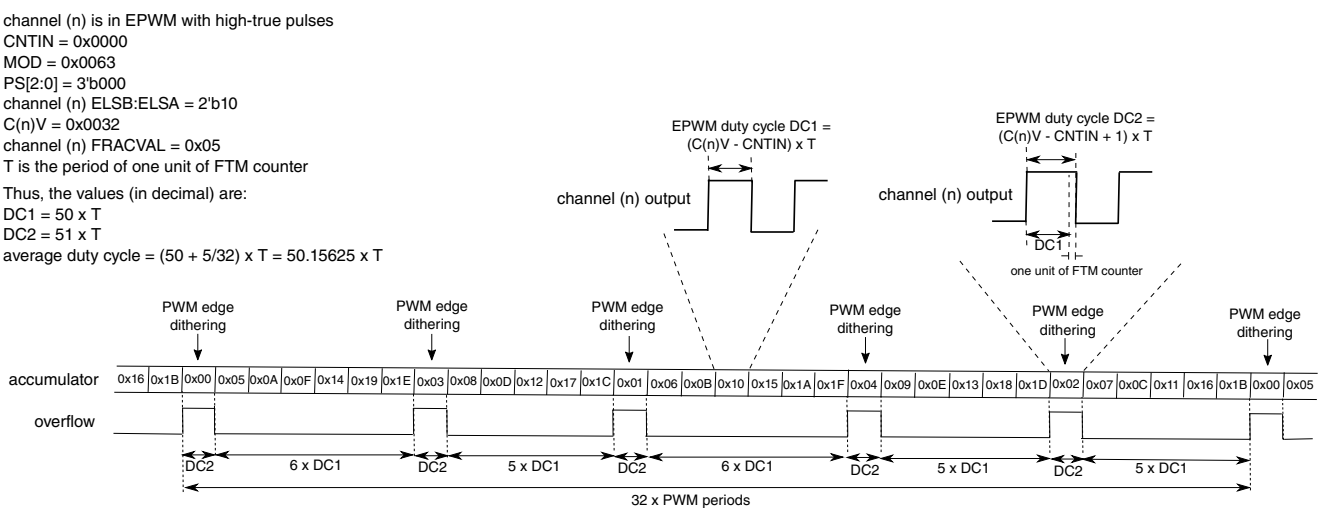


Figure 41-106. Example of Average Duty Cyle when the Channel (n) is in EPWM Mode with PWM Edge Dithering

41.5.33.2.2 CPWM Mode

The PWM edge dithering for channel (n) in CPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in CPWM mode and the PWM edge dithering is enabled, at the end of each CPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, if there was not the overflow of the channel (n) accumulator in the current CPWM period, then the duty cycle is not modified by the PWM edge dithering, that is, the initial edge of CPWM duty cycle happens on channel (n) match (FTM counter = C(n)V) when the FTM counter is decrementing, and the final edge of CPWM duty cycle on channel (n) match when the FTM counter is incrementing.

However, if there was the overflow of the channel (n) accumulator in the current CPWM period, then the initial edge of CPWM duty cycle happens when (FTM counter = C(n)V + 0x0001) and the FTM counter is decrementing, and the final edge of CPWM duty cycle when (FTM counter = C(n)V + 0x0001) and the FTM counter is incrementing.

The figure below shows an example of PWM edge dithering when the channel (n) is in CPWM mode.

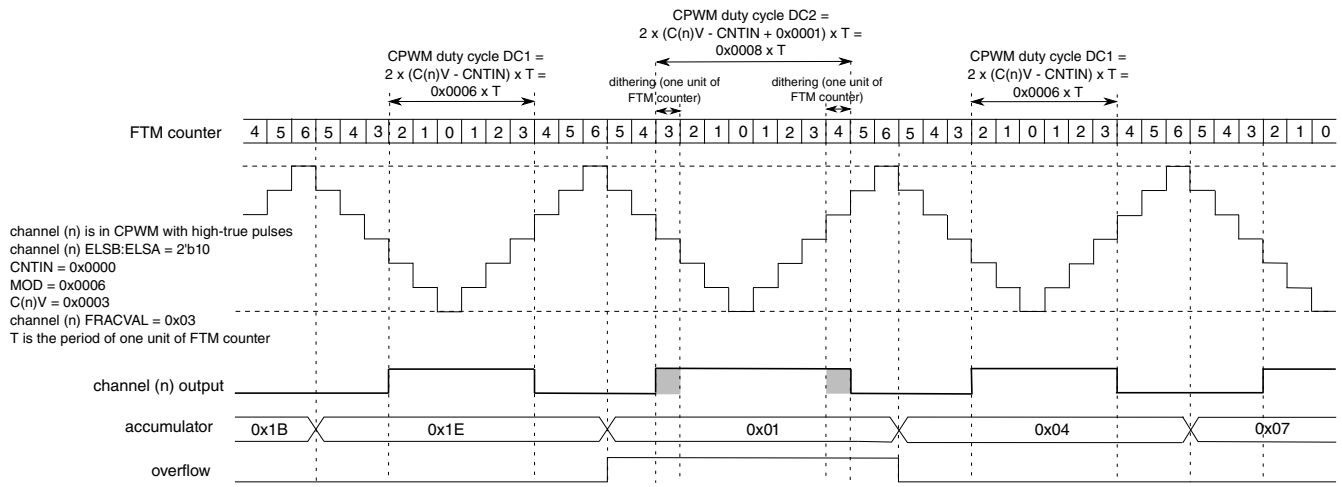


Figure 41-107. Channel (n) is in CPWM Mode with PWM Edge Dithering

41.5.33.2.3 Combine Mode

In the Combine mode, the PWM edge dithering can be done:

- in the channel (n) match (FTM counter = C(n)V) edge or
- in the channel (n+1) match (FTM counter = C(n+1)V edge.

The channel (n) match edge dithering is enabled when a non-zero value is written to the channel (n) FRACVAL.

For the channel (n) match edge dithering, the channel (n) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n) FRACVAL value is added to the channel (n) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n) accumulator in the current PWM period, the channel (n) match edge is not modified, that is, it happens on channel (n) match. However, if there was the overflow of the channel (n) accumulator, the channel (n) match edge happens when (FTM counter = C(n)V + 0x0001).

The figure below shows an example of the channel (n) match edge dithering when the channels (n) and (n+1) are in Combine mode.

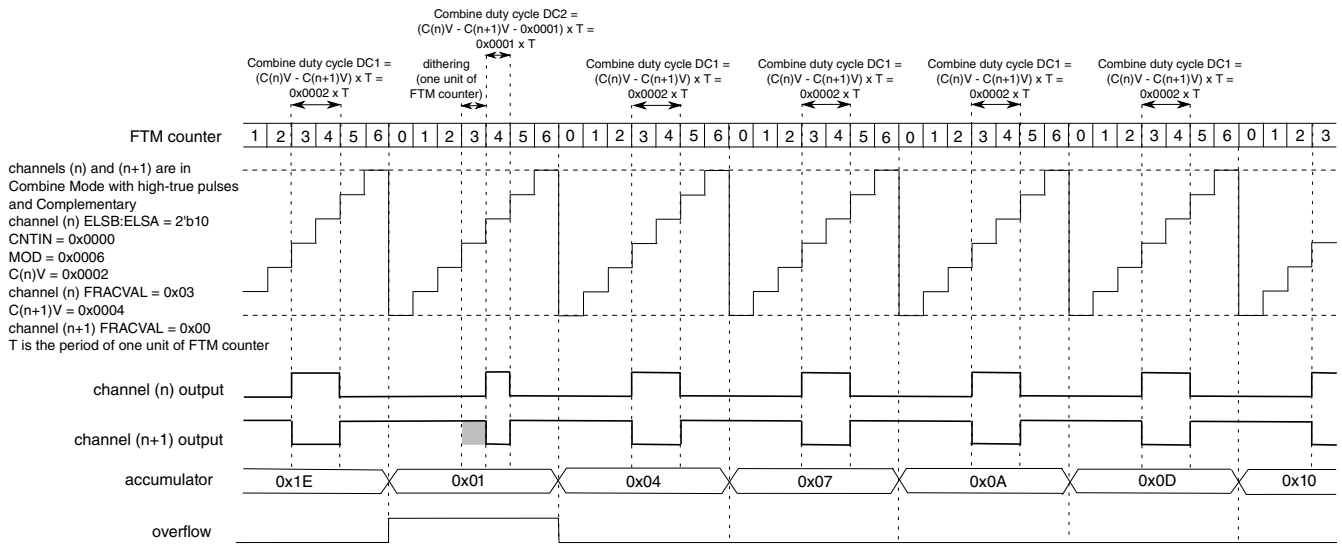


Figure 41-108. Channel (n) Match Edge Dithering in Combine Mode

The channel (n+1) match edge dithering is enabled when a non-zero value is written to the channel (n+1) FRACVAL.

For the channel (n+1) match edge dithering, the channel (n+1) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n+1) FRACVAL value is added to the channel (n+1) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n+1) accumulator in the current PWM period, the channel (n+1) match edge is not modified, that is, it happens on channel (n+1) match. However, if there was the overflow of the channel (n+1) accumulator, the channel (n+1) match edge happens when (FTM counter = C(n+1)V + 0x0001).

The figure below shows an example of the channel (n+1) match edge dithering when the channels (n) and (n+1) are in Combine mode.

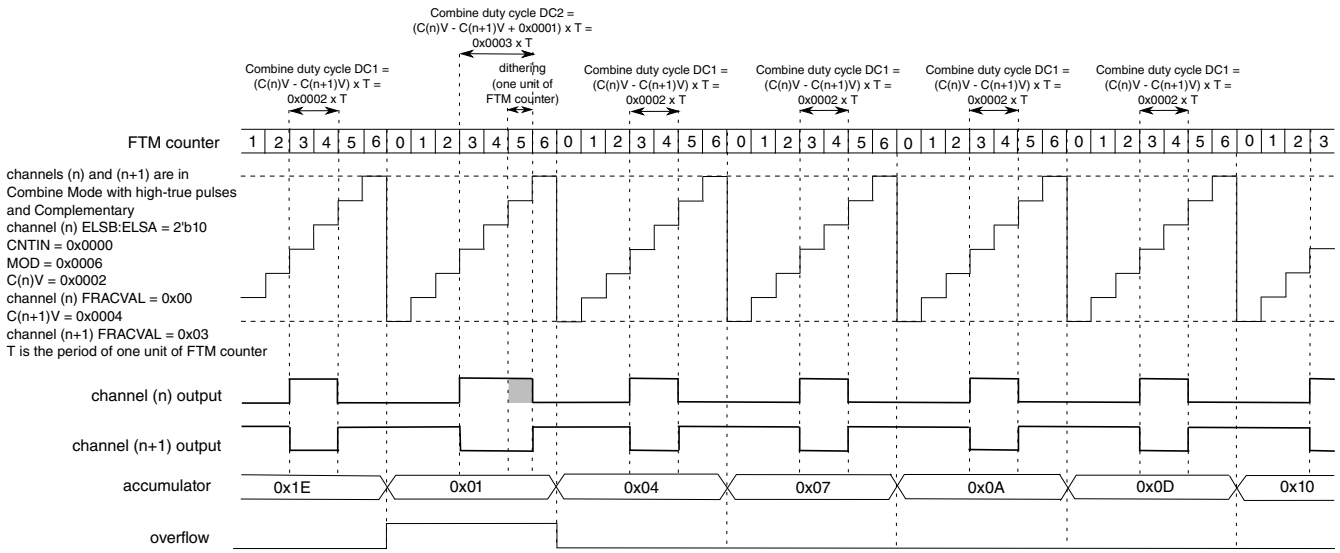


Figure 41-109. Channel (n+1) Match Edge Dithering in Combine Mode

NOTE

It is recommended to use only one PWM Edge Dithering (channel (n) PWM Edge Dithering or channel (n+1) PWM Edge Dithering) at a time.

For the generation of 0% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V < CNTIN or C(n)V > MOD) and (channel (n) FRACVAL is zero) and
- (channel (n+1) FRACVAL is zero).

For the generation of 100% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V = CNTIN) and (channel (n) FRACVAL is zero) and
- (C(n+1)V < CNTIN or C(n+1)V > MOD) and (channel (n +1) FRACVAL is zero).

41.6 Reset overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

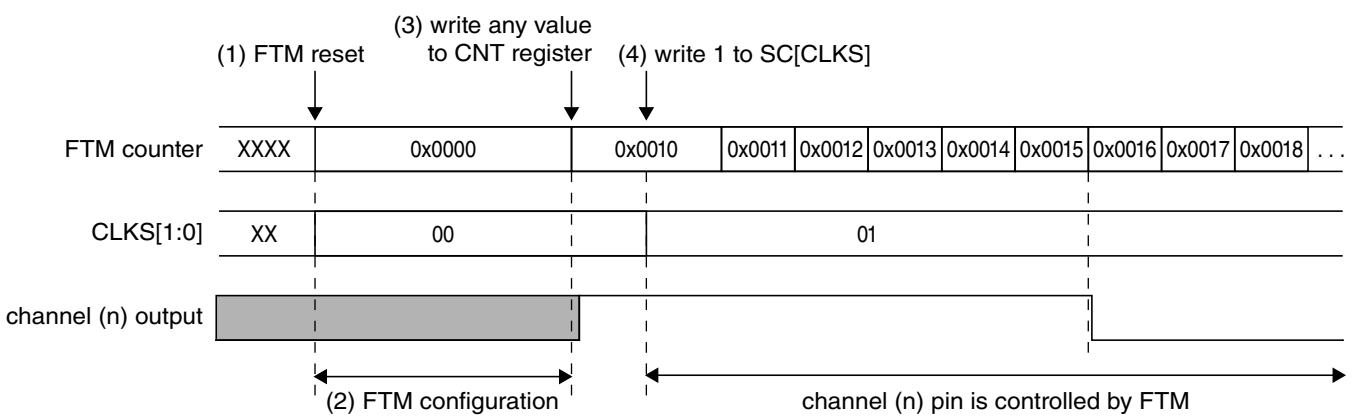
- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 00b);
- the timer overflow interrupt is zero, see Timer Overflow Interrupt;
- the channels interrupts are zero, see Channel (n) Interrupt;
- the fault interrupt is zero, see Fault Interrupt;
- the channels are in input capture mode, see Input Capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by FTM (channel (n) ELSB:ELSA = 0:0) (Channel Modes).

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (see the CLKS its in the register SC), its value is updated to zero and the pins are not controlled by FTM (Channel Modes).

After the reset, the FTM should be configurated (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) (Counter reset).

The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero (Channel Modes).



NOTES:
– CNTIN = 0x0010
– Channel (n) is in low-true combine mode with CNTIN < C(n)V < C(n+1)V < MOD
– C(n)V = 0x0015

Figure 41-110. FTM behavior after reset when the channel (n) is in Combine mode

The following figure shows an example when the channel (n) is in Output Compare mode and the channel (n) output is toggled when there is a match. In the Output Compare mode, the channel output is not updated to its initial value when there is a write to CNT

register (item 3). In this case, use the software output control (Software Output Control Mode) or the initialization (Initialization) to update the channel output to the selected value (item 4).

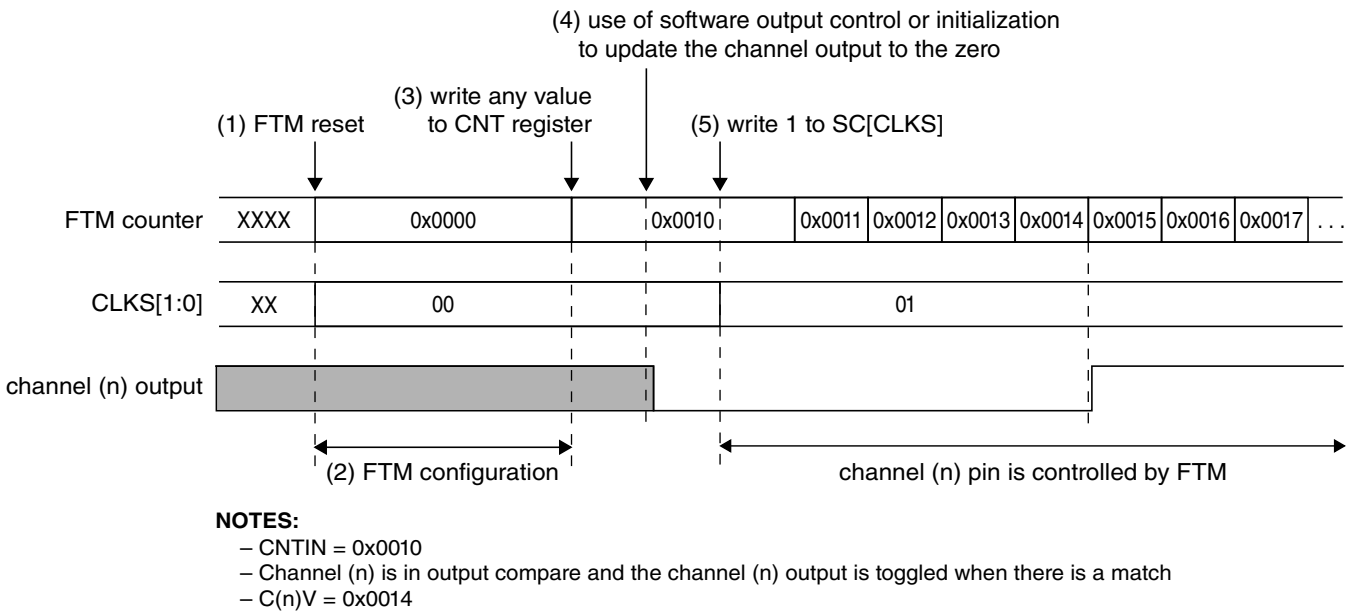


Figure 41-111. FTM behavior after reset when the channel (n) is in Output Compare mode

41.7 FTM Interrupts

41.7.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

41.7.2 Reload Point Interrupt

The Reload Point interrupt is generated when (RIE = 1) and (RF = 1).

41.7.3 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHIE = 1) and (CHF = 1).

41.7.4 Fault Interrupt

The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

41.8 Initialization Procedure

The following initialization procedure is recommended to configure the FlexTimer operation. This procedure can also be used to do a new configuration of the FlexTimer operation.

1. Define the POL bits.
2. Mask the channels outputs using SYNCHOM = 0. Two clocks after the write to OUTMASK, the channels output are in the safe value.
3. (Re)Configuration FTM counter and channels to generation of periodic signals:
   - Disable the clock. If the selected mode is Quadrature Decoder, then disable this mode.
   - Write to MOD.
   - Write to CNTIN.
   - Configure the channels that will be used.
   - Select the high-true and low-true channels modes.
   - Write to CnV for all channels that will be used .
   - (Re)Configure deadtime and fault control.
   - Do not use the SWOC without SW synchronization (see item 6).
   - Do not use the Inverting without SW synchronization (see item 6).
   - Do not use the Initialization.
   - Do not change the polarity control.
   - Do not configure the HW synchronization
4. Write any value to CNT. The FTM Counter is reset and the channels output are updated according to new configuration.
5. Enable the clock. Write to CLKS[1:0] bits a value different from zero. If in the Quadrature Decoder mode, enable this mode.
6. Configure the SW synchronization for SWOC (if it is necessary), Inverting (if it is necessary) and Output Mask (always)
   - Select synchronization for Output Mask Write to SYNC (SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
   - Write to SYNCONF.
     - HW Synchronization can not be enabled (HWSOC = 0, HWINVC = 0, HWOM = 0, HWWRBUF = 0, HWRSTCNT = 0, HWTRIGMODE = 0).
     - SW Synchronization for SWOC (if it is necessary): SWSOC = [0/1] and SWOC = [0/1].

- SW Synchronization for Inverting (if it is necessary): SWINVC = [0/1] and INVC = [0/1].
- SW Synchronization for SWOM (always): SWOM = 1. No enable the SW Synchronization for write buffers (because the writes to registers with write buffer are done using CLKS[1:0] = 2'b00): SWWRBUF = 0 and CNTINC = 0 .
- SW Synchronization for counter reset (always): SWRSTCNT = 1.
- Enhanced synchronization (always): SYNCMODE = 1.

- If the SWOC is used (SWSOC = 1 and SWOC = 1), then write to SWOCTRL register.
- If the Inverting is used (SWINVC = 1 and INVC = 1), then write to INVCTRL register.
- Write to OUTMASK to enable the masked channels.

7. Generate the Software Trigger Write to SYNC (SWSYNC = 1, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
8. Write to PWM_EN to enable the PWM outputs.

41.9 Usage Guide

41.9.1 FTM Interrupts

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

41.9.2 FTM Hall sensor support

For 3 phase motor control sensor-ed applications the use of Hall sensors, generally 3 sensors placed 120 degrees apart around the rotor, are deployed to detect position and speed. Each of the 3 sensors provides a pulse that applied to an input capture pin, can then be analyzed and both speed and position can be deduced. To simplify the calculations required by the CPU on each hall sensor's input, if all 3 inputs are "exclusively OR'd" into one timer channel and the free running counter is refreshed on every edge then this can simplify the speed calculation.

Via the SIM module and SIM_FTMOPT1 register the FTM2CH1SEL bit provides the choice of normal FTM2_CH1 input or the XOR of FTM2_CH0, FTM2_CH1 and FTM1_CH1 pins that will be applied to FTM2_CH1.

NOTE

If the user utilizes FTM1_CH1 to be an input to FTM2_CH1, FTM1_CH0 can still be utilized for other functions.

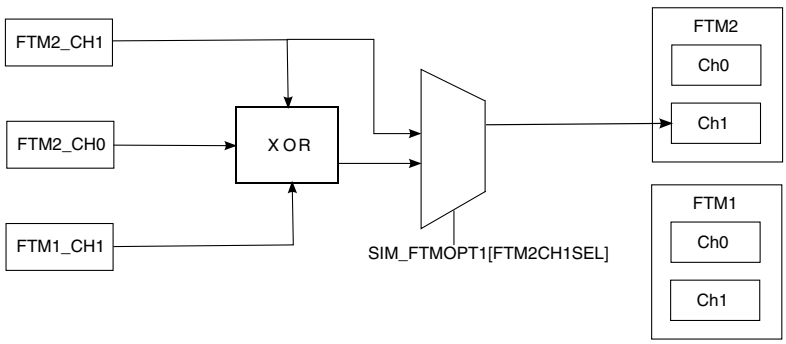


Figure 41-112. FTM Hall Sensor Configuration

41.9.3 FTM Modulation Implementation

FTM0 and FTM3 support a modulation function where the output channels when configured as PWM or Output Compare mode modulate another timer output when the channel signal is asserted. Any of the 8 channels of FTM0 and any of the 8 channels of FTM3 can be configured to support this modulation function.

The SIM_FTMOPT1 register has control bits (FTMxCHySEL) that allow the user to select normal PWM/Output Compare mode on the corresponding FTM timer channel or modulate with FTM1_CH1. The diagram below shows the implementation for FTM0. FTM3 has similar implementation controlled by SIM_FTMOPT1[FTM3CHySEL] on each of its 8 channels with modulation possible via FTM2_CH1. See SIM Block Guide for further information.

When FTM1_CH1 is used to modulate an FTM0 channel, then the user must configure FTM1_CH1 to provide a signal that has a higher frequency than the modulated FTM0 channel output. Also it limits the use of the FTM1_CH0 function, as the FTM1_CH1 will be programmed to provide a 50% duty PWM signal and limit the start and modulus values for the free running counter. FTM2 has a similar restriction when FTM2_CH1 is used for modulating an FTM3 channel.

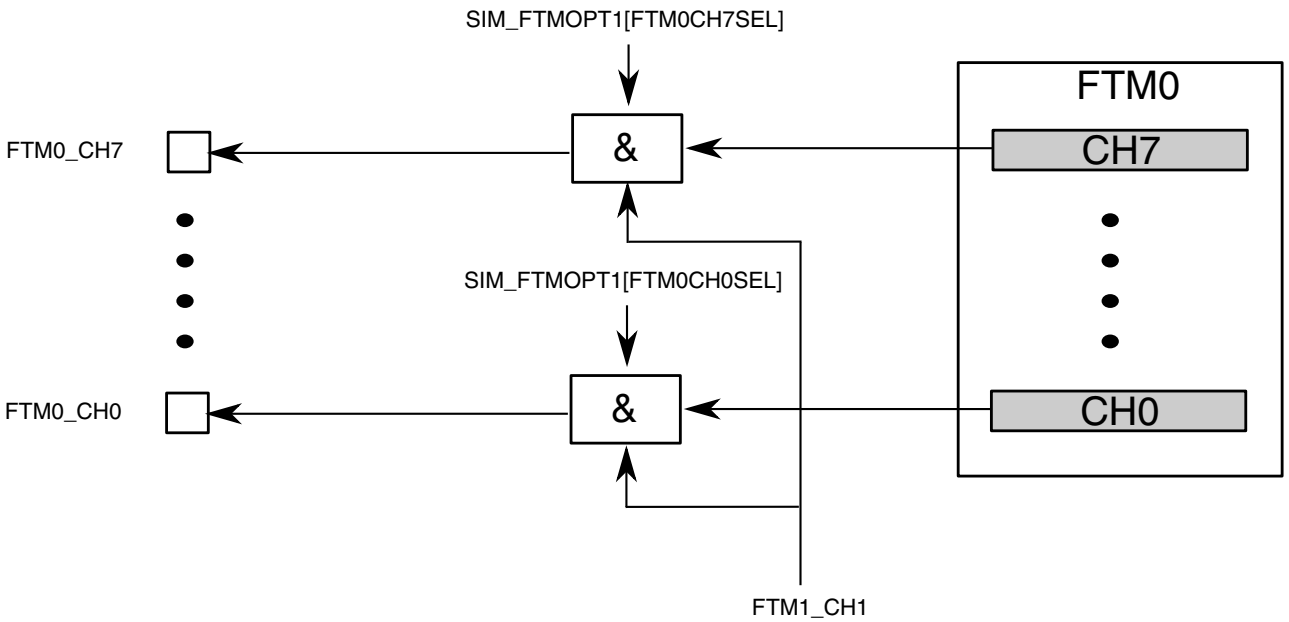
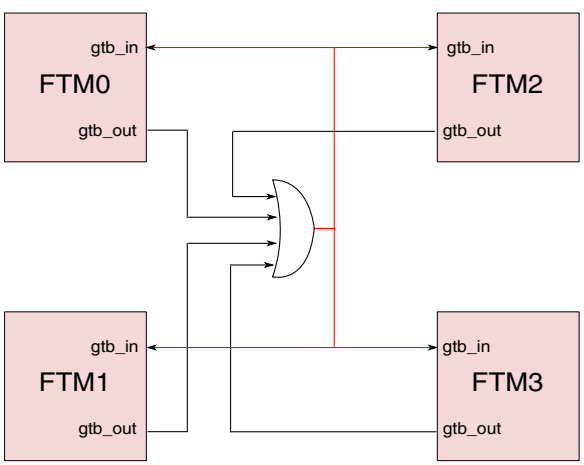


Figure 41-113. FTM Output Modulation

41.9.4 FTM Global Time Base

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

FTM supports global timer base through the GTB feature. Any of the FTM module could be used as the GTB_EN source. The global timer base only allows the FTM counters to start their operation synchronously, it does not automatically provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during misc FTM operation.



41.9.5 FTM BDM and debug halt mode

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

Chapter 42
Low-power Periodic Interrupt Timer (LPIT)

42.1 Chip-specific information for this module

42.1.1 Instantiation Information

This device contains one LPIT module with 4 channels.

42.1.1.1 LPIT/DMA Periodic Trigger Assignments

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

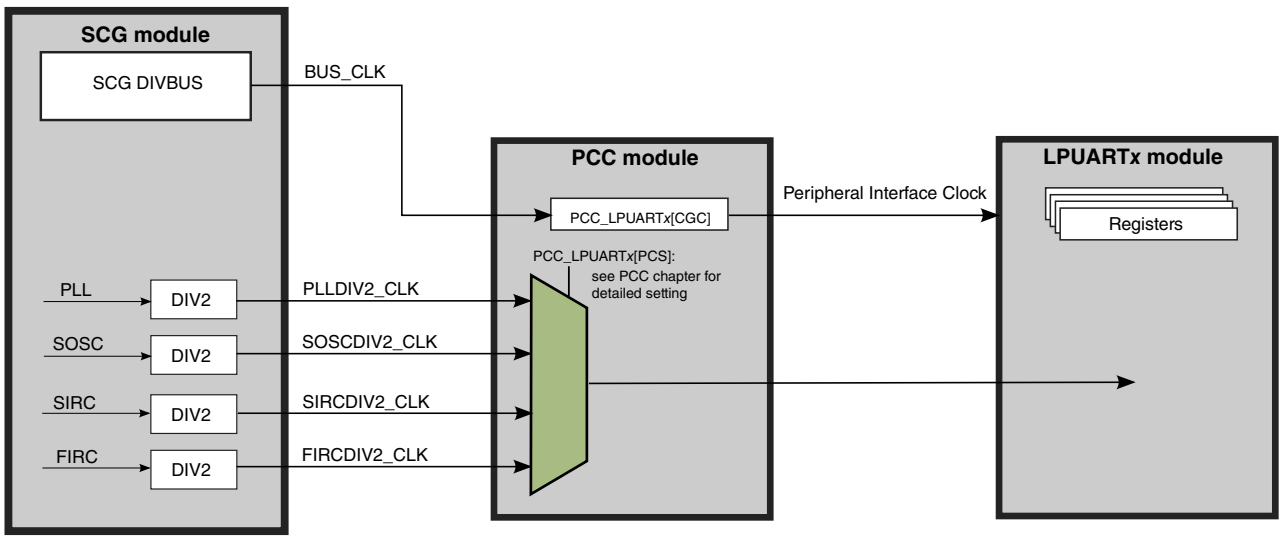
Table 42-1. LPIT channel assignments for periodic DMA triggering

DMA Channel Number	LPIT Channel
0	0
1	1
2	2
3	3

42.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

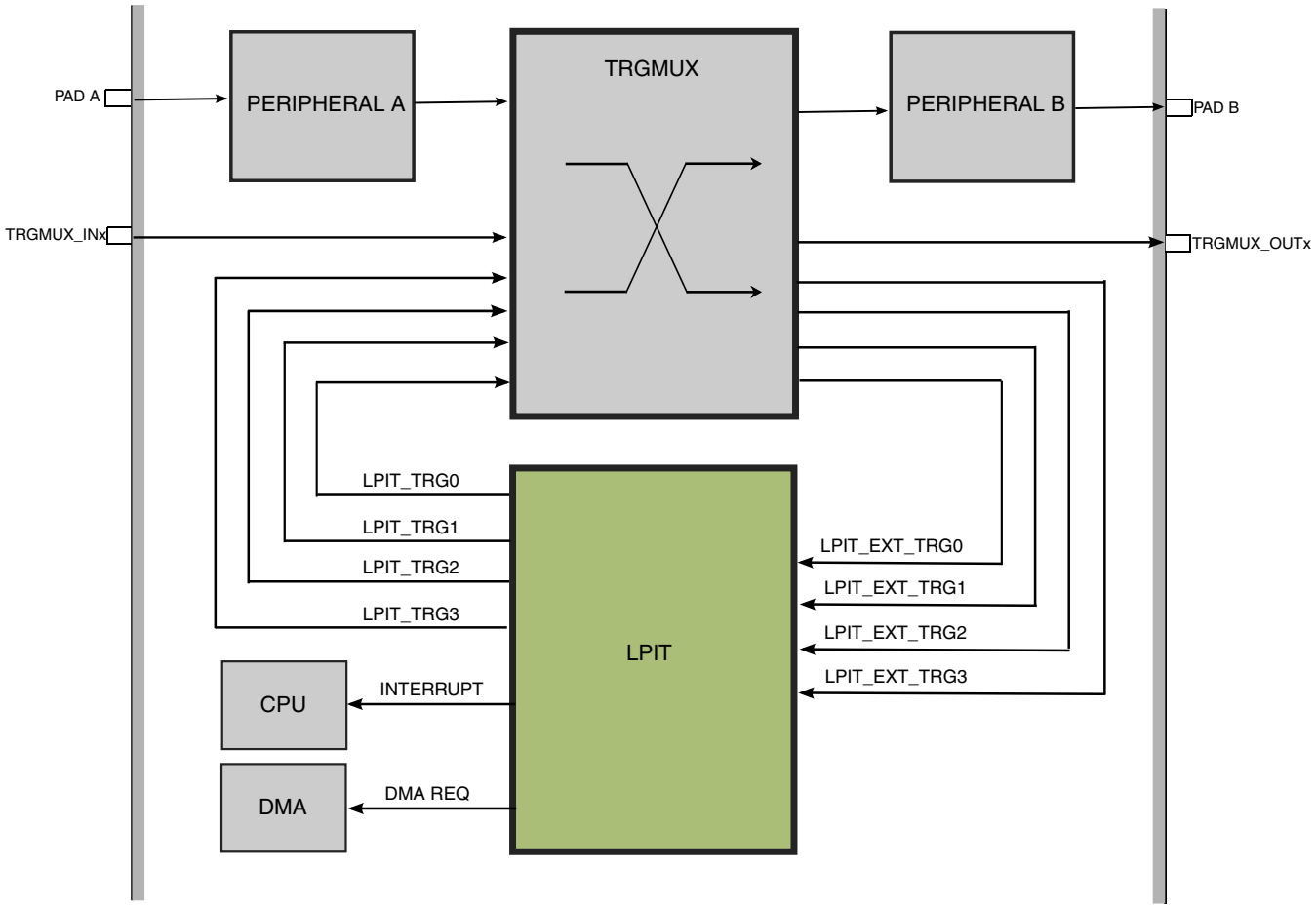
The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPUART Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, FlexIO and LPIT.



42.1.3 Inter-connectivity Information

The LPIT module interconnectivity with other peripherals is based on the TRGMUX.



42.2 Introduction

42.2.1 Overview

The Low Power Periodic Interrupt Timer (LPIT) is a multi-channel timer module generating independent pre-trigger and trigger outputs. These timer channels can operate individually or can be chained together. The LPIT can operate in low power modes if configured to do so. The pre-trigger and trigger outputs can be used to trigger other modules on the device.

Each timer channel can be configured to run independently and made to work in either compare or capture modes. In compare mode, the timers decrement when enabled and generate an output pre-trigger and timeout pulse. The trigger output is 1 clock cycle delayed of the pre-trigger pulse. Each timer channel start, reload and restart can be

controlled via control bits. The timer can be configured to always decrement, or decrement on selected trigger inputs or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations. In capture mode, the timer can be used to perform measurements as the timer value is captured (in the timer value register) when a selected trigger input is asserted. In capture mode, the timer can support once-off or multiple measurements (for example, frequency measurements).

The timer channels operate on an asynchronous clock, which is independent from the register read/write access clock. Clock synchronization between the clock domains ensures normal operations.

42.2.2 Block Diagram

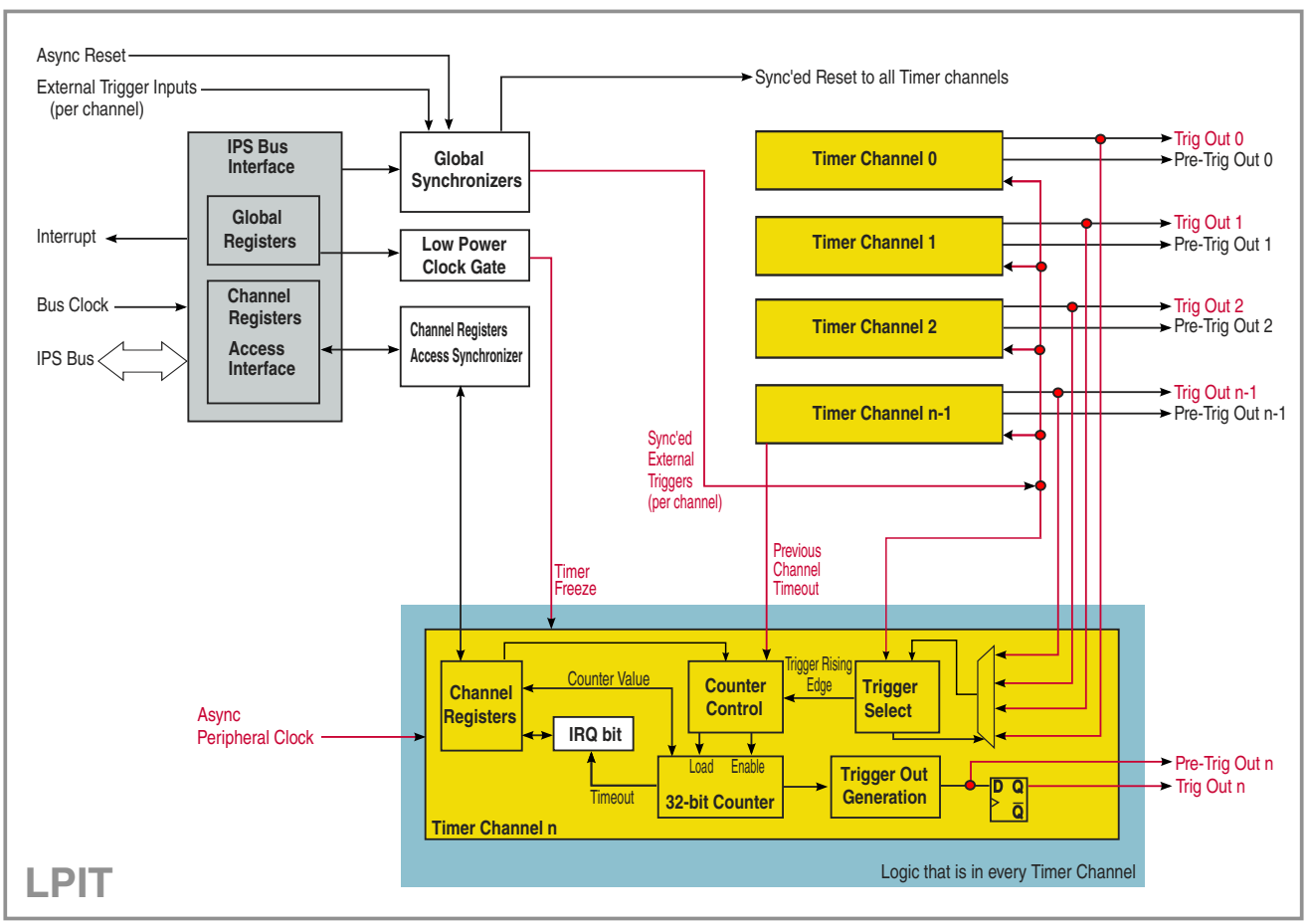


Figure 42-1. Top Level Block Diagram

42.3 Modes of operation

The LPIT module supports the chip modes described in the following table.

Table 42-2. Chip modes supported by the LPIT module

Chip mode	LPIT Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (MCR[DOZE_EN]) is set and the LPIT is using an external or internal clock source which remains operating during stop/wait modes.
Debug	Can continue operating provided the Debug Enable bit (MCR[DBG_EN]) is set.

42.4 Memory Map and Registers

The memory map comprises of 32-bit aligned registers which can be accessed via 8-bit, 16-bit, or 32-bit accesses. Write access to reserved locations will generate a transfer error. Read access to reserved locations will also generate a transfer error and the read data bus will show all 0s. The Memory Map and complete module is in Big Endian format.

The module will not check for correctness of programmed values in the registers and software must ensure that correct values are being written.

LPIT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_7000	Version ID Register (LPIT0_VERID)	32	R	0100_0000h	42.4.1/1126
4003_7004	Parameter Register (LPIT0_PARAM)	32	R	See section	42.4.2/1127
4003_7008	Module Control Register (LPIT0_MCR)	32	R/W	0000_0000h	42.4.3/1127
4003_700C	Module Status Register (LPIT0_MSR)	32	w1c	0000_0000h	42.4.4/1128
4003_7010	Module Interrupt Enable Register (LPIT0_MIER)	32	R/W	0000_0000h	42.4.5/1129
4003_7014	Set Timer Enable Register (LPIT0_SETTEN)	32	R/W	0000_0000h	42.4.6/1131
4003_7018	Clear Timer Enable Register (LPIT0_CLRTEN)	32	W (always reads 0)	0000_0000h	42.4.7/1132
4003_7020	Timer Value Register (LPIT0_TVAL0)	32	R/W	0000_0000h	42.4.8/1133
4003_7024	Current Timer Value (LPIT0_CVAL0)	32	R	FFFF_FFFFh	42.4.9/1134

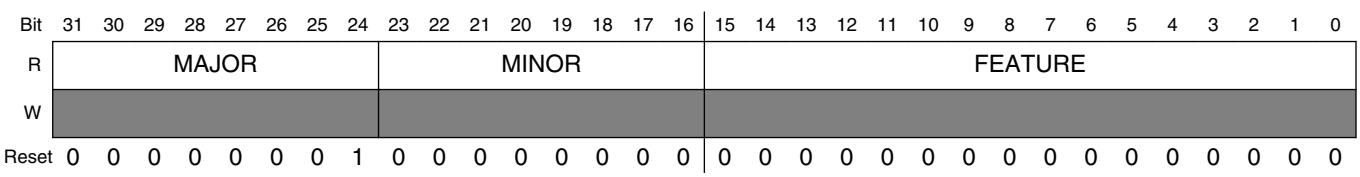
Table continues on the next page...

LPIT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_7028	Timer Control Register (LPIT0_TCTRL0)	32	R/W	0000_0000h	42.4.10/ 1135
4003_7030	Timer Value Register (LPIT0_TVAL1)	32	R/W	0000_0000h	42.4.8/1133
4003_7034	Current Timer Value (LPIT0_CVAL1)	32	R	FFFF_FFFFh	42.4.9/1134
4003_7038	Timer Control Register (LPIT0_TCTRL1)	32	R/W	0000_0000h	42.4.10/ 1135
4003_7040	Timer Value Register (LPIT0_TVAL2)	32	R/W	0000_0000h	42.4.8/1133
4003_7044	Current Timer Value (LPIT0_CVAL2)	32	R	FFFF_FFFFh	42.4.9/1134
4003_7048	Timer Control Register (LPIT0_TCTRL2)	32	R/W	0000_0000h	42.4.10/ 1135
4003_7050	Timer Value Register (LPIT0_TVAL3)	32	R/W	0000_0000h	42.4.8/1133
4003_7054	Current Timer Value (LPIT0_CVAL3)	32	R	FFFF_FFFFh	42.4.9/1134
4003_7058	Timer Control Register (LPIT0_TCTRL3)	32	R/W	0000_0000h	42.4.10/ 1135

42.4.1 Version ID Register (LPITx_VERID)

Address: 4003_7000h base + 0h offset = 4003_7000h



LPITx_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification
FEATURE	Feature Number This read only field returns the feature set number.

42.4.2 Parameter Register (LPITx_PARAM)

This register provides details on the parameter settings that were used while including this module in the device.

Address: 4003_7000h base + 4h offset = 4003_7004h

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

* Notes:
- The reset value is chip-specific.u = Unaffected by reset.

LPITx_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 EXT_TRIG	Number of External Trigger Inputs Number of external triggers implemented.
CHANNEL	Number of Timer Channels Number of timer channels implemented.

42.4.3 Module Control Register (LPITx_MCR)

Address: 4003_7000h base + 8h offset = 4003_7008h

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												DBG_EN	DOZE_EN	SW_RST	M_CEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPITx_MCR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBG_EN	Debug Enable Bit Allows the timer channels to be stopped when the device enters the Debug mode 0 Timer channels are stopped in Debug mode 1 Timer channels continue to run in Debug mode
2 DOZE_EN	DOZE Mode Enable Bit Allows the timer channels to be stopped or continue to run when the device enters the DOZE mode 0 Timer channels are stopped in DOZE mode 1 Timer channels continue to run in DOZE mode
1 SW_RST	Software Reset Bit Resets all channels and registers, except the Module Control Register. Remains set until cleared by software. 0 Timer channels and registers are not reset 1 Timer channels and registers are reset
0 M_CEN	Module Clock Enable Enables the peripheral clock to the module timers. M_CEN bit must be asserted when writing to timer registers. Both clocks (bus clock and peripheral clock) must be enabled, to allow for clock synchronization and update of register bits. **NOTE:** Writing to the MSR, SETTEN , CLRTEN, TCTRL, and TVAL registers while M_CEN = 0, will lead to the assertion of a transfer error for that bus cycle. Writing to CVAL and reserved registers will always generate a transfer error. 0 Protocol clock to timers is disabled 1 Protocol clock to timers is enabled

42.4.4 Module Status Register (LPITx_MSR)

Address: 4003_7000h base + Ch offset = 4003_700Ch

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												TIF3	TIF2	TIF1	TIF0
W													w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPITx_MSR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TIF3	Channel 3 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
2 TIF2	Channel 2 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
1 TIF1	Channel 1 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
0 TIF0	Channel 0 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred

42.4.5 Module Interrupt Enable Register (LPITx_MIER)

Address: 4003_7000h base + 10h offset = 4003_7010h

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												TIE3	TIE2	TIE1	TIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

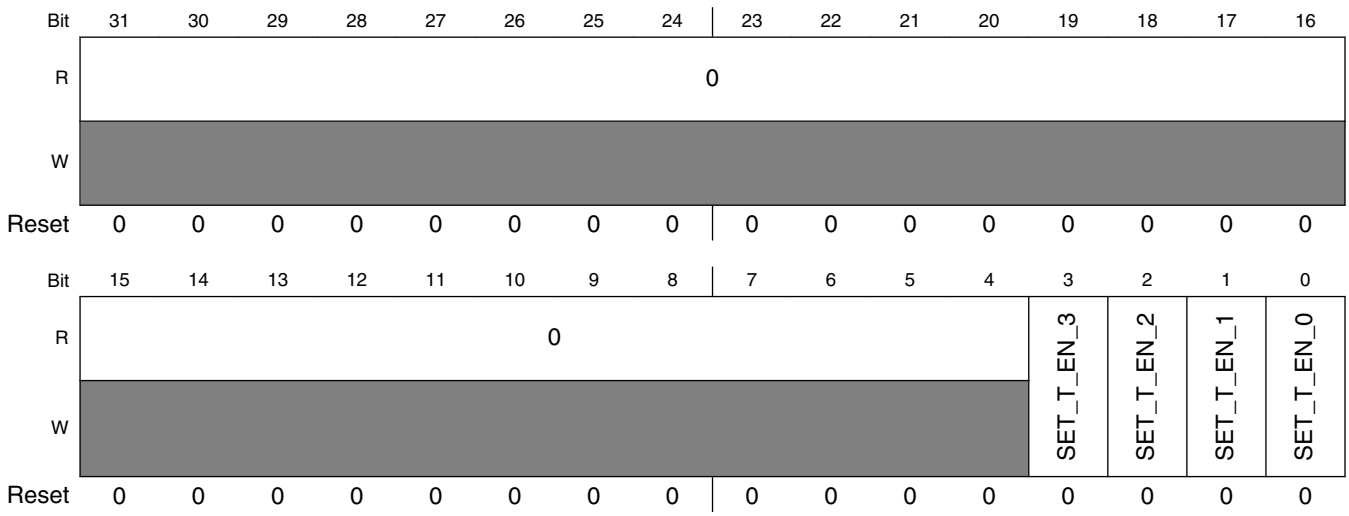
LPITx_MIER field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled

42.4.6 Set Timer Enable Register (LPITx_SETTEN)

This register allows simulataneous enabling of timer channels. Timer channels can be enabled either by writing '1' to T_EN in respective TCTRLn register or setting the corresponding bit in this register. Writing a '0' to this register has no effect. CLRTEN register should be used to disable timer channels simultaneously.

Address: 4003_7000h base + 14h offset = 4003_7014h



LPITx_SETTEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SET_T_EN_3	Set Timer 3 Enable Writing '1' to this bit will enable the timer channel 3. This bit can be used in addition to T_EN bit in TCTRL3 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL3 is set to '0' or '1' is written to the CLR_T_EN_3 bit in CLRTEN register. 0 No effect 1 Enables the Timer Channel 3
2 SET_T_EN_2	Set Timer 2 Enable Writing '1' to this bit will enable the timer channel 2. This bit can be used in addition to T_EN bit in TCTRL2 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL2 is set to '0' or '1' is written to the CLR_T_EN_2 bit in CLRTEN register. 0 No Effect 1 Enables the Timer Channel 2
1 SET_T_EN_1	Set Timer 1 Enable

Table continues on the next page...

LPITx_SETTEN field descriptions (continued)

Field	Description
	Writing '1' to this bit will enable the timer channel 1. This bit can be used in addition to T_EN bit in TCTRL1 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL1 is set to '0' or '1' is written to the CLR_T_EN_1 bit in CLRTEN register. 0 No Effect 1 Enables the Timer Channel 1
0 SET_T_EN_0	Set Timer 0 Enable Writing '1' to this bit will enable the timer channel 0. This bit can be used in addition to T_EN bit in TCTRL0 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL0 is set to 0 or '1' is written to the CLR_T_EN_0 bit in CLRTEN register. 0 No effect 1 Enables the Timer Channel 0

42.4.7 Clear Timer Enable Register (LPITx_CLRTEN)

Address: 4003_7000h base + 18h offset = 4003_7018h

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	0	0	0
W													CLR_T_EN_3	CLR_T_EN_2	CLR_T_EN_1	CLR_T_EN_0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

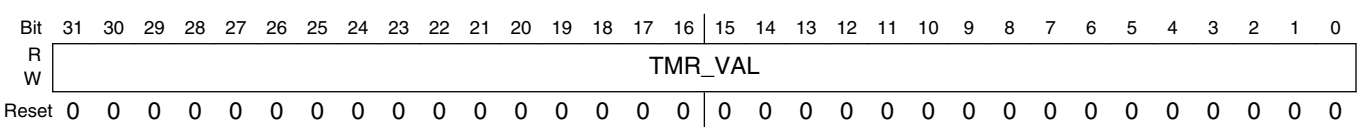
LPITx_CLRTEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 CLR_T_EN_3	Clear Timer 3 Enable Writing a '1' to this bit will disable the timer channel 3. This bit can be used in addition to T_EN bit in TCTRL3 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable Writing a '1' to this bit will disable the timer channel 2. This bit can be used in addition to T_EN bit in TCTRL2 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 2
1 CLR_T_EN_1	Clear Timer 1 Enable Writing a '1' to this bit will disable the timer channel 1. This bit can be used in addition to T_EN bit in TCTRL1 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable Writing a '1' to this bit will disable the timer channel 0. This bit can be used in addition to T_EN bit in TCTRL0 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No action 1 Clear T_EN bit for Timer Channel 0

42.4.8 Timer Value Register (LPITx_TVAL*n*)

In compare modes, these registers select the timeout period for the timer channels. In capture modes, these registers are loaded with the value of the counter when the trigger asserts.

Address: 4003_7000h base + 20h offset + (16d × i), where i=0d to 3d



LPITx_TVAL*n* field descriptions

Field	Description
TMR_VAL	Timer Value

LPITx_TVAL*n* field descriptions (continued)

Field	Description
	In compare modes, sets the timer channel 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 channel; 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 channel must be disabled and enabled again. In capture modes, this register stores the inverse of the counter whenever the trigger asserts. 0 Invalid load value in compare modes >0 Value to be loaded (Compare Mode) or Value of Timer (Capture Mode)

42.4.9 Current Timer Value (LPITx_CVAL*n*)

These registers indicate the current timer counter value.

NOTE

While the timer is running, CVALn register reads may not return the real value.

Address: 4003_7000h base + 24h 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	TMR_CUR_VAL																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

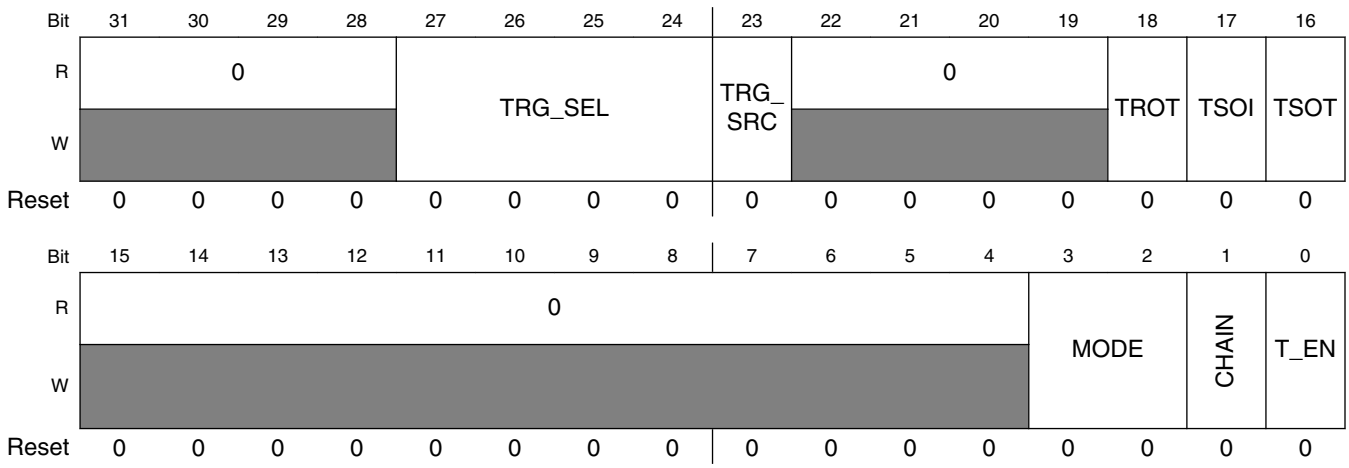
LPITx_CVAL*n* field descriptions

Field	Description
TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

42.4.10 Timer Control Register (LPITx_TCTRL*n*)

These registers contain the control bits for each timer channel

Address: 4003_7000h base + 28h offset + (16d × i), where i=0d to 3d



LPITx_TCTRL*n* 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 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. This field should only be changed when the LPIT timer channel is disabled. The TRG_SRC bit selects between internal and external trigger signals for each channel. The TRG_SEL bits select one trigger from the set of internal or external triggers selected by TRG_SRC. 0 Timer channel 0 trigger source is selected 1 Timer channel 1 trigger source is selected 2 Timer channel 2 trigger source is selected . . . . . . n Timer channel 'n' trigger source is selected
23 TRG_SRC	Trigger Source Selects between internal or external trigger sources. The final trigger is selected by TRG_SEL depending on which trigger source out of internal triggers or external triggers are selected by TRG_SRC. Refer to the chip configuration section for available external trigger options. If a channel does not have an associated external trigger then this bit for that channel should be set to 1. 0 Trigger source selected in external 1 Trigger source selected is the internal trigger
22–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 TROT	Timer Reload On Trigger

Table continues on the next page...

LPITx_TCTRL*n* field descriptions (continued)

Field	Description
	When set, the LPIT timer will reload when a rising edge is detected on the selected trigger input. The trigger input is ignored if the LPIT is disabled during debug mode or DOZE mode (DOZE_EN or DBGEN = 0) 0 Timer will not reload on selected trigger 1 Timer will reload on selected trigger
17 TSOI	Timer Stop On Interrupt This bit controls whether the channel timer will stop after it times out and when it can restart (when TSOT = 0). If TSOT = 1, then the timer will stop on timeout and will restart after a rising edge on the selected trigger is detected. If TSOT = 0, then this bit controls when the timer restarts. 0 Timer does not stop after timeout 1 Timer will stop after timeout and will restart after rising edge on the T_EN bit is detected (i.e. timer channel is disabled and then enabled)
16 TSOT	Timer Start On Trigger This bit controls when the timer starts decrementing. 0 Timer starts to decrement immediately based on restart condition (controlled by TSOI bit) 1 Timer starts to decrement when rising edge on selected trigger is detected
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 MODE	Timer Operation Mode Configures the Channel Timer Mode of Operation. The mode bits control how the timer decrements. See Functional Description for more details. 00 32-bit Periodic Counter 01 Dual 16-bit Periodic Counter 10 32-bit Trigger Accumulator 11 32-bit Trigger Input Capture
1 CHAIN	Chain Channel When enabled, timer channel will decrement when channel N-1 trigger asserts. Channel 0 cannot be chained. 0 Channel Chaining is disabled. Channel Timer runs independently. 1 Channel Chaining is enabled. Timer decrements on previous channel's timeout
0 T_EN	Timer Enable Enables or disables the Timer Channel 0 Timer Channel is disabled 1 Timer Channel is enabled

42.5 Functional description

42.5.1 Initialization

The following steps can be used to initialize the LPIT module

- Enable the protocol clock by setting the M_CEN bit in the MCR register.

NOTE

Writing to certain registers while M_CEN = 0 will lead to assertion of transfer error for that bus access. These registers are MSR, SETTEN, CLRTEN, TVAL, and TCTRL. Writing to CVAL and Reserved registers will generate a transfer error irrespective of M_CEN bit value. Reads to these registers can happen irrespective of M_CEN bit value.

- Wait for 4 protocol clock cycles to allow time for clock synchronization and reset de-assertion.
- For each timer channel that is to be enabled, configure the timer mode of operation (MODE bits), Trigger source selection (TRG_SEL & TRG_SRC) and Trigger control bits (TROT, TSOT, TSOI bits) in the TCTRLn register.
- Configure the channels that are to be chained by setting the CHAIN bit in the corresponding channel's TCTRLn register.
- For channels configured in Compare Mode, set the timer timeout value by programming the appropriate value in TVAL register for those channels.
- Configure TIEn bits in MIER register for those channels which are required to generate interrupt on timer timeout.
- Configure the low power mode functionality of the module by setting the DBG_EN and DOZE_EN bits in the MCR register. This is common to all timer channels.
- Enable the channel timers by setting the corresponding T_EN bit in the corresponding channel's TCRTLn register.
- For channels configured in Capture Mode, the timer value can be read from TVALn register when channel timeout occurs.
- At any time, the current value of the timer for any channel can be read by reading the corresponding channel's CVALn register.
- The timer interrupt flag bits (TIFn) in MSR register get asserted on timer timeout. These bits can be cleared by writing '1' to them.

42.5.2 Timer Modes

The timer mode is configured by setting an appropriate value in the MODE bits in TCTRLn register. The timer modes supported are:

- *32-bit Periodic Counter:* In this mode the counter will load and then decrement down to zero. It will then set the timer interrupt flag and assert the output pre-trigger.
- *Dual 16-bit Periodic Counter:* In this mode, the counter will load and then the lower 16-bits will decrement down to zero, which will assert the output pre-trigger. The upper 16-bits will then decrement down to zero, which will negate the output pre-trigger and set the timer interrupt flag.
- *32-bit Trigger Accumulator:* In this mode, the counter will load on the first trigger rising edge and then decrement down to zero on each trigger rising edge. It will then set the timer interrupt flag and assert the output pre-trigger.
- *32-bit Trigger Input Capture:* In this mode, the counter will load with 0xFFFF_FFFF and then decrement down to zero. If a trigger rising edge is detected, it will store the inverse of the current counter value in the load value register, set the timer interrupt flag and assert the output pre-trigger.

The timer operation is further controlled by Trigger Control bits (TSOT, TSOI, TROT) which control the timer load, reload, start and restart of the timers.

NOTE

- The trigger output is asserted one Protocol Timer Clock cycle later than pre-trigger output. The trigger output and the pre-trigger output de-assert at the same time.
- The pre-trigger output is asserted for two clock cycles and trigger output is asserted for one clock cycle (except in 16-bit Periodic Counter mode where both pre-trigger and trigger are asserted for many cycles depending on TMR_VAL[31:16]).

42.5.3 Trigger Control for Timers

The TSOT, TROT, TSOI and TRG_SEL, TRG_SRC bits control how the trigger input affects the timer operation. The TRG_SEL selects the input trigger for the channel from all other channel's trigger outputs. The TRG_SRC further selects between the selected internal trigger and the external trigger input to the channel.

The selected trigger affects the timer operation based on TROT, TSOI & TSOT bits. The behavior due to these bits is as follows:

- If TSOI = 1, counter stops on TIF assertion. Requires trigger (if TSOT = 1) or T_EN rising edge (if TSOT = 0), to reload and decrement. If TSOI = 0, counter does not stop after timeout.

- If TROT = 1, counter is loaded on each trigger; else, counter is loaded on every T_EN rising edge or timeout rising edge (timeout not used in Capture modes).
- If TSOT = 1, counter will start to decrement on trigger. Subsequent triggers are ignored till a counter timeout. If TSOT = 0, counter decrements immediately from the next clock edge. TSOT has no effect when channel is Chained or in Capture mode.

These bits affect the timer operation differently in different timer modes:

- In 32-bit Periodic Counter and Dual 16-bit Periodic Counter modes, all bits (TSOT, TSOI & TROT) affect the timer operation as described above.
- In 32-bit Trigger Accumulator mode, only TSOI bit controls the timer function. TROT & TSOT bits have no effect on timer operation.
- In 32-bit Input Trigger Capture mode, TSOI and TROT bits control the timer function. TSOT bit has no effect on timer operation.

42.5.4 Channel Chaining

Individual timer channels can be chained together to achieve a larger value of timeout. Chaining the timer channel causes them to work in a *'nested loop'* manner thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

The channels are chained by setting the CHAIN bit in corresponding channel's TCTRLn register. When a channel is chained, that channel's timer decrements on previous channel's timeout pulse, irrespective of the timer mode (MODE bits). The TSOT bit does not have any effect if the channel timer (Channel 'n') is chained to previous channel's timer (Channel 'n-1').

42.6 Usage Guide

42.6.1 Periodic timer/counter

LPIT typical usage is to generate periodic trigger pulses and interrupts.

Example: LPIT channel0 trigger a periodic interrupt every 1 second

- Enable the LPIT module clock;
- Reset the timer channels and registers;
- Setup timer operation in debug and doze modes and enable the module;
- Setup the channel counters operation mode to "32-bit Periodic Counter",and keep default values for the trigger source;

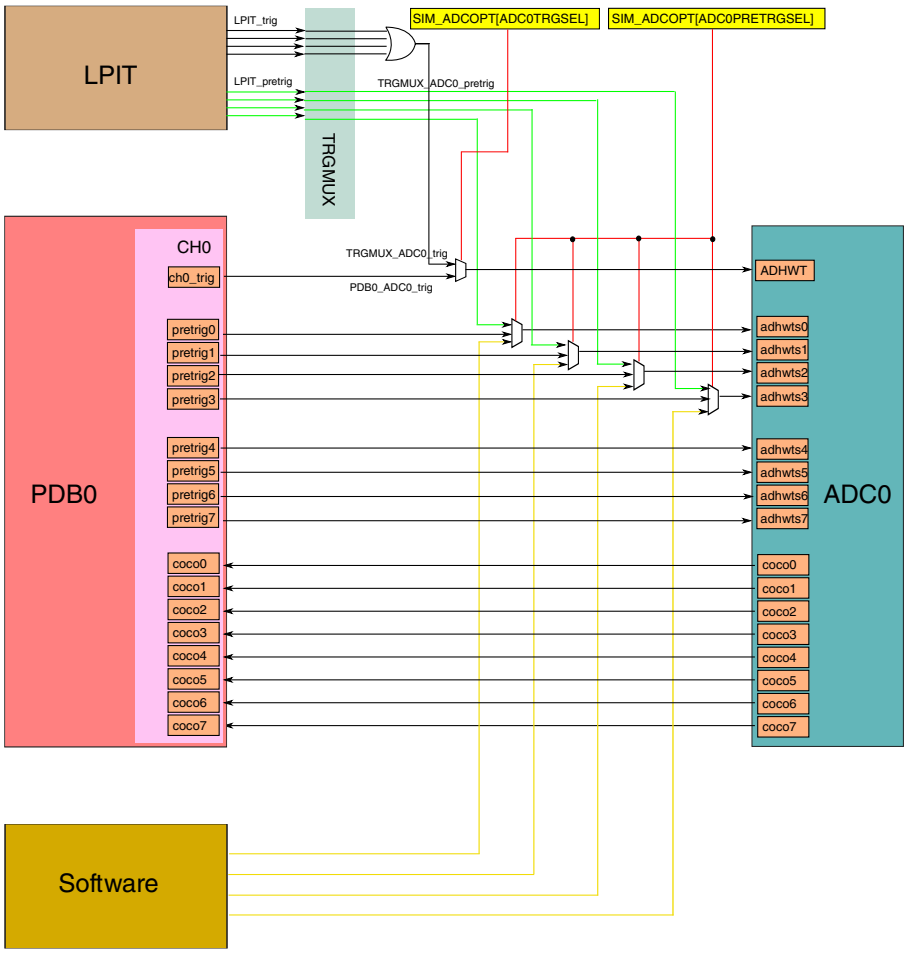
- Set timer period for channel 0 as 1 second;
- Enable channel0 interrupt;
- Starts the timer counting afer all configuration;
- In the channel interrupt rountine, clear the channel flag every 1 second.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPIT0);
LPIT0_MCR |= LPIT_MCR_SW_RST_MASK;
LPIT0_MCR &= ~LPIT_MCR_SW_RST_MASK;
LPIT0_MCR |= (LPIT_MCR_DBG_EN(1) | LPIT_MCR_DOZE_EN(1) | LPIT_MCR_M_CEN_MASK);
LPIT0_TCTRL0 |= LPIT_TCTRL_MODE(0);
LPIT0_TVAL0 = ONE_SECOND_VALUE;
LPIT0_MIER |= LPIT_MIER_TIE0_MASK;
NVIC_EnableIRQ(LPIT0_IRQ);
LPIT0_SETTEN |= LPIT_SETTEN_SET_T_EN_0_MASK;
```

42.6.2 LPIT/ADC Trigger

The LPIT could be used as an alternate ADC hardware trigger source, whose implementation is via TRGMUX. Each LPIT channel supports one pre-trigger and one trigger. The LPIT channels are implemented based on independent counters. When used as ADC trigger source, the channel outputs are ORed together to generate the ADC hardware trigger. The following diagram shows an example of using LPIT triggering ADC0.



Example: LPIT hardware trigger via TRGMUX for ADC conversion

- ADC module initialization and enable its hardware trigger;
- Enable the LPIT module clock;
- Reset the LPIT timer channels and registers;
- Setup timer operation in debug and doze modes and enable LPIT module;
- Setup the LPIT_CH0 and LPIT_CH1 counters mode to "32-bit Periodic Counter",and keep default values for the trigger source;
- Set timer period for LPIT_CH0 and LPIT_CH1,they are used as ADC pre-trigger delay;
- Starts the timer counting after all configuration;
- In SIM register, select TRGMUX output as ADC pre-trigger and trigger source;
- Configure LPIT_CH0 and LPIT_CH1 as ADC hardware trigger by TRGMUX;
- In the ADC interrupt routine, clear the COCO flag and read the conversion value. (If Rn is read, the COCO flag will be cleared automatically.)

The following pseudo-code matches the described setup above:

```
ADC_Config();
CLOCK_EnableClock(LPIT0);
LPIT0_MCR |= LPIT_MCR_SW_RST_MASK;
```

```
LPIT0_MCR &= ~LPIT_MCR_SW_RST_MASK;
LPIT0_MCR |= (LPIT_MCR_DBG_EN(1) | LPIT_MCR_DOZE_EN(1) | LPIT_MCR_M_CEN_MASK);
LPIT0_TCTRL0 |= LPIT_TCTRL_MODE(0);
LPIT0_TCTRL1 |= LPIT_TCTRL_MODE(0);
LPIT0_TVAL0 = ADC_PRETRG_DELAY_VALUE1;
LPIT0_TVAL1 = ADC_PRETRG_DELAY_VALUE2;
LPIT0_SETTEN |= LPIT_SETTEN_SET_T_EN_0_MASK|LPIT_SETTEN_SET_T_EN_1_MASK;
SIM_ADCOPT |= SIM_ADCOPT_ADC0TRGSEL(1)|SIM_ADCOPT_ADC0PRETRGSEL(1);
TRGMUX0_ADC0 = TRGMUX_TRGCFG_SEL0(7)|TRGMUX_TRGCFG_SEL1(8);
```

Chapter 43
Pulse Width Timer (PWT)

43.1 Chip-specific information for this module

43.1.1 Instantiation Information

The Pulse Width Timer (PWT) module on this device consists of one 16-bit counter, which can be used to capture or measure the pulse width mapping on its input channels.

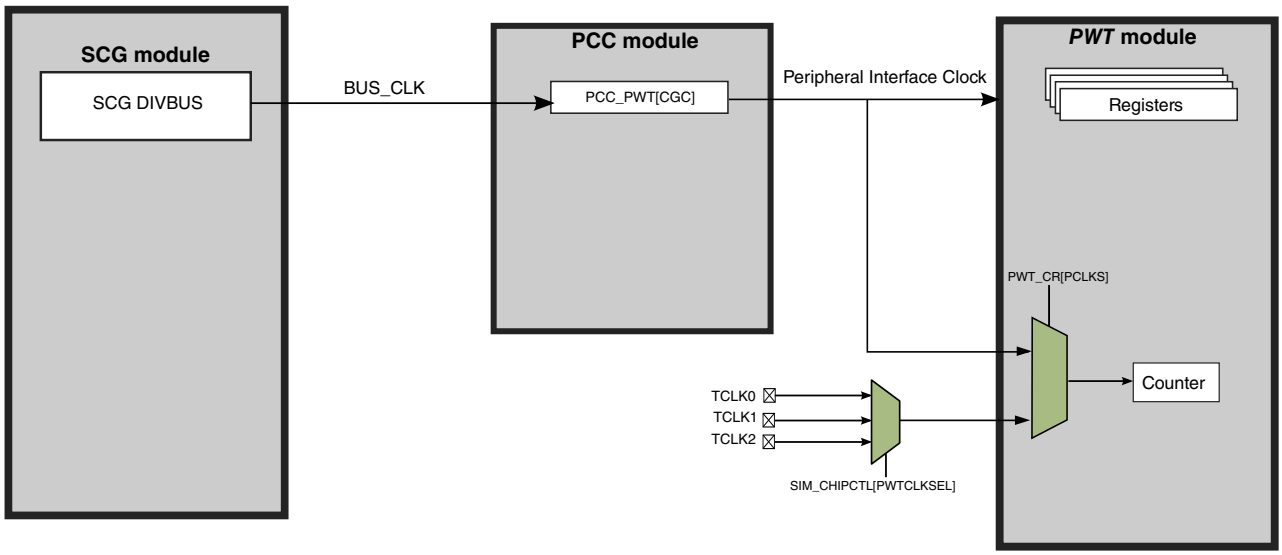
The counter of PWT has two selectable clocks sources, and support up to BUS_CLK with internal timer clock. PWT module supports programmable positive or negative pulse edges, and programmable interrupt generation upon pulse width values or counter overflow.

43.1.2 PWT Clocking Information

Two software selectable clock sources are available for input to pre-scaler divider of PWT module:

- Bus clock
- External clock from pins (TCLKx)

Peripheral Clocking - PWT

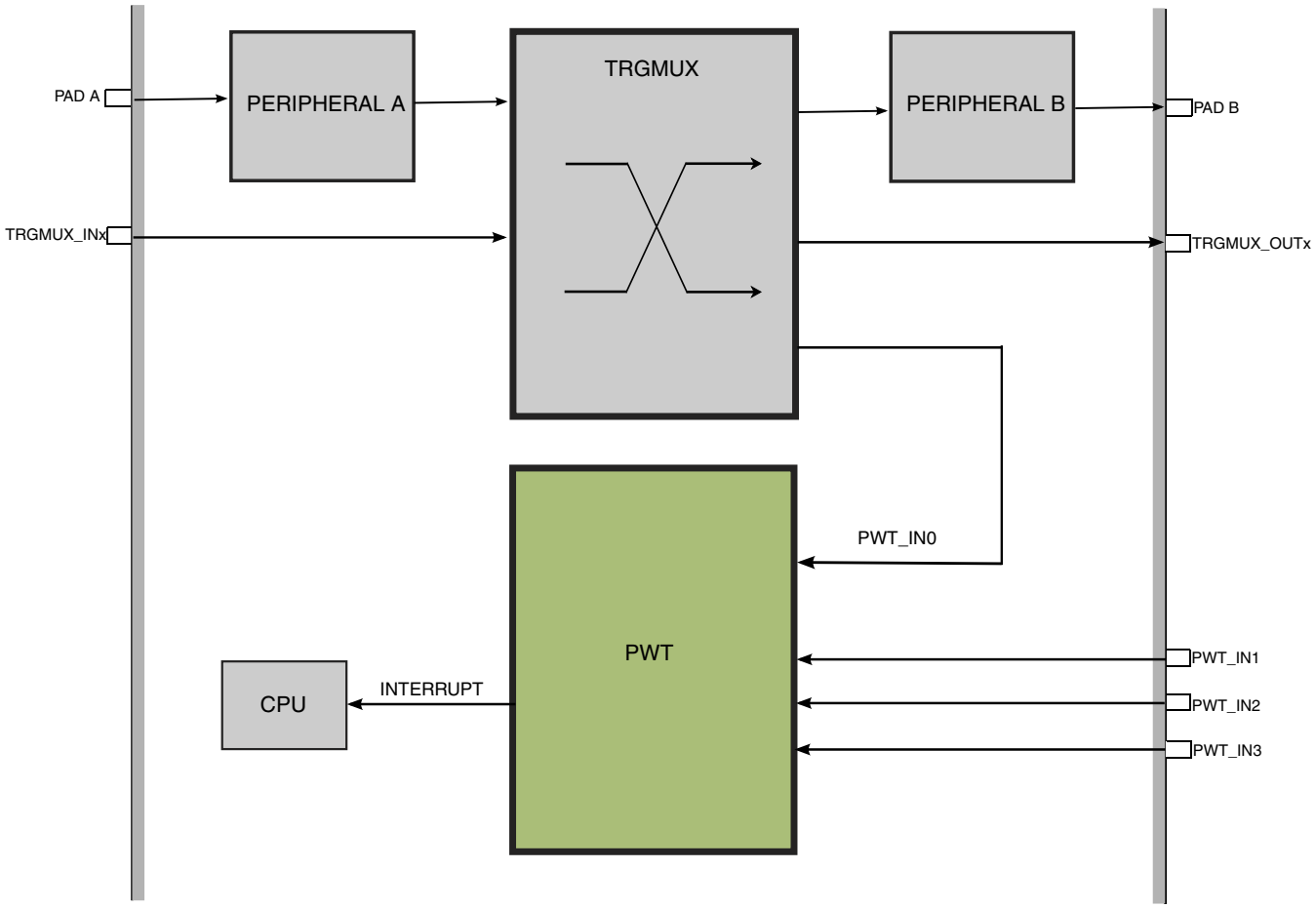


43.1.3 Inter-connectivity Information

PWT module has four input channels, which is connected as shown in the following table:

Table 43-1. PWT input connections

PWT input channel	Connection
0	TRGMUX output
1	PWT_IN1 pin
2	PWT_IN2 pin
3	PWT_IN3 pin



43.2 Introduction

43.2.1 Features

The pulse width timer (PWT) includes the following features:

- Automatic measurement of pulse width with 16 bit resolution
- Separate positive and negative pulse width measurements
- Programmable measuring time between successive alternating edges, rising edges or falling edges
- Programmable pre-scaler from clock input as 16-bit counter time base
- Two selectable clock sources — bus clock and alternative clock
- Four selectable pulse inputs
- Programmable interrupt generation upon pulse width value updated and counter overflows

43.2.2 Modes of operation

This module supports the following mode:

- Run Mode

  When enabled, the pulse width timer module is active.

- Wait Mode

  When enabled, the pulse width timer module is active and can perform the waking up function if the corresponding interrupt is enabled.

- Stop Mode

  The pulse width timer module is halted when entering stop and the register contents and operating status is preserved. If stop exits with reset then the module resets. If stop exits with another source, the module resumes operation based on module status upon exit.

- Active Background Mode

  Upon entering BDM mode, the PWT suspends all counting and pulse edge detection until the microcontroller returns to normal user operating mode. Counting and edge detection resume from the suspended value when normal user operating mode returns as long as the PWTSR bit (PWT software reset) is not written to 1 and the PWT module is still enabled.

43.2.3 Block diagram

The following figure show the block diagram of the PWT.

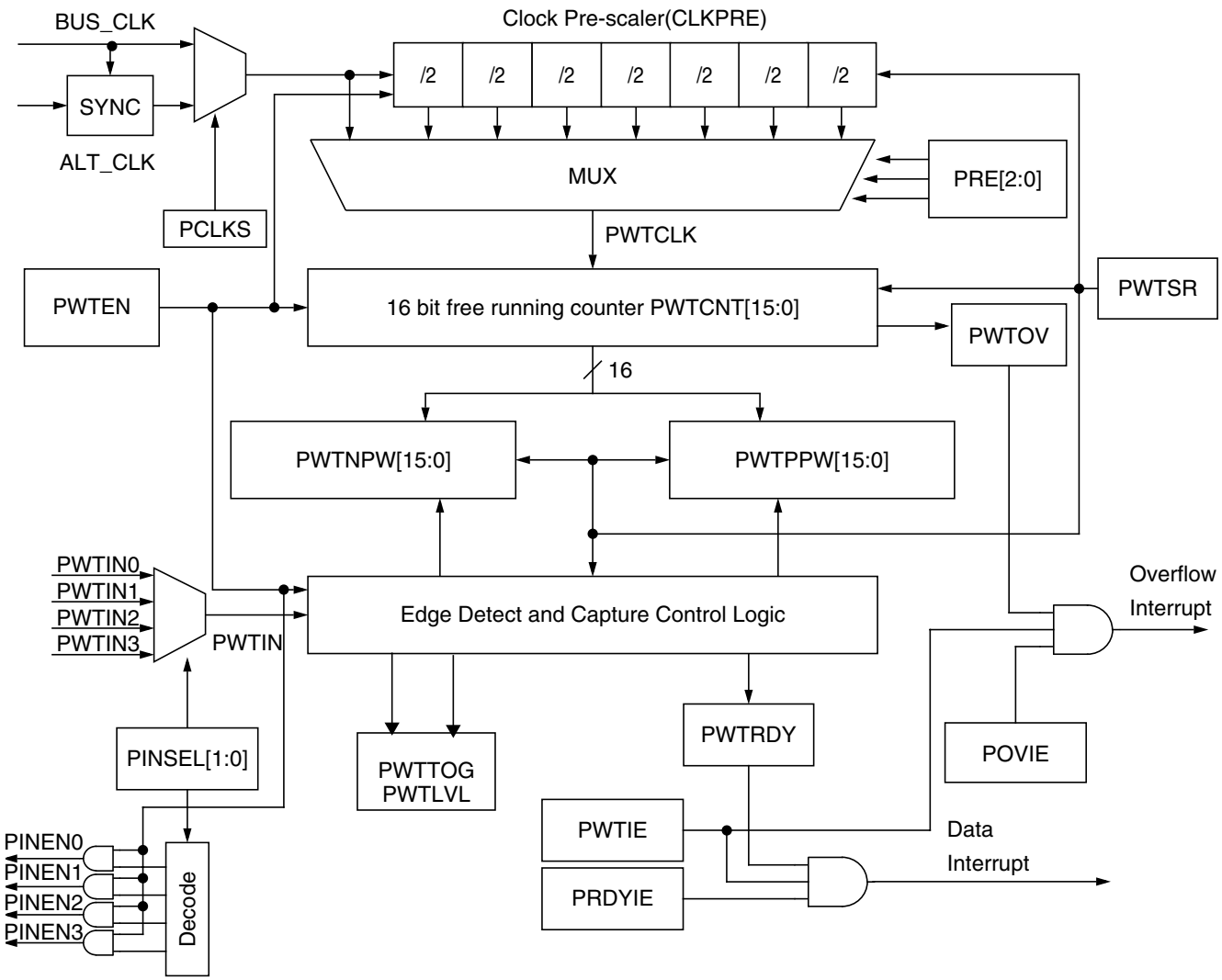


Figure 43-1. Pulse width timer (PWT) block diagram

43.3 External signal description

43.3.1 Overview

PWT has the following signal.

Table 43-2. PWT signal properties

Signal	Pullup	Description	I/O
PWTIN[3:0]	No	Pulse inputs	I
ALTCLK	No	Alternative clock source for the counter	I

43.3.2 PWTIN[3:0] — pulse width timer capture inputs

The input signals are pulse capture inputs which can come from internal or external. The PWT input is selected by PINSEL[1:0] to be routed to the pulse width timer. If the input comes from external and is selected as the PWT input, the input port is enabled for PWT function by PINSEL[1:0] automatically. The minimum pulse width to be measured is 1 PWTCLK cycle, any pulse narrower than this value is ignored by PWT module. The PWTCLK cycle time depends on the PWT clock source selection and pre-scaler rate setting.

43.3.3 ALTCLK— alternative clock source for counter

The PWT has an alternative clock input ALTCLK which can be selected as the clock source of the counter when the PCLKS bit is set. The ALTCLK input must be synchronized by the bus clock. Variations in duty cycle and clock jitter must also be accommodated so that the ALTCLK signal must not exceed one-fourth of the bus frequency. The ALTCLK pin can be shared with a general-purpose port pin. See the Pins and Connections chapter for the pin location and priority of this function.

43.4 Memory Map and Register Descriptions

PWT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_6000	Pulse Width Timer Control and Status Register (PWT_CS)	8	R/W	00h	43.4.1/1149
4005_6001	Pulse Width Timer Control Register (PWT_CR)	8	R/W	00h	43.4.2/1150
4005_6002	Pulse Width Timer Positive Pulse Width Register: High (PWT_PPH)	8	R	00h	43.4.3/1151
4005_6003	Pulse Width Timer Positive Pulse Width Register: Low (PWT_PPL)	8	R	00h	43.4.4/1151
4005_6004	Pulse Width Timer Negative Pulse Width Register: High (PWT_NPH)	8	R	00h	43.4.5/1152
4005_6005	Pulse Width Timer Negative Pulse Width Register: Low (PWT_NPL)	8	R	00h	43.4.6/1152
4005_6006	Pulse Width Timer Counter Register: High (PWT_CNTH)	8	R	00h	43.4.7/1153
4005_6007	Pulse Width Timer Counter Register: Low (PWT_CNTL)	8	R	00h	43.4.8/1153

43.4.1 Pulse Width Timer Control and Status Register (PWT_CS)

Address: 4005_6000h base + 0h offset = 4005_6000h

Bit	7	6	5	4	3	2	1	0
Read	PWTEN	PWTIE	PRDYIE	POVIE	0	FCTLE	PWTRDY	PWTOV
Write					PWTSR			
Reset	0	0	0	0	0	0	0	0

PWT_CS field descriptions

Field	Description
7 PWTEN	PWT Module Enable Enables/disables the PWT module. To avoid unexpected behavior, do not change any PWT configurations as long as PWTEN is set. 0 The PWT is disabled. 1 The PWT is enabled.
6 PWTIE	PWT Module Interrupt Enable Enables the PWT module to generate an interrupt. 0 Disables the PWT to generate interrupt. 1 Enables the PWT to generate interrupt.
5 PRDYIE	PWT Pulse Width Data Ready Interrupt Enable Enables/disables the PWT to generate an interrupt when PWTRDY is set as long as PWTIE is set. 0 Disable PWT to generate interrupt when PWTRDY is set. 1 Enable PWT to generate interrupt when PWTRDY is set.
4 POVIE	PWT Counter Overflow Interrupt Enable Enables/disables the PWT to generate an interrupt when PWTOV is set due to PWT counter overflow. 0 Disable PWT to generate interrupt when PWTOV is set. 1 Enable PWT to generate interrupt when PWTOV is set.
3 PWTSR	PWT Soft Reset Performs a soft reset to the PWT. This field always reads as 0. 0 No action taken. 1 Writing 1 to this field will perform soft reset to PWT.
2 FCTLE	First counter load enable after enable This bit determines if the counter value should be loaded to the corresponding PWTx_PPW{H,L}, PWTx_NPW{H,L} after first enable. 0 Do not load the first counter values to corresponding registers 1 Load the first coutner value to corresponding registers depended by the PWTIN level

Table continues on the next page...

PWT_CS field descriptions (continued)

Field	Description
1 PWTRDY	PWT Pulse Width Valid Indicates that the PWT Pulse Width register(s) has been updated and is ready to be read. This field is cleared by reading PWTRDY and then writing 0 to PWTRDY bit when PWTRDY is set.Writing 1 to this field has no effect. 0 PWT pulse width register(s) is not up-to-date. 1 PWT pulse width register(s) has been updated.
0 PWTOV	PWT Counter Overflow Indicates that the PWT counter has run from 0x0000_0xFFFF to 0x0000_0x0000. This field is cleared by writing 0 to PWTOV when PWTOV is set. Writing 1 to this field has no effect. If another overflow occurs when this field is being cleared, the clearing fails. 0 PWT counter no overflow. 1 PWT counter runs from 0xFFFF to 0x0000.

43.4.2 Pulse Width Timer Control Register (PWT_CR)

Address: 4005_6000h base + 1h offset = 4005_6001h

Bit	7	6	5	4	3	2	1	0
Read	PCLKS	PINSEL		TGL	LVL	PRE		
Write				w1c				
Reset	0	0	0	0	0	0	0	0

PWT_CR field descriptions

Field	Description
7 PCLKS	PWT Clock Source Selection Controls the selection of clock source for the PWT counter. 0 PWT_CLK is selected as the clock source of PWT counter. 1 Alternative clock is selected as the clock source of PWT counter.
6–5 PINSEL	PWT Pulse Inputs Selection Enables the corresponding PWT input port, if this PWT input comes from an external source. 00 PWTIN[0] is enabled. 01 PWTIN[1] is enabled. 10 PWTIN[2] enabled. 11 PWTIN[3] enabled.
4 TGL	PWTIN states Toggled from last state This flag indicates if the selected PWTIN has toggled its state since last time this bit has cleared to 0.

Table continues on the next page...

PWT_CR field descriptions (continued)

Field	Description
	0 The selected PWTIN hasn't changed its original states from last time. 1 The selected PWTIN has toggled its states.
3 LVL	PWTIN Level when Overflows This Read Only bit signalizes the selected PWTIN states when the coutner overflows to read out.
PRE	PWT Clock Prescaler (CLKPRE) Setting Selects the value by which the clock is divided to clock the PWT counter. 000 Clock divided by 1. 001 Clock divided by 2. 010 Clock divided by 4. 011 Clock divided by 8. 100 Clock divided by 16. 101 Clock divided by 32. 110 Clock divided by 64. 111 Clock divided by 128.

43.4.3 Pulse Width Timer Positive Pulse Width Register: High (PWT_PPH)

Address: 4005_6000h base + 2h offset = 4005_6002h

Bit	7	6	5	4	3	2	1	0
Read	PPWH							
Write								
Reset	0	0	0	0	0	0	0	0

PWT_PPH field descriptions

Field	Description
PPWH	Positive Pulse Width[15:8] High byte of captured positive pulse width value.

43.4.4 Pulse Width Timer Positive Pulse Width Register: Low (PWT_PPL)

Address: 4005_6000h base + 3h offset = 4005_6003h

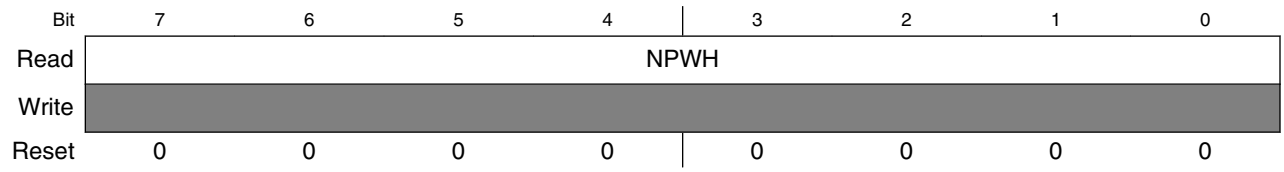
Bit	7	6	5	4	3	2	1	0
Read	PPWL							
Write								
Reset	0	0	0	0	0	0	0	0

PWT_PPL field descriptions

Field	Description
PPWL	Positive Pulse Width[7:0] Low byte of captured positive pulse width value.

43.4.5 Pulse Width Timer Negative Pulse Width Register: High (PWT_NPH)

Address: 4005_6000h base + 4h offset = 4005_6004h

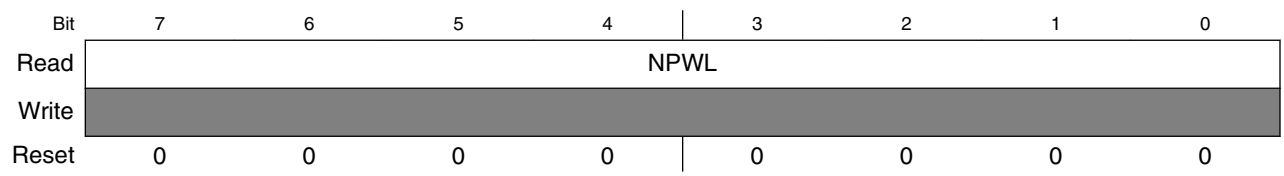


PWT_NPH field descriptions

Field	Description
NPWH	Negative Pulse Width[15:8] High byte of captured negative pulse width value.

43.4.6 Pulse Width Timer Negative Pulse Width Register: Low (PWT_NPL)

Address: 4005_6000h base + 5h offset = 4005_6005h

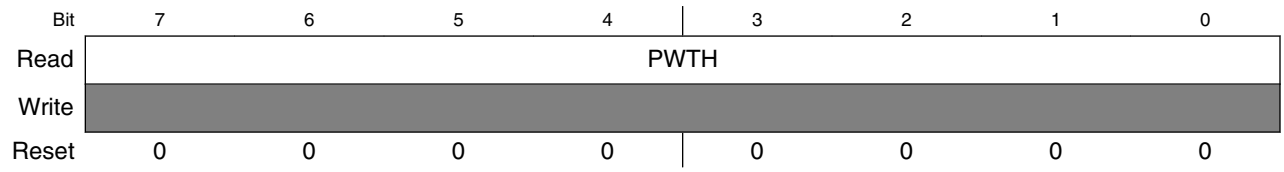


PWT_NPL field descriptions

Field	Description
NPWL	Negative Pulse Width[7:0] Low byte of captured negative pulse width value.

43.4.7 Pulse Width Timer Counter Register: High (PWT_CNTH)

Address: 4005_6000h base + 6h offset = 4005_6006h

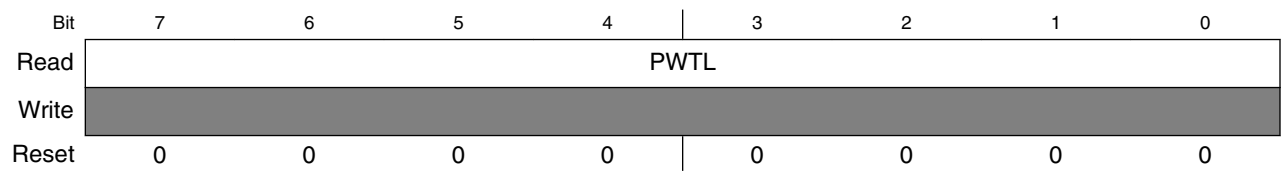


PWT_CNTH field descriptions

Field	Description
PWTH	PWT counter[15:8] High byte of PWT counter register.

43.4.8 Pulse Width Timer Counter Register: Low (PWT_CNTL)

Address: 4005_6000h base + 7h offset = 4005_6007h



PWT_CNTL field descriptions

Field	Description
PWTL	PWT counter[7:0] Low byte of PWT counter register.

43.5 Functional description

43.5.1 PWT counter and PWT clock pre-scaler

The pulse width timer (PWT) measures duration of a pulse or the period of a signal input to the PWTIN by a 16-bit free running counter (PWT_CNTH:L). There is a clock pre-scaler (CLKPRE) in PWT module that provides the frequency divided clock to the PWT_CNTH:L.. The clock pre-scaler can select clock input from bus clock and alternative clock by PWT_CR[PCLKS].

The PWT counter uses the frequency divided clock from CLKPRE for counter advancing. The frequency of pre-scaler is programmable as the clock frequency divided by 1, 2, 4, 8, 16, 32, 64, 128 (depending on the setting of PRE[2:0]).

When PWT_CNT is running, any edge to be measured after the trigger edge causes the value of the PWT_CNT to be uploaded to the appropriate pulse width registers. At the same time, PWT_CNT will be reset to $0000 and the clock pre-scaler output will also be reset together. PWT_CNT will then start advancing again with the input clock. If the PWTxCNT runs from 0xFFFF to 0x0000, the PWTOV bit is set.

43.5.2 Edge detection and capture control

The edge detection and capture control part detects measurement trigger edges and controls when and which pulse width register(s) will be updated.

The edge detection logic determines from which edge appeared on PWTIN the pulse width starts to be measured, when and which pulse width registers should be updated.

The PWTIN can be selected from one of four sources by configuring PINSEL[1:0].

As soon as the PWT is enabled, the 16-bit free counter will begin to count up until a edge transistion on the selected PWTIN. Determined by PWT_CS[FCTLE] and PWTIN state, the counter contents can be uploaded to the corresponding registers.

If PWT_CS[FCTLE] is cleared to 0, the first 16-bit free counter content will just be ignored and not uploaded to neither PWT_PPH:L nor PWT_NPH:L. Otherwise, detemined by current PWTIN state(as signalized by PWT_CR[LVL]), the counter content will be uploaded to PWT_PPH:L if PWT_CR[LVL] is 1 and PWT_NPH:L if PWT_CR[LVL] is 0.

In normal measurement, when the PWT_CS[PWTRDY] is set, software can then read out the positive pulse width and negative pulse width values from PWT_PPH:L and PWT_NPH:L respectively and the selected PWTIN duty ratio can then be calculated. The exception is when overflow happens, software need to check PWT_CR[TGL] and PWT_CR[LVL] to determine if it is low overflow(0 duty ratio) ,high overflow(100% duty ratio), toggled low overflow or toggled high overflow. Below table 1 shows the meaning:

Table 43-3. Abnormal PWTIN duty ratio

Flag	PWT_CR[TGL]	PWT_CR[LVL]	Description
PWT_CS[PWTOV]	0	0	Low overflow
	0	1	High overflow
	1	0	Toggled low overflow

Table continues on the next page...

Table 43-3. Abnormal PWTIN duty ratio (continued)

Flag	PWT_CR[TGL]	PWT_CR[LVL]	Description
	1	1	Toggled high overflow

The following figure illustrates the trigger edge detection and pulse width registers update of PWT.

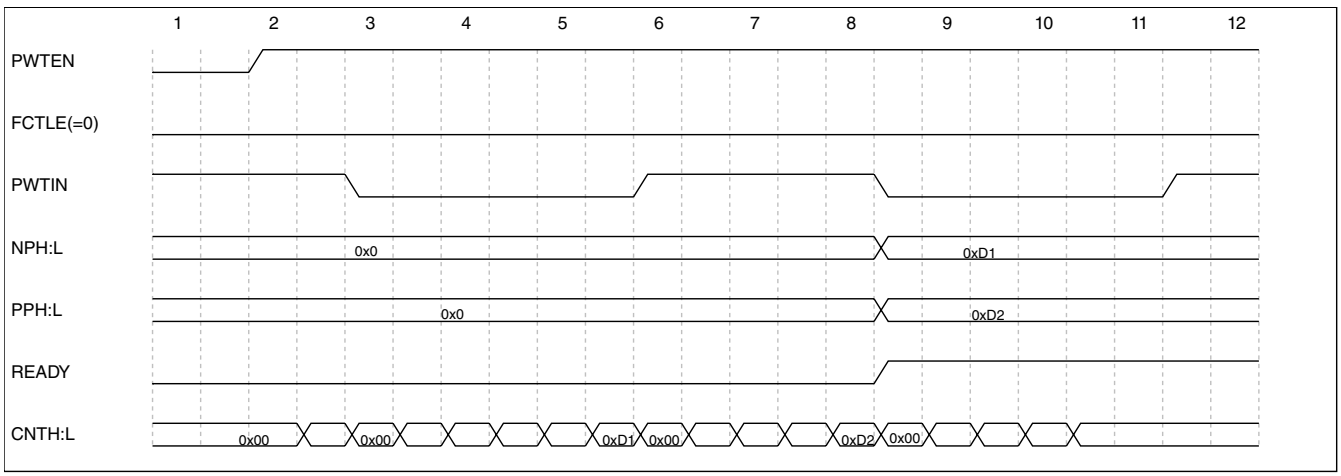


Figure 43-2. PWT normal measurement with FCTLE = 0

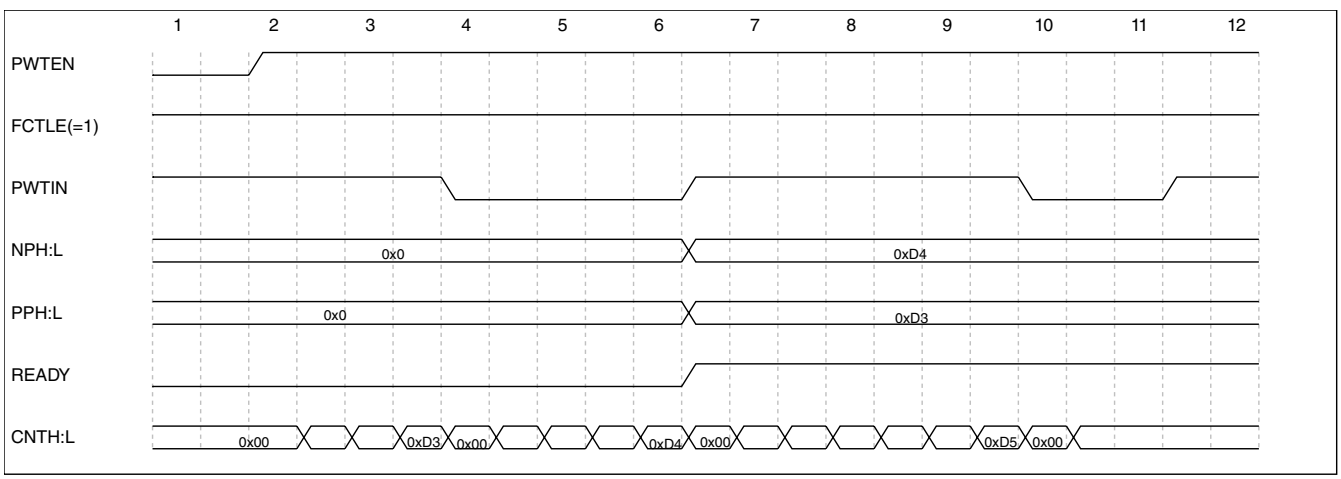


Figure 43-3. PWT normal measurement with FCTLE = 1

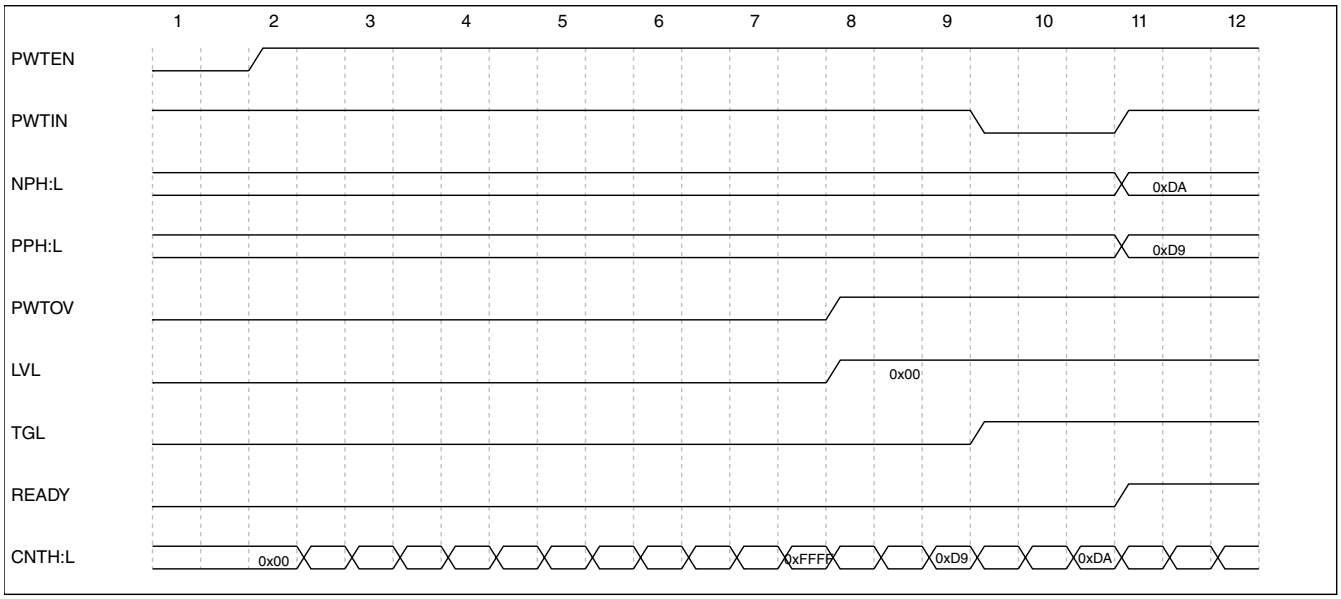


Figure 43-4. PWT measurement overflows at high level with FCTLE = 1

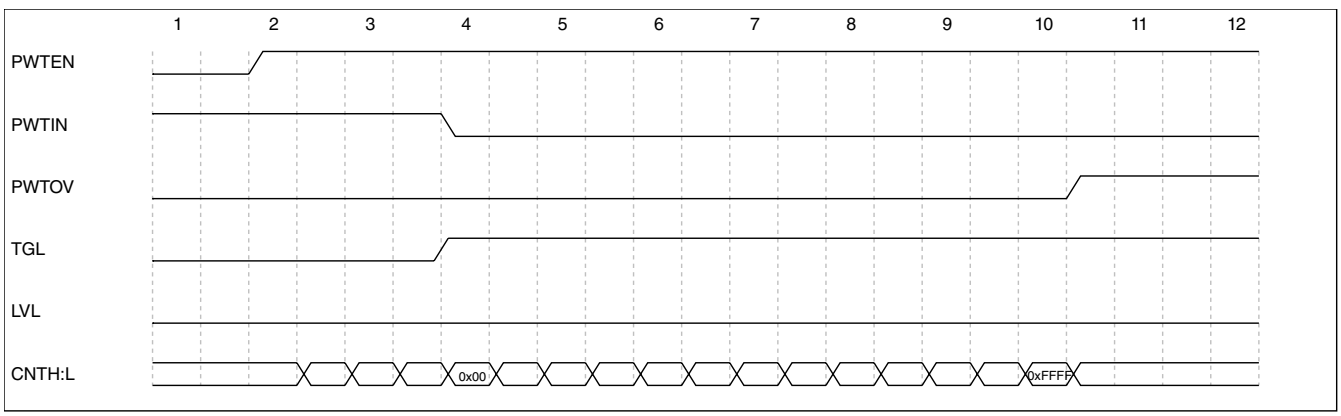


Figure 43-5. PWT measurement overflows with PWTIN toggles

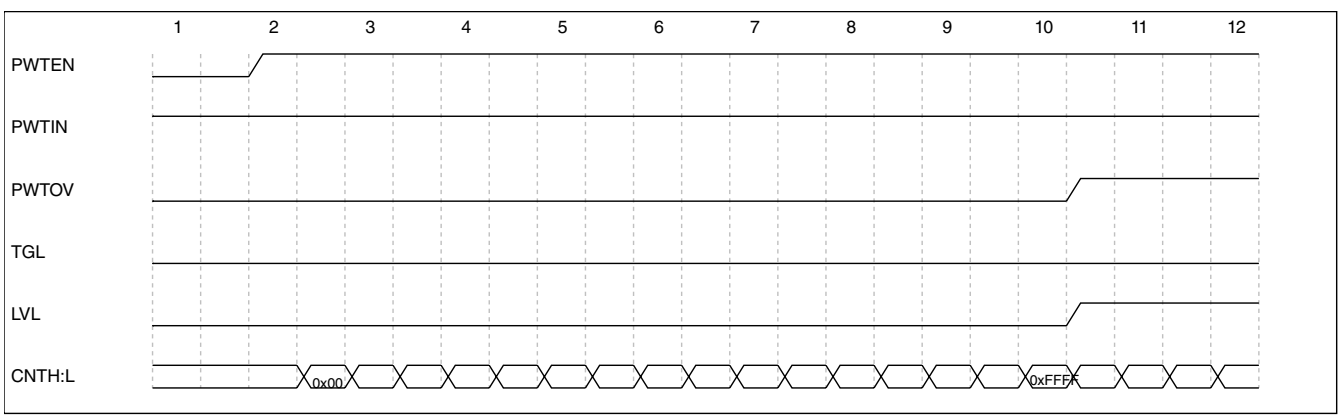


Figure 43-6. PWT measurement overflows without PWTIN toggles

The PWTRDY flag bit indicates that the data can be read in PWTxPPH:L and/or PWTxNPH:L, whenever there is a valid edge transition happened on the selected PWTIN.

When PWTRDY bit is set, the updated pulse width register(s) transfers the data to corresponding 16-bit read buffer(s). The read value of pulse width registers actually comes from the corresponding read buffers, whenever the chip is in normal run mode or BDM mode. Reading followed by writing 0 to the PWTRDY flag clears this bit. Until the PWTRDY bit is cleared, the 16-bit read buffer(s) cannot be updated. But this does not affect the upload of pulse width registers from the PWT counter.

If another pulse measurement is completed and the pulse width registers are updated, the clearing of the PWTRDY flag fails, i.e., the PWTRDY will still be set, but the 16-bit read buffer(s) will be updated again as long as the action is cleared. The user should complete the pulse width data reading before clearing the PWTRDY flag to avoid missing data. This mechanism assures that the second pulse measurement will not be lost in case the MCU does not have enough time to read the first one ready for read. The mechanism is automatically restarted by an MCU reset , writing 1 to PWTSR bit or writing a 0 to PWTEN bit followed by writing a 1 to it.

The following figure illustrates the buffering mechanism of pulse width register:

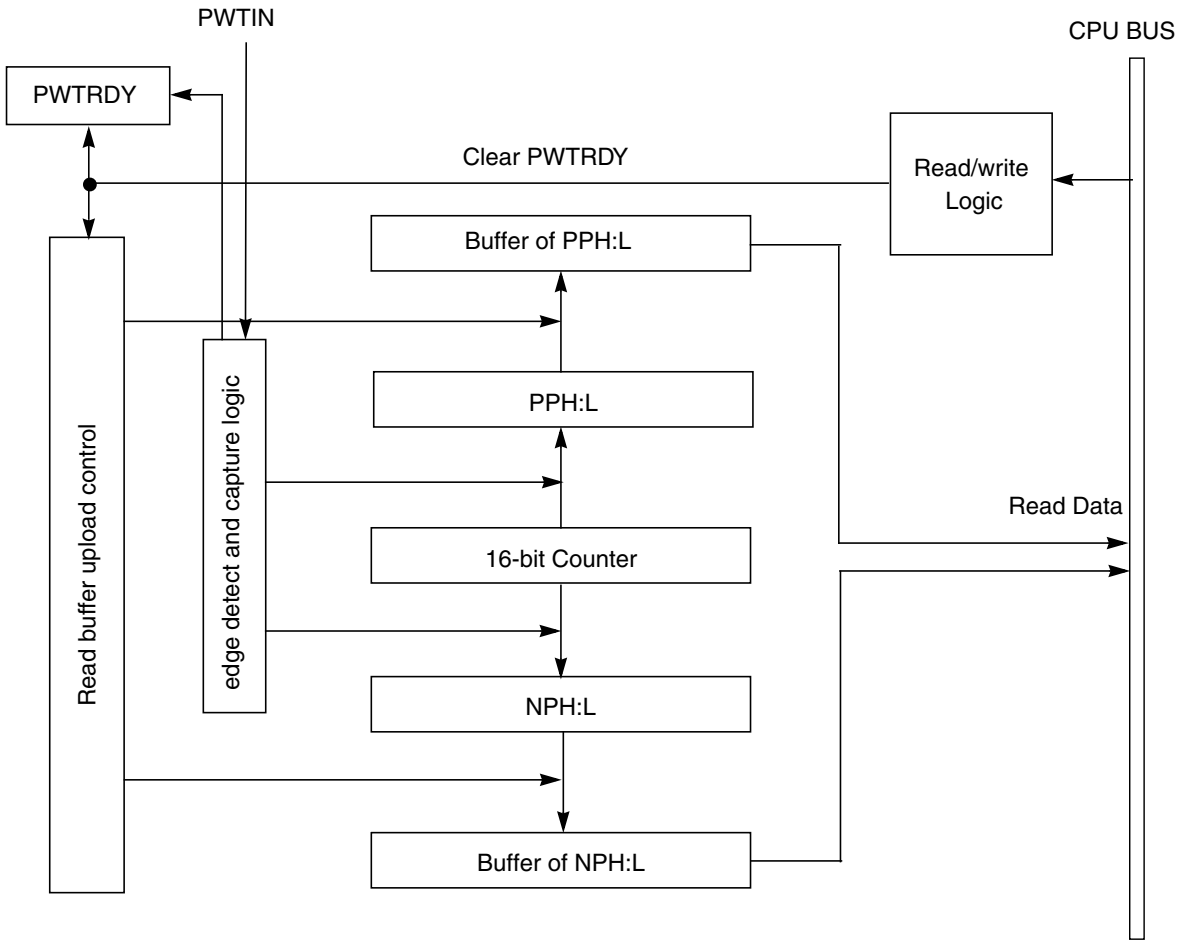


Figure 43-7. Buffering mechanism of pulse width register

When PWT completes any pulse width measurement, a signal is generated to reset PWTxCNTH:L and the clock pre-scaler output after the data has been uploaded to the pulse width registers.. To assure that there is no missing count, the PWTxCNTH:L and the clock pre-scaler output are reset in a bus clock cycle after the completion of a pulse width measurement.

43.6 Reset overview

43.6.1 Description of reset operation

PWT soft reset is built into PWT as a mechanism used to reset/restart the pulse width timer. The PWT soft reset is triggered by writing 1 to the PWTSR bit. (This bit always reads 0). Unlike reset by the CPU, the PWT reset does not restore everything in the PWT to its reset state. The following occurs

1. The PWT counter is set to 0x0000

2. The 16-bit buffer of PWT counter is reset and the reading coherency mechanism is restarted
3. The PWT clock pre-scaler output is reset
4. The edge detection logic is reset
5. The capture logic is reset and the latching mechanism of pulse width registers is also restarted.
6. PWTxPPH, PWTxPPL, PWTxNPH, PWTxNPL are set to 0x00
7. PWTOV, PWTRDY, TGL and LVL status are set to 0
8. All other PWT register settings are not changed

Writing a 0 to PWTEN bit also has the above effects except that the reset state will be held until the PWTEN bit is set to 1.

43.7 Interrupts

43.7.1 Description of interrupt operation

The other major component of the PWT is the interrupts control logic. When the PWTOV bit and POVIE bit of PWTxCS are set, a PWT overflow interrupt can be generated. When PWTRDY bit and PRDYIE bit of PWTxCS are set, a pulse width data ready interrupt can be generated. The PWTIE bit of PWTxCS controls the interrupt generation of the PWT module. The functionality of the PWT is not affected while the interrupt is being generated.

43.7.2 Application examples

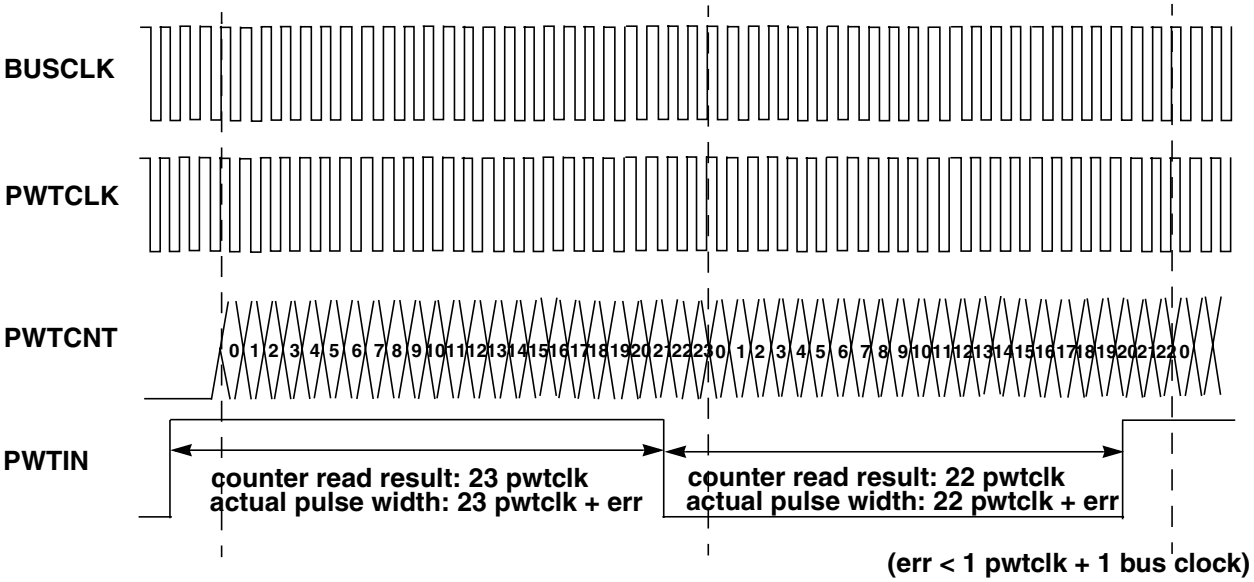


Figure 43-8. Example at PWTCLK is bus clock divided by 1

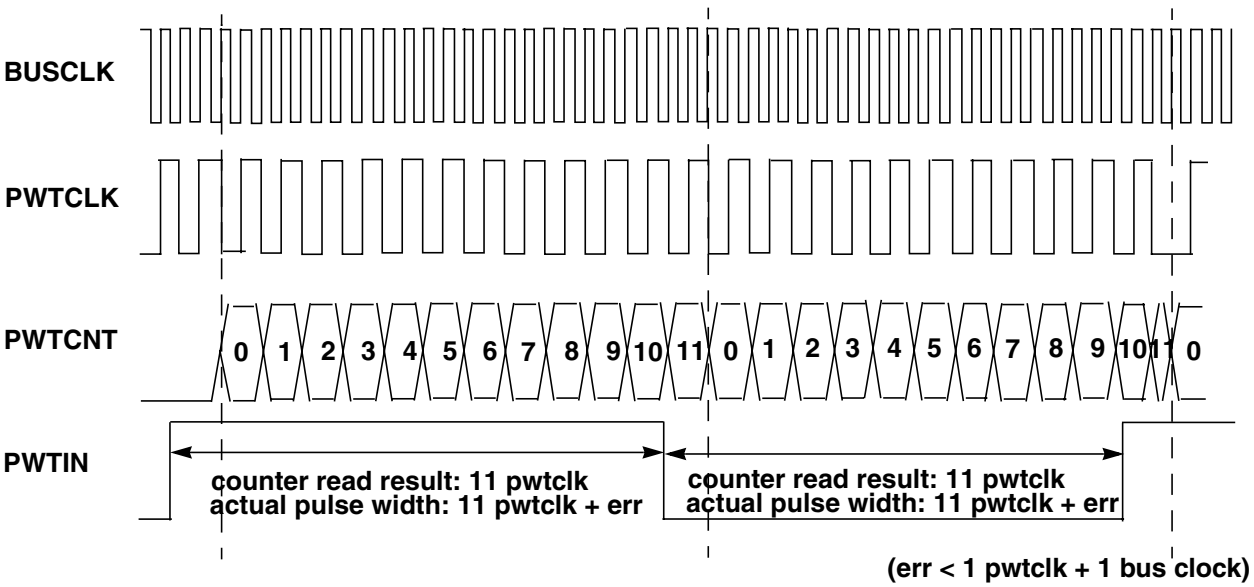


Figure 43-9. Example at PWTCLK is Bus Clock divided by 2

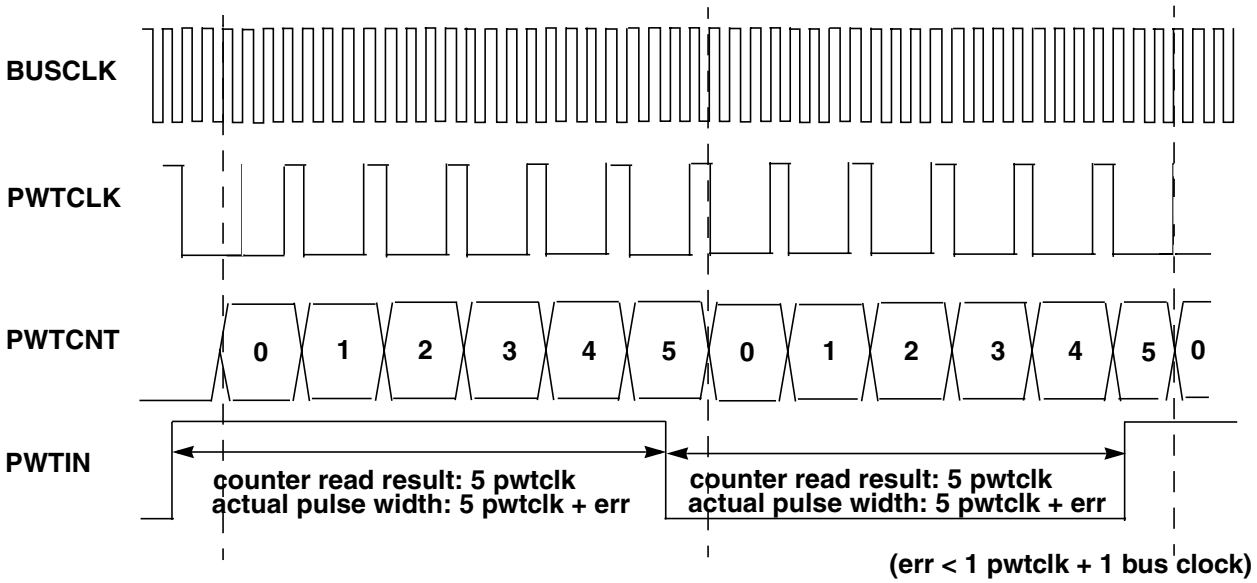


Figure 43-10. Example at PWTCLK is bus clock divided by 4

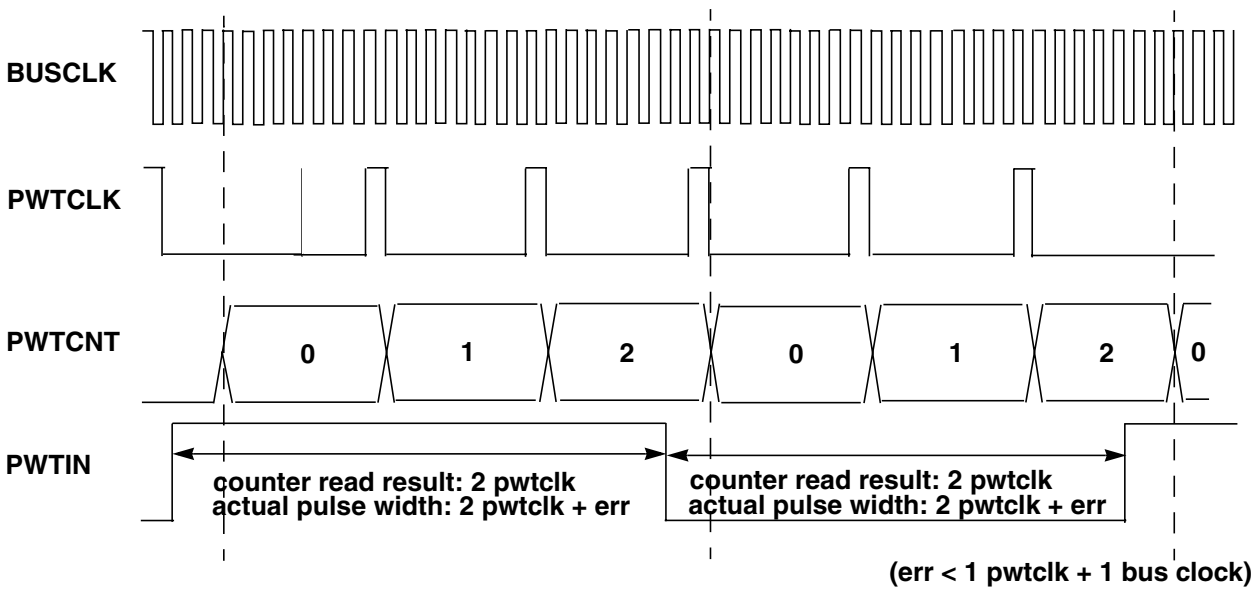


Figure 43-11. Example at PWTCLK is bus clock divided by 8

43.8 Initialization/Application information

Following are the recommended steps to initialize the PWT module:

1. Configure PWTxCR to select clock source, set pre-scaler rate, select PWT input pin and edge detection mode.
2. Set PWTIE, PRDYIE and POVIE bits in PWTxCS if corresponding interrupt is desired to be generated.
3. Set PWTEN bit in PWTxCS to enable the pulse width measurement.

The step 1 and 2 can be sequential or not, but they must be completed before step 3 to ensure all settings are ready before pulse width measurement is enabled.

43.9 Usage Guide

PWT provides an accurate signal frequency measurement for both the positive and negative portions of a periodic signal, useful for applications such as motor control. In conjunction with a Pulse Width Modulated signal it can effectively be used to implement a highly accurate closed loop motor control system, or any other system in which it might be necessary to measure a periodic signal frequency and duty cycle, providing not only accuracy but also high flexibility.

43.9.1 Edge detection, capture control and period measurement

PWT typical usage is external signal input capture and time period measurement.

Example: PWT input channel 1 capture external signal and measure its time period

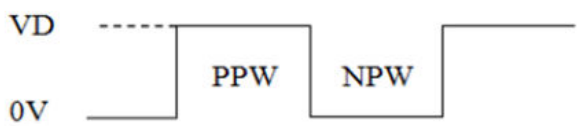


Figure Input pulse capture

The frequency (Hz) is PWT Clock / (PPW ? NPW), PWT Clock = PWT clock source / presacler.

- Enable the PWT module clock;
- Reset the timer channels and registers;
- Configure not to load the first counter values to corresponding registers, enable the PWT interrupt;
- Select bus clock as clock source and enable PWT_IN1 as input source;
- Set the module enable bit to start PWT;
- Wait for the pulse width valid flag (PWTRDY) in interrupt routinue, then get the positive and negative value(PPW, NPW) to calculate the period.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(PWT);
PWT_CS|= PWT_CS_PWTSR_MASK;
PWT_CS|= PWT_CS_FCTLE(0)|PWT_CS_PWTIE_MASK|PWT_CS_PRDYIE_MASK;
```

```
PWT_CR|= PWT_CR_PCLKS(0)|PWT_CR_PRE(0)|PWT_CR_PINSEL(1);
PWT_CS|= PWT_CS_PWTEN_MASK;
EnableIRQ(PWT_IRQ);
```

Chapter 44
Low Power Timer (LPTMR)

44.1 Chip-specific information for this module

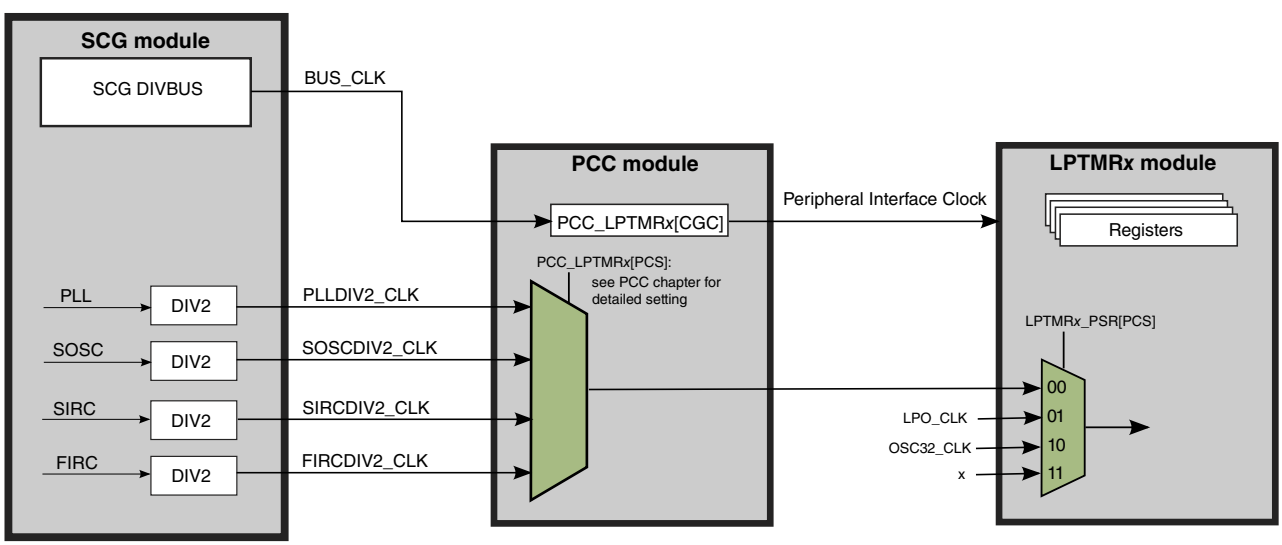
44.1.1 Instantiation Information

This device contains one LPTMR module with 1-channel, 16-bit pulse counter.

44.1.2 LPTMR Clocking Information

The following figure shows the input clock sources available for this module.

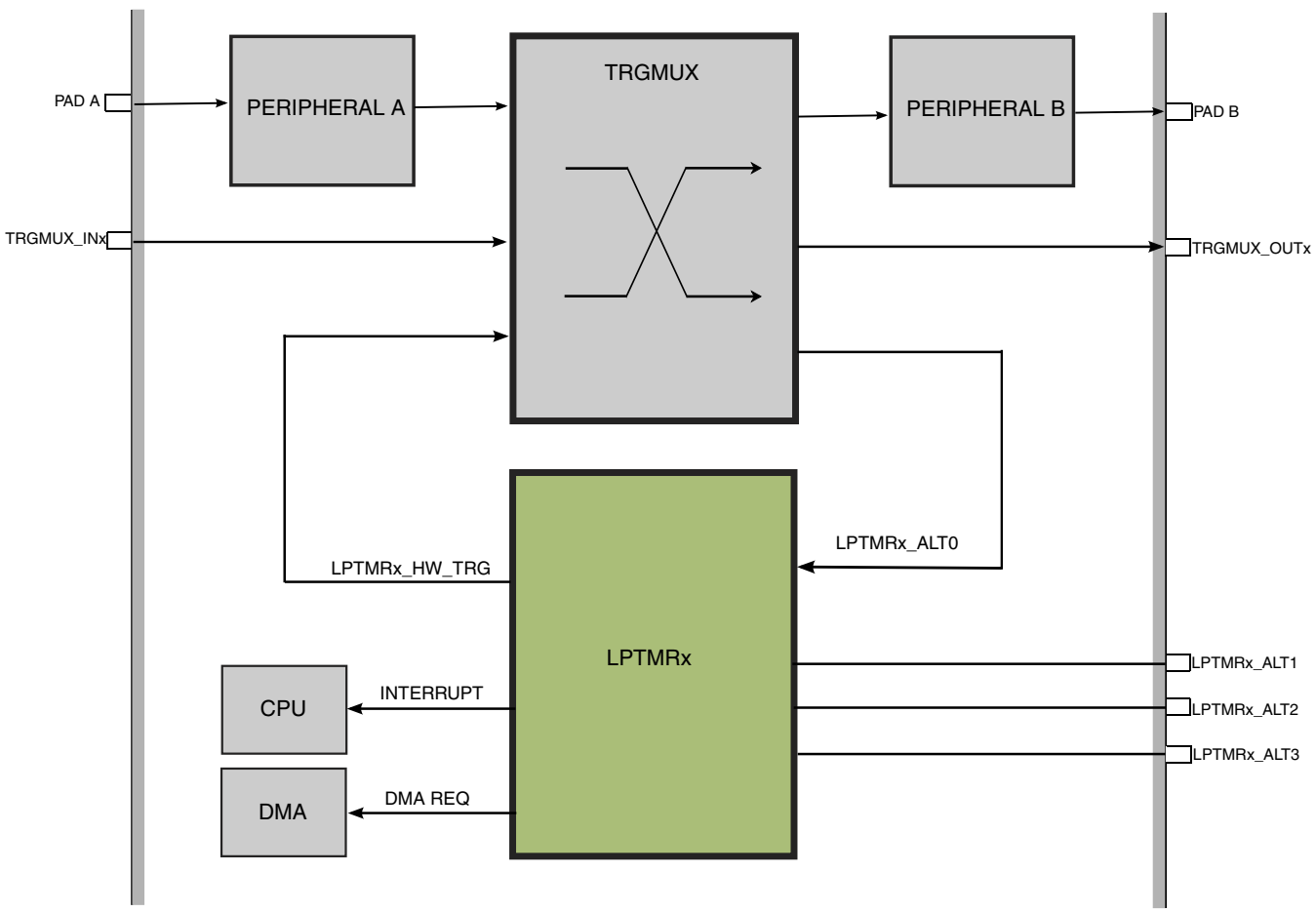
Peripheral Clocking - LPTMR



44.1.3 Inter-connectivity Information

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

LPTMRx_CSR[TPS]	Pulse counter input number	Chip input
00	0	TRGMUX output
01	1	LPTMR0_ALT1 pin
10	2	LPTMR0_ALT2 pin
11	3	LPTMR0_ALT3 pin



44.2 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. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

44.2.1 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

44.2.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 44-1. Modes of operation

Modes	Description
Run	The LPTMR operates normally.
Wait	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Stop	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Debug	The LPTMR operates normally in Pulse Counter mode, but the counter does not increment in Time Counter mode.

44.3 LPTMR signal descriptions

Table 44-2. LPTMR signal descriptions

Signal	I/O	Description
LPTMR_ALT*n*	I	Pulse Counter Input pin

44.3.1 Detailed signal descriptions

Table 44-3. LPTMR interface—detailed signal descriptions

Signal	I/O	Description	
LPTMR_ALT*n*	I	Pulse Counter Input The LPTMR can select one of the input pins to be used in Pulse Counter mode.	
		State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment. Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.
		Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.

44.4 Memory map and register definition

NOTE

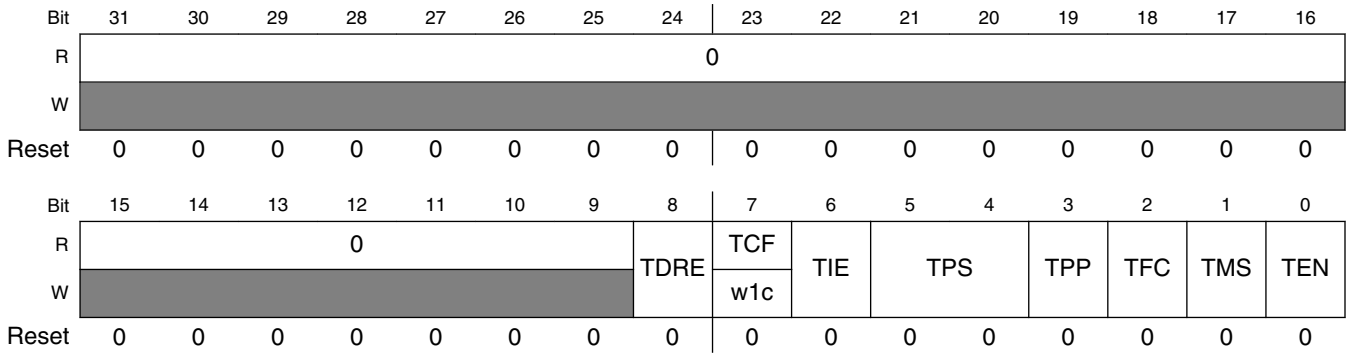
The LPTMR registers are reset only on a POR or LVD event.
See LPTMR power and reset for more details.

LPTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_0000	Low Power Timer Control Status Register (LPTMR0_CSR)	32	R/W	0000_0000h	44.4.1/1169
4004_0004	Low Power Timer Prescale Register (LPTMR0_PSR)	32	R/W	0000_0000h	44.4.2/1170
4004_0008	Low Power Timer Compare Register (LPTMR0_CMR)	32	R/W	0000_0000h	44.4.3/1172
4004_000C	Low Power Timer Counter Register (LPTMR0_CNR)	32	R/W	0000_0000h	44.4.4/1172

44.4.1 Low Power Timer Control Status Register (LPTMRx_CSR)

Address: 4004_0000h base + 0h offset = 4004_0000h



LPTMRx_CSR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 TDRE	Timer DMA Request Enable When TDRE is set, the LPTMR DMA Request is generated whenever TCF is also set and the TCF is cleared when the DMA Controller is done. 0 Timer DMA Request disabled. 1 Timer DMA Request enabled.
7 TCF	Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it. 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 chip configuration information about 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.

Table continues on the next page...

LPTMRx_CSR field descriptions (continued)

Field	Description
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.

44.4.2 Low Power Timer Prescale Register (LPTMRx_PSR)

Address: 4004_0000h base + 4h offset = 4004_0004h

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								PRESCALE					PBYP	PCS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPTMRx_PSR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–3 PRESCALE	Prescale Value

Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
	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. 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.

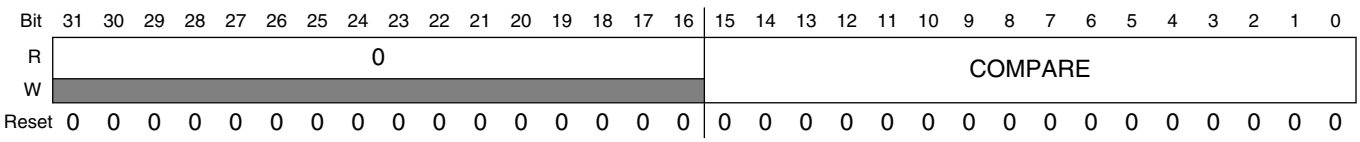
Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
	10 Prescaler/glitch filter clock 2 selected. 11 Prescaler/glitch filter clock 3 selected.

44.4.3 Low Power Timer Compare Register (LPTMRx_CMR)

Address: 4004_0000h base + 8h offset = 4004_0008h

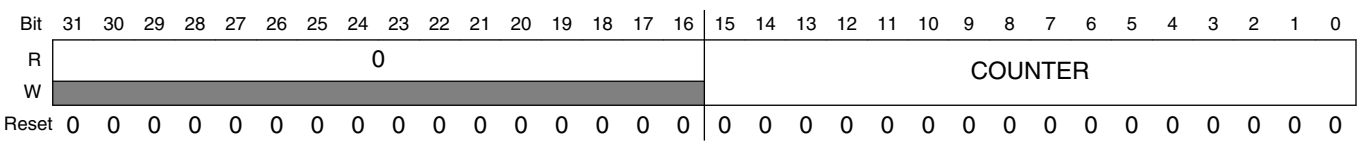


LPTMRx_CMR field descriptions

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.

44.4.4 Low Power Timer Counter Register (LPTMRx_CNR)

Address: 4004_0000h base + Ch offset = 4004_000Ch



LPTMRx_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 The CNR returns the current value of the LPTMR counter at the time this register was last written.

44.5 Functional description

44.5.1 LPTMR power and reset

The LPTMR remains powered in all power 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.

44.5.2 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 may need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency f_{LPTMR} defined in the device datasheet.

44.5.3 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.

44.5.3.1 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.

44.5.3.2 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.

44.5.3.3 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.

44.5.3.4 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.

44.5.4 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.

44.5.5 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.

When the core is halted in Debug mode:

- If configured for Pulse Counter mode, the CNR continues incrementing.
- If configured for Time Counter mode, the CNR stops incrementing.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

44.5.6 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.

44.5.7 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, provided the LPTMR is enabled as a wakeup source.

44.6 Usage Guide

LPTMR is very useful in low power situations. It can be used as a wake-up timer to wake the MCU out of sleep modes after a certain amount of time. If used as pulse counter mode with the glitch filter enabled, then there is no need for a clock to be on. The MCU can wakeup based on counting pulses.

44.6.1 Time Counter mode

The typical usage of LPTMR is as Time Counter mode to generate periodic trigger pulses and interrupts.

Example: LPTMR trigger a periodic interrupt every 1 second

- Enable the LPTMR module clock;

- Configure LPTMR to Timer counter mode by default, use LPO 128K as clock source, bypass the prescaler;
- Set the compare value register to 1 second value;
- Enable timer interrupt ;
- Starts the timer counting after all configuration;
- In the interrupt routine, clear the channel compare flag TCF every 1 second.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPTMR0);
LPTMR0_CSR = 0;
LPTMR0_PSR |= LPTMR_PSR_PBYP_MASK|LPTMR_PSR_PCS(1);
LPTMR0_CMR = ONE_SECOND_VALUE;
LPTMR0_CSR |= LPTMR_CSR_TIE_MASK;
EnableIRQ(LPTMR0_IRQn);
LPTMR0_CSR |= LPTMR_CSR_TEN_MASK;
```

44.6.2 Pulse Counter mode

LPTMR another option is used as Pulse Counter mode to count the input pulses.

Example: LPTMR count the input pulses on LPTMR0_ALT1 pin

- Enable the LPTMR module clock;
- Configure LPTMR to Pulse counter mode, use LPO 128K as clock source, bypass the glitch filter
- Set the compare value register to the value you want to compare the numbers of pulse
- Enable the pulse counter input enable on LPTMR0_ALT1
- Enable timer interrupt
- Starts the pulse counting after all configuration;
- In the interrupt routine, clear the channel compare flag TCF when the counter reaches the value in compare register;

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPTMR0);
LPTMR0_CSR |= LPTMR_CSR_TPS(1)|LPTMR_CSR_TMS_MASK;
LPTMR0_PSR |= LPTMR_PSR_PBYP_MASK|LPTMR_PSR_PCS(1);
LPTMR0_CMR = PULSE_COMPARE_VALUE;
LPTMR0_CSR |= LPTMR_CSR_TIE_MASK;
EnableIRQ(LPTMR0_IRQn);
LPTMR0_CSR |= LPTMR_CSR_TEN_MASK;
```

Chapter 45
Real Time Clock (RTC)

45.1 Chip-specific information for this module

45.1.1 RTC Instantiation

Low range OSC32 (30–40 kHz) will be used as the clock source option for RTC.

NOTE

The wakeup pin is not available for RTC on this device, therefore the related register bitfields are not applicable (e.g. RTC_CR[WPS], RTC_CR[WPE], and RTC_IER[WPON]).

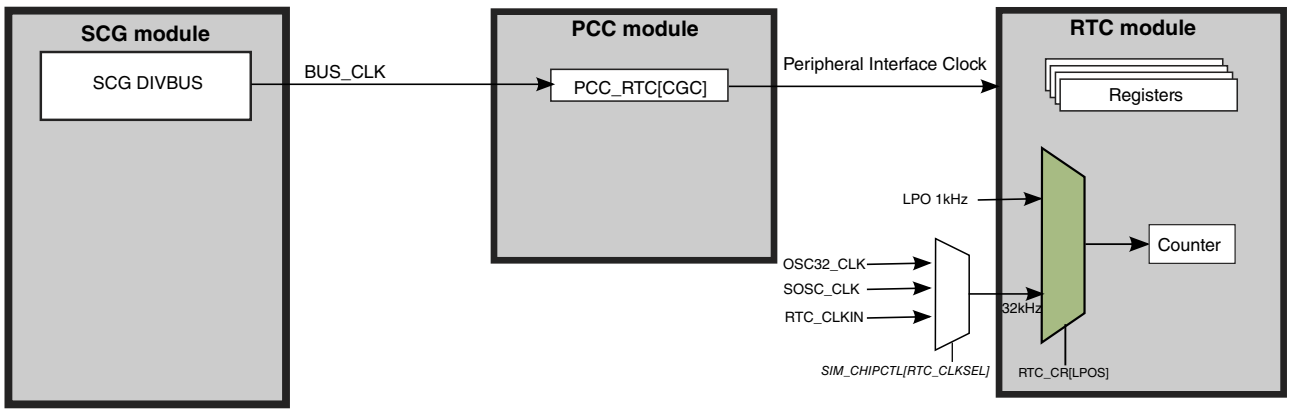
NOTE

Also there is no integrated capacitor for this device, therefore no tunable capacitors (included in the crystal oscillator) can be configured by software.

45.1.2 RTC Clocking Information

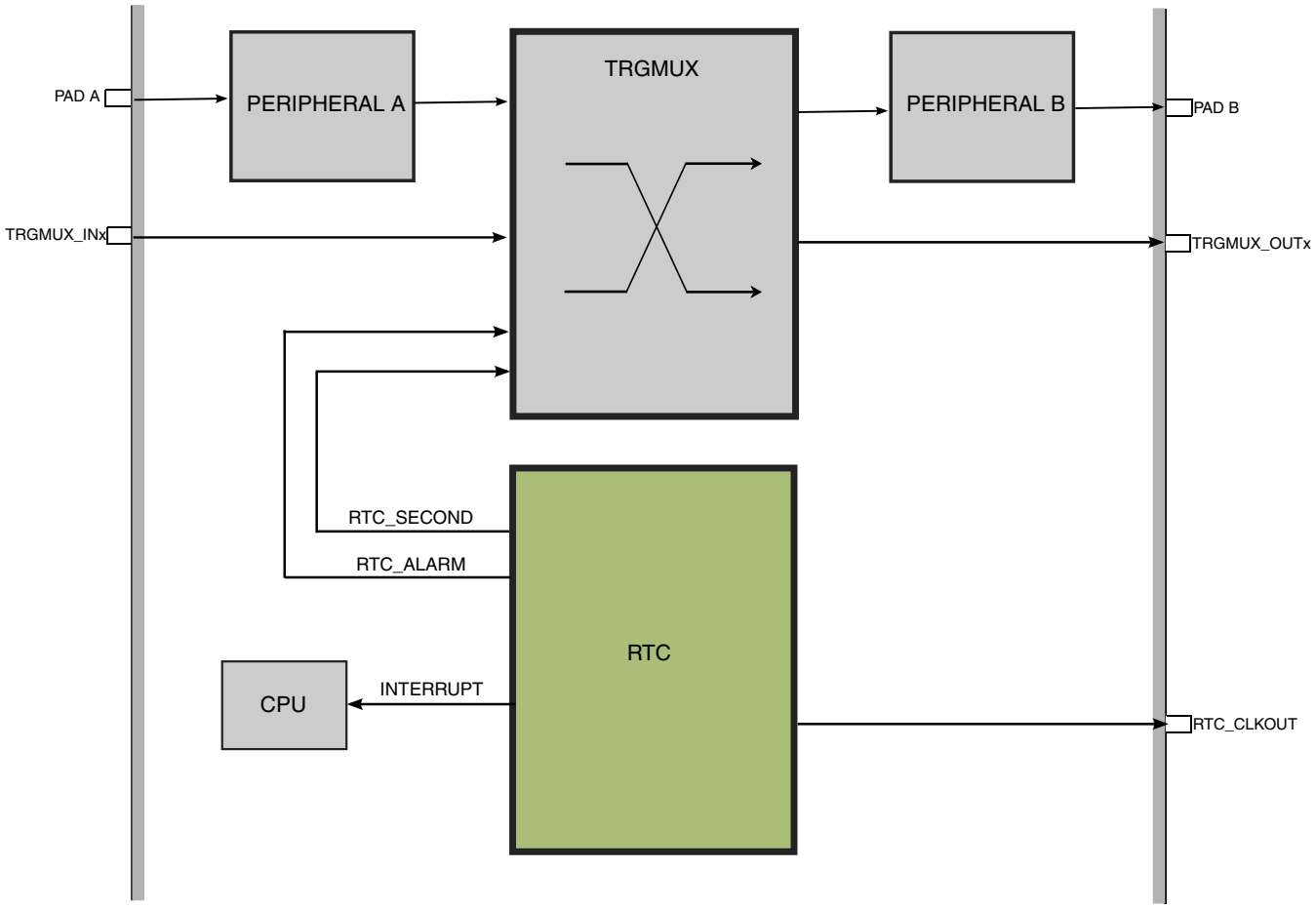
The following figure shows the input clock sources available for this module.

Peripheral Clocking - RTC



45.1.3 Inter-connectivity Information

The SRTC inter-connectivity is shown in following diagram..



45.2 Introduction

45.2.1 Features

The RTC module features include:

- 32-bit seconds counter with roll-over protection and 32-bit alarm
- 16-bit prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm
- Option to increment prescaler using the LPO (prescaler increments by 32 every clock edge)
- Register write protection

- Lock register requires POR or software reset to enable write access
- Access control registers require system reset to enable read and/or write access

- Configurable 1, 2, 4, 8, 16, 32, 64 or 128 Hz square wave output with optional interrupt

45.2.2 Modes of operation

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

45.2.3 RTC signal descriptions

Table 45-1. RTC signal descriptions

Signal	Description	I/O
RTC_CLKOUT	Prescaler square-wave output or 32kHz crystal clock	O

45.2.3.1 RTC clock output

The RTC_CLKOUT signal can output either a square wave prescaler output (configurable to 1, 2, 4, 8, 16, 32, 64 or 128 Hz) or the 32 kHz crystal clock.

45.3 Register definition

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

Write accesses to any register by non-supervisor mode software, when the supervisor access bit in the control register is clear, will terminate with a bus error.

Read accesses by non-supervisor mode software complete as normal.

Writing to a register protected by the write access register or lock register does not generate a bus error, but the write will not complete.

Reading a register protected by the read access register does not generate a bus error, but the register will read zero.

RTC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_D000	RTC Time Seconds Register (RTC_TSR)	32	R/W	0000_0000h	45.3.1/1183
4003_D004	RTC Time Prescaler Register (RTC_TPR)	32	R/W	0000_0000h	45.3.2/1183
4003_D008	RTC Time Alarm Register (RTC_TAR)	32	R/W	0000_0000h	45.3.3/1184
4003_D00C	RTC Time Compensation Register (RTC_TCR)	32	R/W	0000_0000h	45.3.4/1184
4003_D010	RTC Control Register (RTC_CR)	32	R/W	0000_0000h	45.3.5/1186
4003_D014	RTC Status Register (RTC_SR)	32	R/W	0000_0001h	45.3.6/1188
4003_D018	RTC Lock Register (RTC_LR)	32	R/W	0000_00FFh	45.3.7/1189
4003_D01C	RTC Interrupt Enable Register (RTC_IER)	32	R/W	0000_0007h	45.3.8/1190
4003_D800	RTC Write Access Register (RTC_WAR)	32	R/W	0000_00FFh	45.3.9/1192
4003_D804	RTC Read Access Register (RTC_RAR)	32	R/W	0000_00FFh	45.3.10/ 1193

45.3.1 RTC Time Seconds Register (RTC_TSR)

Address: 4003_D000h base + 0h offset = 4003_D000h

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

RTC_TSR field descriptions

Field	Description
TSR	Time Seconds Register When the time counter is enabled, the TSR is read only and increments once a second provided SR[TOF] or SR[TIF] are not set. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled will clear the SR[TOF] and/or the SR[TIF]. Writing to TSR with zero is supported, but not recommended because TSR will read as zero when SR[TIF] or SR[TOF] are set (indicating the time is invalid).

45.3.2 RTC Time Prescaler Register (RTC_TPR)

Address: 4003_D000h base + 4h offset = 4003_D004h

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

RTC_TPR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TPR	Time Prescaler Register When the time counter is enabled, the TPR is read only and increments every 32.768 kHz clock cycle. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TPR can be read or written. The TSR[TSR] increments when bit 14 of the TPR transitions from a logic one to a logic zero.

45.3.3 RTC Time Alarm Register (RTC_TAR)

Address: 4003_D000h base + 8h offset = 4003_D008h

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

RTC_TAR field descriptions

Field	Description
TAR	Time Alarm Register When the time counter is enabled, the SR[TAF] is set whenever the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. Writing to the TAR clears the SR[TAF].

45.3.4 RTC Time Compensation Register (RTC_TCR)

Address: 4003_D000h base + Ch offset = 4003_D00Ch

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

RTC_TCR field descriptions

Field	Description
31–24 CIC	Compensation Interval Counter Current value of the compensation interval counter. If the compensation interval counter equals zero then it is loaded with the contents of the CIR. If the CIC does not equal zero then it is decremented once a second.
23–16 TCV	Time Compensation Value

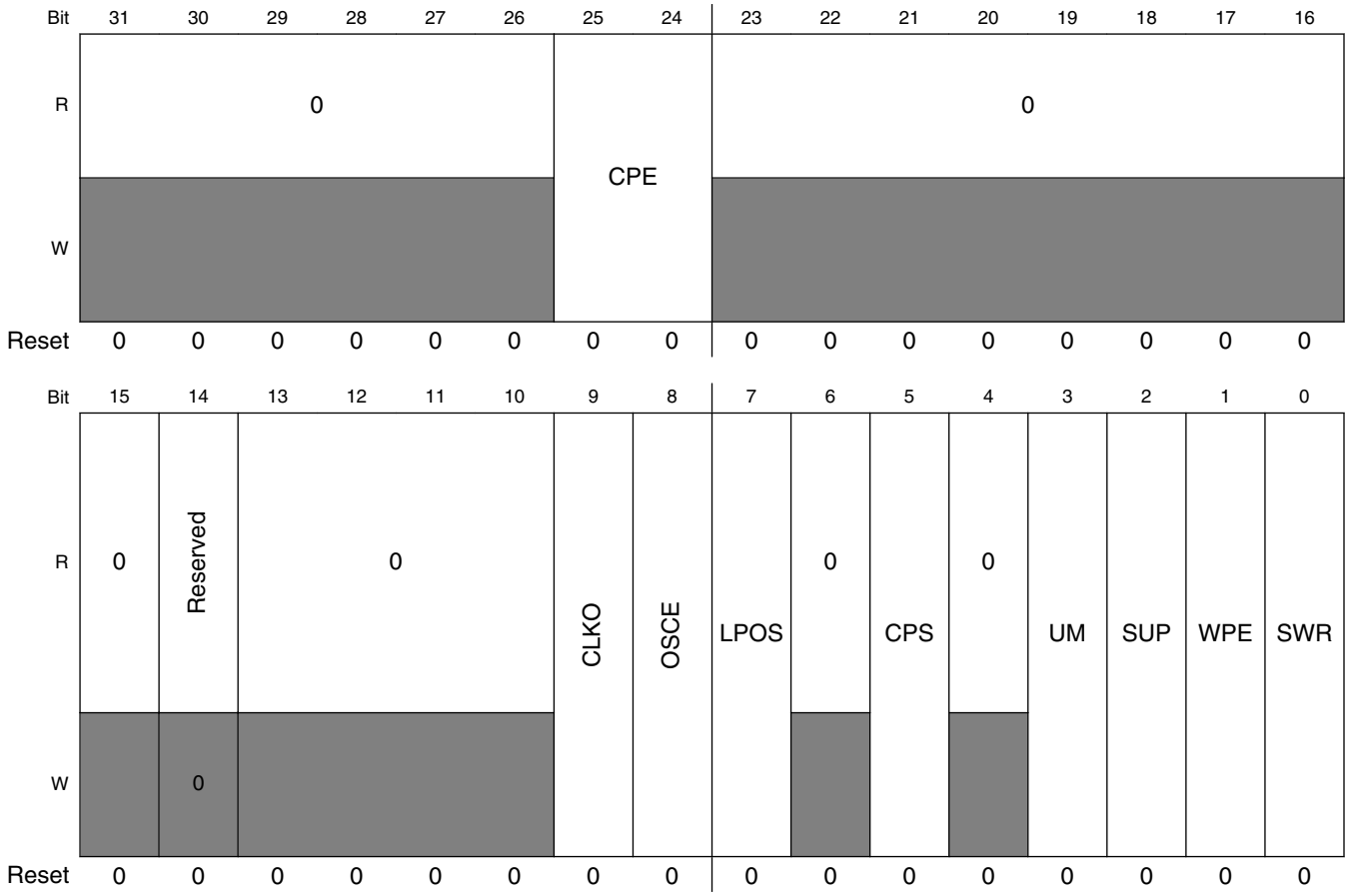
Table continues on the next page...

RTC_TCR field descriptions (continued)

Field	Description
	Current value used by the compensation logic for the present second interval. Updated once a second if the CIC equals 0 with the contents of the TCR field. If the CIC does not equal zero then it is loaded with zero (compensation is not enabled for that second increment).
15–8 CIR	Compensation Interval Register Configures the compensation interval in seconds from 1 to 256 to control how frequently the TCR should adjust the number of 32.768 kHz cycles in each second. The value written should be one less than the number of seconds. For example, write zero to configure for a compensation interval of one second. This register is double buffered and writes do not take affect until the end of the current compensation interval.
TCR	Time Compensation Register Configures the number of 32.768 kHz clock cycles in each second. This register is double buffered and writes do not take affect until the end of the current compensation interval. 80h Time Prescaler Register overflows every 32896 clock cycles. ... ... FFh Time Prescaler Register overflows every 32769 clock cycles. 00h Time Prescaler Register overflows every 32768 clock cycles. 01h Time Prescaler Register overflows every 32767 clock cycles. .... 7Fh Time Prescaler Register overflows every 32641 clock cycles.

45.3.5 RTC Control Register (RTC_CR)

Address: 4003_D000h base + 10h offset = 4003_D010h



RTC_CR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 CPE	Clock Pin Enable **NOTE:** The CPE field should be configured to 01 or 11 (i.e. CPE[0] = 1), if we want the RTC_CLKOUT signal as output. 00 RTC_CLKOUT is disabled. 01 RTC_CLKOUT is enabled. 10 Reserved. 11 Reserved.
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. It must always be written to 0.

Table continues on the next page...

RTC_CR field descriptions (continued)

Field	Description
13–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 CLKO	Clock Output 0 The 32 kHz clock is allowed to output on RTC_CLKOUT. 1 The 32 kHz clock is not allowed to output on RTC_CLKOUT.
8 OSCE	Oscillator Enable 0 32.768 kHz oscillator is disabled. 1 32.768 kHz oscillator is enabled. After setting this bit, wait the oscillator startup time before enabling the time counter to allow the 32.768 kHz clock time to stabilize.
7 LPOS	LPO Select When set, the RTC prescaler increments using the LPO clock and not the RTC 32 kHz clock. The LPO increments the prescaler from bit TPR[5] (TPR[4:0] are ignored), supporting close to 1 second increment of the seconds register. Although compensation is supported when clocked from the LPO, TCR[4:0] of the compensation register are also ignored and only TCR[7:5] set the compensation value (can overflow after 1020 to 1027 cycles). 0 RTC prescaler increments using 32 kHz clock. 1 RTC prescaler increments using LPO, bits [4:0] of the prescaler are bypassed.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 CPS	Clock Pin Select 0 The prescaler output clock (as configured by TSIC) is output on RTC_CLKOUT. 1 The RTC 32kHz clock is output on RTC_CLKOUT.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 UM	Update Mode Allows SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can always be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear. 0 Registers cannot be written when locked. 1 Registers can be written when locked under limited conditions.
2 SUP	Supervisor Access 0 Non-supervisor mode write accesses are not supported and generate a bus error. 1 Non-supervisor mode write accesses are supported.
1 WPE	Wakeup Pin Enable The wakeup pin is optional and not available on all devices. 0 Wakeup pin is disabled. 1 Wakeup pin is enabled and wakeup pin asserts if the RTC interrupt asserts or the wakeup pin is turned on.
0 SWR	Software Reset

Table continues on the next page...

RTC_CR field descriptions (continued)

Field	Description
	0 No effect. 1 Resets all RTC registers except for the SWR bit and the RTC_WAR and RTC_RAR registers . The SWR bit is cleared by POR and by software explicitly clearing it.

45.3.6 RTC Status Register (RTC_SR)

Address: 4003_D000h base + 14h offset = 4003_D014h

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

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

RTC_SR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 TCE	Time Counter Enable When time counter is disabled the TSR register and TPR register are writeable, but do not increment. When time counter is enabled the TSR register and TPR register are not writeable, but increment. 0 Time counter is disabled. 1 Time counter is enabled.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 TAF	Time Alarm Flag Time alarm flag is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. This bit is cleared by writing the TAR register. 0 Time alarm has not occurred. 1 Time alarm has occurred.
1 TOF	Time Overflow Flag Time overflow flag is set when the time counter is enabled and overflows. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0 Time overflow has not occurred. 1 Time overflow has occurred and time counter is read as zero.

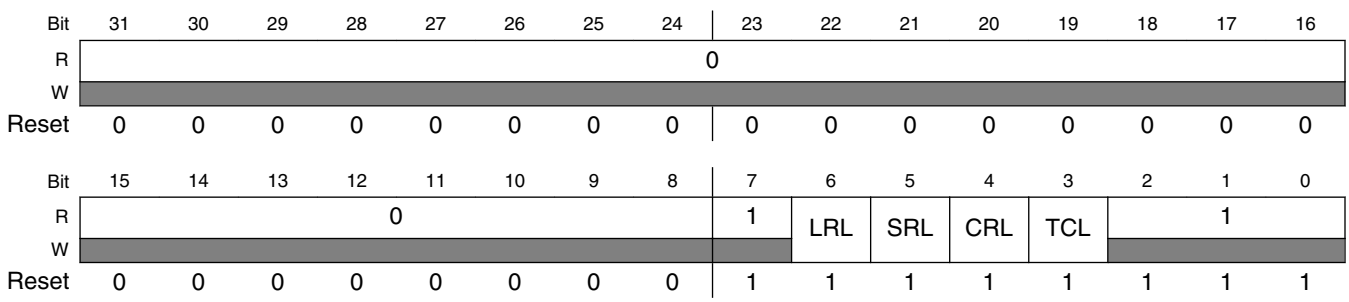
Table continues on the next page...

RTC_SR field descriptions (continued)

Field	Description
0 TIF	Time Invalid Flag The time invalid flag is set on POR or software reset. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0 Time is valid. 1 Time is invalid and time counter is read as zero.

45.3.7 RTC Lock Register (RTC_LR)

Address: 4003_D000h base + 18h offset = 4003_D018h



RTC_LR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
6 LRL	Lock Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Lock Register is locked and writes are ignored. 1 Lock Register is not locked and writes complete as normal.
5 SRL	Status Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Status Register is locked and writes are ignored. 1 Status Register is not locked and writes complete as normal.
4 CRL	Control Register Lock After being cleared, this bit can only be set by POR. 0 Control Register is locked and writes are ignored. 1 Control Register is not locked and writes complete as normal.

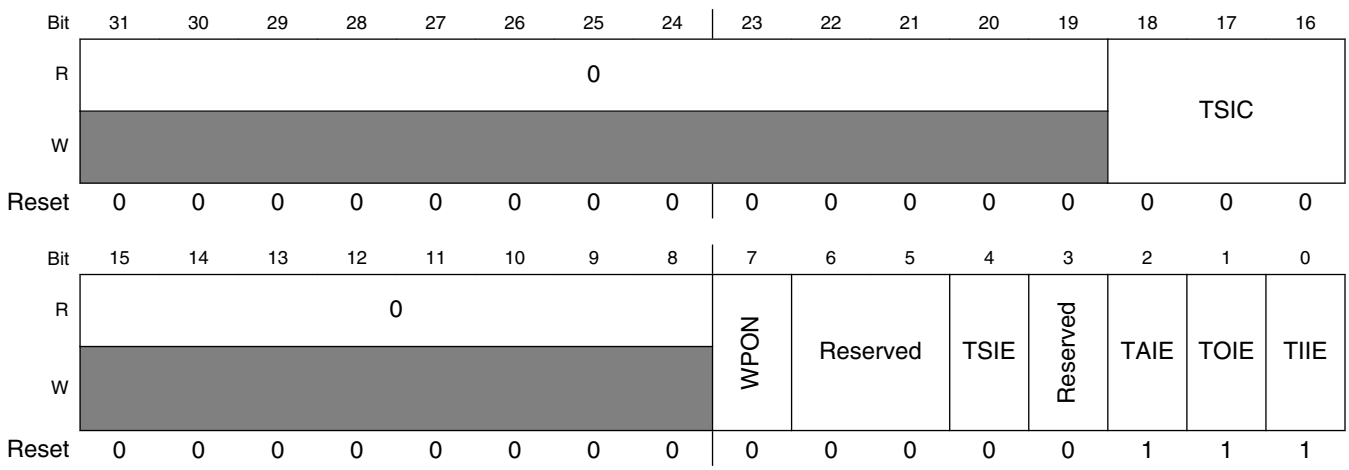
Table continues on the next page...

RTC_LR field descriptions (continued)

Field	Description
3 TCL	Time Compensation Lock After being cleared, this bit can be set only by POR or software reset. 0 Time Compensation Register is locked and writes are ignored. 1 Time Compensation Register is not locked and writes complete as normal.
Reserved	This field is reserved. This read-only field is reserved and always has the value 1.

45.3.8 RTC Interrupt Enable Register (RTC_IER)

Address: 4003_D000h base + 1Ch offset = 4003_D01Ch



RTC_IER field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 TSIC	Timer Seconds Interrupt Configuration Configures the frequency of the RTC Seconds interrupt and the RTC_CLKOUT prescaler output. This field should only be altered when TSIE is clear. 000 1 Hz. 001 2 Hz. 010 4 Hz. 011 8 Hz. 100 16 Hz. 101 32 Hz. 110 64 Hz. 111 128 Hz.

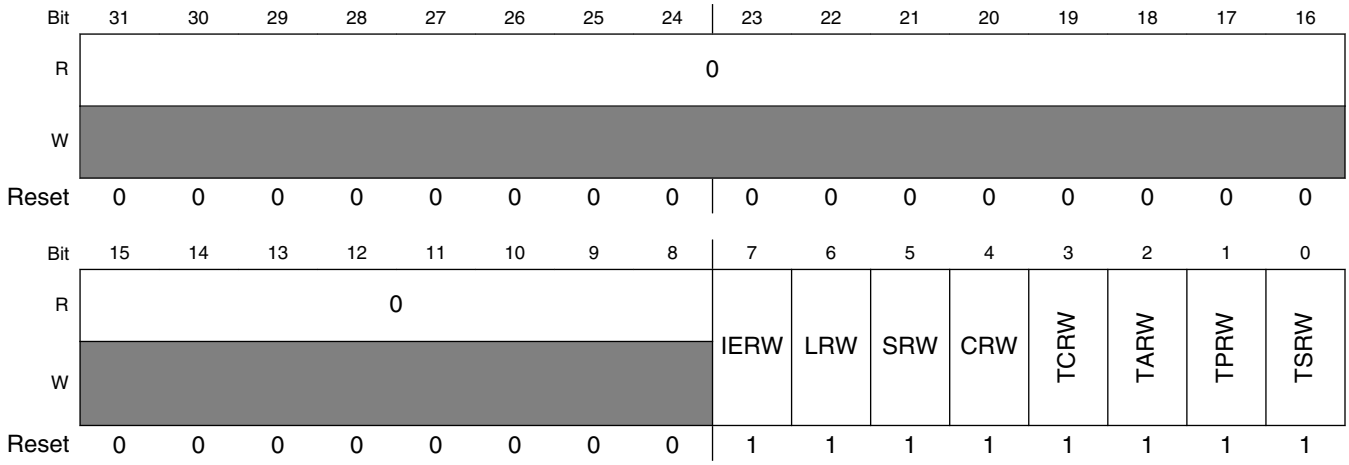
Table continues on the next page...

RTC_IER field descriptions (continued)

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 WPON	Wakeup Pin On The wakeup pin is optional and not available on all devices. Whenever the wakeup pin is enabled and this bit is set, the wakeup pin will assert. 0 No effect. 1 If the wakeup pin is enabled, then the wakeup pin will assert.
6–5 Reserved	This field is reserved.
4 TSIE	Time Seconds Interrupt Enable The seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector. It is generated once a second and requires no software overhead (there is no corresponding status flag to clear). 0 Seconds interrupt is disabled. 1 Seconds interrupt is enabled.
3 Reserved	This field is reserved.
2 TAIE	Time Alarm Interrupt Enable 0 Time alarm flag does not generate an interrupt. 1 Time alarm flag does generate an interrupt.
1 TOIE	Time Overflow Interrupt Enable 0 Time overflow flag does not generate an interrupt. 1 Time overflow flag does generate an interrupt.
0 TIIE	Time Invalid Interrupt Enable 0 Time invalid flag does not generate an interrupt. 1 Time invalid flag does generate an interrupt.

45.3.9 RTC Write Access Register (RTC_WAR)

Address: 4003_D000h base + 800h offset = 4003_D800h



RTC_WAR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 IERW	Interrupt Enable Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Interupt Enable Register are ignored. 1 Writes to the Interrupt Enable Register complete as normal.
6 LRW	Lock Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Lock Register are ignored. 1 Writes to the Lock Register complete as normal.
5 SRW	Status Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Status Register are ignored. 1 Writes to the Status Register complete as normal.
4 CRW	Control Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Control Register are ignored. 1 Writes to the Control Register complete as normal.
3 TCRW	Time Compensation Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset.

Table continues on the next page...

RTC_WAR field descriptions (continued)

Field	Description
	0 Writes to the Time Compensation Register are ignored. 1 Writes to the Time Compensation Register complete as normal.
2 TARW	Time Alarm Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Time Alarm Register are ignored. 1 Writes to the Time Alarm Register complete as normal.
1 TPRW	Time Prescaler Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Time Prescaler Register are ignored. 1 Writes to the Time Prescaler Register complete as normal.
0 TSRW	Time Seconds Register Write After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Writes to the Time Seconds Register are ignored. 1 Writes to the Time Seconds Register complete as normal.

45.3.10 RTC Read Access Register (RTC_RAR)

Address: 4003_D000h base + 804h offset = 4003_D804h

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								IERR	LRR	SRR	CRR	TCRR	TARR	TPRR	TSRR
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

RTC_RAR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 IERR	Interrupt Enable Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Interrupt Enable Register are ignored. 1 Reads to the Interrupt Enable Register complete as normal.
6 LRR	Lock Register Read

Table continues on the next page...

RTC_RAR field descriptions (continued)

Field	Description
	After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Lock Register are ignored. 1 Reads to the Lock Register complete as normal.
5 SRR	Status Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Status Register are ignored. 1 Reads to the Status Register complete as normal.
4 CRR	Control Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Control Register are ignored. 1 Reads to the Control Register complete as normal.
3 TCRR	Time Compensation Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Time Compensation Register are ignored. 1 Reads to the Time Compensation Register complete as normal.
2 TARR	Time Alarm Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Time Alarm Register are ignored. 1 Reads to the Time Alarm Register complete as normal.
1 TPRR	Time Prescaler Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Time Pprescaler Register are ignored. 1 Reads to the Time Prescaler Register complete as normal.
0 TSRR	Time Seconds Register Read After being cleared, this bit is set only by system reset. It is not affected by software reset. 0 Reads to the Time Seconds Register are ignored. 1 Reads to the Time Seconds Register complete as normal.

45.4 Functional description

45.4.1 Power, clocking, and reset

The RTC is an always powered block that remains active in all low power modes.

The time counter within the RTC is clocked by a 32.768 kHz clock sourced from an external crystal using the oscillator. Alternatively, the time counter can be clocked by the LPO and the prescaler will increment by 32 for each LPO clock.

The power-on-reset signal initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers.

45.4.1.1 Oscillator control

The 32.768 kHz crystal oscillator is disabled at POR and must be enabled by software. After enabling the cystal oscillator, wait the oscillator startup time before setting SR[TCE] or using the oscillator clock external to the RTC.

45.4.1.2 Software reset

Writing 1 to CR[SWR] forces the equivalent of a POR to the rest of the RTC module. CR[SWR] is not affected by the software reset and must be cleared by software. The access control registers are not affected by either VBAT POR or the software reset; they are reset by the chip reset.

45.4.1.3 Supervisor access

When the supervisor access control bit is clear, only supervisor mode software can write to the RTC registers, non-supervisor mode software will generate a bus error. Both supervisor and non-supervisor mode software can always read the RTC registers.

45.4.2 Time counter

The time counter consists of a 32-bit seconds counter that increments once every second and a 16-bit prescaler register that increments once every 32.768 kHz clock cycle. There is also the option to clock the prescaler using a 1 kHz LPO that increments the prescaler by 32 on every clock cycle.

Reading the time counter (either seconds or prescaler) while it is incrementing may return invalid data due to synchronization of the read data bus. If it is necessary for software to read the prescaler or seconds counter when they could be incrementing, it is recommended that two read accesses are performed and that software verifies that the same data was returned for both reads.

The time seconds register and time prescaler register can be written only when SR[TCE] is clear. Always write to the prescaler register before writing to the seconds register, because the seconds register increments on the falling edge of bit 14 of the prescaler register.

The time prescaler register increments provided SR[TCE] is set, SR[TIF] is clear, SR[TOF] is clear, and the 32.768 kHz (or 1 kHz) clock source is present. After enabling the oscillator, wait the oscillator startup time before setting SR[TCE] to allow time for the oscillator clock output to stabilize.

If the time seconds register overflows then the SR[TOF] will set and the time prescaler register will stop incrementing. Clear SR[TOF] by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TOF] is set.

SR[TIF] is set on POR and software reset and is cleared by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TIF] is set.

45.4.3 Compensation

The compensation logic provides an accurate and wide compensation range and can correct errors as high as 3906 ppm and as low as 0.12 ppm. The compensation factor must be calculated externally to the RTC and supplied by software to the compensation register. The RTC itself does not calculate the amount of compensation that is required, although the 1 Hz clock is output to an external pin in support of external calibration logic.

Crystal compensation can be supported by using firmware and crystal characteristics to determine the compensation amount. Temperature compensation can be supported by firmware that periodically measures the external temperature via ADC and updates the compensation register based on a look-up table that specifies the change in crystal frequency over temperature.

The compensation logic alters the number of 32.768 kHz clock cycles it takes for the prescaler register to overflow and increment the time seconds counter. The time compensation value is used to adjust the number of clock cycles between -127 and +128. Cycles are added or subtracted from the prescaler register when the prescaler register equals 0x3FFF and then increments. The compensation interval is used to adjust the frequency at which the time compensation value is used, that is, from once a second to once every 256 seconds.

Updates to the time compensation register will not take effect until the next time the time seconds register increments and provided the previous compensation interval has expired. When the compensation interval is set to other than once a second then the compensation is applied in the first second interval and the remaining second intervals receive no compensation.

Compensation is disabled by configuring the time compensation register to zero.

When the prescaler is configured to increment using the 1 kHz LPO, the effective compensation value is divided by 32 and can only adjust the number of clock cycles between -4 and +3.

45.4.4 Time alarm

The Time Alarm register (TAR), SR[TAF], and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit TAR is compared with the 32-bit Time Seconds register (TSR) each time it increments. SR[TAF] will set when TAR equals TSR and TSR increments.

SR[TAF] is cleared by writing TAR. This will usually be the next alarm value, although writing a value that is less than TSR, such as 0, will prevent SR[TAF] from setting again. SR[TAF] cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

45.4.5 Update mode

The Update Mode field in the Control register (CR[UM]) configures software write access to the Time Counter Enable (SR[TCE]) field. When CR[UM] is clear, SR[TCE] can be written only when LR[SRL] is set. When CR[UM] is set, SR[TCE] can also be written when SR[TCE] is clear or when SR[TIF] or SR[TOF] are set. This allows the time seconds and prescaler registers to be initialized whenever time is invalidated, while preventing the time seconds and prescaler registers from being changed on the fly. When LR[SRL] is set, CR[UM] has no effect on SR[TCE].

45.4.6 Register lock

The Lock register (LR) can be used to block write accesses to certain registers until the next POR or software reset. Locking the Control register (CR) will disable the software reset. Locking LR will block future updates to LR.

Write accesses to a locked register are ignored and do not generate a bus error.

45.4.7 Access control

The read access and write access registers are implemented in the chip power domain and reset on the chip reset. They are not affected by the POR or the software reset. They are used to block read or write accesses to each register until the next chip system reset.

45.4.8 Interrupt

The RTC interrupt is asserted whenever a status flag and the corresponding interrupt enable bit are both set. It is always asserted on POR, and software reset. The RTC interrupt is enabled at the chip level by enabling the chip-specific RTC clock gate control bit. The RTC interrupt can be used to wakeup the chip from any low-power mode.

The optional RTC seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector that is generated once a second and requires no software overhead (there is no corresponding status flag to clear). It is enabled in the RTC by the time seconds interrupt enable bit and enabled at the chip level by setting the chip-specific RTC clock gate control bit. The frequency of the seconds interrupt defaults to 1 Hz, but can instead be configured to trigger every 2, 4, 8, 16, 32, 64 or 128 Hz. This interrupt is optional and may not be implemented on all devices.

45.5 Usage Guide

45.5.1 Clock source information

To get an accuracy clock for RTC, an external 32.768 kHz crystal should be connected to EXTAL32/XTAL32 pin, or a 32.768 kHz clock signal to RTC_CLKIN pin. Alternatively, the time counter can be clocked by the LPO 1 kHz and the prescaler will increment by 32 for each LPO clock, which is not that precisely.

45.5.2 Usage examples

This section shows the application examples of initializing the RTC module, setting the data time and alarm.

RTC Module Initialization

The RTC module is reset by a POR or a software reset (The access control registers are not affected by either VBAT POR or the software reset).

Before using the RTC module, a software reset is recommend by setting the RTC_CR[SWR] bit. And the 32.768 kHz external crystal should be enabled to provide clock to RTC.

```
// Reset the RTC by set RTC_CR[SWR] bit, and wait
// for the TIF flag cleared by writing the TSR
while (RTC_SR & RTC_SR_TIF_MASK)
{
    RTC_CR |= RTC_CR_SWR_MASK;
    RTC_CR &= ~RTC_CR_SWR_MASK;
    RTC_TSR = 1;
}

// Setup the update mode and supervisor access mode
// enable 32.768 kHz oscillator timer
RTC_CR = RTC_CR_CPE(0) | RTC_CR_LPOS(0) | RTC_CR_CPS(1) |
         RTC_CR_UM(0) | RTC_CR_SUP(0)| RTC_CR_OSCE(1);
// disable all the interrupts first
RTC_IER = 0;
// stop timer first
RTC_SR &= ~RTC_SR_TCE_MASK;
```

Set Date Time

After RTC initialized, user can set the date time before starting the timer. Please make sure the timer is stopped when setting the date time by RTC_TSR register.

```
// stop timer first
RTC_SR &= ~RTC_SR_TCE_MASK;
// convert the date time to secs first, then write to RTC_TSR register
RTC_TSR = datetime_in_secs;
// start the timer
RTC_SR |= RTC_SR_TCE_MASK;
```

Set Alarm

To set an alarm and trigger alarm interrupt, user should enable the alarm interrupt, write the alarm seconds into RTC_TAR.

```
uint32_t datetime_in_secs;

// assume the timer is running
// enable the interrupt
RTC_IER |= RTC_IER_TAIE_MASK;
// enable the RTC IRQ in NVIC
NVIC_EnableIRQ(RTC_IRQn);

// get the current date time in secs
datetime_in_secs = RTC_TSR;
datatime_in_secs += 10;
// set alarm 10s later
RTC_TAR = datetime_in_secs;
```

After 10 seconds, the RTC Alarm IRQ would be triggered and IRQ Handler called. In the IRQ Handler, user should first clear the interrupt status:

```
if (RTC_SR & RTC_SR_TAF_MASK)
{
    // clear the TAF flag by writing the RTC_TAR register
    RTC_TAR = 0;
}
// Then doing the alarm task in this IRQ Handler
……
```

45.5.3 RTC_CLKOUT signal

When the RTC is enabled and the port control module selects the RTC_CLKOUT function, the RTC_CLKOUT signal output either a square wave prescaler output (configurable to 1, 2, 4, 8, 16, 32, 64 or 128 Hz) or 32 kHz output derived from RTC oscillator as shown below.

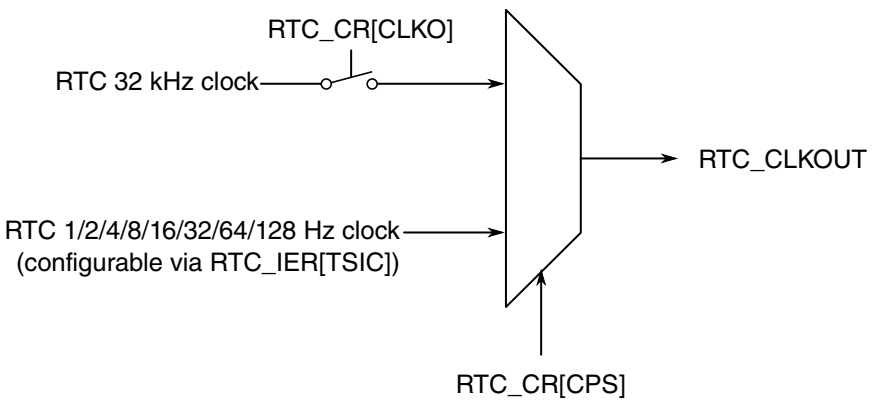


Figure 45-1. RTC_CLKOUT generation

NOTE

When using LPO 1kHz as RTC clock source, it cannot directly output to pad. But RTC can normally output 1/2/4/8/…/64/128 Hz clock using prescaler.

Chapter 46
Low Power Serial Peripheral Interface (LPSPI)

46.1 Chip-specific information for this module

46.1.1 Instantiation Information

This device contains two LPSPI modules. The LPSPI can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

Table 46-1. LPSPI Configuration

	TX FIFO (word/32bit)	RX FIFO (word/32bit)	Chip Selects
LPSPI0	4	4	4
LPSPI1	4	4	4

NOTE

The TX/RX FIFO "word" does not refer to system bus width 32-bit, and it varies for different communication module. For example:

- LPSPI: 32-bit
- LPI2C: 8-bit (except CMD)
- LPUART: 10-bit

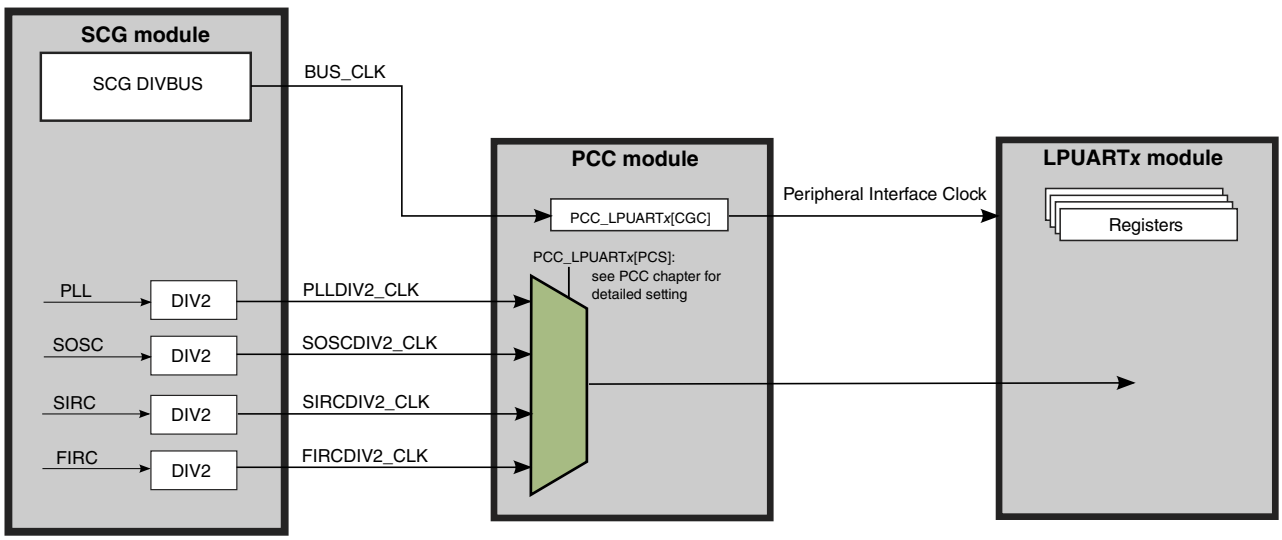
NOTE

The exact number of chip select for each module is depending on the package, not all of the chip selects are available on different packages.

46.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

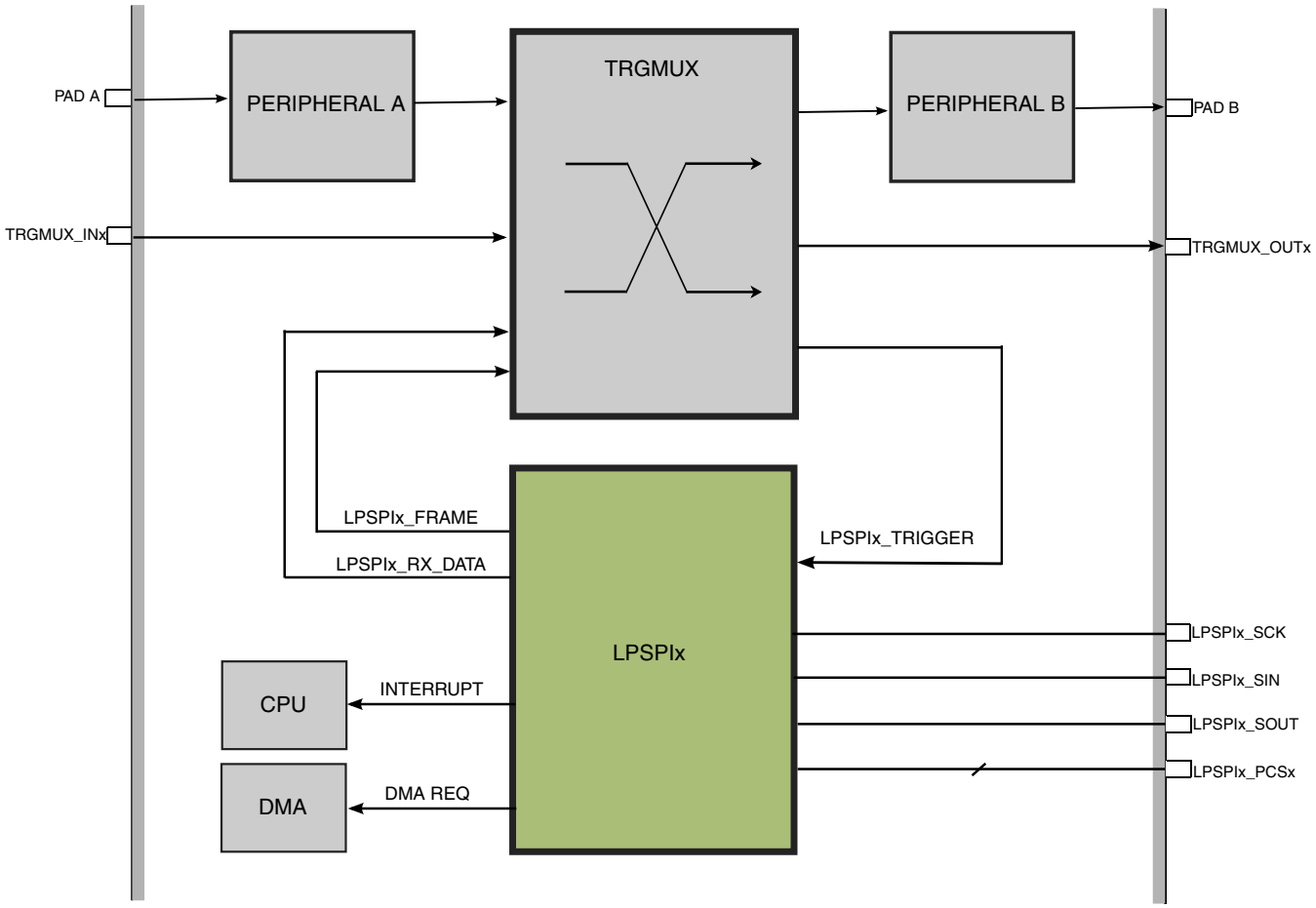
The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPUART Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, FlexIO and LPIT.



46.1.3 Inter-connectivity Information

The LPSPI inter-connectivity is shown in following diagram.



46.2 Introduction

46.2.1 Overview

The LPSPI is a low power Serial Peripheral Interface (SPI) module that supports an efficient interface to an SPI bus as a master and/or a slave. The LPSPI can continue operating in stop modes provided an appropriate clock is available and is designed for low CPU overhead with DMA offloading of FIFO register accesses.

46.2.2 Features

The LPSPI supports the following features:

- Word size = 32 bits
- Command/transmit FIFO of 4 words.
- Receive FIFO of 4 words.
- Host request input can be used to control the start time of an SPI bus transfer.

46.2.3 Block Diagram

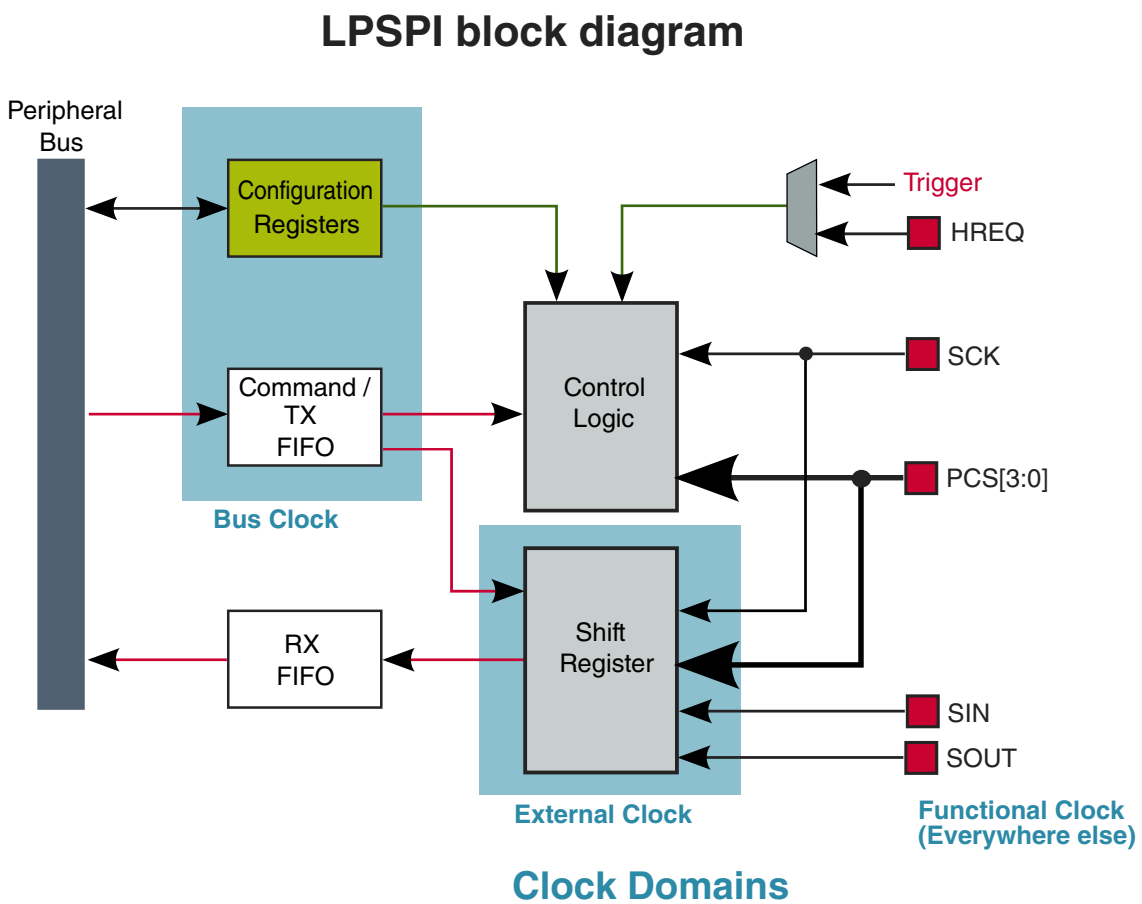


Figure 46-1. Block Diagram

46.2.4 Modes of operation

The LPSPI module supports the chip modes described in the following table.

Table 46-2. Chip modes supported by the LPSPI module

Chip mode	LPSPI Operation
Run	Normal operation

Table continues on the next page...

Table 46-2. Chip modes supported by the LPSPI module (continued)

Chip mode	LPSPI Operation
Stop/Wait	Can continue operating if the Doze Enable bit (CR[DOZEN]) is set and the LPSPI is using an external or internal clock source, which remains operating during stop/ wait modes.
Debug	Can continue operating if the Debug Enable bit (CR[DBGEN]) is set.

46.2.5 Signal Descriptions

Signal	Description	I/O
SCK	Serial clock. Input in slave mode, output in master mode.	I/O
PCS[0]	Peripheral Chip Select. Input in slave mode, output in master mode.	I/O
PCS[1] / HREQ	Peripheral Chip Select or Host Request. Host Request pin is selected when HREN=1 and HRSEL=0. Input in either slave mode or when used as Host Request, output in master mode.	I/O
PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
SOUT / DATA[0]	Serial Data Output. Can be configured as serial data input signal. Used as data pin 0 in quad-data and dual-data transfers.	I/O
SIN / DATA[1]	Serial Data Input. Can be configured as serial data output signal. Used as data pin 1 in quad-data and dual-data transfers.	I/O

46.3 Memory Map and Registers

LPSPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_C000	Version ID Register (LPSPI0_VERID)	32	R	0100_0004h	46.3.1/1207
4002_C004	Parameter Register (LPSPI0_PARAM)	32	R	See section	46.3.2/1208
4002_C010	Control Register (LPSPI0_CR)	32	R/W	0000_0000h	46.3.3/1209
4002_C014	Status Register (LPSPI0_SR)	32	R/W	0000_0001h	46.3.4/1210
4002_C018	Interrupt Enable Register (LPSPI0_IER)	32	R/W	0000_0000h	46.3.5/1212
4002_C01C	DMA Enable Register (LPSPI0_DER)	32	R/W	0000_0000h	46.3.6/1213
4002_C020	Configuration Register 0 (LPSPI0_CFGR0)	32	R/W	0000_0000h	46.3.7/1214
4002_C024	Configuration Register 1 (LPSPI0_CFGR1)	32	R/W	0000_0000h	46.3.8/1215
4002_C030	Data Match Register 0 (LPSPI0_DMR0)	32	R/W	0000_0000h	46.3.9/1217
4002_C034	Data Match Register 1 (LPSPI0_DMR1)	32	R/W	0000_0000h	46.3.10/ 1217
4002_C040	Clock Configuration Register (LPSPI0_CCR)	32	R/W	0000_0000h	46.3.11/ 1218
4002_C058	FIFO Control Register (LPSPI0_FCR)	32	R/W	0000_0000h	46.3.12/ 1219
4002_C05C	FIFO Status Register (LPSPI0_FSR)	32	R	0000_0000h	46.3.13/ 1219
4002_C060	Transmit Command Register (LPSPI0_TCR)	32	R/W	0000_001Fh	46.3.14/ 1220
4002_C064	Transmit Data Register (LPSPI0_TDR)	32	W	0000_0000h	46.3.15/ 1223
4002_C070	Receive Status Register (LPSPI0_RSR)	32	R	0000_0002h	46.3.16/ 1224
4002_C074	Receive Data Register (LPSPI0_RDR)	32	R	0000_0000h	46.3.17/ 1225
4002_D000	Version ID Register (LPSPI1_VERID)	32	R	0100_0004h	46.3.1/1207
4002_D004	Parameter Register (LPSPI1_PARAM)	32	R	See section	46.3.2/1208
4002_D010	Control Register (LPSPI1_CR)	32	R/W	0000_0000h	46.3.3/1209
4002_D014	Status Register (LPSPI1_SR)	32	R/W	0000_0001h	46.3.4/1210
4002_D018	Interrupt Enable Register (LPSPI1_IER)	32	R/W	0000_0000h	46.3.5/1212
4002_D01C	DMA Enable Register (LPSPI1_DER)	32	R/W	0000_0000h	46.3.6/1213
4002_D020	Configuration Register 0 (LPSPI1_CFGR0)	32	R/W	0000_0000h	46.3.7/1214
4002_D024	Configuration Register 1 (LPSPI1_CFGR1)	32	R/W	0000_0000h	46.3.8/1215
4002_D030	Data Match Register 0 (LPSPI1_DMR0)	32	R/W	0000_0000h	46.3.9/1217
4002_D034	Data Match Register 1 (LPSPI1_DMR1)	32	R/W	0000_0000h	46.3.10/ 1217

Table continues on the next page...

LPSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_D040	Clock Configuration Register (LPSPI1_CCR)	32	R/W	0000_0000h	46.3.11/ 1218
4002_D058	FIFO Control Register (LPSPI1_FCR)	32	R/W	0000_0000h	46.3.12/ 1219
4002_D05C	FIFO Status Register (LPSPI1_FSR)	32	R	0000_0000h	46.3.13/ 1219
4002_D060	Transmit Command Register (LPSPI1_TCR)	32	R/W	0000_001Fh	46.3.14/ 1220
4002_D064	Transmit Data Register (LPSPI1_TDR)	32	W	0000_0000h	46.3.15/ 1223
4002_D070	Receive Status Register (LPSPI1_RSR)	32	R	0000_0002h	46.3.16/ 1224
4002_D074	Receive Data Register (LPSPI1_RDR)	32	R	0000_0000h	46.3.17/ 1225

46.3.1 Version ID Register (LPSPIx_VERID)

.

Address: Base address + 0h offset

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

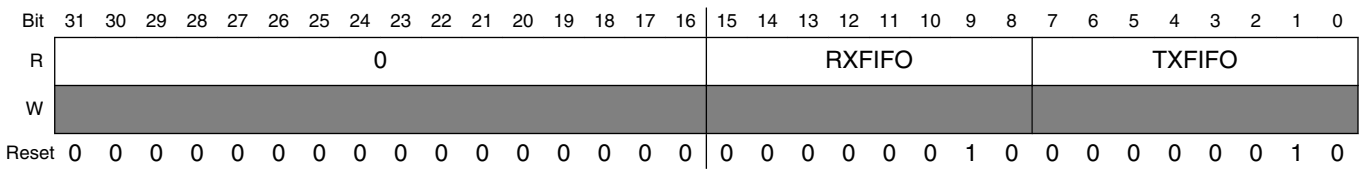
LPSPIx_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Module Identification Number This read only field returns the feature set number. 0x0004 Standard feature set supporting 32-bit shift register.

46.3.2 Parameter Register (LPSPIx_PARAM)

.

Address: Base address + 4h offset

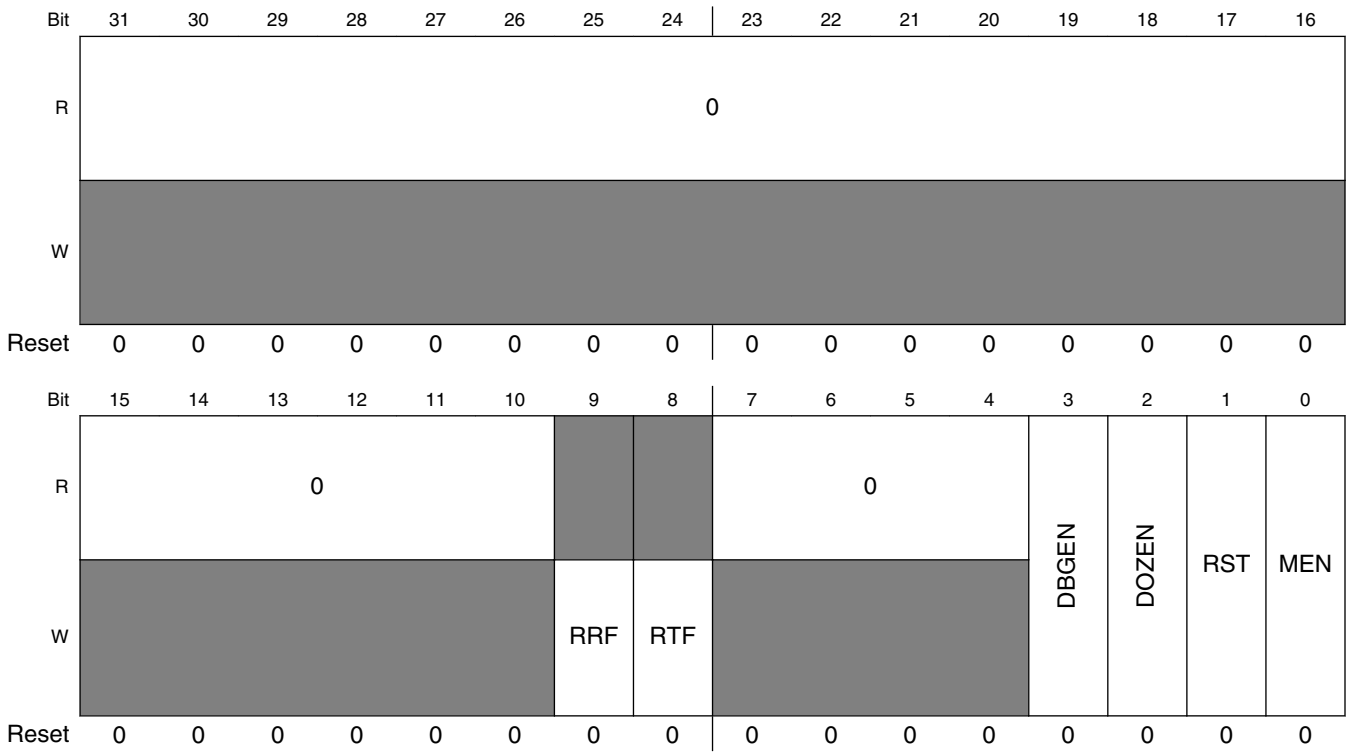


LPSPIx_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 RXFIFO	Receive FIFO Size The number of words in the receive FIFO is 2^RXFIFO.
TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

46.3.3 Control Register (LPSPIx_CR)

Address: Base address + 10h offset



LPSPIx_CR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive FIFO is reset.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit FIFO is reset.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBGEN	Debug Enable 0 Module is disabled in debug mode. 1 Module is enabled in debug mode.
2 DOZEN	Doze mode enable Enables or disables Doze mode

Table continues on the next page...

LPSPIx_CR field descriptions (continued)

Field	Description
	0 Module is enabled in Doze mode. 1 Module is disabled in Doze mode.
1 RST	Software Reset Reset all internal logic and registers, except the Control Register. Remains set until cleared by software. 0 Master logic is not reset. 1 Master logic is reset.
0 MEN	Module Enable 0 Module is disabled. 1 Module is enabled.

46.3.4 Status Register (LPSPIx_SR)

Address: Base address + 14h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			w1c	w1c	w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

LPSPIx_SR field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 MBF	Module Busy Flag 0 LPSPI is idle. 1 LPSPI is busy.
23–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields as configured by MATCFG. 0 Have not received matching data. 1 Have received matching data.

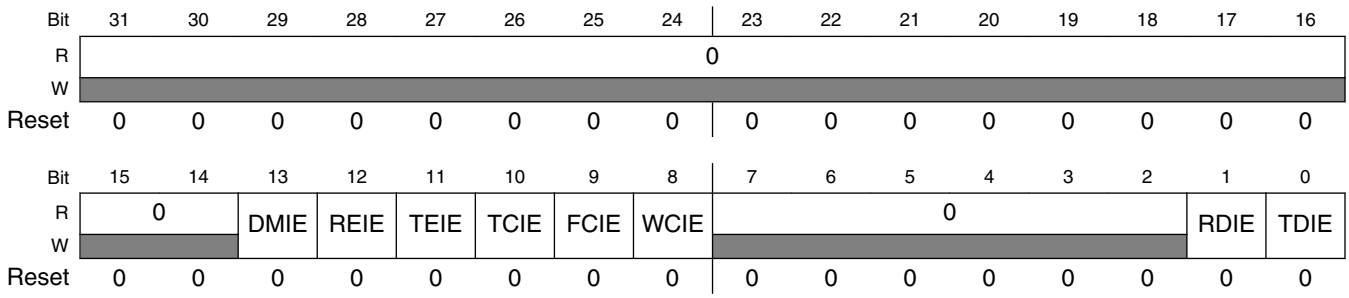
Table continues on the next page...

LPSPIx_SR field descriptions (continued)

Field	Description
12 REF	Receive Error Flag This flag will set when the Receiver FIFO overflows. 0 Receive FIFO has not overflowed. 1 Receive FIFO has overflowed.
11 TEF	Transmit Error Flag This flag will set when the Transmit FIFO underruns. 0 Transmit FIFO underrun has not occurred. 1 Transmit FIFO underrun has occurred
10 TCF	Transfer Complete Flag This flag will set in master mode when the LPSPI returns to idle state with the transmit FIFO empty. 0 All transfers have not completed. 1 All transfers have completed.
9 FCF	Frame Complete Flag This flag will set at the end of each frame transfer, when the PCS negates. 0 Frame transfer has not completed. 1 Frame transfer has completed.
8 WCF	Word Complete Flag This flag will set when the last bit of a received word is sampled. 0 Transfer word not completed. 1 Transfer word completed.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. 0 Transmit data not requested. 1 Transmit data is requested.

46.3.5 Interrupt Enable Register (LPSPIx_IER)

Address: Base address + 18h offset



LPSPIx_IER field descriptions

Field	Description
31–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 DMIE	Data Match Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 REIE	Receive Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
11 TEIE	Transmit Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 TCIE	Transfer Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 FCIE	Frame Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
8 WCIE	Word Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
0 TDIE	Transmit Data Interrupt Enable

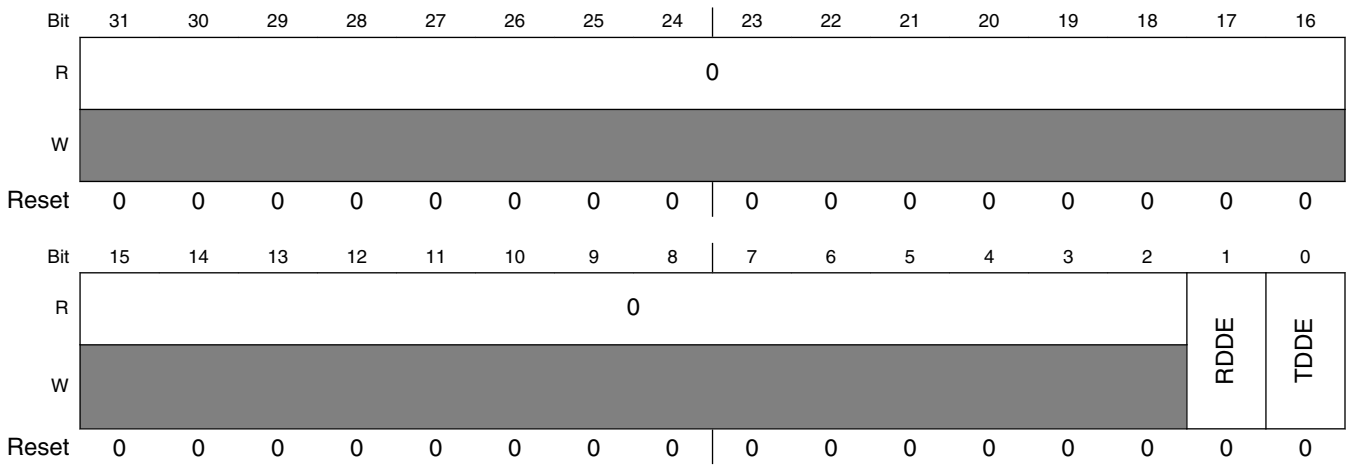
Table continues on the next page...

LPSPIx_IER field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled

46.3.6 DMA Enable Register (LPSPIx_DER)

Address: Base address + 1Ch offset

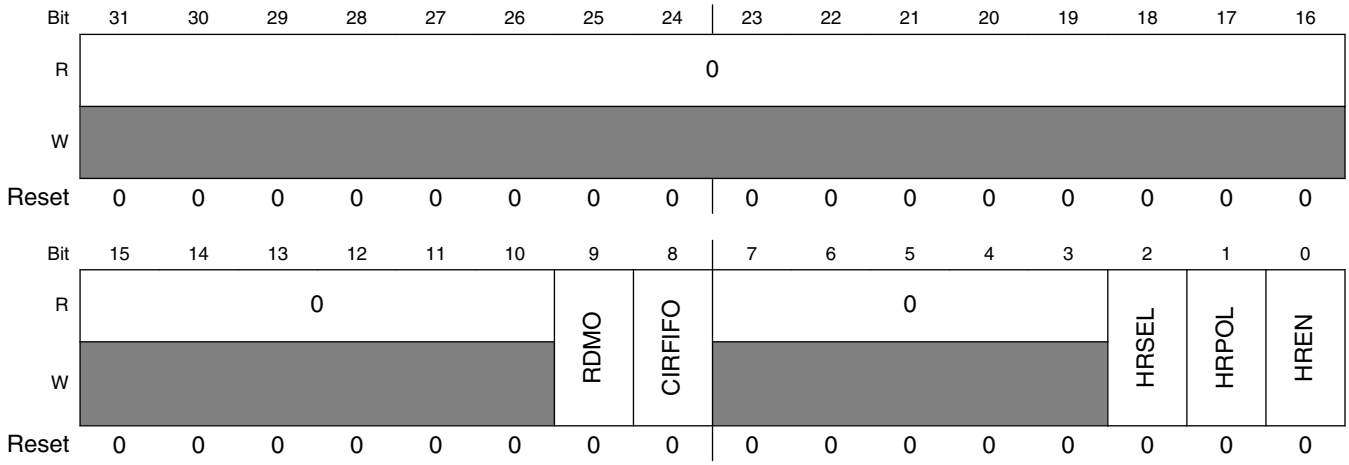


LPSPIx_DER field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

46.3.7 Configuration Register 0 (LPSPIx_CFGR0)

Address: Base address + 20h offset



LPSPIx_CFGR0 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause DMF to set is discarded. Once DMF is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing DMF to ensure no receive data is lost. 0 Received data is stored in the receive FIFO as normal. 1 Received data is discarded unless the DMF is set.
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as normal, but once the LPSPI is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This will cause the contents of the transmit FIFO to be cycled through repeatedly. 0 Circular FIFO is disabled. 1 Circular FIFO is enabled.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HRSEL	Host Request Select Selects the source of the host request input. When the host request function is enabled with the LPSPI_HREQ pin, the LPSPI_PCS[1] function is disabled. 0 Host request input is pin LPSPI_HREQ. 1 Host request input is input trigger.
1 HRPOL	Host Request Polarity

Table continues on the next page...

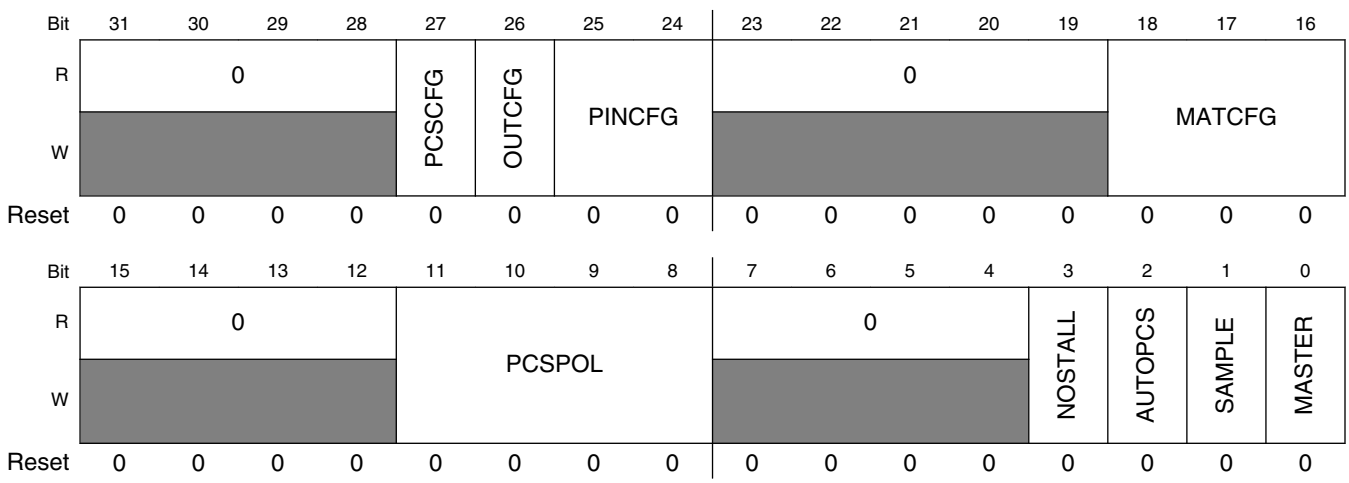
LPSPIx_CFGR0 field descriptions (continued)

Field	Description
	Configures the polarity of the host request pin. 0 Active low. 1 Active high.
0 HREN	Host Request Enable When enabled in master mode, the LPSPI will only initiate a SPI bus transfer if the host request input is asserted. 0 Host request is disabled. 1 Host request is enabled.

46.3.8 Configuration Register 1 (LPSPIx_CFGR1)

The CFGR1 should only be written when the LPSPI is disabled.

Address: Base address + 24h offset



LPSPIx_CFGR1 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 PCSCFG	Peripheral Chip Select Configuration PCSCFG must be set if performing 4-bit transfers. 0 PCS[3:2] are enabled. 1 PCS[3:2] are disabled.
26 OUTCFG	Output Config Configures if the output data is tristated between accesses (LPSPI_PCS is negated).

Table continues on the next page...

LPSPIx_CFGR1 field descriptions (continued)

Field	Description
	0 Output data retains last value when chip select is negated. 1 Output data is tristated when chip select is negated.
25–24 PINCFG	Pin Configuration Configures which pins are used for input and output data during single bit transfers. 00 SIN is used for input data and SOUT for output data. 01 SIN is used for both input and output data. 10 SOUT is used for both input and output data. 11 SOUT is used for input data and SIN for output data.
23–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. 000 Match disabled. 001 Reserved 010 Match enabled (1st data word equals MATCH0 OR MATCH1). 011 Match enabled (any data word equals MATCH0 OR MATCH1). 100 Match enabled (1st data word equals MATCH0 AND 2nd data word equals MATCH1). 101 Match enabled (any data word equals MATCH0 AND next data word equals MATCH1) 110 Match enabled (1st data word AND MATCH1 equals MATCH0 AND MATCH1) 111 Match enabled (any data word AND MATCH1 equals MATCH0 AND MATCH1).
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 PCSPOL	Peripheral Chip Select Polarity Configures the polarity of each Peripheral Chip Select pin. 0 The PCSx is active low. 1 The PCSx is active high.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 NOSTALL	No Stall In master mode, the LPSPI will stall transfers when the transmit FIFO is empty or receive FIFO is full ensuring that no transmit FIFO underrun or receive FIFO overrun can occur. Setting this bit will disable this functionality. 0 Transfers will stall when transmit FIFO is empty or receive FIFO is full. 1 Transfers will not stall, allowing transmit FIFO underrun or receive FIFO overrun to occur.
2 AUTOPCS	Automatic PCS The LPSPI slave normally requires the PCS to negate between frames for correct operation. Setting this bit will cause the LPSPI to generate an internal PCS signal at the end of each transfer word when CPHA=1. When this bit is set, the SCK must remain idle for at least 4 LPSPI functional clock cycles (divided by PRESCALE configuration) between each word to ensure correct operation. This bit is ignored in master mode.

Table continues on the next page...

LPSPIx_CFGR1 field descriptions (continued)

Field	Description
	0 Automatic PCS generation disabled. 1 Automatic PCS generation enabled.
1 SAMPLE	Sample Point When set, the LPSPI master will sample the input data on a delayed LPSPI_SCK edge. This improves the setup time when sampling data. The input data setup time in master mode with delayed LPSPI_SCK edge is equal to the input data setup time in slave mode. This bit is ignored in slave mode. 0 Input data sampled on SCK edge. 1 Input data sampled on delayed SCK edge.
0 MASTER	Master Mode Configures the LPSPI in master or slave mode. This bit directly controls the direction of the LPSPI_SCK and LPCPI_PCS pins. 0 Slave mode. 1 Master mode.

46.3.9 Data Match Register 0 (LPSPIx_DMR0)

Address: Base address + 30h offset

Bit	31	30	29	28	27	26	25	24	23	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	MATCH0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPIx_DMR0 field descriptions

Field	Description
MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

46.3.10 Data Match Register 1 (LPSPIx_DMR1)

Address: Base address + 34h offset

Bit	31	30	29	28	27	26	25	24	23	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	MATCH1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPIx_DMR1 field descriptions

Field	Description
MATCH1	Match 1 Value

LPSPIx_DMR1 field descriptions (continued)

Field	Description
	Compared against the received data when receive data match is enabled.

46.3.11 Clock Configuration Register (LPSPIx_CCR)

The CCR is only used in master mode and cannot be changed when the LPSPI is enabled.

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	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	SCKPCS								PCSSCK								DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPIx_CCR field descriptions

Field	Description
31–24 SCKPCS	SCK to PCS Delay Configures the delay in master mode from the last SCK edge to the PCS negation. The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
23–16 PCSSCK	PCS to SCK Delay Configures the delay in master mode from the PCS assertion to the first SCK edge. The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
15–8 DBT	Delay Between Transfers Configures the delay in master mode from the PCS negation to the next PCS assertion. The delay is equal to (DBT + 2) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 2 cycles. Note that half the delay occurs before PCS assertion and the other half of the delay occurs after PCS negation; the full command word can only update in the middle. Also configures the delay in master mode from the last SCK edge of a transfer word and the first SCK edge of the next transfer word in a continuous transfer. The delay is equal to (DBT + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
SCKDIV	SCK Divider Configures the divide ratio of the SCK pin in master mode. The SCK period is equal to (SCKDIV+2) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum period is 2 cycles. If the period is an odd number of cycles, then the first half of the period will be one cycle longer than the second half of the period.

46.3.12 FIFO Control Register (LPSPIx_FCR)

Address: Base address + 58h offset

Bit	31	30	29	28	27	26	25	24	23	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								RXWATER								0								TXWATER							
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

LPSPIx_FCR 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 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

46.3.13 FIFO Status Register (LPSPIx_FSR)

Address: Base address + 5Ch offset

Bit	31	30	29	28	27	26	25	24	23	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								RXCOUNT								0								TXCOUNT							
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

LPSPIx_FSR 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 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

46.3.14 Transmit Command Register (LPSPIx_TCR)

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO in the order they are written. Command Register writes will be tagged and cause the command register to update once that entry reaches the top of the FIFO. This allows changes to the command word and the transmit data itself to be interleaved. Changing the command word will cause all subsequent SPI bus transfer to be performed using the new command word.

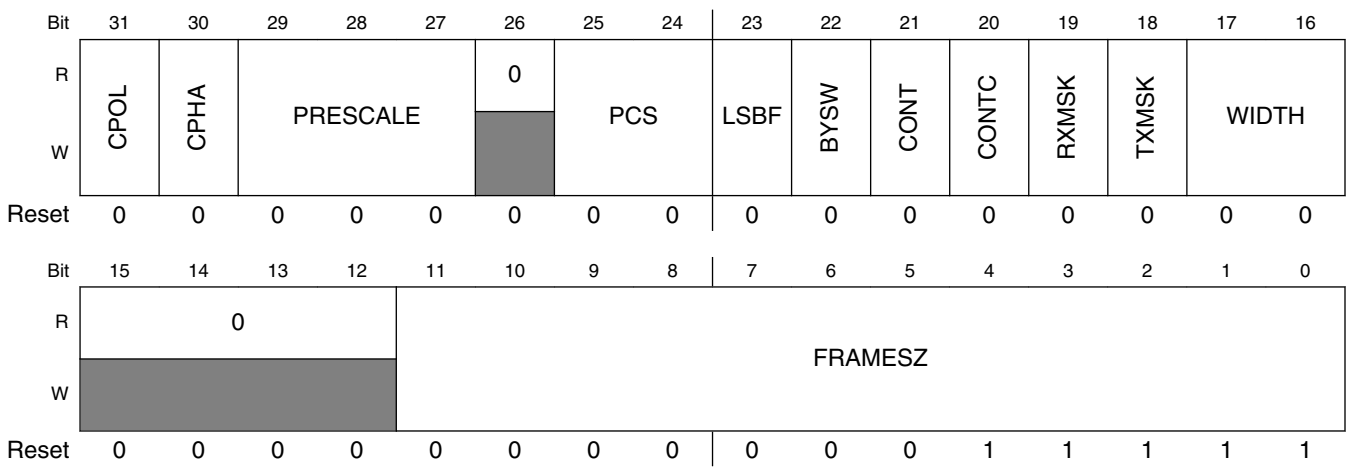
In master mode, writing a new command word does not initiate a new transfer, unless TXMSK is set. Transfers are initiated by transmit data in the transmit FIFO, or a new command word with TXMSK set. Hardware will clear TXMSK when the LPSPI_PCS negates.

In master mode if the command word is changed before an existing frame has completed, then the existing frame will terminate and the command word will then update. The command word can be changed during a continuous transfer, provided CONTC of the new command word is set and the command word is written on a frame size boundary.

In slave mode, the command word should be changed only when the LPSPI is idle and there is no SPI bus transfer.

Reading the Transmit Command Register will return the current state of the command register.

Address: Base address + 60h offset



LPSPIx_TCR field descriptions

Field	Description
31 CPOL	Clock Polarity This field is only updated between frames.

Table continues on the next page...

LPSPIx_TCR field descriptions (continued)

Field	Description
	0 The inactive state value of SCK is low. 1 The inactive state value of SCK is high.
30 CPHA	Clock Phase This field is only updated between frames. 0 Data is captured on the leading edge of SCK and changed on the following edge. 1 Data is changed on the leading edge of SCK and captured on the following edge.
29–27 PRESCALE	Prescaler Value Prescaler applied to the clock configuration register for all SPI bus transfers. This field is only updated between frames. 000 Divide by 1. 001 Divide by 2. 010 Divide by 4. 011 Divide by 8. 100 Divide by 16. 101 Divide by 32. 110 Divide by 64. 111 Divide by 128.
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. This field is only updated between frames. 00 Transfer using LPSPI_PCS[0] 01 Transfer using LPSPI_PCS[1] 10 Transfer using LPSPI_PCS[2] 11 Transfer using LPSPI_PCS[3]
23 LSBF	LSB First 0 Data is transferred MSB first. 1 Data is transferred LSB first.
22 BYSW	Byte Swap Byte swap will swap the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and each received data word stored to the FIFO (or compared with match registers). 0 Byte swap disabled. 1 Byte swap enabled.
21 CONT	Continuous Transfer In master mode, continuous transfer will keep the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. In slave mode, when continuous transfer is enabled the LPSPI will only transmit the first FRAMESZ bits, after which it will transmit received data assuming a 32-bit shift register.

Table continues on the next page...

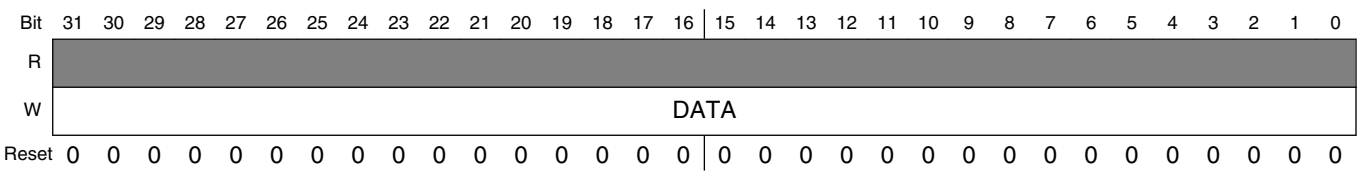
LPSPIx_TCR field descriptions (continued)

Field	Description
	0 Continuous transfer disabled. 1 Continuous transfer enabled.
20 CONTC	Continuing Command In master mode, this bit allows the command word to be changed within a continuous transfer. The initial command word must enable continuous transfer (CONT=1), the continuing command must set this bit (CONTC=1) and the continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. 0 Command word for start of new transfer. 1 Command word for continuing transfer.
19 RXMSK	Receive Data Mask When set, receive data is masked (receive data is not stored in receive FIFO). 0 Normal transfer. 1 Receive data is masked.
18 TXMSK	Transmit Data Mask When set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In master mode, this bit will initiate a new transfer which cannot be aborted by another command word and the bit will be cleared by hardware at the end of the transfer. 00 Normal transfer. 01 Mask transmit data.
17–16 WIDTH	Transfer Width Either RXMSK or TXMSK must be set for 2-bit or 4-bit transfers. 00 Single bit transfer. 01 Two bit transfer. 10 Four bit transfer. 11 Reserved.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). The minimum frame size is 8 bits. If the frame size is larger than 32 bits, data will be loaded from the transmit FIFO and stored to the receive FIFO every 32 bits. If the size of the transfer word is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits (e.g.: a 72-bit transfer will load/store 32-bits from the FIFO and then another 32-bits from the FIFO and then the final 8-bits from the FIFO).

46.3.15 Transmit Data Register (LPSPIx_TDR)

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO in the order it was written.

Address: Base address + 64h offset

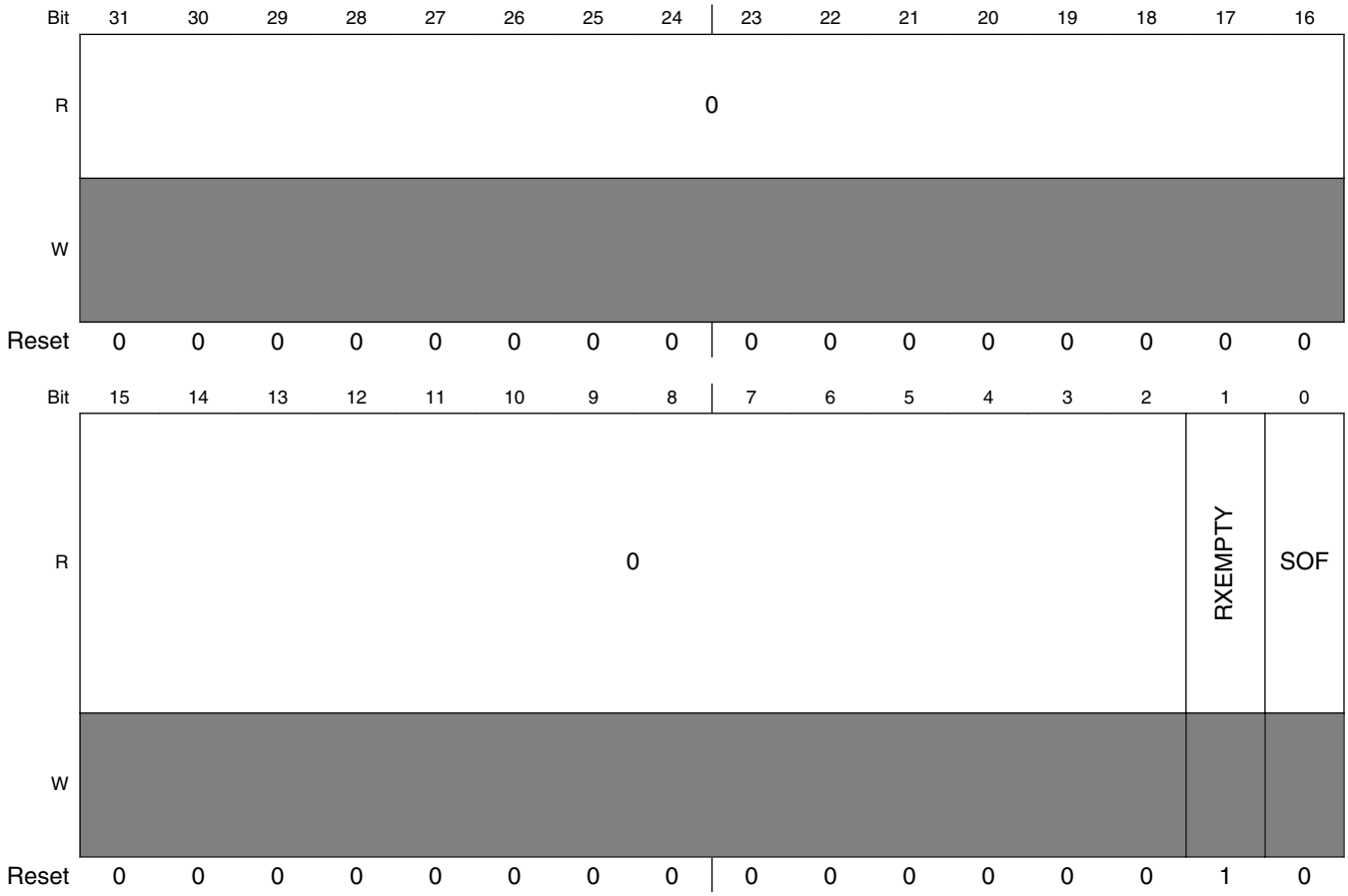


LPSPIx_TDR field descriptions

Field	Description
DATA	Transmit Data Both 8-bit and 16-bit writes of transmit data will zero extend the data written and push the data into the transmit FIFO.

46.3.16 Receive Status Register (LPSPIx_RSR)

Address: Base address + 70h offset



LPSPIx_RSR field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RXEMPTY	RX FIFO Empty 0 RX FIFO is not empty. 1 RX FIFO is empty.
0 SOF	Start Of Frame Indicates that this is the first data word received after LPSPI_PCS assertion. 0 Subsequent data word received after LPSPI_PCS assertion. 1 First data word received after LPSPI_PCS assertion.

46.3.17 Receive Data Register (LPSPIx_RDR)

Address: Base address + 74h offset

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

LPSPIx_RDR field descriptions

Field	Description
DATA	Receive Data

46.4 Functional description

46.4.1 Clocking and Resets

46.4.1.1 Functional clock

The LPSPI functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support SPI bus transfers in both master and slave modes. If the functional clock is disabled in slave mode, the LPSPI can transfer a single word before the functional clock needs to be enabled. The LPSPI divides the functional clock by a prescaler and the resulting frequency must be at least two times faster than the SPI clock frequency.

46.4.1.2 External clock

The LPSPI shift register is clocked directly by the external pins. This allows the LPSPI slave to remain operational in low power modes, even when the LPSPI functional clock is disabled, although this is limited to a single word transfer.

46.4.1.3 Bus clock

The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPSPI registers, including FIFOs.

46.4.1.4 Chip reset

The logic and registers for the LPSPI are reset to their default state on a chip reset.

46.4.1.5 Software reset

The LPSPI implements a software reset bit in the Control Register. The CR[RST] will reset all logic and registers to their default state, except for the CR itself.

46.4.1.6 FIFO reset

The LPSPI implements write-only control bits that resets the transmit/command FIFO (CR[RTF] and receive FIFO (CR[RRF]). A FIFO is empty after being reset.

46.4.2 Master Mode

46.4.2.1 Transmit and Command FIFO

The transmit and command FIFO is a combined FIFO that includes both transmit data and command words. Command words are stored to the transmit/command FIFO by writing the transmit command register. Transmit data words are stored to the transmit/command FIFO by writing the transmit data register.

When a command word is at the top of the transmit/command FIFO, the following actions can occur:

- If the LPSPI is between frames, the command word is pulled from the FIFO and controls all subsequent transfers.
- If the LPSPI is busy and either the existing CONT bit is clear or the new CONTC value is clear, the SPI frame will complete at the end of the existing word, ignoring

the FRAMESZ configuration. The command word is then pulled from the FIFO and controls all subsequent transfers (or until the next update to the command word).

- If the LPSPI is busy and the existing CONT bit is set and the new CONTC value is set, the command word is pulled from the FIFO during the last LPSPI_SCK pulse of the existing frame (based on FRAMESZ configuration) and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When CONTC is set, only the lower 24-bits of the command word are updated.

The current state of the existing command word can be read by reading the transmit command register. It requires at least three LPSPI functional clock cycles for the transmit command register to update after it is written (assuming an empty FIFO) and the LPSPI must be enabled (CR[MEN] is set).

Writing the transmit command register does not initiate a SPI bus transfer, unless the TXMSK bit is set. When TXMSK is set, a new command word will not be loaded until the end of the existing frame (based on FRAMESZ configuration) and the TXMSK bit will be cleared at the end of the transfer.

The following table describes the attributes that are controlled by the command word.

Table 46-3. LPSPI Command Word

Field	Description	Modify During Transfer
CPOL	Configures polarity of the LPSPI_SCK pin. Any change of CPOL value will cause a transition on the LPSPI_SCK pin.	N
CPHA	Configures clock phase of transfer.	N
PRESCALE	Configures prescaler used to divide the LPSPI functional clock to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS allows the LPSPI to connect to different slave devices at different frequencies.	N
PCS	Configures which LPSPI_PCS asserts for the transfer, the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.	N
LSBF	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted/received first.	Y
BYSW	Enables byte swap on each 32-bit word when transmitting and receiving data. Can be useful when interfacing to devices that organize data as big endean.	Y

Table continues on the next page...

Table 46-3. LPSPI Command Word (continued)

Field	Description	Modify During Transfer
CONT	Configures for a continuous transfer that keeps PCS asserted between frames (as configured by FRAMESZ). A new command word is required to cause PCS to negate. Also supports changing the command word at frame size boundaries.	Y
CONTC	Indicates this is a new command word for the existing continuous transfer. This bit is ignored when not written to the transmit/command FIFO on a frame boundary.	Y
RXMSK	Masks the receive data and does not store to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.	Y
TXMSK	Masks the transmit data, so that data is not pulled from transmit FIFO and the output data pin is tristated (unless configured by OUTCFG). Useful for half-duplex transfers.	Y
WIDTH	Configures the number of bits shifted on each LPSPI_SCK pulse. Single bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. Two and four bit transfers are useful for interfacing to QuadSPI memory devices and only support half-duplex data formats (at least one of TXMSK or RXMSK must also be set).	Y
FRAMESZ	Configures the number of bits in each frame to FRAMESZ+1. The minimum frame size is 8-bits and the maximum frame size is 4096-bits. If the frame size is less than or equal to 32-bits, the word size and frame size are identical. If the frame size is greater than 32-bits, then the word size is 32-bits for each word except the last (the last word contains the remainder bits if the frame size is not divisible by 32). The minimum word size is 2-bits, a frame size of 33-bits (or similar) is not supported.	Y

The LPSPI initiates a SPI bus transfer when data is written to the transmit FIFO, the HREQ pin is asserted (or disabled) and the LPSPI is enabled. The SPI bus transfer uses the attributes configured in the transmit command register and timing parameters from the clock configuration register to perform the transfer. The SPI bus transfer ends once

the FRAMESZ configuration is reached, or at the end of a word when a new transmit command word is at the top of the transmit/command FIFO. The HREQ input is only checked the next time the LPSPI goes idle (completes the current transfer and transmit/command register is empty).

The transmit/command FIFO also supports a Circular FIFO feature. This allows the LPSPI master to (periodically) repeat a short data transfer that can fit within the transmit/command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled, the current state of the FIFO read pointer is saved and the status flags do not update. Once the transmit/command FIFO is considered empty and the LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit/command FIFO are not permanently pulled from the FIFO while circular FIFO mode is enabled.

46.4.2.2 Receive FIFO and Data Match

The receive FIFO is used to store receive data during SPI bus transfers. When RXMSK is set, receive data is discarded instead of storing in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame. Receive data that is already discarded due to RXMSK bit cannot cause the data match to set and will delay the match on first received data word until after all discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

46.4.2.3 Timing Parameters

The following table lists the timing parameters that are used for all SPI bus transfers, these timing parameters are relative to the LPSPI functional clock divided by the PRESCALE configuration. Although the Clock Configuration Register cannot be changed when the LPSPI is busy, the PRESCALE configuration can be altered between transfers using the command register, to support interfacing to different slave devices at different frequencies.

Table 46-4. LPSPI Timing Parameters

Field	Description	Min	Max
SCKDIV	Configures the LPSPI_SCK clock period to (SCKDIV+2) cycles. When configured to an odd number of cycles, the first half of the LPSPI_SCK cycle is one cycle longer than the second half.	0 (2 cycles)	255 (257 cycles)
DBT	Configures the minimum delay between PCS negation and the next PCS assertion to (DBT + 2) cycles. When the command word is updated between transfers, there is a minimum of (DBT/2)+1 cycles between the command word update and any change on LPSPI_PCS pins.	0 (2 cycles)	255 (257 cycles)
DBT	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (DBT + 1) cycles. This is useful where the external slave requires a large delay between different words of a SPI bus transfer.	0 (1 cycle)	255 (256 cycles)
PCSSCK	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	Configures the minimum delay between the last SCK edge and the PCS assertion to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

46.4.2.4 Pin Configuration

The LPSPI_SIN and LPSPI_SOUT pins can be configured via the PINCFG configuration to swap directions or even support half-duplex transfers on the same pin.

The OUTCFG configuration can be used to determine if output data pin (eg: LPSPI_SOUT) will tristate when the LPSPI_PCS is negated, or if it will simply retain the last value. When configuring for half-duplex transfers using the same data pin in single bit transfer mode, or any transfer in 2-bit and 4-bit transfer modes, then the output data pins must be configured to tristate when LPSPI_PCS is negated.

The PCSCFG configuration is used to disable LPSPI_PCS[3:2] functions and to use them for quad-data transfers. This option must be enabled when performing quad-data transfers.

46.4.3 Slave Mode

LPSPI slave mode uses the same shift register and logic as the master mode, but does not use the clock configuration register and the transmit command register must remain static during SPI bus transfers.

46.4.3.1 Transmit and Command FIFO

The transmit command register should be initialized before enabling the LPSPI in slave mode, although the command register will not update until after the LPSPI is enabled. Once enabled, the transmit command register should only be changed if the LPSPI is idle. The following table lists how the command register functions in slave mode.

Table 46-5. LPSPI Command Word in Slave Mode

Field	Description
CPOL	Configures polarity of the external LPSPI_SCK input.
CPHA	Configures clock phase of transfer.
PRESCALE	Configures LPSPI functional clock prescaler.
PCS	Configures which LPSPI_PCS is used, the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.
LSBF	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted/received first.
BYSW	Enables byte swap on each 32-bit word when transmitting and receiving data. Can be useful when interfacing to devices that organize data as big endean.
CONT	When set, only the first FRAMSZ bits will be transmitted/ received by the LPSPI.
CONTC	This bit is reserved in slave mode.
RXMSK	Masks the receive data and does not store to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.
TXMSK	Masks the transmit data, so that data is not pulled from transmit FIFO and the output data pin is tristated (unless configured by OUTCFG). Useful for half-duplex transfers.
WIDTH	Configures the number of bits shifted on each LPSPI_SCK pulse. Single bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. Two and four bit transfers are useful for interfacing to QuadSPI memory

Table continues on the next page...

Table 46-5. LPSPI Command Word in Slave Mode (continued)

Field	Description
	devices and only support half-duplex data formats (at least one of TXMSK or RXMSK must also be set).
FRAMESZ	Configures the number of bits in each frame to FRAMESZ+1. The minimum frame size is 8-bits and the maximum frame size is 4096-bits. If the frame size is less than or equal to 32-bits, the word size and frame size are identical. If the frame size is greater than 32-bits, then the word size is 32-bits for each word except the last (the last word contains the remainder bits if the frame size is not divisible by 32). The minimum word size is 2-bits, a frame size of 33-bits (or similar) is not supported.

The transmit FIFO must be filled with transmit data before the LPSPI_PCS input asserts, otherwise the transmit error flag will set.

46.4.3.2 Receive FIFO and Data Match

The receive FIFO is used to store receive data during SPI bus transfers. When RXMSK is set, receive data is discarded instead of storing in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame. Receive data that is already discarded due to RXMSK bit cannot cause the data match to set and will delay the match on first received data word until after all discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

46.4.3.3 Clocked Interface

The LPSPI supports interfacing to external masters that provide only clock and data pins (LPSPI_PCS is not required). This requires using CPHA=1, configuring the LPSPI_PCS input to be always asserted (configure PCSPOL) and setting the AUTOPCS bit. When AUTOPCS is set, a minimum of four LPSPI functional clock cycles (divided by PRESCALE configuration) is required between the last LPSPI_SCK edge of one word and the first LPSPI_SCK edge of the next word.

46.4.4 Interrupts and DMA Requests

The following table illustrates the status flags that can generate the LPSPI interrupt and LPSPI transmit/receive DMA requests.

Table 46-6. LPSPI Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit FIFO, as configured by TXWATER.	Y	TX	Y
RDF	Data can be read from the receive FIFO, as configured by RXWATER.	Y	RX	Y
WCF	Word complete, last bit of word has been sampled.	Y	N	Y
FCF	Frame complete, PCS has negated .	Y	N	Y
TCF	Transfer complete, PCS has negated and transmit/command FIFO is empty.	Y	N	Y
TEF	Transmit error flag, indicates transmit/ command FIFO underrun. This bit cannot set in master mode when NOSTALL is clear.	Y	N	Y
REF	Receive error flag, indicates receive FIFO overflow. This bit cannot set in master mode when NOSTALL is clear.	Y	N	Y
DMF	Data match flag, received data has matched the configured data match value.	Y	N	Y
MBF	LPSPI is busy performing a SPI bus transfer.	N	N	N

46.4.5 Peripheral Triggers

The connection of the LPSPI peripheral triggers with other peripherals are device specific.

46.4.5.1 Output Triggers

The LPSPI generates two output triggers that can be connected to other peripherals on the device. The frame output trigger asserts at the end of each frame (when PCS negates) and remains asserted until PCS next asserts. The word output trigger asserts at the end of each received word and remains asserted for one LPSPI_SCK period.

46.4.5.2 Input Trigger

The LPSPI input trigger can be selected in place of the LPSPI_HREQ input to control the start of a LPSPI bus transfer. The input trigger must assert for longer than one LPSPI functional clock cycle to be detected.

Chapter 47
Low Power Inter-Integrated Circuit (LPI2C)

47.1 Chip-specific information for this module

47.1.1 Instantiation Information

This device has two LPI2C modules. The LPI2C can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

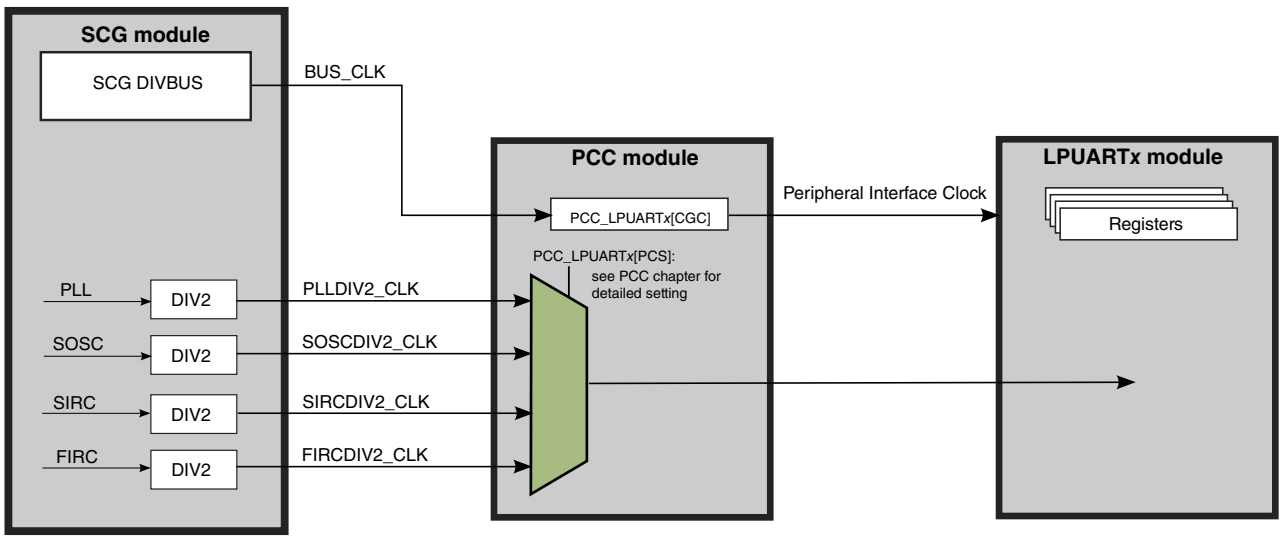
Table 47-1. LPI2C Configuration

	TX FIFO (word/8bit)	RX FIFO (word/8bit)	SMBus	Slave mode enable
LPI2C0	4	4	Yes	Yes
LPI2C1	4	4	Yes	Yes

47.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

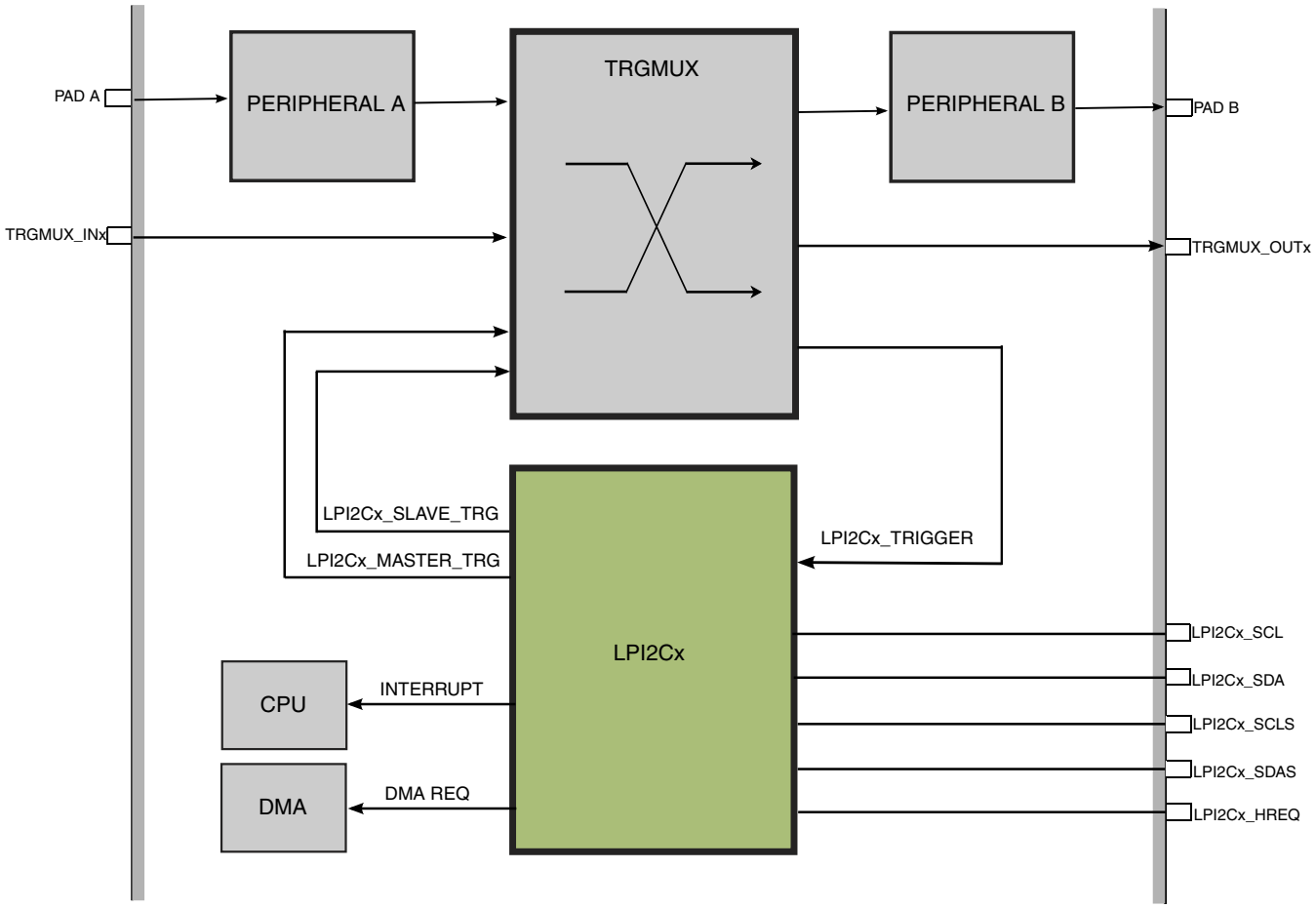
The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPUART Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, FlexIO and LPIT.



47.1.3 Inter-connectivity Information

The LPI2C inter-connectivity is shown in following diagram.



47.2 Introduction

47.2.1 Overview

The LPI2C is a low power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I2C bus as a master and/or a slave. The LPI2C can continue operating in stop modes provided an appropriate clock is available and is designed for low CPU overhead with DMA offloading of FIFO register accesses. The LPI2C implements logic support for standard-mode, fast-mode, fast-mode plus and ultra-fast modes of operation. The LPI2C module also complies with the System Management Bus (SMBus) Specification, version 2.

47.2.2 Features

The LPI2C supports the following features of the I2C specification:

- Standard, Fast, Fast+ and Ultra Fast modes are supported.
- HS-mode supported in slave mode.
- HS-mode supported for master mode, provided SCL pin implements current source pull-up (device specific).
- Multi-master support including synchronization and arbitration.
- Clock stretching.
- General call, 7-bit and 10-bit addressing.
- Software reset, START byte and Device ID require software support.

The LPI2C master supports the following features:

- Command/transmit FIFO of 4 words.
- Receive FIFO of 4 words.
- Command FIFO will wait for idle I2C bus before initiating transfer
- Command FIFO can initiate (repeated) START and STOP conditions and one or more master-receiver transfers.
- STOP condition can be generated from command FIFO or automatically when the transmit FIFO is empty.
- Host request input can be used to control the start time of an I2C bus transfer.
- Flexible receive data match can generate interrupt on data match and/or discard unwanted data.
- Flag and optional interrupt to signal Repeated START condition, STOP condition, loss of arbitration, unexpected NACK and command word errors.
- Supports configurable bus idle timeout and pin stuck low timeout.

The LPI2C slave supports the following features:

- Separate I2C slave registers to minimize software overhead due to master/slave switching.
- Support for 7-bit or 10-bit addressing, address range, SMBus alert and general call address.
- Transmit data register supporting interrupt or DMA requests.
- Receive data register supporting interrupt or DMA requests.
- Software controllable ACK or NACK, with optional clock stretching on ACK/NACK bit.
- Configurable clock stretching to avoid transmit FIFO underrun and receive FIFO overrun.
- Flag and optional interrupt at end of packet, STOP condition or bit error detection.

47.2.3 Block Diagram

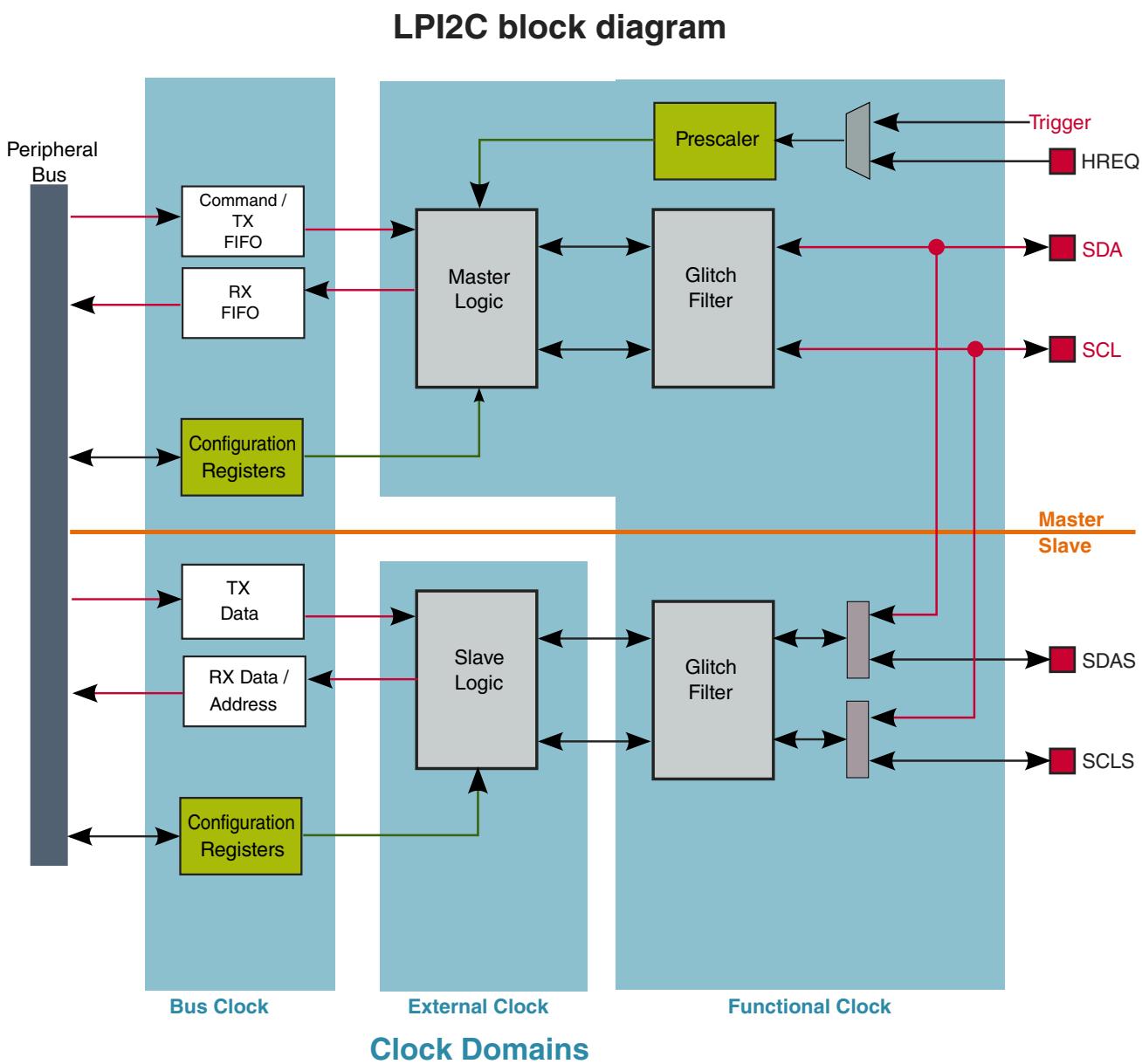


Figure 47-1. LPI2C block diagram

47.2.4 Modes of operation

The LPI2C module supports the chip modes described in the following table.

Table 47-2. Chip modes supported by the LPI2C module

Chip mode	LPI2C Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (MCR[DOZEN]) is set and the LPI2C is using an external or internal clock source which remains operating during stop/wait modes.
Debug	Can continue operating provided the Debug Enable bit (MCR[DBGEN]) is set.

47.2.5 Signal Descriptions

Signal	Description	I/O
SCL	LPI2C clock line. In 4-wire mode, this is the SCL input pin.	I/O
SDA	LPI2C data line. In 4-wire mode, this is the SDA input pin.	I/O
HREQ	Host request, can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
SCLS	Secondary I2C clock line. In 4-wire mode, this is the SCL output pin. If LPI2C master/slave are configured to use separate pins, this the LPI2C slave SCL pin.	I/O
SDAS	Secondary I2C data line. In 4-wire mode, this is the SDA output pin. If LPI2C master/slave are configured to use separate pins, this the LPI2C slave SDA pin.	I/O

47.3 Memory Map and Registers

LPI2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6000	Version ID Register (LPI2C0_VERID)	32	R	See section	47.3.1/1243
4006_6004	Parameter Register (LPI2C0_PARAM)	32	R	See section	47.3.2/1243
4006_6010	Master Control Register (LPI2C0_MCR)	32	R/W	0000_0000h	47.3.3/1244
4006_6014	Master Status Register (LPI2C0_MSR)	32	R/W	0000_0001h	47.3.4/1245
4006_6018	Master Interrupt Enable Register (LPI2C0_MIER)	32	R/W	0000_0000h	47.3.5/1247
4006_601C	Master DMA Enable Register (LPI2C0_MDER)	32	R/W	0000_0000h	47.3.6/1249

Table continues on the next page...

LPI2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6020	Master Configuration Register 0 (LPI2C0_MCFGR0)	32	R/W	0000_0000h	47.3.7/1250
4006_6024	Master Configuration Register 1 (LPI2C0_MCFGR1)	32	R/W	0000_0000h	47.3.8/1251
4006_6028	Master Configuration Register 2 (LPI2C0_MCFGR2)	32	R/W	0000_0000h	47.3.9/1253
4006_602C	Master Configuration Register 3 (LPI2C0_MCFGR3)	32	R/W	0000_0000h	47.3.10/ 1254
4006_6040	Master Data Match Register (LPI2C0_MDMR)	32	R/W	0000_0000h	47.3.11/ 1254
4006_6048	Master Clock Configuration Register 0 (LPI2C0_MCCR0)	32	R/W	0000_0000h	47.3.12/ 1255
4006_6050	Master Clock Configuration Register 1 (LPI2C0_MCCR1)	32	R/W	0000_0000h	47.3.13/ 1256
4006_6058	Master FIFO Control Register (LPI2C0_MFCR)	32	R/W	0000_0000h	47.3.14/ 1257
4006_605C	Master FIFO Status Register (LPI2C0_MFSR)	32	R	0000_0000h	47.3.15/ 1257
4006_6060	Master Transmit Data Register (LPI2C0_MTDR)	32	W	0000_0000h	47.3.16/ 1258
4006_6070	Master Receive Data Register (LPI2C0_MRDR)	32	R	0000_4000h	47.3.17/ 1259
4006_6110	Slave Control Register (LPI2C0_SCR)	32	R/W	0000_0000h	47.3.18/ 1260
4006_6114	Slave Status Register (LPI2C0_SSR)	32	R/W	0000_0000h	47.3.19/ 1261
4006_6118	Slave Interrupt Enable Register (LPI2C0_SIER)	32	R/W	0000_0000h	47.3.20/ 1264
4006_611C	Slave DMA Enable Register (LPI2C0_SDER)	32	R/W	0000_0000h	47.3.21/ 1265
4006_6124	Slave Configuration Register 1 (LPI2C0_SCFGR1)	32	R/W	0000_0000h	47.3.22/ 1266
4006_6128	Slave Configuration Register 2 (LPI2C0_SCFGR2)	32	R/W	0000_0000h	47.3.23/ 1268
4006_6140	Slave Address Match Register (LPI2C0_SAMR)	32	R/W	0000_0000h	47.3.24/ 1269
4006_6150	Slave Address Status Register (LPI2C0_SASR)	32	R	0000_4000h	47.3.25/ 1270
4006_6154	Slave Transmit ACK Register (LPI2C0_STAR)	32	R/W	0000_0000h	47.3.26/ 1271
4006_6160	Slave Transmit Data Register (LPI2C0_STDR)	32	W	0000_0000h	47.3.27/ 1271
4006_6170	Slave Receive Data Register (LPI2C0_SRDR)	32	R	0000_4000h	47.3.28/ 1272
4006_7000	Version ID Register (LPI2C1_VERID)	32	R	See section	47.3.1/1243
4006_7004	Parameter Register (LPI2C1_PARAM)	32	R	See section	47.3.2/1243

Table continues on the next page...

LPI2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_7010	Master Control Register (LPI2C1_MCR)	32	R/W	0000_0000h	47.3.3/1244
4006_7014	Master Status Register (LPI2C1_MSR)	32	R/W	0000_0001h	47.3.4/1245
4006_7018	Master Interrupt Enable Register (LPI2C1_MIER)	32	R/W	0000_0000h	47.3.5/1247
4006_701C	Master DMA Enable Register (LPI2C1_MDER)	32	R/W	0000_0000h	47.3.6/1249
4006_7020	Master Configuration Register 0 (LPI2C1_MCFGR0)	32	R/W	0000_0000h	47.3.7/1250
4006_7024	Master Configuration Register 1 (LPI2C1_MCFGR1)	32	R/W	0000_0000h	47.3.8/1251
4006_7028	Master Configuration Register 2 (LPI2C1_MCFGR2)	32	R/W	0000_0000h	47.3.9/1253
4006_702C	Master Configuration Register 3 (LPI2C1_MCFGR3)	32	R/W	0000_0000h	47.3.10/ 1254
4006_7040	Master Data Match Register (LPI2C1_MDMR)	32	R/W	0000_0000h	47.3.11/ 1254
4006_7048	Master Clock Configuration Register 0 (LPI2C1_MCCR0)	32	R/W	0000_0000h	47.3.12/ 1255
4006_7050	Master Clock Configuration Register 1 (LPI2C1_MCCR1)	32	R/W	0000_0000h	47.3.13/ 1256
4006_7058	Master FIFO Control Register (LPI2C1_MFCR)	32	R/W	0000_0000h	47.3.14/ 1257
4006_705C	Master FIFO Status Register (LPI2C1_MFSR)	32	R	0000_0000h	47.3.15/ 1257
4006_7060	Master Transmit Data Register (LPI2C1_MTDR)	32	W	0000_0000h	47.3.16/ 1258
4006_7070	Master Receive Data Register (LPI2C1_MRDR)	32	R	0000_4000h	47.3.17/ 1259
4006_7110	Slave Control Register (LPI2C1_SCR)	32	R/W	0000_0000h	47.3.18/ 1260
4006_7114	Slave Status Register (LPI2C1_SSR)	32	R/W	0000_0000h	47.3.19/ 1261
4006_7118	Slave Interrupt Enable Register (LPI2C1_SIER)	32	R/W	0000_0000h	47.3.20/ 1264
4006_711C	Slave DMA Enable Register (LPI2C1_SDER)	32	R/W	0000_0000h	47.3.21/ 1265
4006_7124	Slave Configuration Register 1 (LPI2C1_SCFGR1)	32	R/W	0000_0000h	47.3.22/ 1266
4006_7128	Slave Configuration Register 2 (LPI2C1_SCFGR2)	32	R/W	0000_0000h	47.3.23/ 1268
4006_7140	Slave Address Match Register (LPI2C1_SAMR)	32	R/W	0000_0000h	47.3.24/ 1269
4006_7150	Slave Address Status Register (LPI2C1_SASR)	32	R	0000_4000h	47.3.25/ 1270
4006_7154	Slave Transmit ACK Register (LPI2C1_STAR)	32	R/W	0000_0000h	47.3.26/ 1271
4006_7160	Slave Transmit Data Register (LPI2C1_STDR)	32	W	0000_0000h	47.3.27/ 1271

Table continues on the next page...

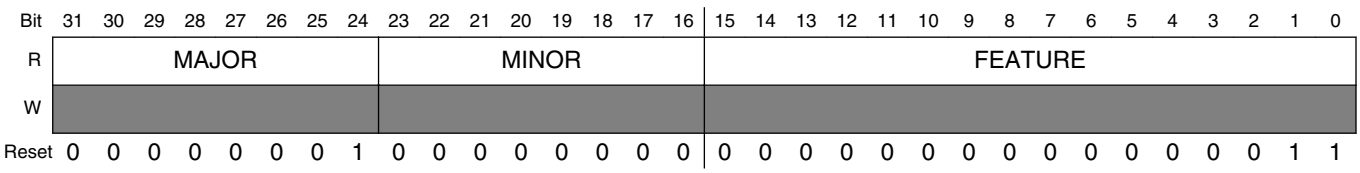
LPI2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_7170	Slave Receive Data Register (LPI2C1_SRDR)	32	R	0000_4000h	47.3.28/ 1272

47.3.1 Version ID Register (LPI2Cx_VERID)

.

Address: Base address + 0h offset



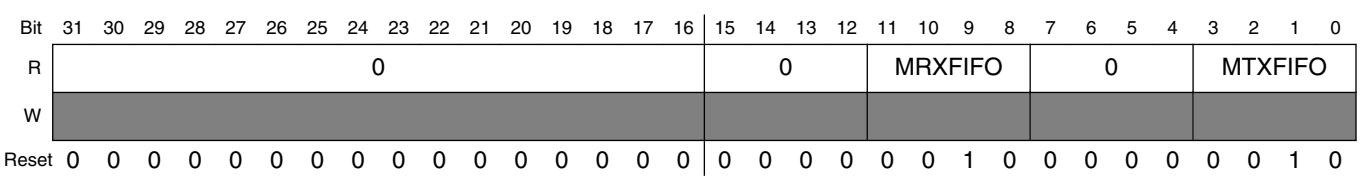
LPI2Cx_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0002 Master only with standard feature set. 0x0003 Master and slave with standard feature set.

47.3.2 Parameter Register (LPI2Cx_PARAM)

.

Address: Base address + 4h offset

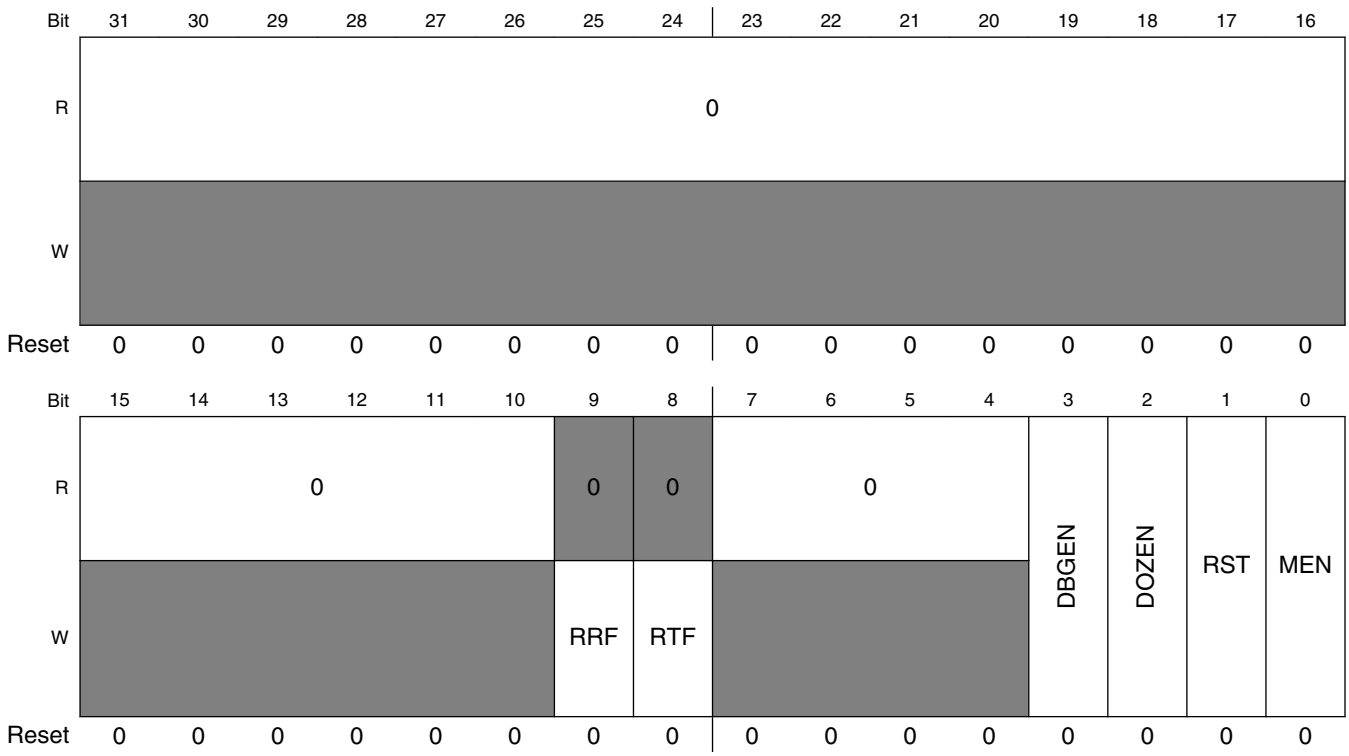


LPI2Cx_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 MRXFIFO	Master Receive FIFO Size The number of words in the master receive FIFO is 2^MRXFIFO.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MTXFIFO	Master Transmit FIFO Size The number of words in the master transmit FIFO is 2^MTXFIFO.

47.3.3 Master Control Register (LPI2Cx_MCR)

Address: Base address + 10h offset



LPI2Cx_MCR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

LPI2Cx_MCR field descriptions (continued)

Field	Description
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive FIFO is reset.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit FIFO is reset.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBGEN	Debug Enable 0 Master is disabled in debug mode. 1 Master is enabled in debug mode.
2 DOZEN	Doze mode enable Enables or disables Doze mode for the master. 0 Master is enabled in Doze mode. 1 Master is disabled in Doze mode.
1 RST	Software Reset Reset all internal master logic and registers, except the Master Control Register. Remains set until cleared by software. 0 Master logic is not reset. 1 Master logic is reset.
0 MEN	Master Enable 0 Master logic is disabled. 1 Master logic is enabled.

47.3.4 Master Status Register (LPI2Cx_MSR)

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	MBF	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	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W		w1c	w1c	w1c	w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

LPI2Cx_MSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 BBF	Bus Busy Flag 0 I2C Bus is idle. 1 I2C Bus is busy.
24 MBF	Master Busy Flag 0 I2C Master is idle. 1 I2C Master is busy.
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields as configured by MATCFG. Received data that is discarded due to CMD field does not cause this flag to set. 0 Have not received matching data. 1 Have received matching data.
13 PLTF	Pin Low Timeout Flag Will set when the SCL and/or SDA input is low for more than PINLOW cycles, even when the LPI2C master is idle. Software is responsible for resolving the pin low condition. This flag cannot be cleared for as long as the pin low timeout continues and must be cleared before the LPI2C can initiate a START condition. 0 Pin low timeout has not occurred or is disabled. 1 Pin low timeout has occurred.
12 FEF	FIFO Error Flag Detects an attempt to send or receive data without first generating a (repeated) START condition. This can occur if the transmit FIFO underflows when the AUTOSTOP bit is set. When this flag is set, the LPI2C master will send a STOP condition (if busy) and will not initiate a new START condition until this flag has been cleared. 0 No error. 1 Master sending or receiving data without START condition.
11 ALF	Arbitration Lost Flag This flag will set if the LPI2C master transmits a logic one and detects a logic zero on the I2C bus, or if it detects a START or STOP condition while it is transmitting data. When this flag sets, the LPI2C master will release the bus (go idle) and will not initiate a new START condition until this flag has been cleared. 0 Master has not lost arbitration. 1 Master has lost arbitration.
10 NDF	NACK Detect Flag This flag will set if the LPI2C master detects a NACK when transmitting an address or data. If a NACK is expected for a given address (as configured by the command word) then the flag will set if a NACK is not generated. When set, the master will transmit a STOP condition and will not initiate a new START condition until this flag has been cleared.

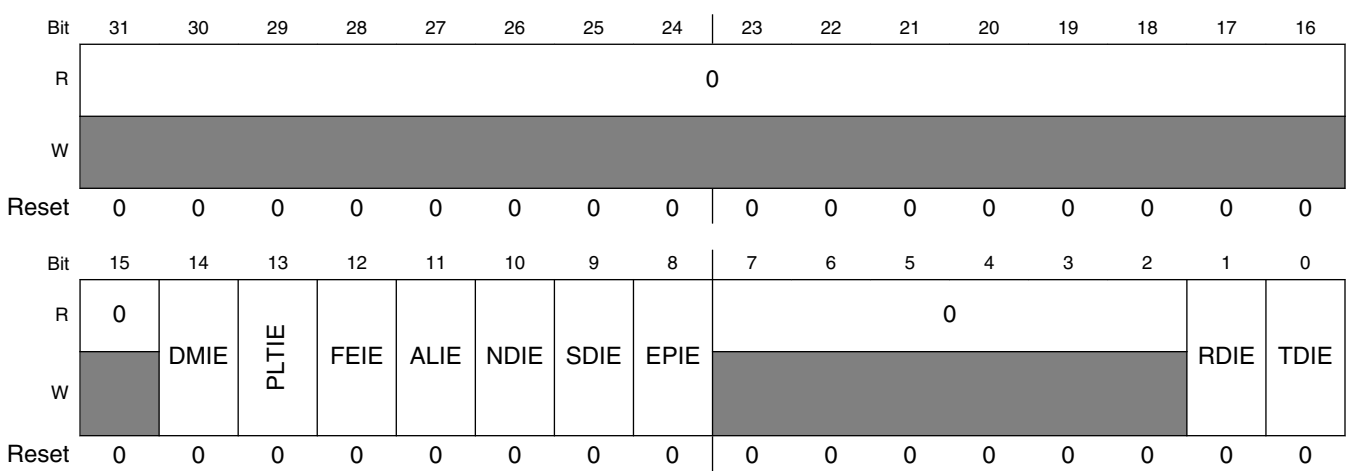
Table continues on the next page...

LPI2Cx_MSR field descriptions (continued)

Field	Description
	0 Unexpected NACK not detected. 1 Unexpected NACK was detected.
9 SDF	STOP Detect Flag This flag will set when the LPI2C master generates a STOP condition. 0 Master has not generated a STOP condition. 1 Master has generated a STOP condition.
8 EPF	End Packet Flag This flag will set when the LPI2C master generates either a repeated START or a STOP condition. It does not set when the master first generates a START condition. 0 Master has not generated a STOP or Repeated START condition. 1 Master has generated a STOP or Repeated START condition.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. 0 Transmit data not requested. 1 Transmit data is requested.

47.3.5 Master Interrupt Enable Register (LPI2Cx_MIER)

Address: Base address + 18h offset

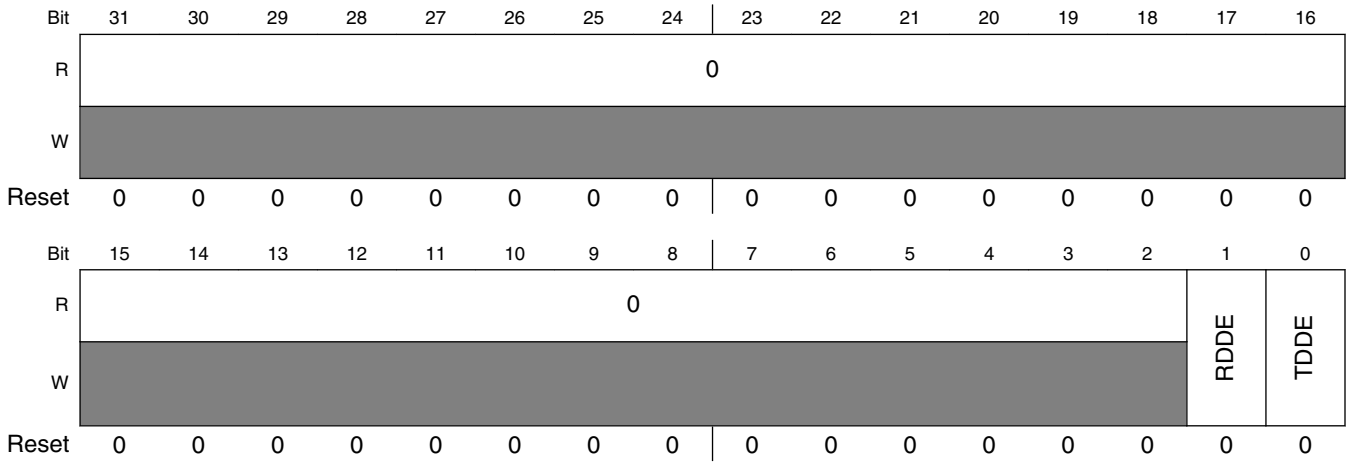


LPI2Cx_MIER field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DMIE	Data Match Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
13 PLTIE	Pin Low Timeout Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 FEIE	FIFO Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
11 ALIE	Arbitration Lost Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 NDIE	NACK Detect Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 SDIE	STOP Detect Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
8 EPIE	End Packet Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
0 TDIE	Transmit Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled

47.3.6 Master DMA Enable Register (LPI2Cx_MDER)

Address: Base address + 1Ch offset

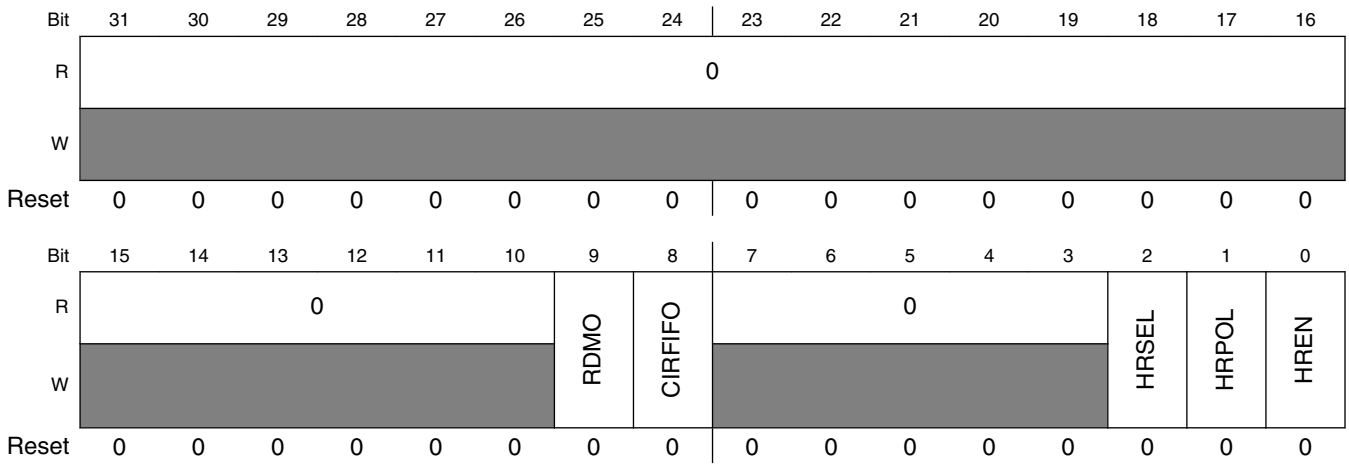


LPI2Cx_MDER field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

47.3.7 Master Configuration Register 0 (LPI2Cx_MCFGR0)

Address: Base address + 20h offset



LPI2Cx_MCFGR0 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause DMF to set is discarded. Once DMF is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing DMF to ensure no receive data is lost. 0 Received data is stored in the receive FIFO as normal. 1 Received data is discarded unless the RMF is set.
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as normal, but once the LPI2C master is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This will cause the contents of the transmit FIFO to be cycled through repeatedly. If AUTOSTOP is set, a STOP condition will be sent whenever the transmit FIFO is empty and the read pointer is restored. 0 Circular FIFO is disabled. 1 Circular FIFO is enabled.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HRSEL	Host Request Select Selects the source of the host request input. 0 Host request input is pin LPI2C_HREQ. 1 Host request input is input trigger.
1 HRPOL	Host Request Polarity

Table continues on the next page...

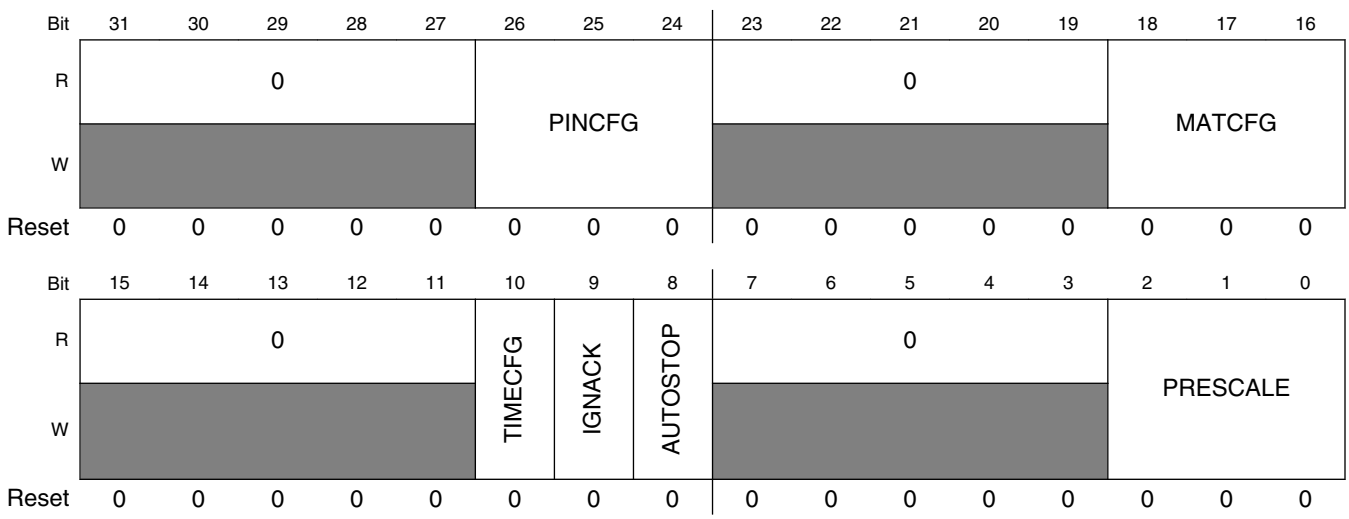
LPI2Cx_MCFGR0 field descriptions (continued)

Field	Description
	Configures the polarity of the host request input pin. 0 Active low. 1 Active high.
0 HREN	Host Request Enable When enabled, the LPI2C master will only initiate a START condition if the host request input is asserted and the bus is idle. A repeated START is not affected by the host request. 0 Host request input is disabled. 1 Host request input is enabled.

47.3.8 Master Configuration Register 1 (LPI2Cx_MCFGR1)

The MCFGR1 should only be written when the I2C Master is disabled.

Address: Base address + 24h offset



LPI2Cx_MCFGR1 field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26–24 PINCFG	Pin Configuration Configures the pin mode. 000 LPI2C configured for 2-pin open drain mode. 001 LPI2C configured for 2-pin output only mode (ultra-fast mode). 010 LPI2C configured for 2-pin push-pull mode. 011 LPI2C configured for 4-pin push-pull mode. 100 LPI2C configured for 2-pin open drain mode with separate LPI2C slave.

Table continues on the next page...

LPI2Cx_MCFGR1 field descriptions (continued)

Field	Description
	101 LPI2C configured for 2-pin output only mode (ultra-fast mode) with separate LPI2C slave. 110 LPI2C configured for 2-pin push-pull mode with separate LPI2C slave. 111 LPI2C configured for 4-pin push-pull mode (inverted outputs).
23–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. 000 Match disabled. 001 Reserved. 010 Match enabled (1st data word equals MATCH0 OR MATCH1). 011 Match enabled (any data word equals MATCH0 OR MATCH1). 100 Match enabled (1st data word equals MATCH0 AND 2nd data word equals MATCH1). 101 Match enabled (any data word equals MATCH0 AND next data word equals MATCH1). 110 Match enabled (1st data word AND MATCH1 equals MATCH0 AND MATCH1). 111 Match enabled (any data word AND MATCH1 equals MATCH0 AND MATCH1).
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 TIMECFG	Timeout Configuration 0 Pin Low Timeout Flag will set if SCL is low for longer than the configured timeout. 1 Pin Low Timeout Flag will set if either SCL or SDA is low for longer than the configured timeout.
9 IGNACK	When set, the received NACK field is ignored and assumed to be ACK. This bit is required to be set in Ultra-Fast Mode. 0 LPI2C Master will receive ACK and NACK normally. 1 LPI2C Master will treat a received NACK as if it was an ACK.
8 AUTOSTOP	Automatic STOP Generation When enabled, a STOP condition is generated whenever the LPI2C master is busy and the transmit FIFO is empty. The STOP condition can also be generated using a transmit FIFO command. 0 No effect. 1 STOP condition is automatically generated whenever the transmit FIFO is empty and LPI2C master is busy.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C master logic, except the digital glitch filters. 000 Divide by 1. 001 Divide by 2. 010 Divide by 4. 011 Divide by 8. 100 Divide by 16. 101 Divide by 32.

Table continues on the next page...

LPI2Cx_MCFGR1 field descriptions (continued)

Field	Description
	110 Divide by 64. 111 Divide by 128.

47.3.9 Master Configuration Register 2 (LPI2Cx_MCFGR2)

The MCFGR2 should only be written when the I2C Master is disabled.

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	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				FILTSDA				0				FILTSCL				0				BUSIDLE											
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

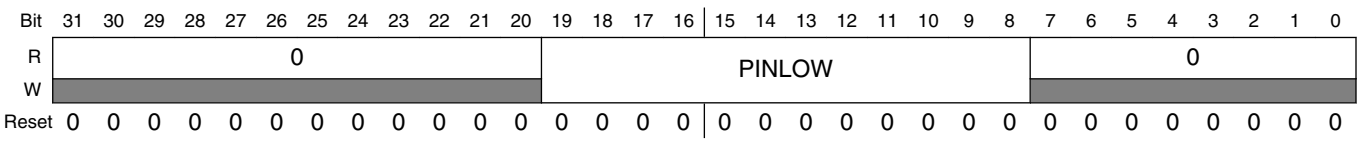
LPI2Cx_MCFGR2 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 FILTSDA	Glitch Filter SDA Configures the I2C master digital glitch filters for SDA input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSDA cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in High Speed mode.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 FILTSCL	Glitch Filter SCL Configures the I2C master digital glitch filters for SCL input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSCL cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in High Speed mode.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period in clock cycles. If both SCL and SDA are high for longer than BUSIDLE cycles, then the I2C bus is assumed to be idle and the master can generate a START condition. When set to zero, this feature is disabled.

47.3.10 Master Configuration Register 3 (LPI2Cx_MCFGR3)

The MCFGR3 should only be written when the I2C Master is disabled.

Address: Base address + 2Ch offset

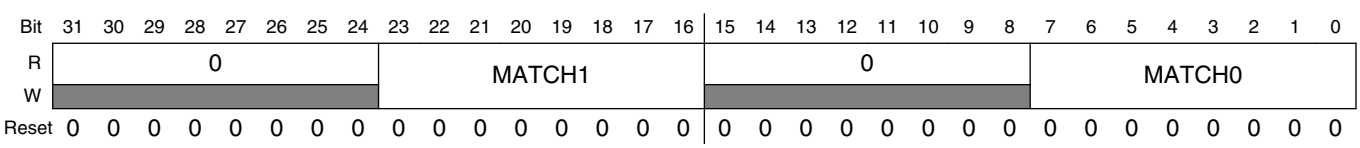


LPI2Cx_MCFGR3 field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–8 PINLOW	Pin Low Timeout Configures the pin low timeout flag in clock cycles. If SCL and/or SDA is low for longer than (PINLOW * 256) cycles then PLTF is set. When set to zero, this feature is disabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

47.3.11 Master Data Match Register (LPI2Cx_MDMR)

Address: Base address + 40h offset



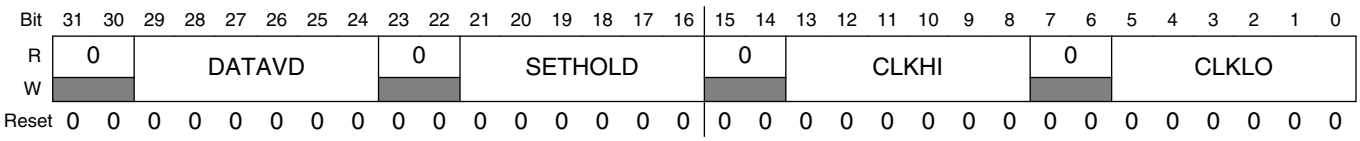
LPI2Cx_MDMR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 MATCH1	Match 1 Value Compared against the received data when receive data match is enabled.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

47.3.12 Master Clock Configuration Register 0 (LPI2Cx_MCCR0)

The MCCR0 cannot be changed when the I2C master is enabled and is used for standard, fast, fast-mode plus and ultra-fast transfers.

Address: Base address + 48h offset



LPI2Cx_MCCR0 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master as the setup and hold time for a (repeated) START condition and setup time for a STOP condition. The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. This value is also used for the minimum bus free time between a STOP and a START condition.

47.3.13 Master Clock Configuration Register 1 (LPI2Cx_MCCR1)

The MCCR1 cannot be changed when the I2C master is enabled and is used for high speed mode transfers. The separate clock configuration for high speed mode allows arbitration to take place in Fast mode (with timing configured by MCCR0), before switching to high speed mode (with timing configured by MCCR1).

Address: Base address + 50h offset

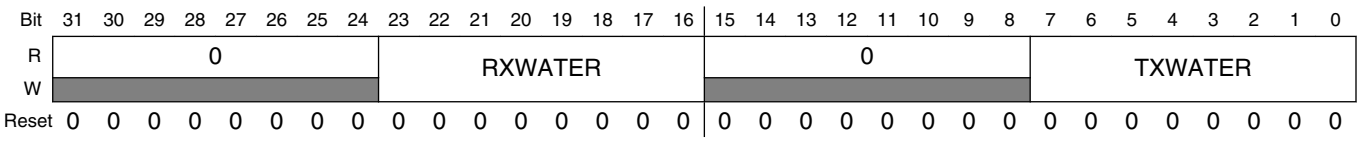
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DATAVD						0		SETHOLD						0		CLKHI						0		CLKLO					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2Cx_MCCR1 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master as the setup and hold time for a (repeated) START condition and setup time for a STOP condition. The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. This value is also used for the minimum bus free time between a STOP and a START condition.

47.3.14 Master FIFO Control Register (LPI2Cx_MFCR)

Address: Base address + 58h offset

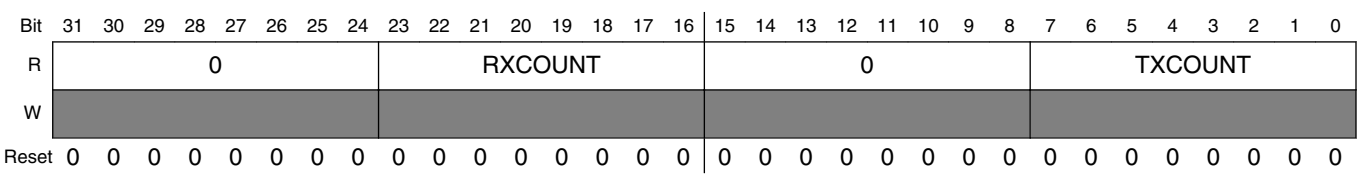


LPI2Cx_MFCR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

47.3.15 Master FIFO Status Register (LPI2Cx_MFSR)

Address: Base address + 5Ch offset



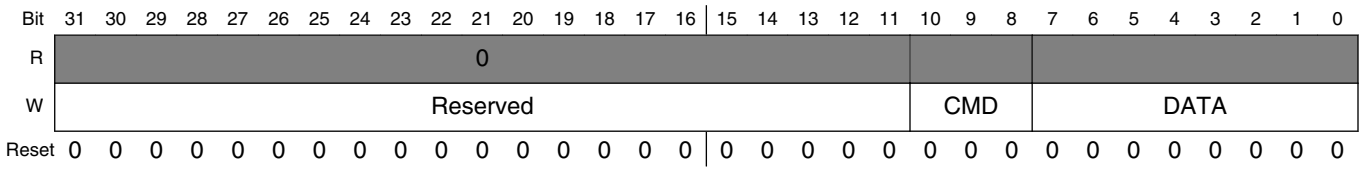
LPI2Cx_MFSR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

47.3.16 Master Transmit Data Register (LPI2Cx_MTDR)

An 8-bit write to the CMD field will store the data in the Command FIFO, but does not increment the FIFO write pointer. An 8-bit write to the DATA field will zero extend the CMD field unless the CMD field has been written separately since the last FIFO write, it also increments the FIFO write pointer. A 16-bit or 32-bit will write both the CMD and DATA fields and increment the FIFO.

Address: Base address + 60h offset

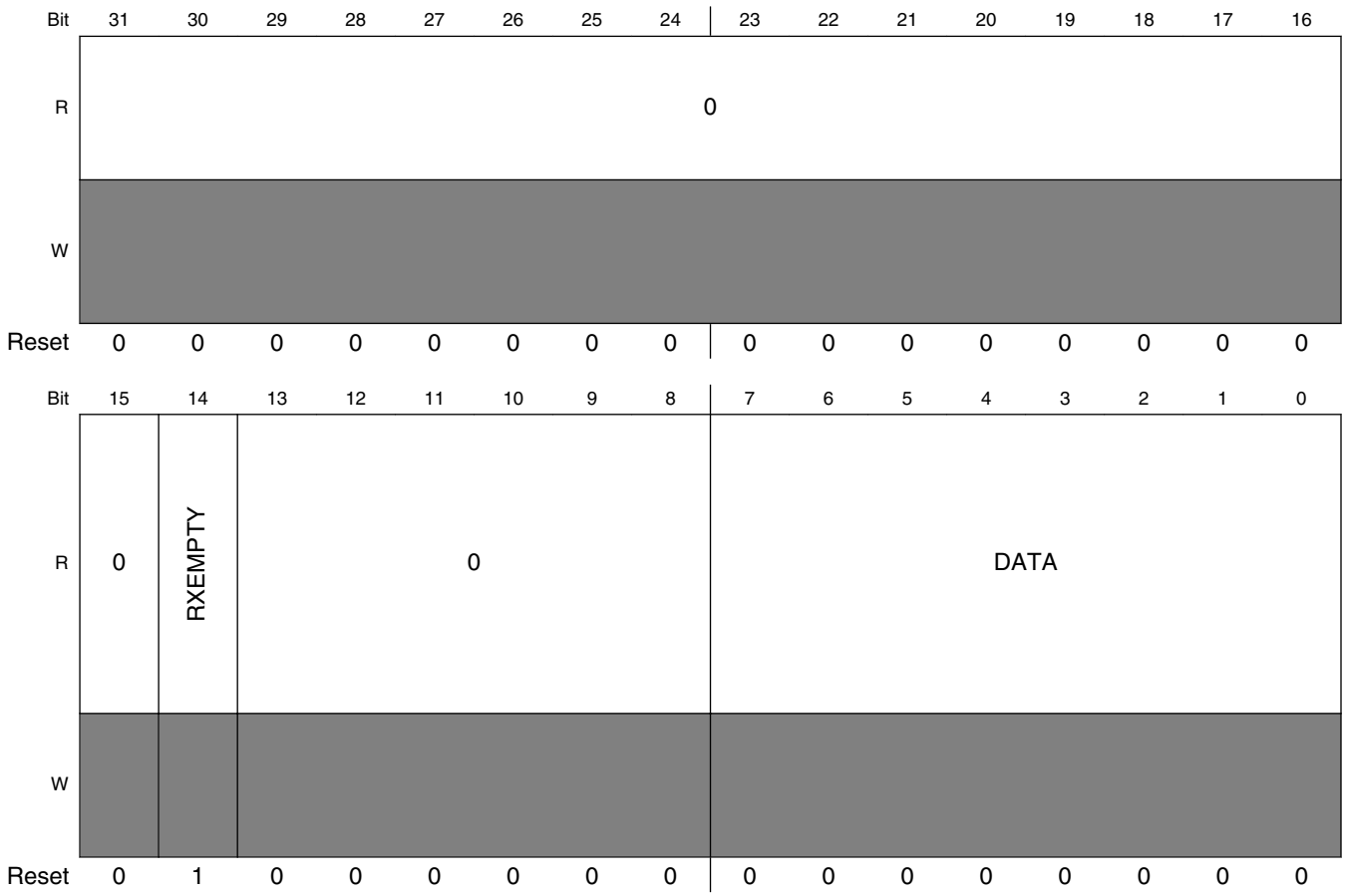


LPI2Cx_MTDR field descriptions

Field	Description
31–11 Reserved	This field is reserved.
10–8 CMD	Command Data 000 Transmit DATA[7:0]. 001 Receive (DATA[7:0] + 1) bytes. 010 Generate STOP condition. 011 Receive and discard (DATA[7:0] + 1) bytes. 100 Generate (repeated) START and transmit address in DATA[7:0]. 101 Generate (repeated) START and transmit address in DATA[7:0]. This transfer expects a NACK to be returned. 110 Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. 111 Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. This transfer expects a NACK to be returned.
DATA	Transmit Data Performing an 8-bit write to DATA will zero extend the CMD field.

47.3.17 Master Receive Data Register (LPI2Cx_MRDR)

Address: Base address + 70h offset

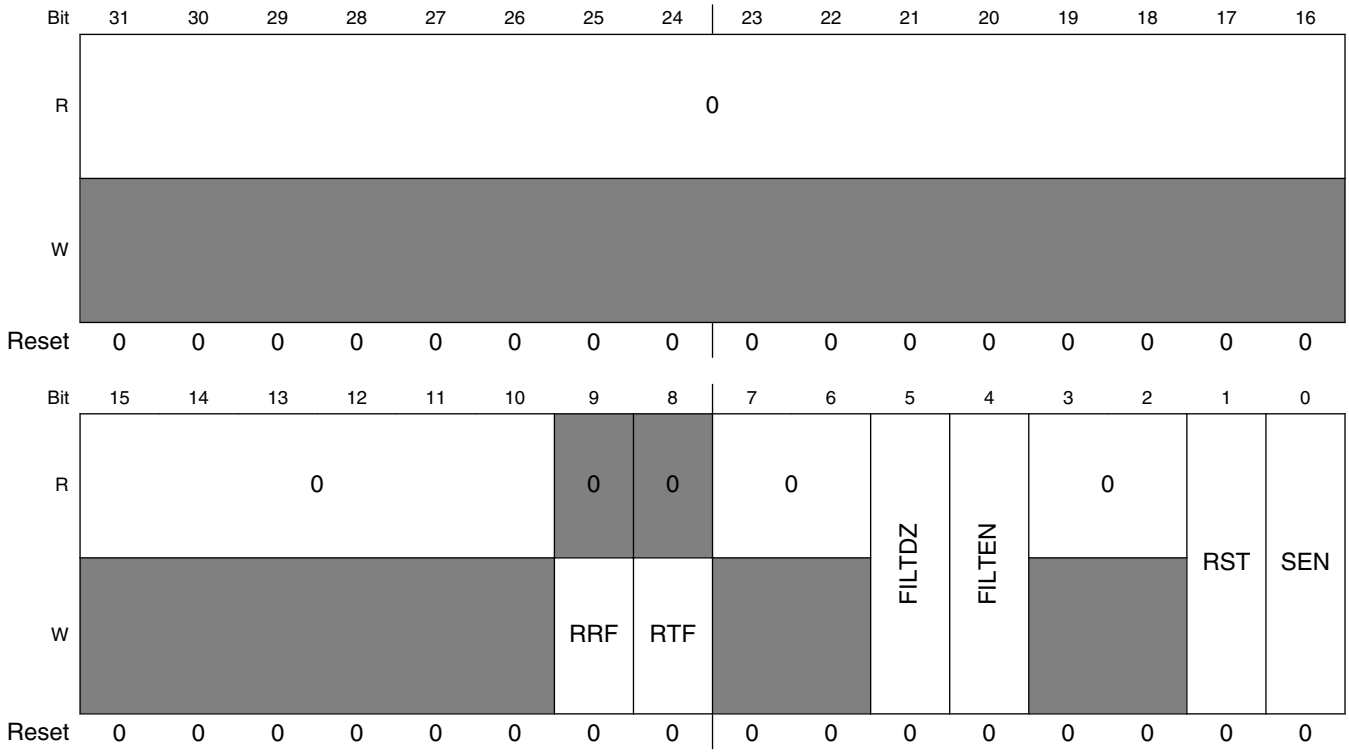


LPI2Cx_MRDR field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 RXEMPTY	RX Empty 0 Receive FIFO is not empty. 1 Receive FIFO is empty.
13–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA	Receive Data Reading this register returns the data received by the I2C master that has not been discarded. Receive data can be discarded due to the CMD field or the master can be configured to discard non-matching data.

47.3.18 Slave Control Register (LPI2Cx_SCR)

Address: Base address + 110h offset



LPI2Cx_SCR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive Data Register is now empty.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit Data Register is now empty.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 FILTDZ	Filter Doze Enable 0 Filter remains enabled in Doze mode. 1 Filter is disabled in Doze mode.
4 FILTEN	Filter Enable 0 Disable digital filter and output delay counter for slave mode. 1 Enable digital filter and output delay counter for slave mode.

Table continues on the next page...

LPI2Cx_SCR field descriptions (continued)

Field	Description
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RST	Software Reset 0 Slave logic is not reset. 1 Slave logic is reset.
0 SEN	Slave Enable 0 Slave mode is disabled. 1 Slave mode is enabled.

47.3.19 Slave Status Register (LPI2Cx_SSR)

Address: Base address + 114h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2Cx_SSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 BBF	Bus Busy Flag

Table continues on the next page...

LPI2Cx_SSR field descriptions (continued)

Field	Description
	0 I2C Bus is idle. 1 I2C Bus is busy.
24 SBF	Slave Busy Flag 0 I2C Slave is idle. 1 I2C Slave is busy.
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SARF	SMBus Alert Response Flag This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 SMBus Alert Response disabled or not detected. 1 SMBus Alert Response enabled and detected.
14 GCF	General Call Flag This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Slave has not detected the General Call Address or General Call Address disabled. 1 Slave has detected the General Call Address.
13 AM1F	Address Match 1 Flag Indicates that the received address has matched the ADDR1 field or ADDR0 to ADDR1 range as configured by ADDRCFG. This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Have not received ADDR1 or ADDR0/ADDR1 range matching address. 1 Have received ADDR1 or ADDR0/ADDR1 range matching address.
12 AM0F	Address Match 0 Flag Indicates that the received address has matched the ADDR0 field as configured by ADDRCFG. This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Have not received ADDR0 matching address. 1 Have received ADDR0 matching address.
11 FEF	FIFO Error Flag FIFO error flag can only set when clock stretching is disabled. 0 FIFO underflow or overflow not detected. 1 FIFO underflow or overflow detected.
10 BEF	Bit Error Flag This flag will set if the LPI2C slave transmits a logic one and detects a logic zero on the I2C bus. The slave will ignore the rest of the transfer until the next (repeated) START condition. 0 Slave has not detected a bit error. 1 Slave has detected a bit error.

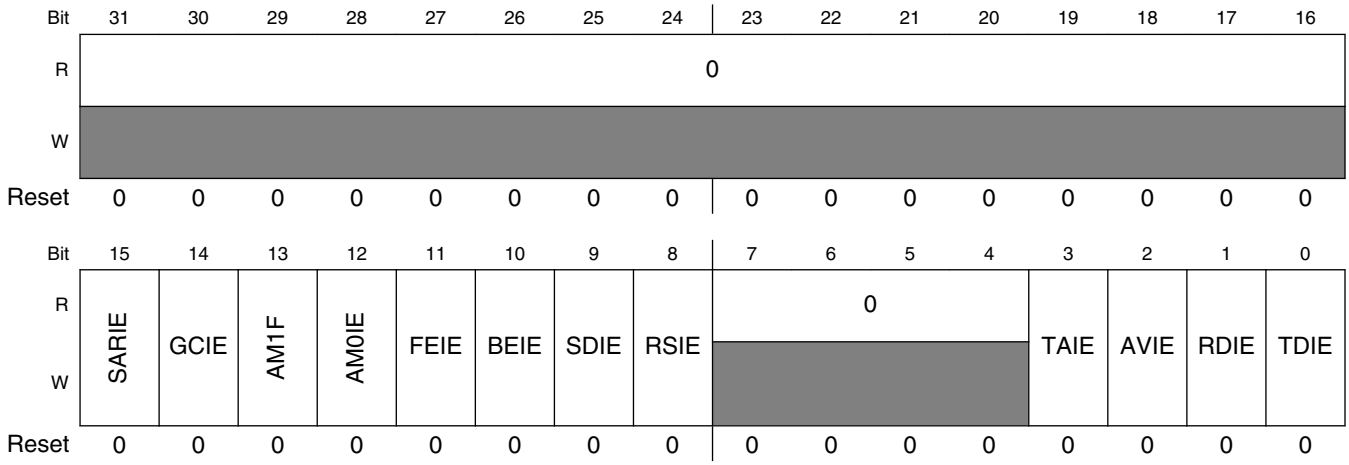
Table continues on the next page...

LPI2Cx_SSR field descriptions (continued)

Field	Description
9 SDF	STOP Detect Flag This flag will set when the LPI2C slave detects a STOP condition, provided the LPI2C slave matched the last address byte. 0 Slave has not detected a STOP condition. 1 Slave has detected a STOP condition.
8 RSF	Repeated Start Flag This flag will set when the LPI2C slave detects a repeated START condition, provided the LPI2C slave matched the last address byte. It does not set when the slave first detects a START condition. 0 Slave has not detected a Repeated START condition. 1 Slave has detected a Repeated START condition.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TAF	Transmit ACK Flag This flag is cleared by writing the transmit ACK register. 0 Transmit ACK/NACK is not required. 1 Transmit ACK/NACK is required.
2 AVF	Address Valid Flag This flag is cleared by reading the address status register. When RXCFG is set, this flag is also cleared by reading the receive data register. 0 Address Status Register is not valid. 1 Address Status Register is valid.
1 RDF	Receive Data Flag This flag is cleared by reading the receive data register. When RXCFG is set, this flag is not cleared when reading the receive data register and AVF is set. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag This flag is cleared by writing the transmit data register. When TXCFG is clear, it is also cleared if a NACK or Repeated START or STOP condition is detected. 0 Transmit data not requested. 1 Transmit data is requested.

47.3.20 Slave Interrupt Enable Register (LPI2Cx_SIER)

Address: Base address + 118h offset



LPI2Cx_SIER field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SARIE	SMBus Alert Response Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
14 GCIE	General Call Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
13 AM1F	Address Match 1 Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 AM0IE	Address Match 0 Interrupt Enable 0 Interrupt enabled. 1 Interrupt disabled.
11 FEIE	FIFO Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 BEIE	Bit Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 SDIE	STOP Detect Interrupt Enable

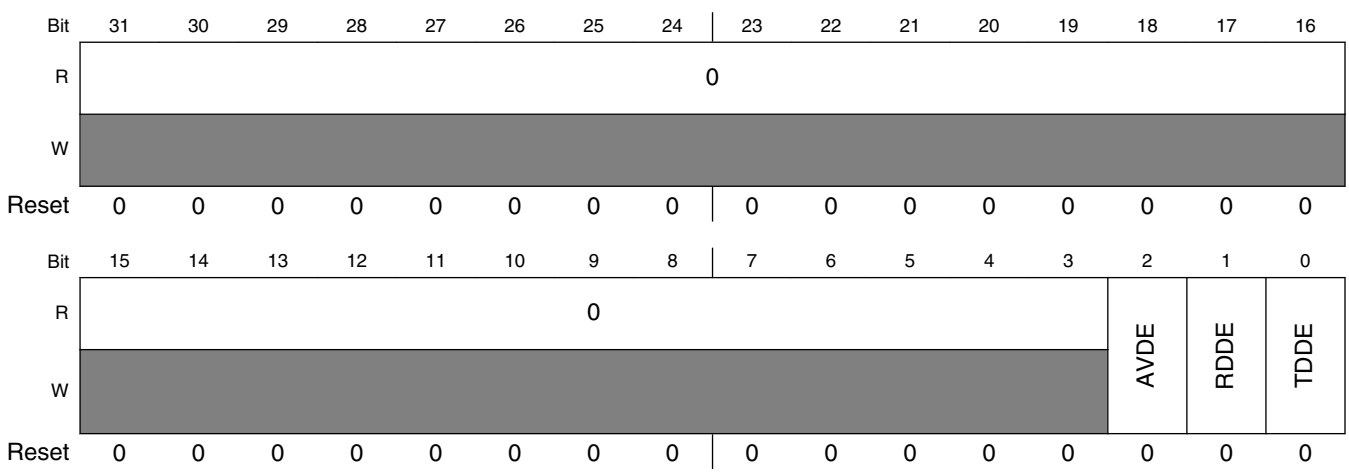
Table continues on the next page...

LPI2Cx_SIER field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled.
8 RSIE	Repeated Start Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TAIE	Transmit ACK Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
2 AVIE	Address Valid Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
0 TDIE	Transmit Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled

47.3.21 Slave DMA Enable Register (LPI2Cx_SDER)

Address: Base address + 11Ch offset



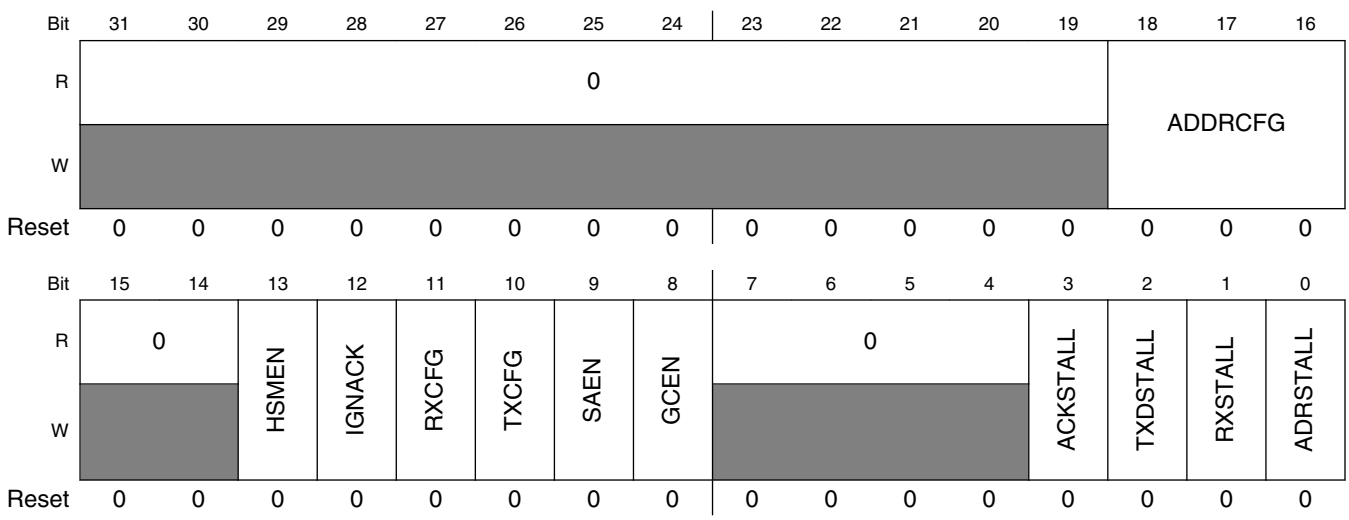
LPI2Cx_SDER field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 AVDE	Address Valid DMA Enable The Address Valid DMA request is shared with the Receive Data DMA request. If both are enabled, then set RXCFG to allow the DMA to read the address from the Receive Data Register. 0 DMA request disabled. 1 DMA request enabled.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

47.3.22 Slave Configuration Register 1 (LPI2Cx_SCFGR1)

The SCFGR1 should only be written when the I2C Slave is disabled.

Address: Base address + 124h offset



LPI2Cx_SCFGR1 field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 ADDRCFG	Address Configuration

Table continues on the next page...

LPI2Cx_SCFGR1 field descriptions (continued)

Field	Description
	Configures the condition that will cause an address to match. 000 Address match 0 (7-bit). 001 Address match 0 (10-bit). 010 Address match 0 (7-bit) or Address match 1 (7-bit). 011 Address match 0 (10-bit) or Address match 1 (10-bit). 100 Address match 0 (7-bit) or Address match 1 (10-bit). 101 Address match 0 (10-bit) or Address match 1 (7-bit). 110 From Address match 0 (7-bit) to Address match 1 (7-bit). 111 From Address match 0 (10-bit) to Address match 1 (10-bit).
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 HSMEN	High Speed Mode Enable Enables detection of the High-speed Mode master code of slave address 0000_1XX, but does not cause an address match on this code. When set and any Hs-mode master code is detected, the FILTEN and ACKSTALL bits are ignored until the next STOP condition is detected. 0 Disables detection of Hs-mode master code. 1 Enables detection of Hs-mode master code.
12 IGNACK	Ignore NACK When set, the LPI2C slave will continue transfers after a NACK is detected. This bit is required to be set in Ultra-Fast Mode. 0 Slave will end transfer when NACK detected. 1 Slave will not end transfer when NACK detected.
11 RXCFG	Receive Data Configuration 0 Reading the receive data register will return receive data and clear the receive data flag. 1 Reading the receive data register when the address valid flag is set will return the address status register and clear the address valid flag. Reading the receive data register when the address valid flag is clear will return receive data and clear the receive data flag.
10 TXCFG	Transmit Flag Configuration The transmit data flag will always assert before a NACK is detected at the end of a slave-transmit transfer. This can cause an extra word to be written to the transmit data FIFO. When TXCFG=0, the transmit data register is automatically emptied when a slave-transmit transfer is detected. This cause the transmit data flag to assert whenever a slave-transmit transfer is detected and negate at the end of the slave-transmit transfer. When TXCFG=1, the transmit data flag will assert whenver the transit data register is empty and negate when the transmit data register is full. This allows the transmit data register to be filled before a slave-transmit transfer is detected, but can cause the transmit data register to be written before a NACK is detected on the last byte of a slave transmit transfer. 0 Transmit Data Flag will only assert during a slave-transmit transfer when the transmit data register is empty. 1 Transmit Data Flag will assert whenever the transmit data register is empty.
9 SAEN	SMBus Alert Enable

Table continues on the next page...

LPI2Cx_SCFGR1 field descriptions (continued)

Field	Description
	0 Disables match on SMBus Alert. 1 Enables match on SMBus Alert.
8 GCEN	General Call Enable 0 General Call address is disabled. 1 General call address is enabled.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during slave-transmit address byte(s) and slave-receiver address and data byte(s) to allow software to write the Transmit ACK Register before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the 9th bit and is therefore not compatible with high speed mode. When ACKSTALL is enabled, there is no need to set either RXSTALL or ADRSTALL 0 Clock stretching disabled. 1 Clock stretching enabled.
2 TXDSTALL	TX Data SCL Stall Enables SCL clock stretching when the transmit data flag is set during a slave-transmit transfer. Clock stretching occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when receive data flag is set during a slave-receive transfer. Clock stretching occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when the address valid flag is asserted. Clock stretching only occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.

47.3.23 Slave Configuration Register 2 (LPI2Cx_SCFGR2)

The SCFGR2 should only be written when the I2C Slave is disabled.

Address: Base address + 128h offset

Bit	31	30	29	28	27	26	25	24	23	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				FILTSDA				0				FILTSCL				0		DATAVD						0				CLKHOLD			
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

LPI2Cx_SCFGR2 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 FILTSDA	Glitch Filter SDA Configures the I2C slave digital glitch filters for SDA input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSDA+3 cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration, and is disabled in high speed mode.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 FILTSCL	Glitch Filter SCL Configures the I2C slave digital glitch filters for SCL input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSCL+3 cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration, and is disabled in high speed mode.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C slave equal to FILTSCL+DATAVD+3 cycles. This data valid delay must be configured to less than the minimum SCL low period. The I2C slave data valid delay time is not affected by the PRESCALE configuration, and is disabled in high speed mode.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C slave, when clock stretching is enabled. The minimum hold time is equal to CLKHOLD+3 cycles. The I2C slave clock hold time is not affected by the PRESCALE configuration, and is disabled in high speed mode.

47.3.24 Slave Address Match Register (LPI2Cx_SAMR)

Address: Base address + 140h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					ADDR1										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					ADDR0										0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2Cx_SAMR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26–17 ADDR1	Address 1 Value Compared against the received address to detect the Slave Address. In 10-bit mode, the first address byte is compared to { 11110, ADDR1[10:9] } and the second address byte is compared to ADDR1[8:1]. In 7-bit mode, the address is compared to ADDR1[7:1].
16–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–1 ADDR0	Address 0 Value Compared against the received address to detect the Slave Address. In 10-bit mode, the first address byte is compared to { 11110, ADDR0[10:9] } and the second address byte is compared to ADDR0[8:1]. In 7-bit mode, the address is compared to ADDR0[7:1].
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

47.3.25 Slave Address Status Register (LPI2Cx_SASR)

Address: Base address + 150h 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	ANV	0			RADDR										
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2Cx_SASR field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 ANV	Address Not Valid 0 RADDR is valid. 1 RADDR is not valid.
13–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RADDR	Received Address RADDR updates whenever the AMF is set and the AMF is cleared by reading this register. In 7-bit mode, the address byte is store in RADDR[7:0]. In 10-bit mode, the first address byte is { 11110, RADDR[10:9],

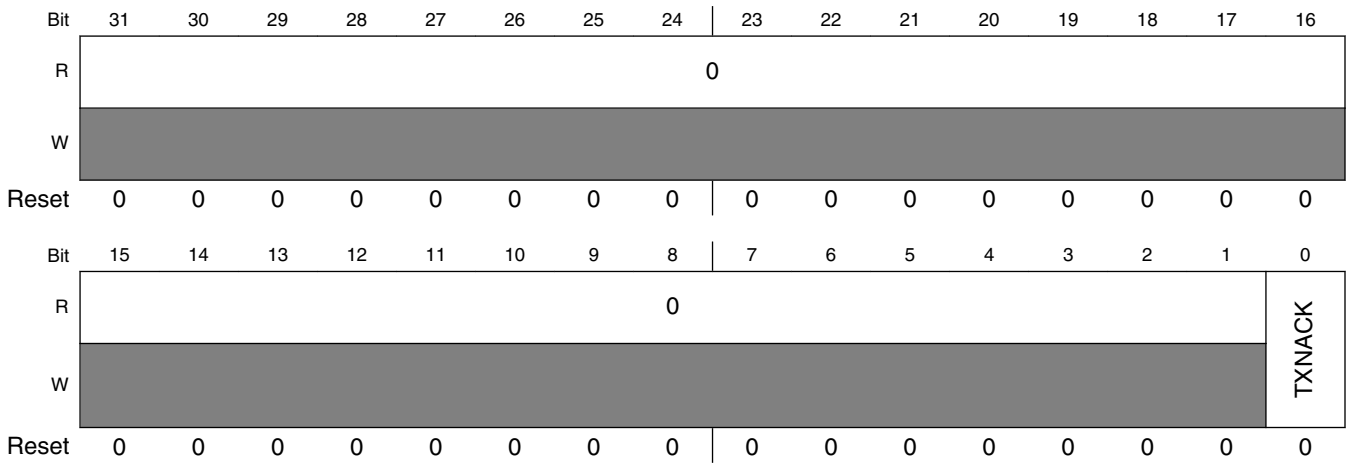
Table continues on the next page...

LPI2Cx_SASR field descriptions (continued)

Field	Description
	RADDR[0] } and the second address byte is RADDR[8:1]. The R/W bit is therefore always stored in RADDR[0].

47.3.26 Slave Transmit ACK Register (LPI2Cx_STAR)

Address: Base address + 154h offset

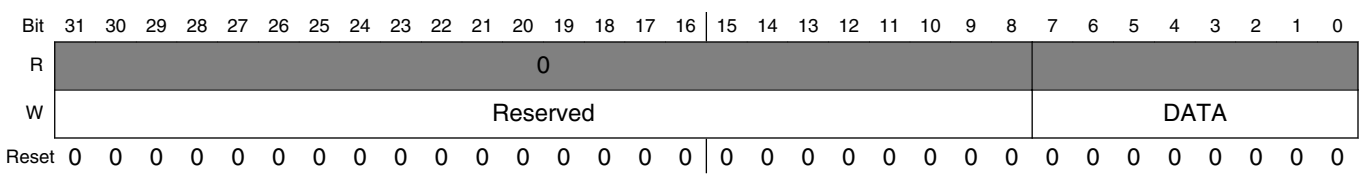


LPI2Cx_STAR field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TXNACK	Transmit NACK When NACKSTALL is set, must be written once for each matching address byte and each received word. Can also be written when LPI2C Slave is disabled or idle to configure the default ACK/NACK. 0 Transmit ACK for received word. 1 Transmit NACK for received word.

47.3.27 Slave Transmit Data Register (LPI2Cx_STDR)

Address: Base address + 160h offset

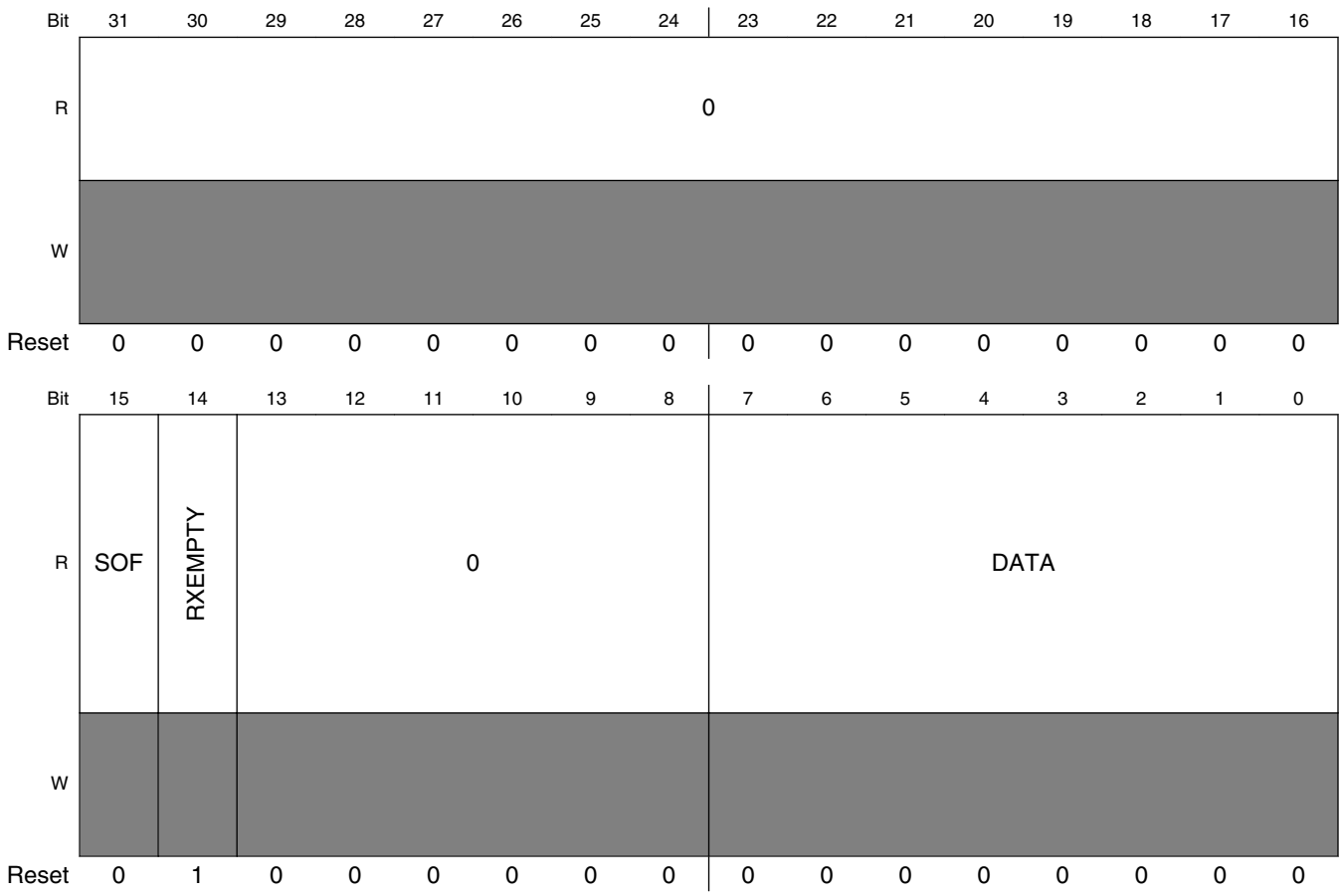


LPI2Cx_STDR field descriptions

Field	Description
31–8 Reserved	This field is reserved.
DATA	Transmit Data Writing this register will store I2C slave transmit data in the transmit register.

47.3.28 Slave Receive Data Register (LPI2Cx_SRDR)

Address: Base address + 170h offset



LPI2Cx_SRDR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SOF	Start Of Frame 0 Indicates this is not the first data word since a (repeated) START or STOP condition. 1 Indicates this is the first data word since a (repeated) START or STOP condition.

Table continues on the next page...

LPI2Cx_SRDR field descriptions (continued)

Field	Description
14 RXEMPTY	RX Empty 0 The Receive Data Register is not empty. 1 The Receive Data Register is empty.
13–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA	Receive Data Reading this register returns the data received by the I2C slave.

47.4 Functional description

47.4.1 Clocking and Resets

47.4.1.1 Functional clock

The LPI2C functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support I2C bus transfers by the LPI2C master. It is also used by the LPI2C slave to support digital filter and data hold time configurations. The LPI2C master divides the functional clock by a prescaler and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.

47.4.1.2 External clock

The LPI2C slave logic is clocked directly from the external pins LPI2C_SCL and LPI2C_SDA (or LPI2C_SCLS and LPI2C_SDAS if master and slave are implemented on separate pins). This allows the LPI2C slave to remain operational, even when the LPI2C functional clock is disabled. Note that the LPI2C slave digital filter must be disabled if the LPI2C functional clock is disabled and this can effect compliance with some of the timing parameters of the I2C specification, such as the data hold time.

47.4.1.3 Bus clock

The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C master and slave registers.

47.4.1.4 Chip reset

The logic and registers for the LPI2C master and slave are reset to their default state on a chip reset.

47.4.1.5 Software reset

The LPI2C master implements a software reset bit in its Control Register. The MCR[RST] will reset all master logic and registers to their default state, except for the MCR itself.

The LPI2C slave implements a software reset bit in its Control Register. The SCR[RST] will reset all slave logic and registers to their default state, except for the SCR itself.

47.4.1.6 FIFO reset

The LPI2C master implements write-only control bits that resets the transmit FIFO (MCR[RTF] and receive FIFO (MCR[RRF]). A FIFO is empty after being reset.

The LPI2C slave implements write-only control bits that resets the transmit data register (SCR[RTF] and receive data register (SCR[RRF]). A data register is empty after being reset.

47.4.2 Master Mode

The LPI2C master logic operates independently from the slave logic to perform all master mode transfers on the I2C bus.

47.4.2.1 Transmit and Command FIFO

The transmit FIFO stores command data to initiate the various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- START or Repeated START condition with address byte and expecting ACK or NACK.
- Transmit data (this is the default for zero extended byte writes to the transmit FIFO).
- Receive 1-256 bytes of data (can also be configured to discard receive data and not store in receive FIFO).
- STOP condition (can also be configured to send STOP condition when transmit FIFO is empty).

Multiple transmit and receive commands can be inserted between the START condition and STOP conditon, transmit and receive commands must not be interleaved in order to comply with the I2C specification. The receive data command and the receive data and discard command can be interleaved to ensure only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C master supports 10-bit addressing through a (repeated) START condition, followed by a transmit byte with the second address byte, followed by any number of data bytes with the master-transmit data.

A START or Repeated START condition that is expecting a NACK (for example, hs-mode master code) must be followed by a STOP or (repeated) START condition.

47.4.2.2 Master Operation

Whenever the LPI2C is enabled, it monitors the I2C bus to detect when the I2C bus is idle (MSR[BBF]). The I2C bus is no longer considered idle if either SCL or SDA are low and becomes idle if a STOP condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). Once the I2C bus is idle, the transmit FIFO is not empty, and the host request is either asserted or disabled, then the LPI2C master will initiate a transfer on the I2C bus. This involves the following steps:

- Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler.
- Transmit a START condition and address byte using the timing configuration in MCCR0, if a high speed mode transfer is configured then timing configuration from MCCR1 is used instead.
- Perform master-transmit or master-receive transfers, as configured by the transmit FIFO.
- Transmit a Repeated START or STOP condition as configured by the transmit FIFO and/or MCFGR1[AUTOSTOP]. A repeated START can change which timing configuration register is used.

When the LPI2C master is disabled (either due to MCR[MEN] being clear or automatically due to mode entry), the LPI2C will continue to empty the transmit FIFO until a STOP condition is transmitted. However, it will no longer stall the I2C bus waiting for the transmit or receive FIFO and once the transmit FIFO is empty it will generate a STOP condition automatically.

The LPI2C master can stall the I2C bus under certain conditions, this will result in SCL pulled low continuously on the first bit of a byte until the condition is removed:

- LPI2C master is enabled and busy, transmit FIFO is empty, and MCFGR1[AUTOSTOP] is clear.
- LPI2C master is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and receive FIFO is full.

47.4.2.3 Receive FIFO and Data Match

The receive FIFO is used to store receive data during master-receiver transfers. Receive data can also be configured to discard receive data instead of storing in the receive FIFO, this is configured by the command word in the transmit FIFO.

Receive data supports a receive data match function that can match received data against one of two bytes or against a masked data byte. The data match function can also be configured to compare only the first one or two received data words since the last (repeated) START condition. Receive data that is already discarded due to the command word cannot cause the data match to set and will delay the match on first received data word until after the discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

47.4.2.4 Timing Parameters

The following timing parameters can be configured by the LPI2C master. Parameters are configured separately for high speed mode (MCCR1) and other modes (MCCR0). This allows the high speed mode master code to be sent using the regular timing parameters and then switch to the high speed mode timing (following a repeated START) until the next STOP condition.

The LPI2C master timing parameters in LPI2C functional clock cycles are configured as follows. They must be configured to meet the I2C timing specification for the required mode.

- Bus idle time is always (MCCR0[CLKLO] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SDA rising edge.
- START or repeated START hold time is equal to (MCCR0/1[SETHOLD] + 1) multiplied by the prescaler.
- START, or repeated START, or STOP setup time is equal to (MCCR0/1[SETHOLD] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SCL rising edge.
- SCL low time (before clock stretching) is equal to (MCCR0/1[CLKLO] + 1) multiplied by the prescaler.
- SCL high time is equal to (MCCR0/1[CLKHI] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SCL rising edge.
- SDA output delay is equal to (MCCR0/1[DATAVD] + 1) multiplied by the prescaler.

The time taken to detect an external rising edge depends on a number of factors including the bus loading and external pull-up resistor sizing. The minimum delay equals two plus the pin input digital filter setting (which are configured separately for SCL and SDA), divided by the prescaler (since the pin input digital filters are not affected by the prescaler setting).

The following timing restrictions must be enforced to avoid unexpected START or STOP conditions on the I2C bus or unexpected START or STOP conditions detected by the LPI2C master. They can be summarized as SDA cannot change when SCL is high outside of a transmitted (repeated) START or STOP condition.

Table 47-3. Timing Parameters

Timing Parameter	Minimum	Maximum	Comment
CLKLO	0x03	-	CLKLO must also be greater than delay through the SCL filter.
CLKHI	0x01	-	
SETHOLD	0x02	-	
DATAVD	0x01	CLKLO - [(FILTSDA+2) / (2 ^ PRESCALER)]	DATAVD must be less than CLKLO minus delay through the SDA filter.
FILTSCL	0x00	[CLKLO × (2 ^ PRESCALER)] - 3	
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALER)] - 3	Does not apply if compensating for board level skew between SCL and SDA.
BUSIDLE	(CLKLO+SETHOLD+2) × 2	-	Must also be greater than CLKHI+1.

The timing parameters must be configured to meet the requirements of the I2C specification, this will depend on the mode being supported, the frequency of the LPI2C functional clock. Some example configurations are provided below.

Table 47-4. LPI2C Example Timing Configurations

I2C Mode	Clock Frequency	Baud Rate	PRESCALER	FILTSCL/ FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast	8 MHz	400 kbps	0x0	0x0/0x0	0x04	0x0B	0x05	0x02
Fast+	8 MHz	1 Mbps	0x0	0x0/0x0	0x02	0x03	0x01	0x01
Fast	48 MHz	400 kbps	0x2	0x1/0x1	0x07	0x11	0x0B	0x03
Fast+	48 MHz	1 Mbps	0x2	0x1/0x1	0x03	0x06	0x04	0x04
Fast+	48 MHz	1 Mbps	0x0	0x1/0x1	0x1D	0x18	0x13	0x0F
HS-mode	48 MHz	3.2 Mbps	0x0	0x0/0x0	0x07	0x08	0x03	0x01
Fast	60 MHz	400 kbps	0x1	0x2/0x2	0x11	0x28	0x21	0x08
Fast+	60 MHz	1 Mbps	0x1	0x2/0x2	0x07	0x0F	0x0B	0x01
HS-mode	60 MHz	3.33 Mbps	0x1	0x0/0x0	0x04	0x03	0x04	0x01
Ultrafast	60 MHz	5 Mbps	0x0	0x0/0x0	0x02	0x05	0x03	0x01

The formula to calculate number of cycles per bit is as follows:

*Baud rate divide = ((CLKLO+CLKHI+2)*2^PRESCALER) + ROUNDDOWN((2+FILTSCL)/2^PRESCALER)*

This assumes SCL will pull high within 1 cycle of the LPI2C functional clock, this will depend on the pullup resistor and loading on the SCL pin.

47.4.2.5 Error Conditions

The LPI2C master will monitor for errors while it is active, the following conditions will generate an error flag and block a new START condition from being sent until the flag is cleared by software:

- START or STOP condition detected and not generated by LPI2C master (sets MSR[ALF]).
- Transmitting data on SDA and different value being received (sets MSR[ALF]).
- NACK detected when transmitting data, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- NACK detected and expecting ACK for address byte, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- ACK detected and expecting NACK for address byte, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).

- Transmit FIFO requesting to transmit or receive data without a START condition (sets MSR[FEF]).
- SCL (or SDA if MCFGR1[TIMECFG] is set) is low for (MCFGR2[TIMELOW] * 256) prescaler cycles without a pin transition (sets MSR[PLTF]).

Software must respond to the MSR[PTLF] flag to terminate the existing command either cleanly (by clearing MCR[MEN]) or abruptly (by setting MCR[SWRST]).

The MCFGR2[BUSIDLE] field can be used to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE+1) prescaler cycles. The I2C bus is normally considered idle when the LPI2C master is first enabled, but when BUSIDLE is configured greater than zero then SCL and/or SDA must be high for (BUSIDLE+1) prescaler cycles before the I2C bus is first considered idle.

47.4.2.6 Pin Configuration

The LPI2C master defaults to open-drain configuration of the LPI2C_SDA and LPI2C_SCL pins. Support for true open drain is device specific and requires the pins where LPI2C pins are muxed to support true open drain. Support for high speed mode is also device specific and requires the LPI2C_SCL pin to support the current source pull-up required in the I2C specification.

The LPI2C master also supports the output only push-pull function required for I2C ultra-fast mode using the LPI2C_SDA and LPI2C_SCL pins. Support for ultra-fast mode also requires the IGNACK bit to be set.

A push-pull 2 wire configuration is also available to the LPI2C master that may support a partial high speed mode provided the LPI2C is the only master and all I2C pins on the bus are at the same voltage. This will configure the LPI2C_SCL pin as push-pull for every clock except the 9th clock pulse to allow high speed mode compatible slaves to perform clock stretching. In this mode, the LPI2C_SDA pin is tristated for master-receive data bits and master-transmit ACK/NACK bits.

The push-pull 4 wire configuration separates the SCL input and output and the SDA input and output onto separate pins, with SCL/SDA used as the input pins and SCLS/SDAS used as the output pins with configurable polarity. This simplifies the external connections when connecting the I2C bus to external level shifters. The LPI2C master logic and LPI2C slave logic are not able to connect to separate I2C buses when using this configuration.

47.4.3 Slave Mode

The LPI2C slave logic operates independently from the master logic to perform all slave mode transfers on the I2C bus.

47.4.3.1 Address Match

The LPI2C slave can be configured to match one of two addresses using either 7-bit or 10-bit addressing modes for each address, or to match a range of addresses in either 7-bit or 10-bit addressing modes. Separately, it can be configured to match the General Call Address or the SMBus Alert Address and generate appropriate flags. The LPI2C slave can also be configured to detect the high speed mode master code and to disable the digital filters and output valid delay time until the next STOP condition is detected.

Once a valid address is matched, the LPI2C slave will automatically perform slave-transmit or slave-receive transfers until a NACK is detected (unless IGNACK is set), a bit error is detected (the LPI2C slave is driving SDA, but a different value is sampled), or a (repeated) START or STOP condition is detected.

47.4.3.2 Transmit and Receive

The transmit and receive data registers are double buffered and only update during a slave-transmit and slave-receive transfer respectively. The slave address that was received can be configured to be read from either the receive data register (for example, when using DMA to transfer data) or from the address status register. The transmit data register can be configured to only request data once a slave-transmit transfer is detected or to request new data whenever the transmit data register is empty.

The transmit data register should only be written when the transmit data flag is set. The receive data register should only be read when the received data flag is set (or the address valid flag is set and RXCFG=1). The address status register should only be read when the address valid flag is set.

47.4.3.3 Clock Stretching

The LPI2C slave supports many configurable options for when clock stretching is performed. The following conditions can be configured to perform clock stretching.

- During 9th clock pulse of address byte and address valid flag is set.
- During 9th clock pulse of slave-transmit transfer and transmit data flag is set.
- During 9th clock pulse of slave-receive transfer and receive data flag is set.

- During 8th clock pulse of address byte or slave-receive transfer and transmit ACK flag is set. This is disabled in high speed mode.
- Clock stretching can also be extended for CLKHOLD cycles to allow additional setup time to sample the SDA pin externally. This is disabled in high speed mode.

Unless extended by the CLKHOLD configuration, clock stretching will extend for one peripheral bus clock cycle after SDA updates when clock stretching is enabled.

47.4.3.4 Timing Parameters

The LPI2C slave can configure the following timing parameters, these parameters are disabled when SCR[FILTEN] is clear, when SCR[FILTDZ] is set in Doze mode, and when LPI2C slave detects high speed mode. When disabled, the LPI2C slave is clocked directly from the I2C bus and may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

- SDA data valid time from SCL negation to SDA update.
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally.
- SCL glitch filter time.
- SDA glitch filter time.

The LPI2C slave imposes the following restrictions on the timing parameters.

- FILTSDA must be configured to greater than or equal to FILTSCL (unless compensating for board level skew between SDA and SCL).
- DATAVD must be configured less than the minimum SCL low period.

47.4.3.5 Error Conditions

The LPI2C slave can detect the following error conditions.

- Bit error flag will set when the LPI2C slave is driving SDA, but samples a different value than what is expected.
- FIFO error flag will set due to a transmit data underrun or a receive data overrun. Clock stretching can be enabled to eliminate the possibility of underrun and overrun occurring.
- FIFO error flag will also set due to an address overrun when RXCFG is set, otherwise an address overrun is not flagged. Clock stretching can be enabled to eliminate the possibility of overrun occurring.

The LPI2C slave does not implement a timeout due to SCL and/or SDA being stuck low. If this detection is required, the LPI2C master logic should be used and software can reset the LPI2C slave when this condition is detected.

47.4.4 Interrupts and DMA Requests

The LPI2C master and slave interrupts may be combined depending on the device.

The LPI2C master and slave transmit DMA requests may be combined depending on the device.

The LPI2C master and slave receive DMA requests may be combined depending on the device.

47.4.4.1 Master mode

The following table illustrates the master mode sources that can generate the LPI2C master interrupt and LPI2C master transmit/receive DMA requests.

Table 47-5. Master Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit FIFO, as configured by TXWATER.	Y	TX	Y
RDF	Data can be read from the receive FIFO, as configured by RXWATER.	Y	RX	Y
EPF	Master has transmitted Repeated START or STOP condition.	Y	N	Y
SDF	Master has transmitted STOP condition.	Y	N	Y
NDF	Master detected NACK during address byte when expecting ACK, master detected ACK during address byte and expecting NACK, or master detected NACK during master-transmitter data byte.	Y	N	Y
ALF	Master lost arbitration due to START/STOP condition detected at	Y	N	Y

Table continues on the next page...

Table 47-5. Master Interrupts and DMA Requests (continued)

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
	wrong time, or Master was transmitting data but received different data than what was transmitted.			
FEF	Master expecting START condition in command FIFO and next entry in FIFO is not START condition.	Y	N	Y
PLTF	Pin low timeout is enabled and SCL (or SDA if configured) is low for longer than the configured timeout.	Y	N	Y
DMF	Received data matches the configured data match, and receive data not discarded due to command FIFO entry.	Y	N	Y
MBF	LPI2C master is busy transmitting/receiving data.	N	N	N
BBF	LPI2C master is enabled and activity detected on I2C bus, but STOP condition has not been detected and bus idle timeout (if enabled) has not occurred.	N	N	N

47.4.4.2 Slave mode

The following table illustrates the slave mode sources that can generate the LPI2C slave interrupt and the LPI2C slave transmit/receive DMA requests.

Table 47-6. Slave Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit data register.	Y	TX	Y
RDF	Data can be read from the receive data register.	Y	RX	Y

Table continues on the next page...

Table 47-6. Slave Interrupts and DMA Requests (continued)

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
AVF	Address can be read from the address status register.	Y	RX	Y
TAF	ACK/NACK can be written to the transmit ACK register.	Y	N	Y
RSF	Slave has detected an address match followed by a Repeated START condition.	Y	N	Y
SDF	Slave has detected an address match followed by a STOP condition.	Y	N	Y
BEF	Slave was transmitting data, but received different data than what was transmitted.	Y	N	Y
FEF	Transmit data underrun, receive data overrun or address status overrun (when RXCFG=1). This flag can only set when clock stretching is disabled.	Y	N	Y
AM0F	Slave detected address match with ADDR0 field.	Y	N	N
AM1F	Slave detected address match with ADDR1 field or address range.	Y	N	N
GCF	Slave detected address match with general call address.	Y	N	N
SARF	Slave detected address match with SMBus alert address.	Y	N	N
SBF	LPI2C slave is busy receiving address byte or transmitting/receiving data.	N	N	N
BBF	LPI2C slave is enabled and START condition detected on I2C bus, but STOP condition has not been detected.	N	N	N

47.4.5 Peripheral Triggers

The connection of the LPI2C peripheral triggers with other peripherals are device specific.

47.4.5.1 Master Output Trigger

The LPI2C master generates an output trigger that can be connected to other peripherals on the device. The master output trigger asserts on both a Repeated START or STOP condition and remains asserted for one cycle of the LPI2C functional clock divided by the prescaler.

47.4.5.2 Slave Output Trigger

The LPI2C slave generates an output trigger that can be connected to other peripherals on the device. The slave output trigger asserts on both a Repeated START or STOP condition that occurs following a slave address match. It remains asserted until the next slave SCL pin negation.

47.4.5.3 Input Trigger

The LPI2C input trigger can be selected in place of the LPI2C_HREQ pin to control the start of a LPI2C master bus transfer. The input trigger must assert for longer than one LPI2C functional clock cycle to be detected.

47.5 Usage Guide

For master:

- Configure functional clock source
- Reset LPI2C module by LPI2C0_MCR[RST]
- Configure baudrate
- Set Tx/Rx FIFO watermark by LPI2C0_MFCR
- Enable Master mode by set LPI2C0_MCR[MEN]

For slave:

- Configure functional clock source
- Set the slave address into LPI2C0_SAMR
- Configure the TDF only be set in the Slave-Transmit condition by LPI2C0_SCFGR1[TXCFG]

- Enable the TX Data SCL Stall and RX SCL Stall for clock stretching on SCL
- Enable Slave mode by set LPI2C0_SCR[SEN]

Chapter 48
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

48.1 Chip-specific information for this module

48.1.1 Instantiation Information

This device has three LPUART modules. The LPUART can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

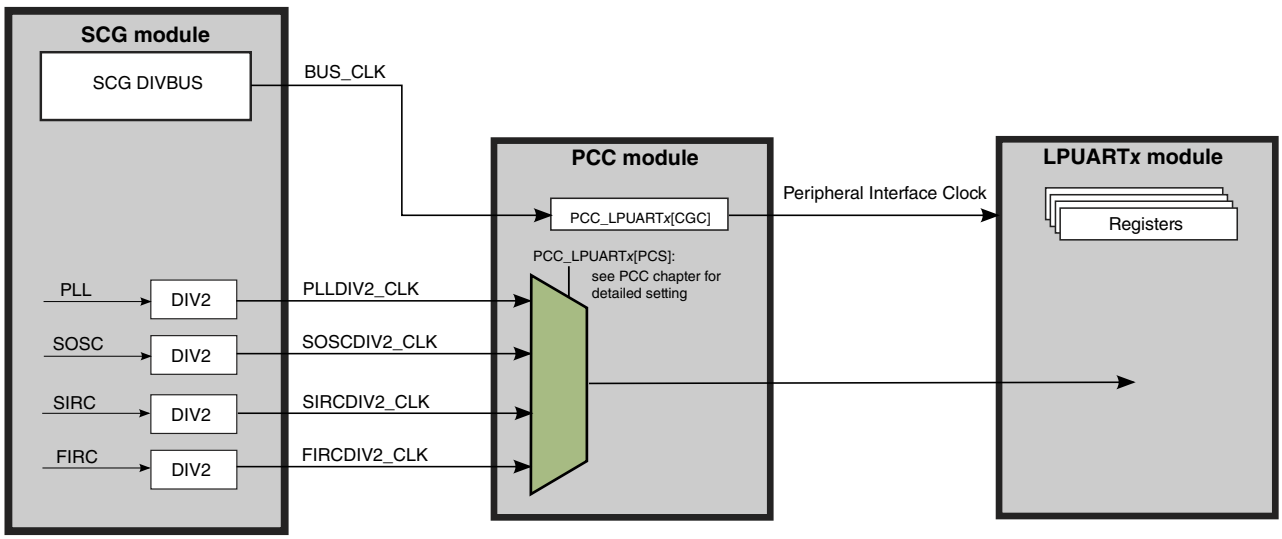
Table 48-1. LPUART Configuration

	TX FIFO (word/10bit)	RX FIFO (word/10bit)	Single-wire mode
LPUART0	4	4	Yes
LPUART1	4	4	Yes
LPUART2	4	4	Yes

48.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

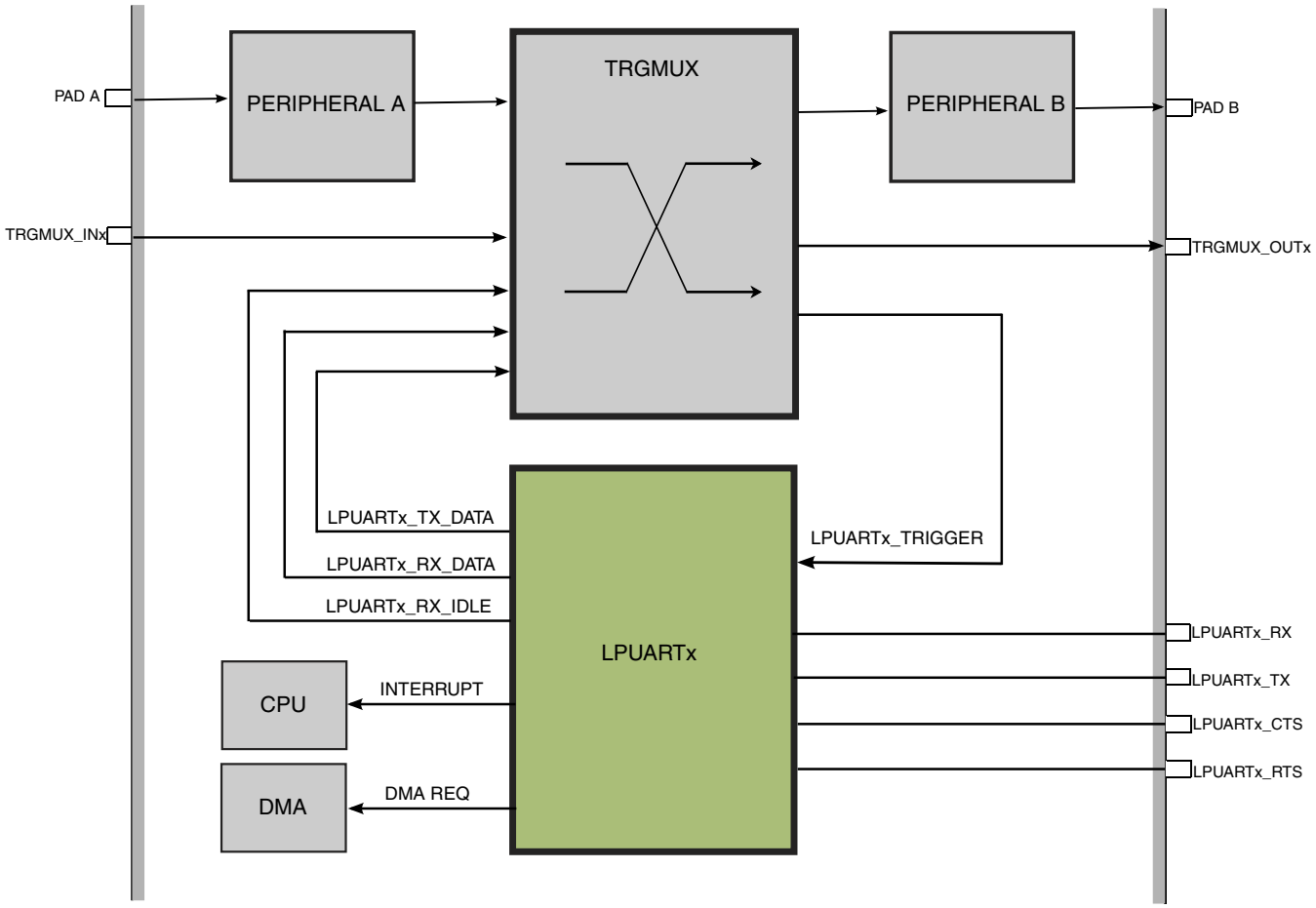
The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPUART Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, FlexIO and LPIT.



48.1.3 Inter-connectivity Information

The LPUART inter-connectivity is shown in following diagram.



48.2 Introduction

48.2.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete

    - Receive data register full
    - Receive overrun, parity error, framing error, and noise error
    - Idle receiver detect
    - Active edge on receive pin
    - Break detect supporting LIN
    - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
    - Idle line wakeup
    - Address mark wakeup
    - Receive data match
- Automatic address matching to reduce ISR overhead:
    - Address mark matching
    - Idle line address matching
    - Address match start, address match end
- Optional 13-bit break character generation / 11-bit break character detection
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64 or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- Independent FIFO structure for transmit and receive
    - Separate configurable watermark for receive and transmit requests
    - Option for receiver to assert request after a configurable number of idle characters if receive FIFO is not empty

48.2.2 Modes of operation

48.2.2.1 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.

48.2.2.2 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.

48.2.2.3 Debug mode

The LPUART remains functional in debug mode.

48.2.3 Signal Descriptions

Signal	Description	I/O
TXD	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is disabled or transmit direction is configured for receive data.	I/O
RXD	Receive data.	I
CTS_B	Clear to send.	I
RTS_B	Request to send.	O

48.2.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

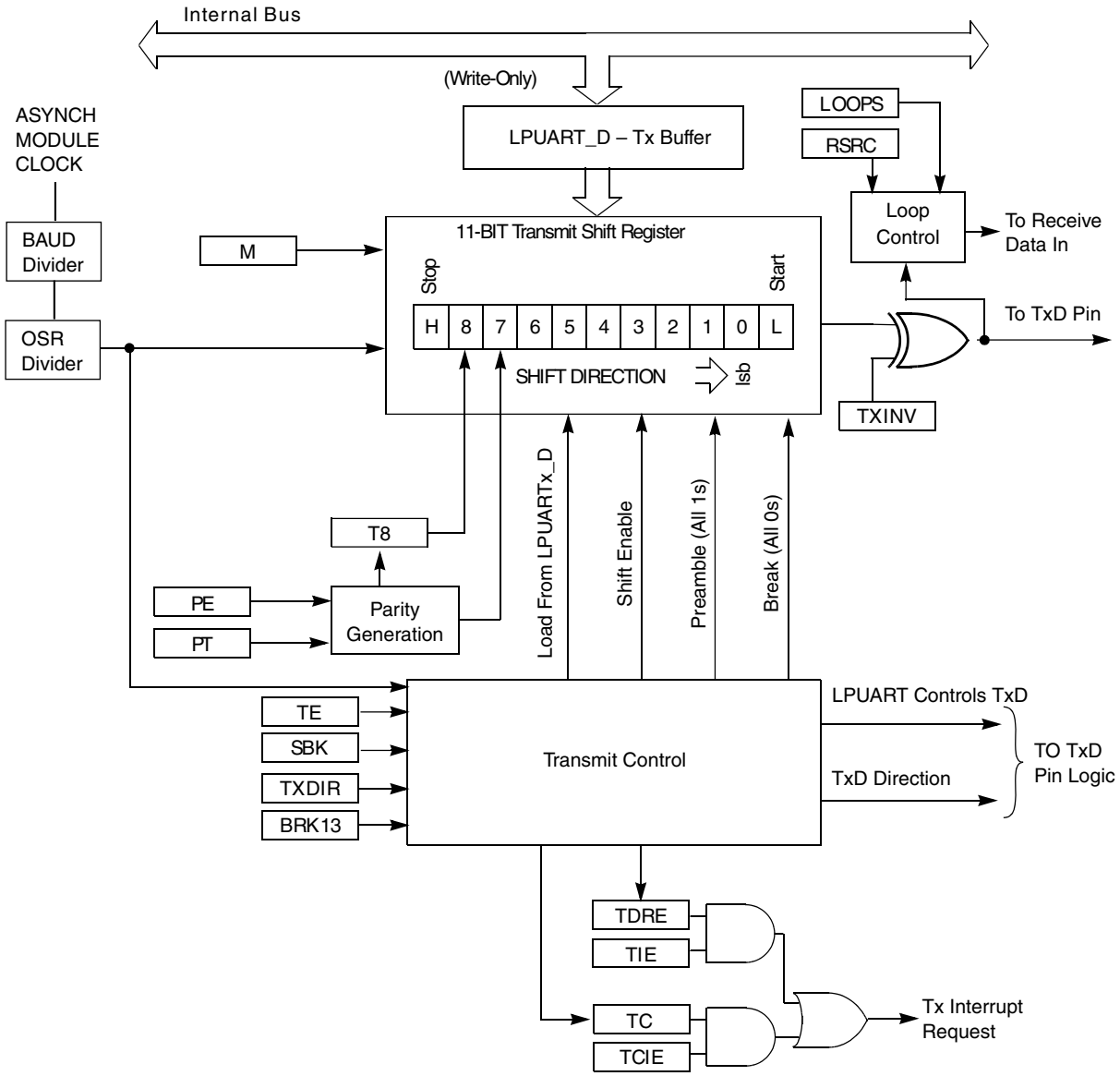


Figure 48-1. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

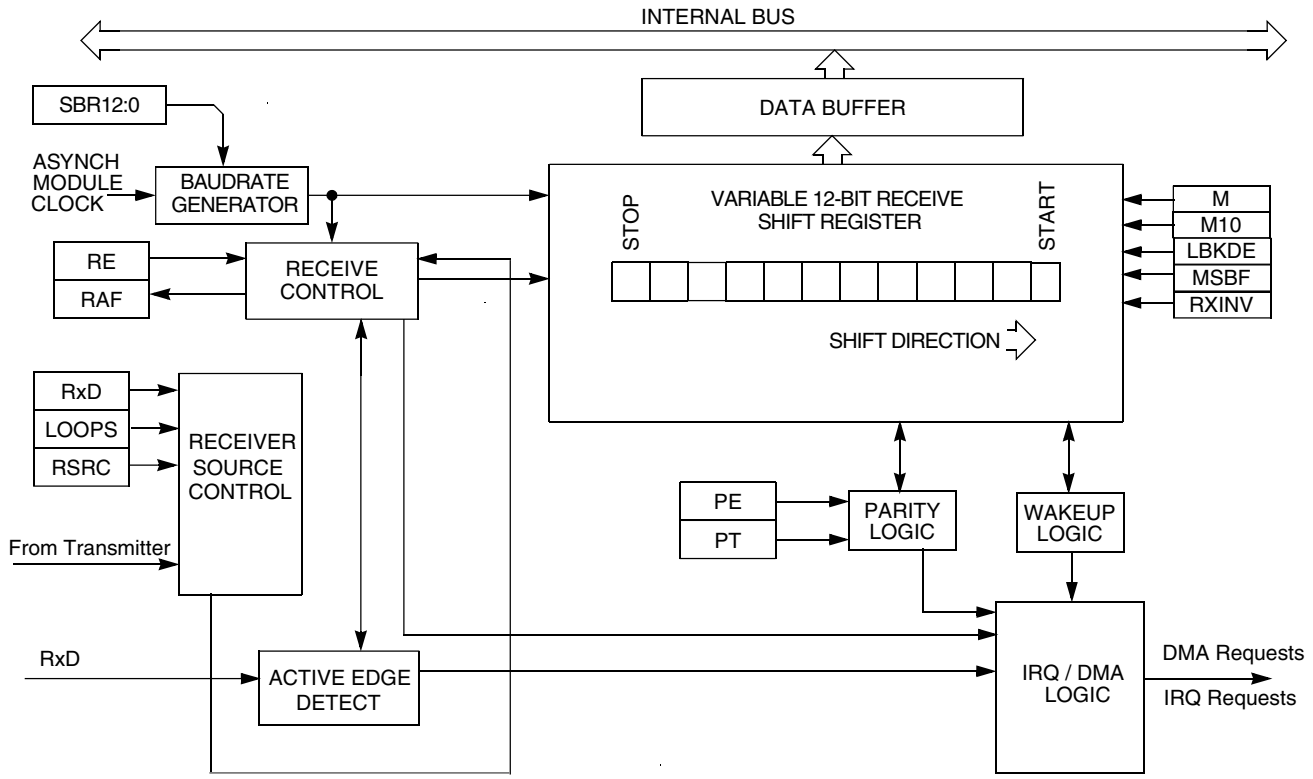


Figure 48-2. LPUART receiver block diagram

48.3 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.

48.3.1 LPUART Register Descriptions

These registers may not be applicable to all instances of LPUART. For more details on the registers supported on each module instance, please refer to "The LPUART as implemented on the chip."

48.3.1.1 LPUART Memory Map

Absolute address	Register	Width (In bits)	Access	Reset value
4006A000h	Version ID (LPUART0_VERID)	32	RO	04010003h
4006A004h	Parameter (LPUART0_PARAM)	32	RO	00000202h
4006A008h	LPUART Global (LPUART0_GLOBAL)	32	RW	00000000h
4006A00Ch	LPUART Pin Configuration (LPUART0_PINCFG)	32	RW	00000000h
4006A010h	LPUART Baud Rate (LPUART0_BAUD)	32	RW	0F000004h
4006A014h	LPUART Status (LPUART0_STAT)	32	RW	00C00000h
4006A018h	LPUART Control (LPUART0_CTRL)	32	RW	00000000h
4006A01Ch	LPUART Data (LPUART0_DATA)	32	RW	00001000h
4006A020h	LPUART Match Address (LPUART0_MATCH)	32	RW	00000000h
4006A024h	LPUART Modem IrDA (LPUART0_MODIR)	32	RW	00000000h
4006A028h	LPUART FIFO (LPUART0_FIFO)	32	RW	00C00011h
4006A02Ch	LPUART Watermark (LPUART0_WATER)	32	RW	00000000h
4006B000h	Version ID (LPUART1_VERID)	32	RO	04010003h
4006B004h	Parameter (LPUART1_PARAM)	32	RO	00000202h
4006B008h	LPUART Global (LPUART1_GLOBAL)	32	RW	00000000h
4006B00Ch	LPUART Pin Configuration (LPUART1_PINCFG)	32	RW	00000000h
4006B010h	LPUART Baud Rate (LPUART1_BAUD)	32	RW	0F000004h
4006B014h	LPUART Status (LPUART1_STAT)	32	RW	00C00000h
4006B018h	LPUART Control (LPUART1_CTRL)	32	RW	00000000h
4006B01Ch	LPUART Data (LPUART1_DATA)	32	RW	00001000h
4006B020h	LPUART Match Address (LPUART1_MATCH)	32	RW	00000000h
4006B024h	LPUART Modem IrDA (LPUART1_MODIR)	32	RW	00000000h
4006B028h	LPUART FIFO (LPUART1_FIFO)	32	RW	00C00011h
4006B02Ch	LPUART Watermark (LPUART1_WATER)	32	RW	00000000h
4006C000h	Version ID (LPUART2_VERID)	32	RO	04010003h
4006C004h	Parameter (LPUART2_PARAM)	32	RO	00000202h
4006C008h	LPUART Global (LPUART2_GLOBAL)	32	RW	00000000h
4006C00Ch	LPUART Pin Configuration (LPUART2_PINCFG)	32	RW	00000000h
4006C010h	LPUART Baud Rate (LPUART2_BAUD)	32	RW	0F000004h
4006C014h	LPUART Status (LPUART2_STAT)	32	RW	00C00000h
4006C018h	LPUART Control (LPUART2_CTRL)	32	RW	00000000h
4006C01Ch	LPUART Data (LPUART2_DATA)	32	RW	00001000h
4006C020h	LPUART Match Address (LPUART2_MATCH)	32	RW	00000000h
4006C024h	LPUART Modem IrDA (LPUART2_MODIR)	32	RW	00000000h
4006C028h	LPUART FIFO (LPUART2_FIFO)	32	RW	00C00011h
4006C02Ch	LPUART Watermark (LPUART2_WATER)	32	RW	00000000h

48.3.1.2 Version ID (VERID)

48.3.1.2.1 Address

Register	Offset
VERID	Base address + 0h offset

48.3.1.2.2 Diagram

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

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

48.3.1.2.3 Fields

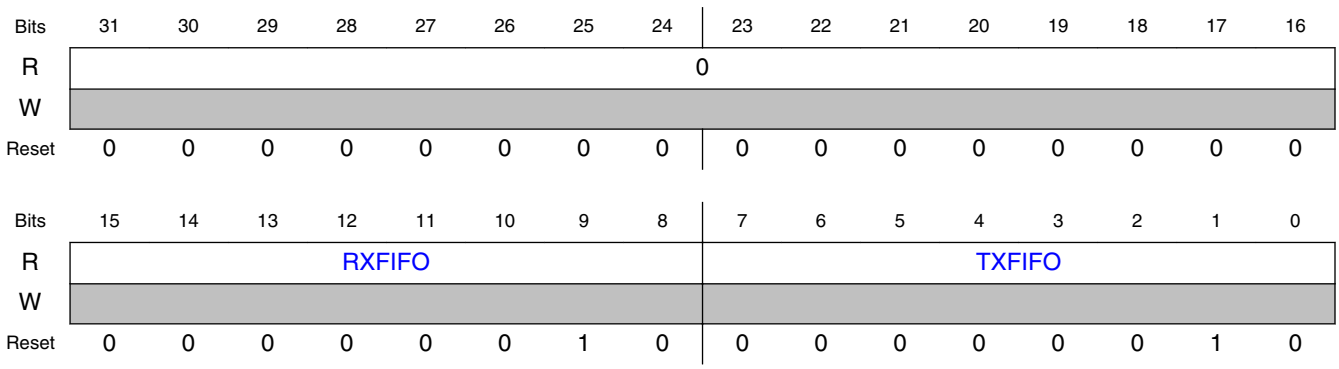
Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number This read only field returns the feature set number. 0000000000000001 - Standard feature set. 0000000000000011 - Standard feature set with MODEM/IrDA support.

48.3.1.3 Parameter (PARAM)

48.3.1.3.1 Address

Register	Offset
PARAM	Base address + 4h offset

48.3.1.3.2 Diagram



48.3.1.3.3 Fields

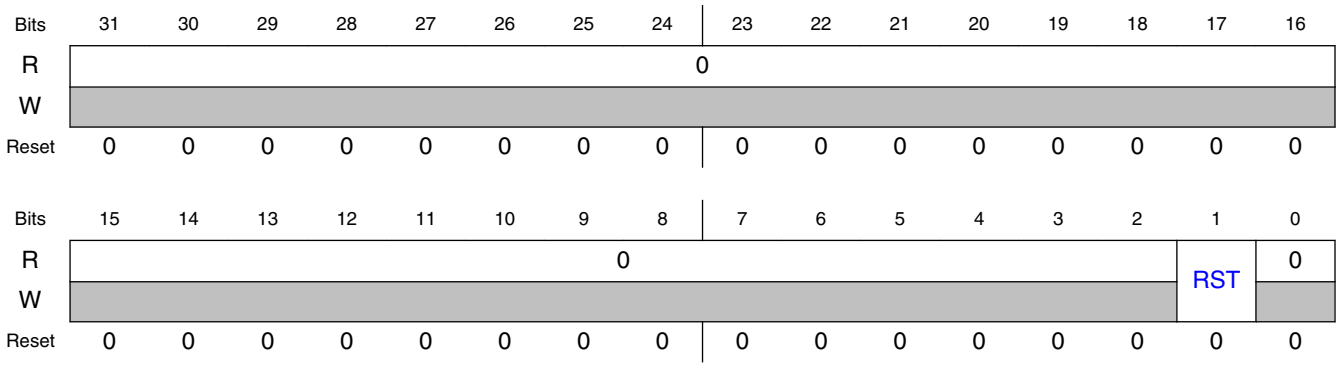
Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size The number of words in the receive FIFO is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

48.3.1.4 LPUART Global (GLOBAL)

48.3.1.4.1 Address

Register	Offset
GLOBAL	Base address + 8h offset

48.3.1.4.2 Diagram



48.3.1.4.3 Fields

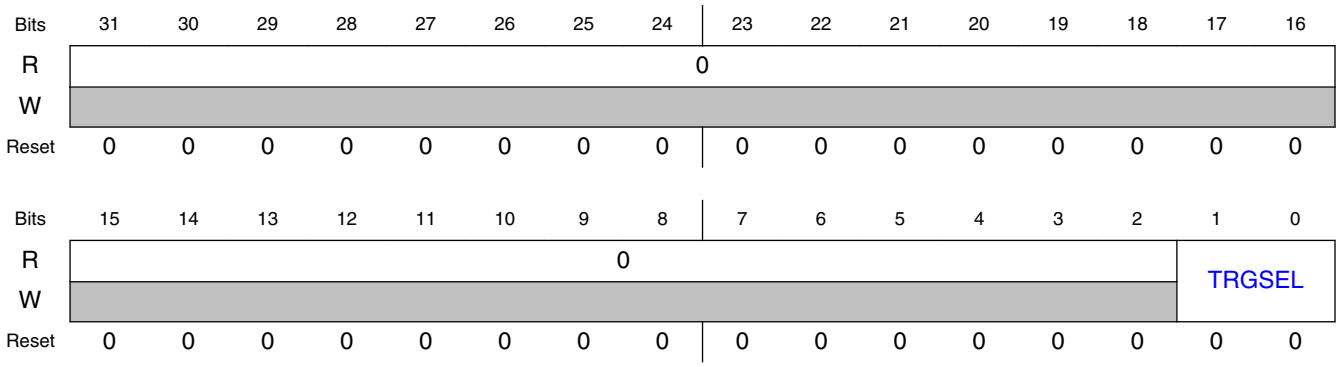
Field	Function
31-2 —	Reserved
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software. 0 - Module is not reset. 1 - Module is reset.
0 —	Reserved

48.3.1.5 LPUART Pin Configuration (PINCFG)

48.3.1.5.1 Address

Register	Offset
PINCFG	Base address + Ch offset

48.3.1.5.2 Diagram



48.3.1.5.3 Fields

Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. 00 - Input trigger is disabled. 01 - Input trigger is used instead of RXD pin input. 10 - Input trigger is used instead of CTS_B pin input. 11 - Input trigger is used to modulate the TXD pin output.

48.3.1.6 LPUART Baud Rate (BAUD)

48.3.1.6.1 Address

Register	Offset
BAUD	Base address + 10h offset

48.3.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAEN1	MAEN2	M10	OSR					TDMAE	0	RDMAE	RIDMAE	MATCFG		BOTHEDGE	RESYNCDIS
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

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

48.3.1.6.3 Fields

Field	Function
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 7-bit to 9-bit data characters. 1 - Receiver and transmitter use 10-bit data characters.
28-24 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver between 4x (00011) and 32x (11111). Writing an invalid oversampling ratio (i.e., a value not between 4x and 32x) 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.
22 —	Reserved
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 RIDMAE	Receiver Idle DMA Enable

Table continues on the next page...

Field	Function
	RIDMAE configures the receiver idle flag, LPUART_STAT[IDLE], to generate a DMA request. When this bit is set, reading LPUART_DATA when either DATA[RXEMPT] or DATA[IDLINE] bit is set, will generate an End Of Packet response until the completion of the existing DMA transfer. During an End of Packet response, reading the LPUART_DATA register will return 0x0000_33FF and does not pull data from the FIFO. 0 - DMA request disabled. 1 - DMA request enabled.
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. 0 - One stop bit. 1 - Two stop bits.
12-0 SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) * SBR)". The 13-bit baud rate setting [SBR12:SBR0] must only be updated when the transmitter and receiver are both disabled (LPUART_CTRL[RE] and LPUART_CTRL[TE] are both 0).

48.3.1.7 LPUART Status (STAT)

48.3.1.7.1 Address

Register	Offset
STAT	Base address + 14h offset

48.3.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	w1c	w1c										w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0													
W	w1c	w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.3.1.7.3 Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0 - No LIN break character has been detected. 1 - LIN break character has been detected.
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the RXD pin occurs. RXEDGIF is cleared by writing a 1 to it. 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 RXD input for all cases: data bits, start and stop bits, break, and idle.

Table continues on the next page...

Field	Function
	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 9 to 13 bit times. 1 - Break character is transmitted with length of 12 to 15 bit times.
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer. 0 - LIN break detect is disabled, normal break character can be detected. 1 - LIN break detect is enabled. LIN break character is detected at length of 11 bit times (if M = 0) or 12 (if M = 1) or 13 (M10 = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. 0 - LPUART receiver idle waiting for a start bit. 1 - LPUART receiver active (RXD input not idle).
23 TDRE	Transmit Data Register Empty Flag When the transmit FIFO is enabled, TDRE will set when the number of datawords in the transmit FIFO (LPUART_DATA) is equal to or less than the number indicated by LPUART_WATER[TXWATER]). To clear TDRE, write to the LPUART data register (LPUART_DATA) until the number of words in the transmit FIFO is greater than the number indicated by LPUART_WATER[TXWATER]. When the transmit FIFO is disabled,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

Table continues on the next page...

Field	Function
	When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by LPUART_WATER[RXWATER]. To clear RDRF, read LPUART_DATA until the number of datawords in the receive data buffer is equal to or less than the number indicated by LPUART_WATER[RXWATER]. When the receive FIFO is disabled, 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. 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.

Table continues on the next page...

Field	Function
	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
13-0 —	Reserved

48.3.1.8 LPUART Control (CTRL)

48.3.1.8.1 Address

Register	Offset
CTRL	Base address + 18h offset

48.3.1.8.2 Function

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

48.3.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0		M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.3.1.8.4 Fields

Field	Function
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading 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.
29 TXDIR	TXD Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TXD pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the TXD pin. 0 - TXD pin is an input in single-wire mode. 1 - TXD pin is an output in single-wire mode.
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 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.

Table continues on the next page...

Field	Function
	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.
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 TXD 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 9 to 13 bits, or 12 to 15 bits if LPUART_STAT[BRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the 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 0 - MA1F interrupt disabled 1 - MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0 - MA2F interrupt disabled 1 - MA2F interrupt enabled

Table continues on the next page...

Field	Function
13-12 —	Reserved
11 M7	7-Bit Mode Select This bit should only be changed when the transmitter and receiver are both disabled. 0 - Receiver and transmitter use 8-bit to 10-bit data characters. 1 - Receiver and transmitter use 7-bit data characters.
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 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 RXD pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 - Normal operation - RXD and TXD 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 RXD pin. 1 - Single-wire LPUART mode where the TXD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 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. 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.

Table continues on the next page...

Field	Function
	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.

48.3.1.9 LPUART Data (DATA)

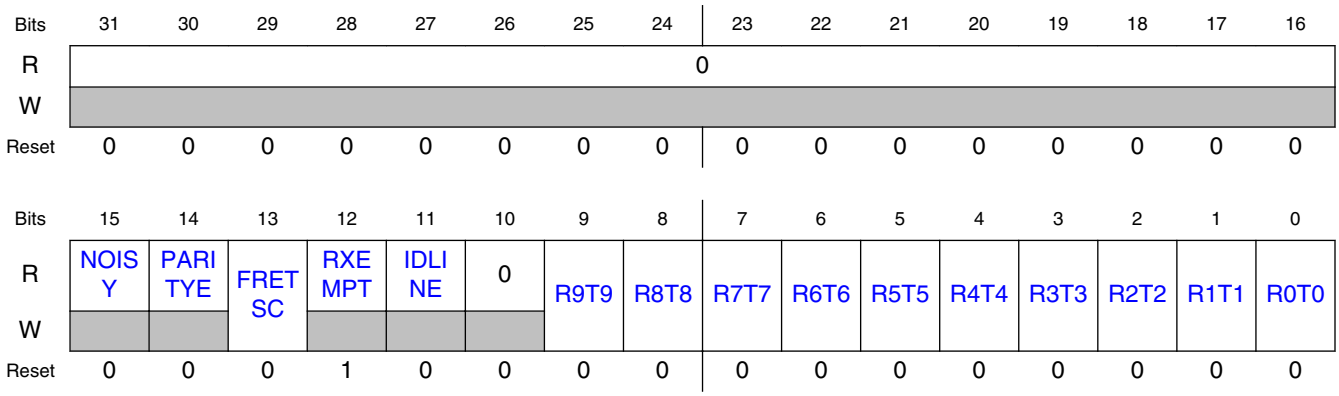
48.3.1.9.1 Address

Register	Offset
DATA	Base address + 1Ch offset

48.3.1.9.2 Function

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.

48.3.1.9.3 Diagram



48.3.1.9.4 Fields

Field	Function
31-16 —	Reserved
15 NOISY	NOISY The current received dataword contained in DATA[R9:R0] was received with noise. 0 - The dataword was received without noise. 1 - The data was received with noise.
14 PARITYE	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

Table continues on the next page...

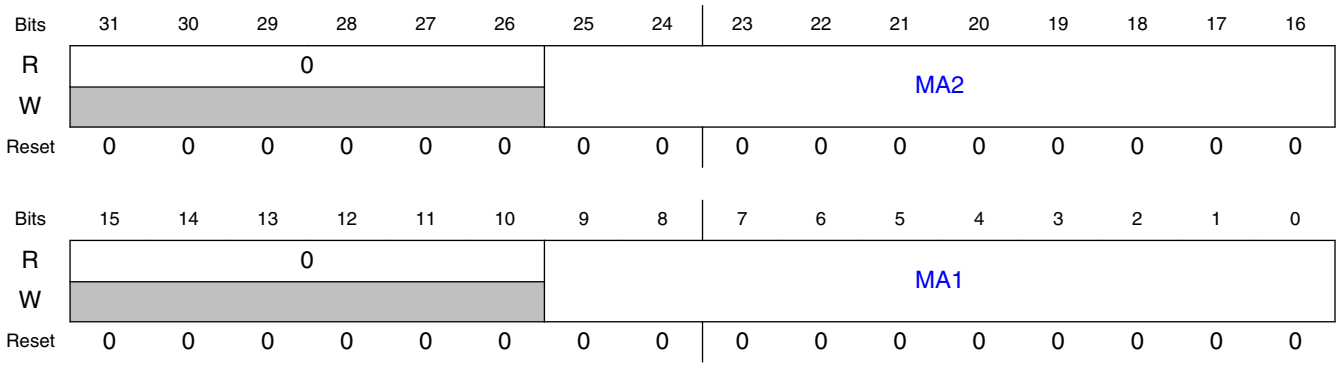
Field	Function
9 R9T9	R9T9 Read receive data buffer 9 or write transmit data buffer 9.
8 R8T8	R8T8 Read receive data buffer 8 or write transmit data buffer 8.
7 R7T7	R7T7 Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	R6T6 Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	R5T5 Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	R4T4 Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	R3T3 Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	R2T2 Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	R1T1 Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	R0T0 Read receive data buffer 0 or write transmit data buffer 0.

48.3.1.10 LPUART Match Address (MATCH)

48.3.1.10.1 Address

Register	Offset
MATCH	Base address + 20h offset

48.3.1.10.2 Diagram



48.3.1.10.3 Fields

Field	Function
31-26 —	Reserved
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
9-0 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.

48.3.1.11 LPUART Modem IrDA (MODIR)

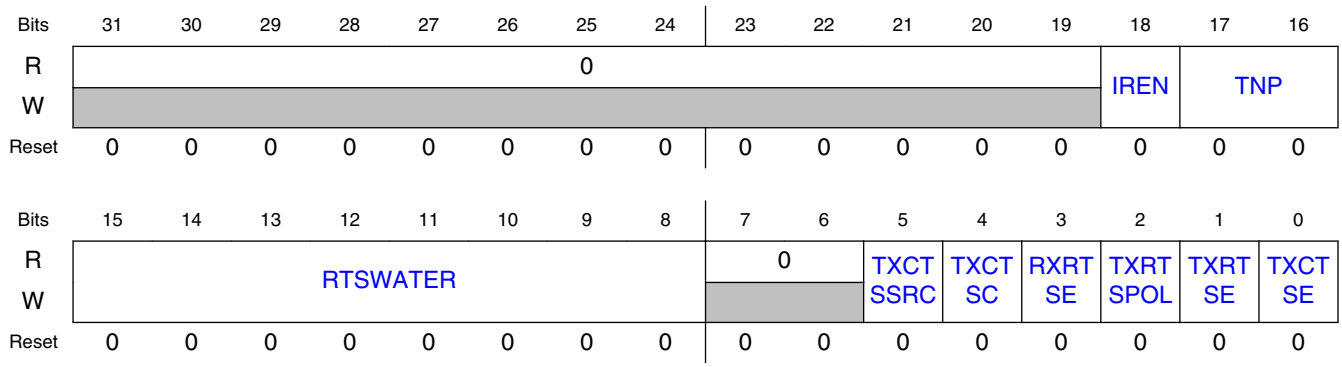
48.3.1.11.1 Address

Register	Offset
MODIR	Base address + 24h offset

48.3.1.11.2 Function

The MODEM register controls options for setting the modem configuration.

48.3.1.11.3 Diagram



48.3.1.11.4 Fields

Field	Function
31-19 —	Reserved
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. 0 - IR disabled. 1 - IR enabled.
17-16 TNP	Transmitter narrow pulse Configures whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse when IR is enabled. The IR pulse width should be configured to less than half of the oversampling ratio. Common pulse widths are 3/16, 1/16, 1/32 or 1/4 of the bit length. These can be configured by selecting the appropriate oversample ratio and pulse width. 00 - 1/OSR. 01 - 2/OSR. 10 - 3/OSR. 11 - 4/OSR.
15-8 RTSWATER	Receive RTS Configuration Configures the point at which the RX RTS output negates based on the number of additional characters that can be stored in the Receive FIFO. When configured to 0 with receiver controlling RTS, it negates when the the start bit is detetected for the character that will cause the FIFO to become full, and it asserts when the FIFO is not full and the character being received would not cause the FIFO to become full.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0 - CTS input is the CTS_B pin. 1 - CTS input is the inverted Receiver Match result.
4	Transmit CTS Configuration

Table continues on the next page...

Field	Function
TXCTSC	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. 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.

48.3.1.12 LPUART FIFO (FIFO)

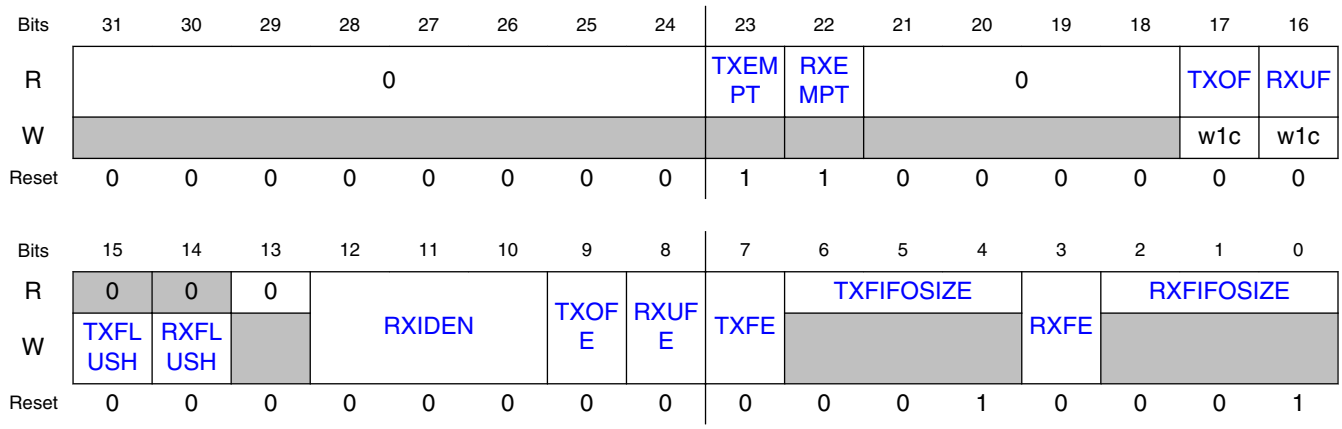
48.3.1.12.1 Address

Register	Offset
FIFO	Base address + 28h offset

48.3.1.12.2 Function

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the data buffer/FIFO is empty.

48.3.1.12.3 Diagram



48.3.1.12.4 Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register. 0 - Transmit buffer is not empty. 1 - Transmit buffer is empty.
22 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0 - Receive buffer is not empty. 1 - Receive buffer is empty.
21-18 —	Reserved
17 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of TXOFE. However, an interrupt will be issued to the host only if TXOFE is set. This flag is cleared by writing a 1. 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.
16 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of RXUFE. However, an interrupt will be issued to the host only if RXUFE is set. This flag is cleared by writing a 1. 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.
15	Transmit FIFO/Buffer Flush

Table continues on the next page...

Field	Function
TXFLUSH	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.
14 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0 - No flush operation occurs. 1 - All data in the receive FIFO/buffer is cleared out.
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable When set, enables the assertion of RDRF when the receiver is idle for a number of idle characters and the FIFO is not empty. 000 - Disable RDRF assertion due to partially filled FIFO when receiver is idle. 001 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 1 character. 010 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 2 characters. 011 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 4 characters. 100 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 8 characters. 101 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 16 characters. 110 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 32 characters. 111 - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 64 characters.
9 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0 - TXOF flag does not generate an interrupt to the host. 1 - TXOF flag generates an interrupt to the host.
8 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.
7 TXFE	Transmit FIFO Enable When this field is set, the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 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 - Transmit FIFO/Buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable

Table continues on the next page...

Field	Function
	When this field is set, the built in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0 - Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 - Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
2-0 RXFIFOSIZE	Receive FIFO. Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000 - Receive FIFO/Buffer depth = 1 dataword. 001 - Receive FIFO/Buffer depth = 4 datawords. 010 - Receive FIFO/Buffer depth = 8 datawords. 011 - Receive FIFO/Buffer depth = 16 datawords. 100 - Receive FIFO/Buffer depth = 32 datawords. 101 - Receive FIFO/Buffer depth = 64 datawords. 110 - Receive FIFO/Buffer depth = 128 datawords. 111 - Receive FIFO/Buffer depth = 256 datawords.

48.3.1.13 LPUART Watermark (WATER)

48.3.1.13.1 Address

Register	Offset
WATER	Base address + 2Ch offset

48.3.1.13.2 Function

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

48.3.1.13.3 Diagram

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

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

48.3.1.13.4 Fields

Field	Function
31-24 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
23-16 RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE] and must be greater than 0.
15-8 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
7-0 TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

48.4 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.

48.4.1 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.

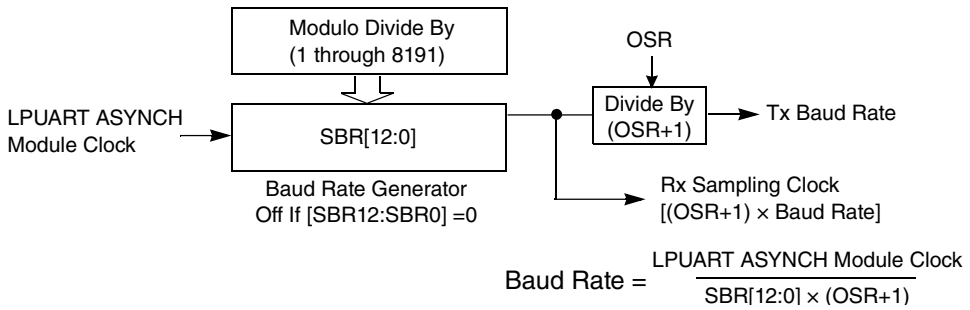


Figure 48-3. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

48.4.2 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 (TXD) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the LPUART data register.

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13 bits long depending on the setting in the CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at LPUART_DATA.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

48.4.2.1 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, 9-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. When the LPUART is the receiving device, 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 TXD 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_CTRL[M7], LPUART_BAUD[M10] and LPUART_BAUD[SNBS] bits as shown below.

Table 48-2. Break character length

BRK13	M	M10	M7	SBNS	Break character length
0	0	0	0	0	10 bit times
0	0	0	0	1	11 bit times
0	0	0	1	0	9 bit times
0	0	0	1	1	10 bit times
0	1	0	X	0	11 bit times
0	1	0	X	1	12 bit times
0	X	1	X	0	12 bit times
0	X	1	X	1	13 bit times
1	0	0	0	0	13 bit times
1	0	0	0	1	13 bit times
1	0	0	1	0	12 bit times
1	0	0	1	1	12 bit times
1	1	0	X	0	14 bit times
1	1	0	X	1	14 bit times
1	X	1	X	0	15 bit times
1	X	1	X	1	15 bit times

48.4.2.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the clear-to-send operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and TXD remains in the mark state until CTS_B is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can also be enabled even if the same LPUART receiver's RTS_B signal is disabled.

48.4.2.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, RTS_B asserts one bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmitter data buffer has any characters. RTS_B deasserts one

bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

48.4.2.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is 3-stated unless the LPUART is driving. The RTS_B signal can be used by the transmitter to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

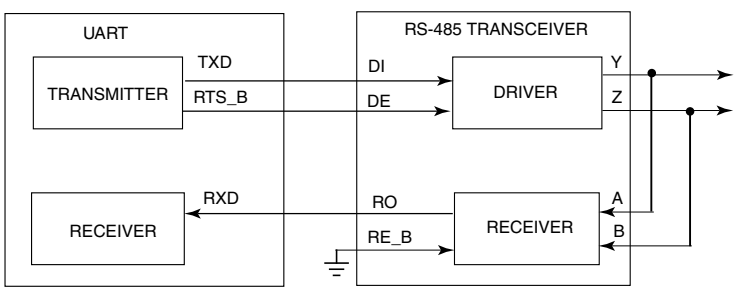


Figure 48-4. Transceiver driver enable using RTS_B

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. This option can be refined further by operating the LPUART in single wire mode, freeing the RXD pin for other uses.

48.4.3 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, seven to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit or 10-bit data mode, refer to Data Modes. For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (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.

48.4.3.1 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 RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (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.

48.4.3.2 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 48-3. Receiver Wakeup Options

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, IDLE flag not set
1	0	00	01	Receiver wakeup on idle line, IDLE flag set

Table continues on the next page...

Table 48-3. Receiver Wakeup Options (continued)

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
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

48.4.3.2.1 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], LPUART_CTRL[M7] and LPUART_BAUD[M10] control bit selects 7-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, 9 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.

48.4.3.2.2 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.

48.4.3.2.3 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.

48.4.3.2.4 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 RXD 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.

48.4.3.2.5 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 RXD 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.

48.4.3.2.6 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 RXD 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.

48.4.3.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert RTS_B.

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts RTS_B if the number of characters in the receiver data register is full or a start bit is detected that will cause the receiver data register to be full.
- The receiver asserts RTS_B when the number of characters in the receiver data register is not full and has not detected a start bit that will cause the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receiver data buffer is overrun.
- If the receiver request-to-send functionality is disabled, the receiver RTS_B remains deasserted.

48.4.3.4 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.

48.4.3.4.1 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.

48.4.3.4.2 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.

48.4.3.4.3 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.

48.4.3.4.4 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.

48.4.4 Additional LPUART functions

The following sections describe additional LPUART functions.

48.4.4.1 Data Modes

The LPUART transmitter and receiver can be configured to operate in 7-bit data mode by setting LPUART_CTRL[M7], 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.

48.4.4.2 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.

48.4.4.3 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 RXD pin is not used by the LPUART.

48.4.4.4 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 TXD pin (the RXD pin is not used).

In single-wire mode, the LPUART_CTRL[TXDIR] bit controls the direction of serial data on the TXD pin. When LPUART_CTRL[TXDIR] is cleared, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so an external device can send serial data to the receiver. When LPUART_CTRL[TXDIR] is set, the TXD pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

48.4.5 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 LPUART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

48.4.5.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD signal. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent 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.

48.4.5.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow 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.

48.4.6 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 TXD at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (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 RXD 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 RXD 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 49
Flexible I/O (FlexIO)

49.1 Chip-specific Information for this Module

49.1.1 Instantiation Information

Table 49-1. FlexIO Configuration

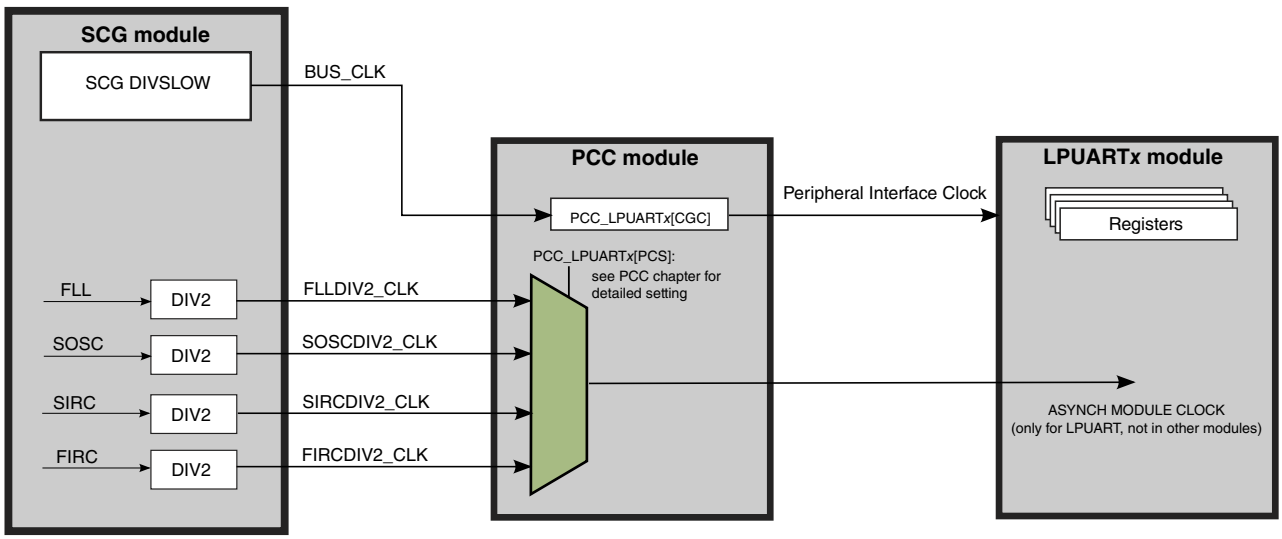
	Timers	Shifters	Pins
Number	4	4	8

49.1.2 FlexIO Clocking Information

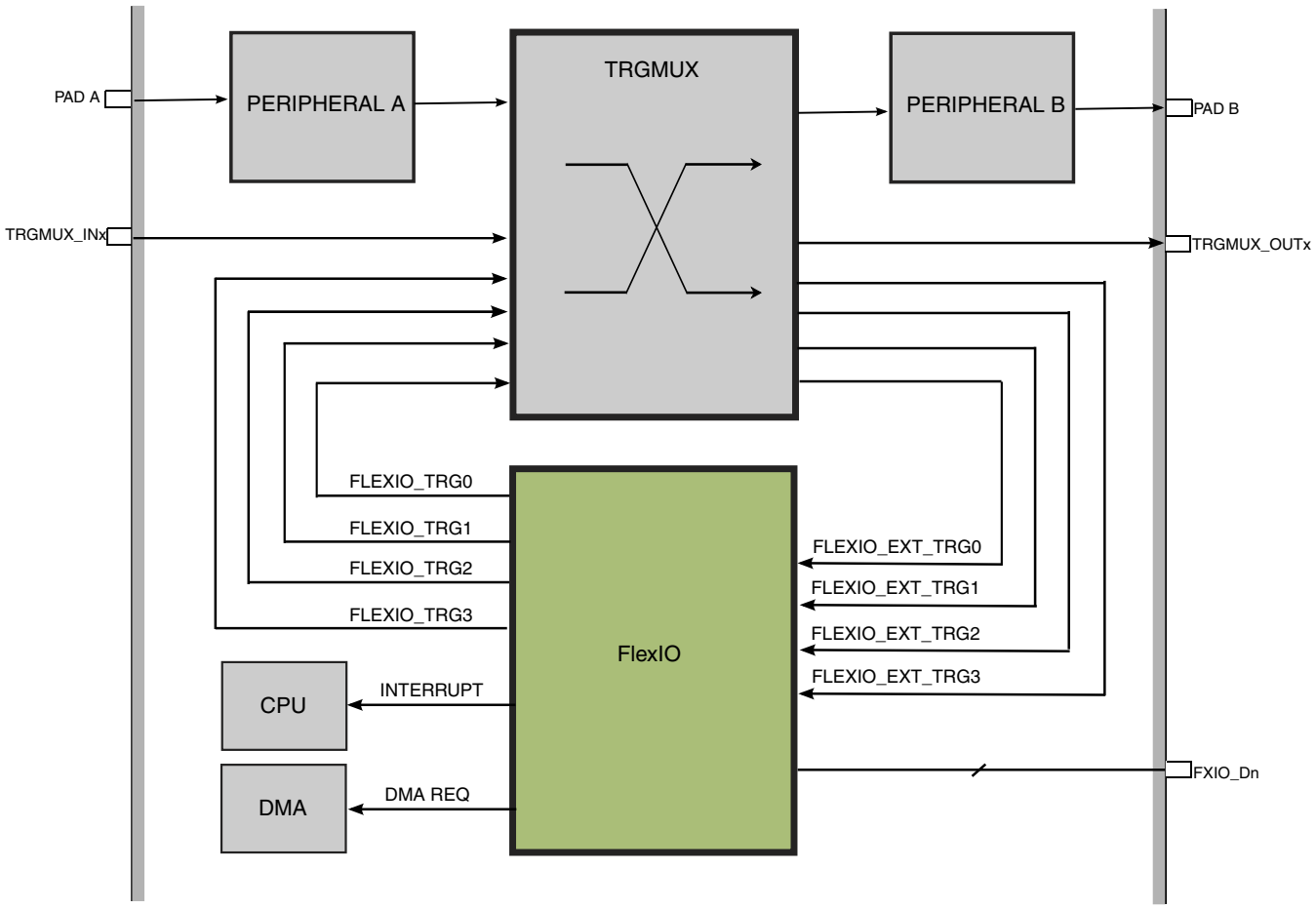
The FlexIO blocks are clocked from a single FlexIO clock that can be selected from OSCCLK, SCGIRCLK, SCGFIRCLK, or SCGFCLK. The selected source is controlled by the PCC_FLEXIO register in the PCC module. You have to select a clock for FlexIO and enable the clock gate before accessing any of the FlexIO registers.

Peripheral Clocking - LPUART

Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, and LPIT.



49.1.3 Inter-connectivity Information



FlexIO has a selectable trigger input source controlled by FlexIO_TIMCTLn[TRGSEL] (4-bit field) to use for starting the counter and/or reloading the counter. The trigger signal is from the FlexIO module itself which is called internal triggers, or from other modules which is called external triggers. The external triggers selection is controlled by the TRGMUX_FLEXIO register in the TRGMUX module. For this device, the external triggers can be selected from any of the TRGMUX trigger sources.

49.2 Introduction

49.2.1 Overview

The FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial communication protocols
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions

These functions are provided by the FlexIO while adhering to the following key objectives:

- Low software/CPU overhead: less overhead than software bit-banging, more overhead than dedicated peripheral IP.
- Area/Power efficient implementation: more efficient than integrating multiple peripherals for each desired protocol.

49.2.2 Features

The FlexIO module is capable of supporting a wide range of protocols including, but not limited to:

- UART
- I2C
- SPI
- I2S
- PWM/Waveform generation

The following key features are provided:

- Array of 32-bit shift registers with transmit, receive and data match modes
- Double buffered shifter operation for continuous data transfer
- Shifter concatenation to support large transfer sizes
- Automatic start/stop bit generation
- Interrupt, DMA or polled transmit/receive operation
- Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during stop modes
- Highly flexible 16-bit timers with support for a variety of internal or external trigger, reset, enable and disable conditions

49.2.3 Block Diagram

The following diagram gives a high-level overview of the configuration of FlexIO timers and shifters.

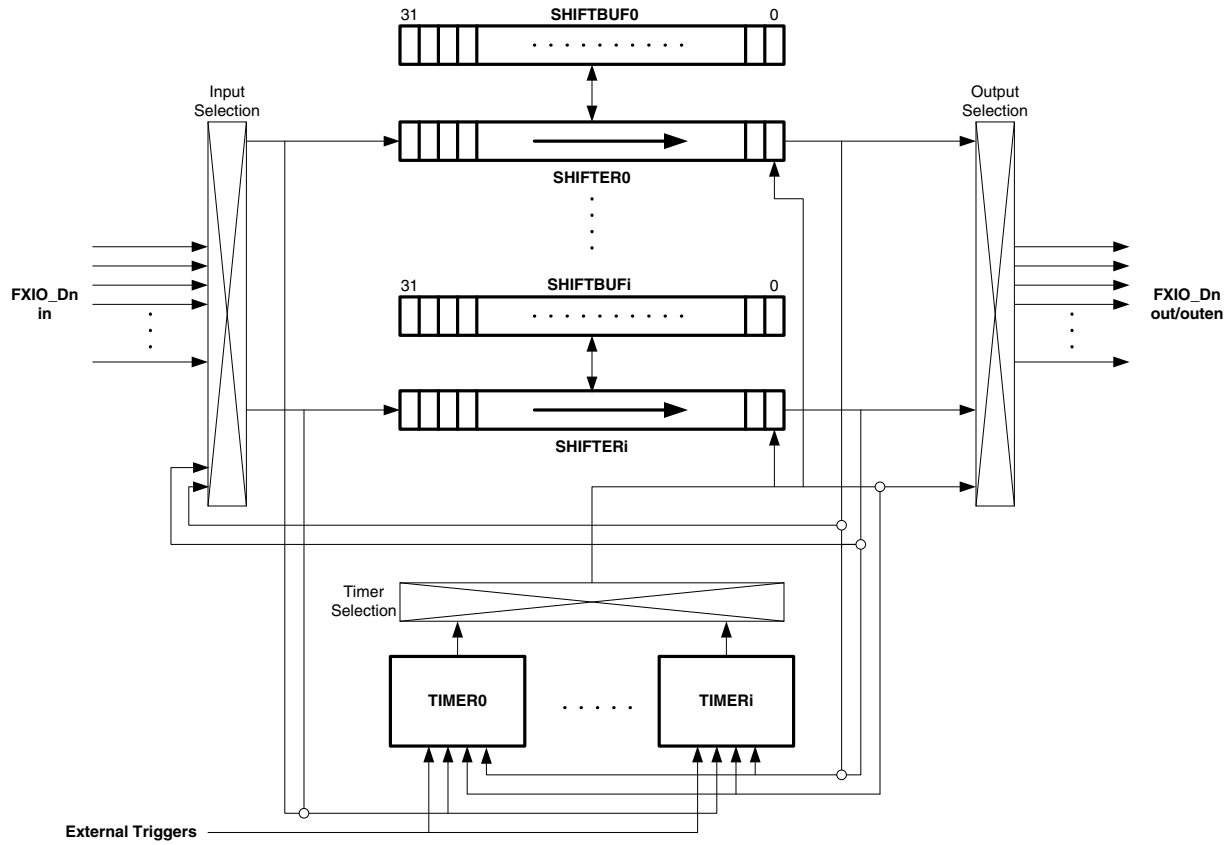


Figure 49-1. FlexIO block diagram

49.2.4 Modes of operation

The FlexIO module supports the chip modes described in the following table.

Table 49-2. Chip modes supported by the FlexIO module

Chip mode	FlexIO Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (CTRL[DOZEN]) is set and the FlexIO is using an external or internal clock source which remains operating during stop/wait modes.
Debug	Can continue operating provided the Debug Enable bit (CTRL[DBGE]) is set.

49.2.5 FlexIO Signal Descriptions

Signal	Description	I/O
FXIO_Dn (n=0...7)	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

49.3 Memory Map/Register Definition

This section includes the memory map and register definition.

NOTE

The FlexIO functional clock must be enabled before accessing any FlexIO registers. Accessing FlexIO registers with FlexIO functional clock disabled will result in transfer error.

FLEXIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A000	Version ID Register (FLEXIO_VERID)	32	R	0101_0000h	49.3.1/1340
4005_A004	Parameter Register (FLEXIO_PARAM)	32	R	See section	49.3.2/1341
4005_A008	FlexIO Control Register (FLEXIO_CTRL)	32	R/W	0000_0000h	49.3.3/1341
4005_A00C	Pin State Register (FLEXIO_PIN)	32	R	0000_0000h	49.3.4/1342
4005_A010	Shifter Status Register (FLEXIO_SHIFTSTAT)	32	w1c	0000_0000h	49.3.5/1343
4005_A014	Shifter Error Register (FLEXIO_SHIFTERR)	32	w1c	0000_0000h	49.3.6/1344
4005_A018	Timer Status Register (FLEXIO_TIMSTAT)	32	w1c	0000_0000h	49.3.7/1344
4005_A020	Shifter Status Interrupt Enable (FLEXIO_SHIFTSIEN)	32	R/W	0000_0000h	49.3.8/1345
4005_A024	Shifter Error Interrupt Enable (FLEXIO_SHIFTEIEN)	32	R/W	0000_0000h	49.3.9/1346
4005_A028	Timer Interrupt Enable Register (FLEXIO_TIMIEN)	32	R/W	0000_0000h	49.3.10/ 1346
4005_A030	Shifter Status DMA Enable (FLEXIO_SHIFTSDEN)	32	R/W	0000_0000h	49.3.11/ 1347
4005_A080	Shifter Control N Register (FLEXIO_SHIFTCTL0)	32	R/W	0000_0000h	49.3.12/ 1347
4005_A084	Shifter Control N Register (FLEXIO_SHIFTCTL1)	32	R/W	0000_0000h	49.3.12/ 1347
4005_A088	Shifter Control N Register (FLEXIO_SHIFTCTL2)	32	R/W	0000_0000h	49.3.12/ 1347
4005_A08C	Shifter Control N Register (FLEXIO_SHIFTCTL3)	32	R/W	0000_0000h	49.3.12/ 1347
4005_A100	Shifter Configuration N Register (FLEXIO_SHIFTCFG0)	32	R/W	0000_0000h	49.3.13/ 1349

Table continues on the next page...

FLEXIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A104	Shifter Configuration N Register (FLEXIO_SHIFTCFG1)	32	R/W	0000_0000h	49.3.13/ 1349
4005_A108	Shifter Configuration N Register (FLEXIO_SHIFTCFG2)	32	R/W	0000_0000h	49.3.13/ 1349
4005_A10C	Shifter Configuration N Register (FLEXIO_SHIFTCFG3)	32	R/W	0000_0000h	49.3.13/ 1349
4005_A200	Shifter Buffer N Register (FLEXIO_SHIFTBUF0)	32	R/W	0000_0000h	49.3.14/ 1350
4005_A204	Shifter Buffer N Register (FLEXIO_SHIFTBUF1)	32	R/W	0000_0000h	49.3.14/ 1350
4005_A208	Shifter Buffer N Register (FLEXIO_SHIFTBUF2)	32	R/W	0000_0000h	49.3.14/ 1350
4005_A20C	Shifter Buffer N Register (FLEXIO_SHIFTBUF3)	32	R/W	0000_0000h	49.3.14/ 1350
4005_A280	Shifter Buffer N Bit Swapped Register (FLEXIO_SHIFTBUFBIS0)	32	R/W	0000_0000h	49.3.15/ 1351
4005_A284	Shifter Buffer N Bit Swapped Register (FLEXIO_SHIFTBUFBIS1)	32	R/W	0000_0000h	49.3.15/ 1351
4005_A288	Shifter Buffer N Bit Swapped Register (FLEXIO_SHIFTBUFBIS2)	32	R/W	0000_0000h	49.3.15/ 1351
4005_A28C	Shifter Buffer N Bit Swapped Register (FLEXIO_SHIFTBUFBIS3)	32	R/W	0000_0000h	49.3.15/ 1351
4005_A300	Shifter Buffer N Byte Swapped Register (FLEXIO_SHIFTBUFBYS0)	32	R/W	0000_0000h	49.3.16/ 1351
4005_A304	Shifter Buffer N Byte Swapped Register (FLEXIO_SHIFTBUFBYS1)	32	R/W	0000_0000h	49.3.16/ 1351
4005_A308	Shifter Buffer N Byte Swapped Register (FLEXIO_SHIFTBUFBYS2)	32	R/W	0000_0000h	49.3.16/ 1351
4005_A30C	Shifter Buffer N Byte Swapped Register (FLEXIO_SHIFTBUFBYS3)	32	R/W	0000_0000h	49.3.16/ 1351
4005_A380	Shifter Buffer N Bit Byte Swapped Register (FLEXIO_SHIFTBUFBBS0)	32	R/W	0000_0000h	49.3.17/ 1352
4005_A384	Shifter Buffer N Bit Byte Swapped Register (FLEXIO_SHIFTBUFBBS1)	32	R/W	0000_0000h	49.3.17/ 1352
4005_A388	Shifter Buffer N Bit Byte Swapped Register (FLEXIO_SHIFTBUFBBS2)	32	R/W	0000_0000h	49.3.17/ 1352
4005_A38C	Shifter Buffer N Bit Byte Swapped Register (FLEXIO_SHIFTBUFBBS3)	32	R/W	0000_0000h	49.3.17/ 1352
4005_A400	Timer Control N Register (FLEXIO_TIMCTL0)	32	R/W	0000_0000h	49.3.18/ 1352
4005_A404	Timer Control N Register (FLEXIO_TIMCTL1)	32	R/W	0000_0000h	49.3.18/ 1352
4005_A408	Timer Control N Register (FLEXIO_TIMCTL2)	32	R/W	0000_0000h	49.3.18/ 1352

Table continues on the next page...

FLEXIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A40C	Timer Control N Register (FLEXIO_TIMCTL3)	32	R/W	0000_0000h	49.3.18/ 1352
4005_A480	Timer Configuration N Register (FLEXIO_TIMCFG0)	32	R/W	0000_0000h	49.3.19/ 1354
4005_A484	Timer Configuration N Register (FLEXIO_TIMCFG1)	32	R/W	0000_0000h	49.3.19/ 1354
4005_A488	Timer Configuration N Register (FLEXIO_TIMCFG2)	32	R/W	0000_0000h	49.3.19/ 1354
4005_A48C	Timer Configuration N Register (FLEXIO_TIMCFG3)	32	R/W	0000_0000h	49.3.19/ 1354
4005_A500	Timer Compare N Register (FLEXIO_TIMCMP0)	32	R/W	0000_0000h	49.3.20/ 1356
4005_A504	Timer Compare N Register (FLEXIO_TIMCMP1)	32	R/W	0000_0000h	49.3.20/ 1356
4005_A508	Timer Compare N Register (FLEXIO_TIMCMP2)	32	R/W	0000_0000h	49.3.20/ 1356
4005_A50C	Timer Compare N Register (FLEXIO_TIMCMP3)	32	R/W	0000_0000h	49.3.20/ 1356

49.3.1 Version ID Register (FLEXIO_VERID)

.

Address: 4005_A000h base + 0h offset = 4005_A000h

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

FLEXIO_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0000 Standard features implemented. 0x0001 Supports state, logic and parallel modes.

49.3.2 Parameter Register (FLEXIO_PARAM)

.

Address: 4005_A000h base + 4h offset = 4005_A004h

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

FLEXIO_PARAM field descriptions

Field	Description
31–24 TRIGGER	Trigger Number Number of external triggers implemented.
23–16 PIN	Pin Number Number of Pins implemented.
15–8 TIMER	Timer Number Number of Timers implemented.
SHIFTER	Shifter Number Number of Shifters implemented.

49.3.3 FlexIO Control Register (FLEXIO_CTRL)

.

Address: 4005_A000h base + 8h offset = 4005_A008h

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

FLEXIO_CTRL field descriptions

Field	Description
31 DOZEN	Doze Enable Disables FlexIO operation in Doze modes. This field is ignored and the FlexIO always disabled in low-leakage stop modes. 0 FlexIO enabled in Doze modes. 1 FlexIO disabled in Doze modes.
30 DBGE	Debug Enable Enables FlexIO operation in Debug mode. 0 FlexIO is disabled in debug modes. 1 FlexIO is enabled in debug modes
29–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 FASTACC	Fast Access Enables fast register accesses to FlexIO registers, but requires the FlexIO clock to be at least twice the frequency of the bus clock. 0 Configures for normal register accesses to FlexIO 1 Configures for fast register accesses to FlexIO
1 SWRST	Software Reset The FlexIO Control Register is not affected by the software reset, all other logic in the FlexIO is affected by the software reset and register accesses are ignored until this bit is cleared. This register bit will remain set until cleared by software, and the reset has cleared in the FlexIO clock domain. 0 Software reset is disabled 1 Software reset is enabled, all FlexIO registers except the Control Register are reset.
0 FLEXEN	FlexIO Enable 0 FlexIO module is disabled. 1 FlexIO module is enabled.

49.3.4 Pin State Register (FLEXIO_PIN)

Address: 4005_A000h base + Ch offset = 4005_A00Ch

Bit	31	30	29	28	27	26	25	24	23	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																								PDI							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

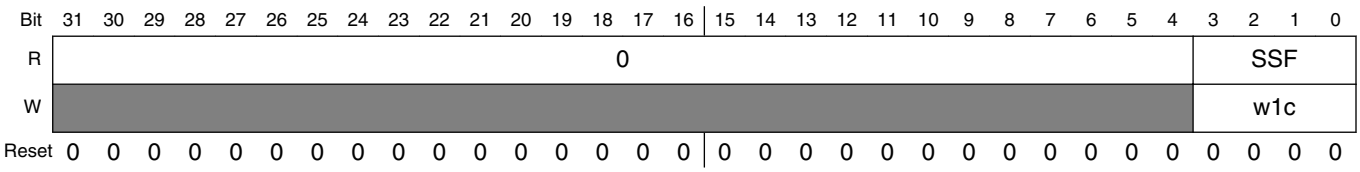
FLEXIO_PIN field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PDI	Pin Data Input Returns the input data on each of the FlexIO pins.

49.3.5 Shifter Status Register (FLEXIO_SHIFTSTAT)

.

Address: 4005_A000h base + 10h offset = 4005_A010h



FLEXIO_SHIFTSTAT field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from Shifter (SHIFTBUF is full), and the status flag is cleared when SHIFTBUF register is read. For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the Shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit, and the status flag is cleared when the SHIFTBUF register is written. For SMOD=Match Store, the status flag is set when a match has occured between SHIFTBUF and Shifter, and the status flag is cleared when the SHIFTBUF register is read. For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and Shifter. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous. 0 Status flag is clear 1 Status flag is set

49.3.6 Shifter Error Register (FLEXIO_SHIFTERR)

.

Address: 4005_A000h base + 14h offset = 4005_A014h

Bit	31	30	29	28	27	26	25	24	23	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																												SEF			
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

FLEXIO_SHIFTERR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: For SMOD=Receive, indicates Shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF Overrun), or indicates that the received start or stop bit does not match the expected value. For SMOD=Transmit, indicates Shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF Underrun). For SMOD=Match Store, indicates a match event occured before the previous match data was read from SHIFTBUF (SHIFTBUF Overrun). For SMOD=Match Continuous, the error flag is set when a match has occured between SHIFTBUF and Shifter. Can be cleared by writing logic one to the flag. For SMOD=Match Continuous, can also be cleared when the SHIFTBUF register is read. 0 Shifter Error Flag is clear 1 Shifter Error Flag is set

49.3.7 Timer Status Register (FLEXIO_TIMSTAT)

.

Address: 4005_A000h base + 18h offset = 4005_A018h

Bit	31	30	29	28	27	26	25	24	23	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																												TSF			
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

FLEXIO_TIMSTAT field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register. In 8-bit PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register.. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register.. 0 Timer Status Flag is clear 1 Timer Status Flag is set

49.3.8 Shifter Status Interrupt Enable (FLEXIO_SHIFTSIEN)

.

Address: 4005_A000h base + 20h offset = 4005_A020h

Bit	31	30	29	28	27	26	25	24	23	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																												SSIE			
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

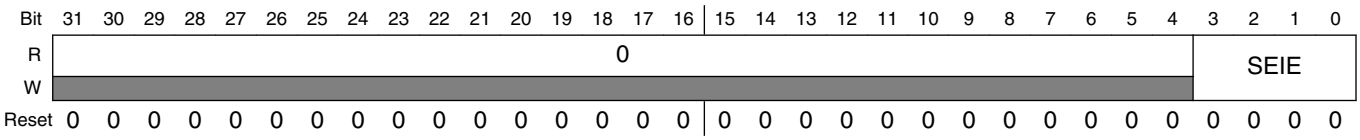
FLEXIO_SHIFTSIEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 0 Shifter Status Flag interrupt disabled 1 Shifter Status Flag interrupt enabled

49.3.9 Shifter Error Interrupt Enable (FLEXIO_SHIFTEIEN)

.

Address: 4005_A000h base + 24h offset = 4005_A024h



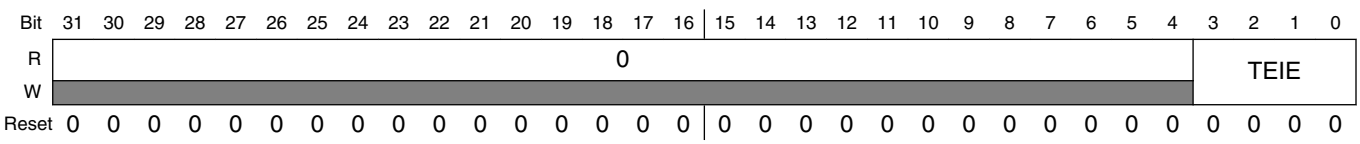
FLEXIO_SHIFTEIEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 0 Shifter Error Flag interrupt disabled 1 Shifter Error Flag interrupt enabled

49.3.10 Timer Interrupt Enable Register (FLEXIO_TIMIEN)

.

Address: 4005_A000h base + 28h offset = 4005_A028h



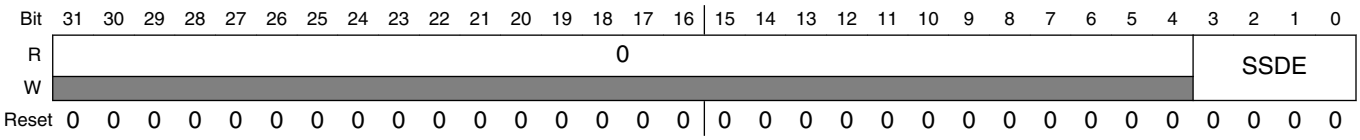
FLEXIO_TIMIEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 0 Timer Status Flag interrupt is disabled 1 Timer Status Flag interrupt is enabled

49.3.11 Shifter Status DMA Enable (FLEXIO_SHIFTSDEN)

.

Address: 4005_A000h base + 30h offset = 4005_A030h



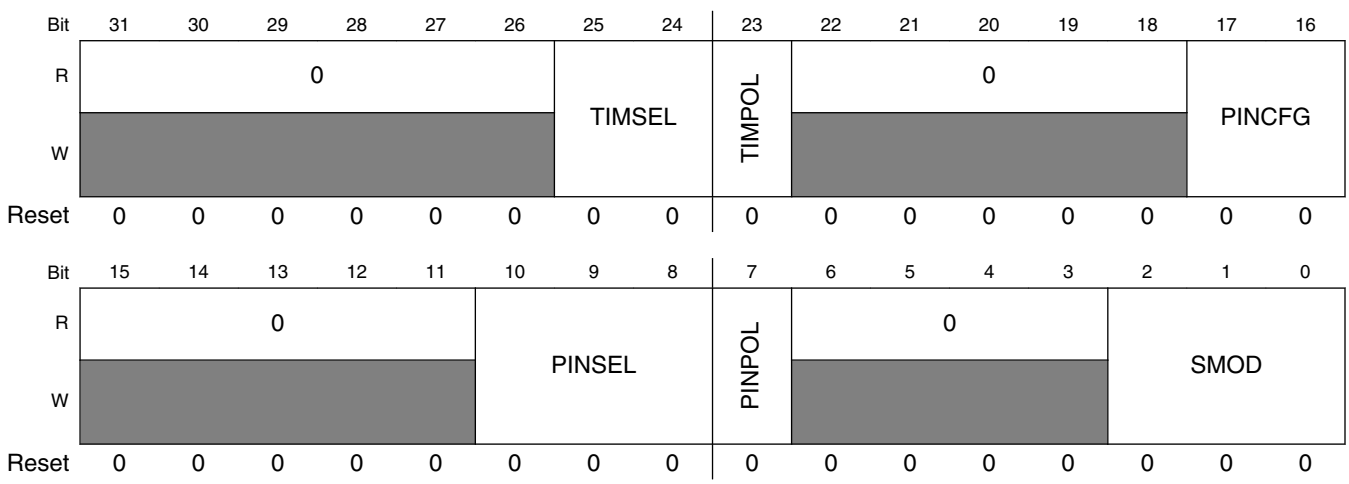
FLEXIO_SHIFTSDEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 0 Shifter Status Flag DMA request is disabled 1 Shifter Status Flag DMA request is enabled

49.3.12 Shifter Control N Register (FLEXIO_SHIFTCTL*n*)

.

Address: 4005_A000h base + 80h offset + (4d × i), where i=0d to 3d



FLEXIO_SHIFTCTL*n* field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 TIMSEL	Timer Select Selects which Timer is used for controlling the logic/shift register and generating the Shift clock.
23 TIMPOL	Timer Polarity 0 Shift on posedge of Shift clock 1 Shift on negedge of Shift clock
22–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 PINCFG	Shifter Pin Configuration 00 Shifter pin output disabled 01 Shifter pin open drain or bidirectional output enable 10 Shifter pin bidirectional output data 11 Shifter pin output
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output.
7 PINPOL	Shifter Pin Polarity 0 Pin is active high 1 Pin is active low
6–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SMOD	Shifter Mode Configures the mode of the Shifter. 000 Disabled. 001 Receive mode. Captures the current Shifter content into the SHIFTBUF on expiration of the Timer. 010 Transmit mode. Load SHIFTBUF contents into the Shifter on expiration of the Timer. 011 Reserved. 100 Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the Timer. 101 Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110 Reserved. 111 Reserved.

49.3.13 Shifter Configuration N Register (FLEXIO_SHIFTCFG*n*)

.

Address: 4005_A000h base + 100h offset + (4d × i), where i=0d to 3d

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

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

FLEXIO_SHIFTCFG*n* field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 INSRC	Input Source Selects the input source for the shifter. 0 Pin 1 Shifter N+1 Output
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.
5–4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. 00 Stop bit disabled for transmitter/receiver/match store 01 Reserved for transmitter/receiver/match store 10 Transmitter outputs stop bit value 0 on store, receiver/match store sets error flag if stop bit is not 0 11 Transmitter outputs stop bit value 1 on store, receiver/match store sets error flag if stop bit is not 1

Table continues on the next page...

FLEXIO_SHIFTCFG*n* field descriptions (continued)

Field	Description
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SSTART	Shifter Start bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. 00 Start bit disabled for transmitter/receiver/match store, transmitter loads data on enable 01 Start bit disabled for transmitter/receiver/match store, transmitter loads data on first shift 10 Transmitter outputs start bit value 0 before loading data on first shift, receiver/match store sets error flag if start bit is not 0 11 Transmitter outputs start bit value 1 before loading data on first shift, receiver/match store sets error flag if start bit is not 1

49.3.14 Shifter Buffer N Register (FLEXIO_SHIFTBUF*n*)

.

Address: 4005_A000h base + 200h offset + (4d × i), where i=0d to 3d

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

FLEXIO_SHIFTBUF*n* field descriptions

Field	Description
SHIFTBUF	Shift Buffer Shift buffer data is used for a variety of functions depending on the SMOD setting: For SMOD=Receive, Shifter data is transferred into SHIFTBUF at the expiration of Timer. For SMOD=Transmit, SHIFTBUF data is transferred into the Shifter before the Timer begins. For SMOD=Match Store/Continuous, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents. The Match is checked either continuosly (Match Continous mode) or when the Timer expires (Match Store mode). SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). In Match Store mode, Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs.

49.3.15 Shifter Buffer N Bit Swapped Register (FLEXIO_SHIFTBUFBIS*n*)

.

Address: 4005_A000h base + 280h offset + (4d × i), where i=0d to 3d

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

FLEXIO_SHIFTBUFBIS*n* field descriptions

Field	Description
SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped. Reads return SHIFTBUF[0:31].

49.3.16 Shifter Buffer N Byte Swapped Register (FLEXIO_SHIFTBUFBYS*n*)

.

Address: 4005_A000h base + 300h offset + (4d × i), where i=0d to 3d

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

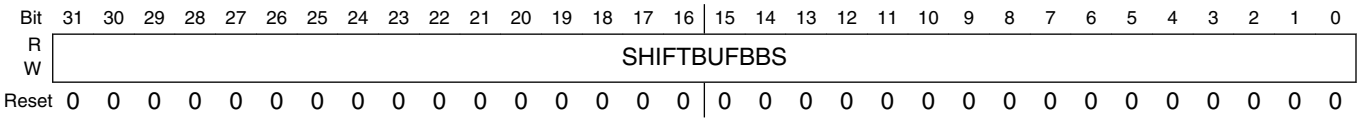
FLEXIO_SHIFTBUFBYS*n* field descriptions

Field	Description
SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are byte swapped. Reads return { SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24] }.

49.3.17 Shifter Buffer N Bit Byte Swapped Register (FLEXIO_SHIFTBUFBBS*n*)

.

Address: 4005_A000h base + 380h offset + (4d × i), where i=0d to 3d



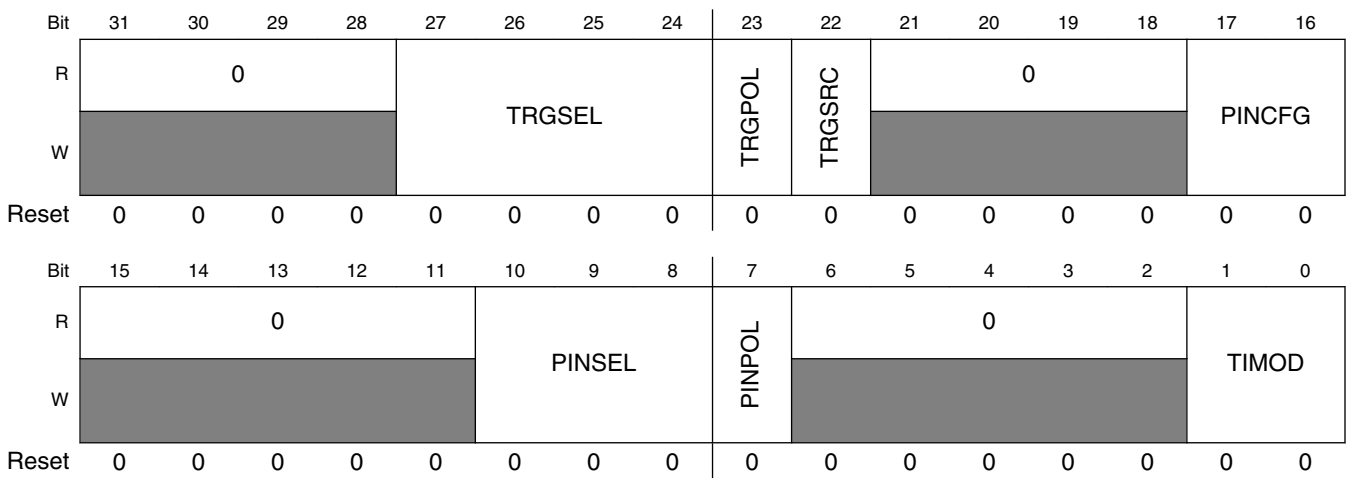
FLEXIO_SHIFTBUFBBS*n* field descriptions

Field	Description
SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped within each byte. Reads return { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }.

49.3.18 Timer Control N Register (FLEXIO_TIMCTL*n*)

.

Address: 4005_A000h base + 400h offset + (4d × i), where i=0d to 3d



FLEXIO_TIMCTL*n* 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 TRGSEL	Trigger Select The valid values for TRGSEL will depend on the FLEXIO_PARAM register.

Table continues on the next page...

FLEXIO_TIMCTL*n* field descriptions (continued)

Field	Description
	• When TRGSRC = 1, the valid values for N will depend on PIN, TIMER, SHIFTER fields in the FLEXIO_PARAM register. • When TRGSRC = 0, the valid values for N will depend on TRIGGER field in FLEXIO_PARAM register. Refer to the chip configuration section for external trigger selection. The internal trigger selection is configured as follows: {N,00} pin 2N input {N,01} shifter N status flag {N,10} pin 2N+1 input {N,11} timer N trigger output
23 TRGPOL	Trigger Polarity 0 Trigger active high 1 Trigger active low
22 TRGSRC	Trigger Source 0 External trigger selected 1 Internal trigger selected
21–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 PINCFG	Timer Pin Configuration 00 Timer pin output disabled 01 Timer pin open drain or bidirectional output enable 10 Timer pin bidirectional output data 11 Timer pin output
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 PINSEL	Timer Pin Select Selects which pin is used by the Timer input or output.
7 PINPOL	Timer Pin Polarity 0 Pin is active high 1 Pin is active low
6–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TIMOD	Timer Mode In 8-bit counter mode, the lower 8-bits of the counter and compare register are used to configure the baud rate of the timer shift clock and the upper 8-bits are used to configure the shifter bit count. In 8-bit PWM mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock and the upper 8-bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. 00 Timer Disabled.

Table continues on the next page...

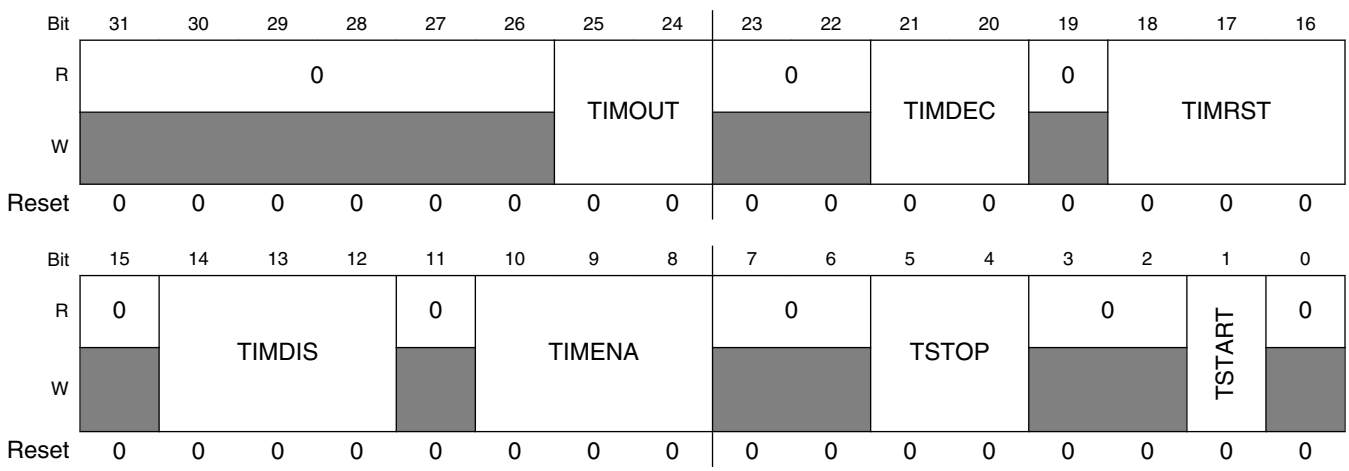
FLEXIO_TIMCTL*n* field descriptions (continued)

Field	Description
	01 Dual 8-bit counters baud/bit mode. 10 Dual 8-bit counters PWM mode. 11 Single 16-bit counter mode.

49.3.19 Timer Configuration N Register (FLEXIO_TIMCFG*n*)

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (eg: Timer 0).

Address: 4005_A000h base + 480h offset + (4d × i), where i=0d to 3d



FLEXIO_TIMCFG*n* field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 TIMOUT	Timer Output Configures the initial state of the Timer Output and whether it is affected by the Timer reset. 00 Timer output is logic one when enabled and is not affected by timer reset 01 Timer output is logic zero when enabled and is not affected by timer reset 10 Timer output is logic one when enabled and on timer reset 11 Timer output is logic zero when enabled and on timer reset
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–20 TIMDEC	Timer Decrement Configures the source of the Timer decrement and the source of the Shift clock. 00 Decrement counter on FlexIO clock, Shift clock equals Timer output. 01 Decrement counter on Trigger input (both edges), Shift clock equals Timer output.

Table continues on the next page...

FLEXIO_TIMCFG*n* field descriptions (continued)

Field	Description
	10 Decrement counter on Pin input (both edges), Shift clock equals Pin input. 11 Decrement counter on Trigger input (both edges), Shift clock equals Trigger input.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset will only reset the lower 8-bits that configure the baud rate. In all other modes, the timer reset will reset the full 16-bits of the counter. 000 Timer never reset 001 Reserved 010 Timer reset on Timer Pin equal to Timer Output 011 Timer reset on Timer Trigger equal to Timer Output 100 Timer reset on Timer Pin rising edge 101 Reserved 110 Timer reset on Trigger rising edge 111 Timer reset on Trigger rising or falling edge
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 TIMDIS	Timer Disable Configures the condition that causes the Timer to be disabled and stop decrementing. 000 Timer never disabled 001 Timer disabled on Timer N-1 disable 010 Timer disabled on Timer compare 011 Timer disabled on Timer compare and Trigger Low 100 Timer disabled on Pin rising or falling edge 101 Timer disabled on Pin rising or falling edge provided Trigger is high 110 Timer disabled on Trigger falling edge 111 Reserved
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 TIMENA	Timer Enable Configures the condition that causes the Timer to be enabled and start decrementing. 000 Timer always enabled 001 Timer enabled on Timer N-1 enable 010 Timer enabled on Trigger high 011 Timer enabled on Trigger high and Pin high 100 Timer enabled on Pin rising edge 101 Timer enabled on Pin rising edge and Trigger high 110 Timer enabled on Trigger rising edge 111 Timer enabled on Trigger rising or falling edge
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

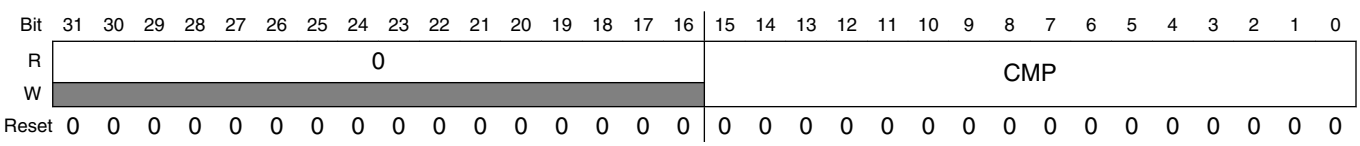
FLEXIO_TIMCFG*n* field descriptions (continued)

Field	Description
5–4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters will output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00 Stop bit disabled 01 Stop bit is enabled on timer compare 10 Stop bit is enabled on timer disable 11 Stop bit is enabled on timer compare and timer disable
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters will output the contents of the start bit when the timer is enabled and the timer counter will reload from the compare register on the first rising edge of the shift clock. 0 Start bit disabled 1 Start bit enabled
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

49.3.20 Timer Compare N Register (FLEXIO_TIMCMP*n*)

.

Address: 4005_A000h base + 500h offset + (4d × i), where i=0d to 3d



FLEXIO_TIMCMP*n* field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset and when the timer decrements down to zero. In dual 8-bit counters baud/bit mode, the lower 8-bits configures the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8-bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2. In dual 8-bit counters PWM mode, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or

Table continues on the next page...

FLEXIO_TIMCMP*n* field descriptions (continued)

Field	Description
	trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2.

49.4 Functional description

49.4.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FlexIO. The timing of shift, load and store events are controlled by the Timer assigned to the Shifter via the SHIFTCTL[TIMSEL] register. The Shifters are designed to support either DMA, interrupt or polled operation. The following block diagram provides a detailed view of the Shifter microarchitecture.

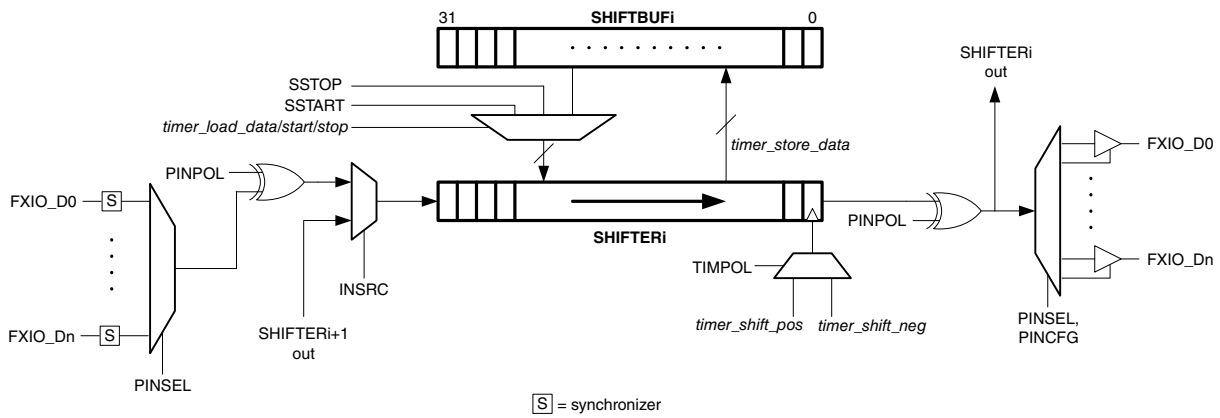


Figure 49-2. Shifter Microarchitecture

49.4.1.1 Transmit Mode

When configured for Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter will load data from the SHIFTBUF register and shift data out when a load event is signalled by the assigned Timer. An optional start/stop bit can also be automatically loaded before/after SHIFTBUF data by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Note that the shifter will immediately load a stop bit when the Shifter is initially configured for Transmit mode if a stop bit is enabled.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been loaded from the SHIFTBUF register into the Shifter or when the Shifter is initially configured into Transmit mode. The flag will clear when new data has been written into the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). The flag can be cleared by writing it with logic 1.

49.4.1.2 Receive Mode

When configured for Receive mode (SHIFTCTL[SMOD]=Receive), the shifter will shift data in and store data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

49.4.1.3 Match Store Mode

When configured for Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter will shift data in, check for a match result and store matched data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs and matched data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the matched data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

49.4.1.4 Match Continuous Mode

When configured for Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter will shift data in and continuously check for a match result whenever a shift event is signalled by the assigned Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs. The flag will clear automatically as soon as there is no longer a match between Shifter data and SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when a match occurs. The flag will clear when there is a read from the SHIFTBUF register or it written with logic 1.

49.4.2 Timer operation

The FlexIO 16-bit timers control the loading, shifting and storing of the shift registers, the counters load the contents of the compare register and decrement down to zero on the FlexIO clock. They can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to enable in response to a trigger, pin or shifter condition; decrement always or only on a trigger or pin edge; reset in response to a trigger or pin condition; and disable on a trigger or pin condition or on a timer compare. Timers can optionally include a start condition and/or stop condition.

Each timer operates independently, although a timer can be configured to enable or disable at the same time as the previous timer (eg: timer1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input or an external trigger input (refer to the chip configuration section for details on the external trigger connections). The trigger configuration is separate from the pin configuration, which can be configured for input, output data or output enable.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD). Once the TIMOD is configured for the desired mode, when the condition configured by timer enable (TIMENA) is detected then the following events occur.

- Timer counter will load the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output will set depending on the TIMOUT configuration.
- Transmit shifters controlled by this timer will either output their start bit value, or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

The Timer will then generate the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC=00 or 01) or equal to the decrement clock (when TIMDEC=10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer will decrement on both rising and falling edges.

When the Timer is configured to reset as configured in the TIMRST field then the Timer counter will load the current value of the Compare Register again, the timer output may also be affected by the reset as configured in TIMOUT.

If the Timer start bit is enabled, the timer counter will reload with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift will not load correctly.

When configured for 8-bit counter mode, whenever the lower 8-bit counter decrements to zero the timer output will toggle, the lower 8-bit counter register will reload from the compare register and the upper 8-bit counter will decrement. For 8-bit PWM mode, the lower 8-bit counter will only decrement when the output is high and the upper 8-bit counter will only decrement when the output is low. The timer output will toggle whenever either lower or upper 8-bit counter decrements to zero.

When the timer decrements to zero, a compare event occurs depending on the timer mode. For 8-bit counter or PWM modes, both halves of the counter must equal zero and the upper half must decrement for the timer compare event to occur, while in 16-bit mode the entire counter must equal zero and decrement. The timer compare event will cause the timer status flag to set, the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load and any configured receive shift registers to store .

When the is Timer is configured to add a stop bit on each compare, the following additional events occur.

- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.
- On the first rising edge of the shifter clock after the compare, the timer counter will reload the current value of the Compare Register.

Transmit shifters must be configured to load on the first shift when the timer is configured to insert a stop bit on each compare.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur.

- Timer counter will reload the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output will clear.
- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter will continue decrementing until the next rising edge of the shift clock is detected, at which point it will finish. A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). Receive shift registers will stop bit enabled will store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge while the timer is in the stop state condition. If there is no configured edge between the timer disable and the next rising edge of the shift clock then the final store and verify do not occur.

49.4.3 Pin operation

The pin configuration for each timer and shifter can be configured to use any FlexIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion would cause logic zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter could be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

When configuring a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FlexIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FlexIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FlexIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

Note that FlexIO pins are also connected internally, configuring a FlexIO shifter or timer to output data on an unused pin will make an internal connection that allows other shifters and timer to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FlexIO clock and therefore incurs a 1 cycle latency.

So when using a Pin input as a Timer Trigger, Timer Clock or Shifter Data Input, the following synchronization delays occur:

1. 0.5 – 1.5 FlexIO clock cycles for external pin
2. 1 FlexIO clock cycle for an internally driven pin

For timing considerations such as output valid time and input setup time for specific applications (SPI Master, SPI Slave, I2C Master, I2S Master, I2S Slave) please refer to the FlexIO Application Information Section.

49.5 Application Information

This section provides examples for a variety of FlexIO module applications.

49.5.1 UART Transmit

UART transmit can be supported using one Timer, one Shifter and one Pin (two Pins if supporting CTS). The start and stop bit insertion is handled automatically and multiple transfers can be supported using DMA controller. The timer status flag can be used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention, before transmitting a break or idle character the SSTART and SSTOP fields should be altered to transmit the required state and the data to transmit must equal 0xFF or 0x00. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the

number of bits to transmit). Note that when performing byte writes to SHIFTBUFn (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered allowing an address mark bit or additional stop bit to remain undisturbed.

FlexIO does not support automatic insertion of parity bits.

Table 49-3. UART Transmit Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open drain by setting PINPOL=0x1 and PINCFG=0x1.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_2222	Configure start bit, stop bit, enable on trigger low and disable on compare. Can support CTS by configuring TIMEN=0x3.
TIMCTLn	0x01C0_0001	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Can support CTS by configuring PINSEL=0x1 (for Pin 1) and PINPOL=0x1.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

49.5.2 UART Receive

UART receive can be supported using one Timer, one Shifter and one Pin (two Timers and two Pins if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers can be supported using the DMA controller. The timer status flag can be used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FlexIO, data is sampled only once in the middle of each bit. Another timer can be used to implement a glitch filter on the incoming data, another Timer can also be used to detect an idle line of programmable length. Break characters will cause the error flag to set and the shifter buffer register will return 0x00.

FlexIO does not support automatic verification of parity bits.

Table 49-4. UART Receiver Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2422	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x0000_0081	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

The UART Receiver with RTS configuration uses a 2nd Timer to generate the RTS output. The RTS will assert when the start bit is detected and negate when the data is read from the shifter buffer register. No start bit will be detected while the RTS is asserted, the received data is simply ignored.

Table 49-5. UART Receiver with RTS Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2522	Configure start bit, stop bit, enable on pin posedge with trigger low and disable on compare. Enable resynchronization

Table continues on the next page...

Table 49-5. UART Receiver with RTS Configuration (continued)

Register	Value	Comments
		to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x03C0_0081	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0030_6100	Enable on Timer N enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(n+1)	0x0143_0083	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

49.5.3 SPI Master

SPI master mode can be supported using two Timers, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of 1 clock cycle between the slave select negating and before the next transfer. Writing to the transmit buffer by either core or DMA is used to initiate each transfer.

Due to synchronization delays, the setup time for the serial input data is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 49-6. SPI Master (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.

Table continues on the next page...

Table 49-6. SPI Master (CPHA=0) Configuration (continued)

Register	Value	Comments
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0. Set PINPOL to invert the output shift clock.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 49-7. SPI Master (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0021	Start bit loads data on first shift.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1.

Table continues on the next page...

Table 49-7. SPI Master (CPHA=1) Configuration (continued)

Register	Value	Comments
		Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep slave select asserted for as long as there is data in the transmit buffer.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

49.5.4 SPI Slave

SPI slave mode can be supported using one Timer, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external slave select asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

Table 49-8. SPI Slave (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6000	Configure enable on trigger rising edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 49-9. SPI Slave (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6602	Configure start bit, enable on trigger rising edge, disable on trigger falling edge, initial clock state is logic 0 and decrement on pin input.

Table continues on the next page...

Table 49-9. SPI Slave (CPHA=1) Configuration (continued)

Register	Value	Comments
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

49.5.5 I2C Master

I2C master mode can be supported using two Timers, two Shifters and two Pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word, when receiving the transmitter must transmit 0xFF to tristate the output. FlexIO inserts a stop bit after every word to generate/verify the ACK/NACK. FlexIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START/ STOP), so the compare register needs to be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin equal to output (although this increases both the clock high and clock low periods by at least 1 FlexIO clock cycle each). The second timer uses the SCL input pin to control the transmit/receive shift registers, this enforces an SDA data hold time by an extra 2 FlexIO clock cycles.

Both the transmit and receive shifters need to be serviced for each word in the transfer, the transmit shifter must transmit 0xFF when receiving and the receive shifter returns the data actually present on the SDA pin. The transmit shifter will load 1 additional word on the last falling edge of SCL pin, this word should be 0x00 if generating a STOP condition or 0xFF if generating a repeated START condition. During the last word of a master-receiver transfer, the transmit SSTOP bit should be set by software to generate a NACK.

The receive shift register will assert an error interrupt if a NACK is detected, but software is responsible for generating the STOP or repeated START condition. If a NACK is detected during master-transmit, the interrupt routine should immediately write the transmit shifter register with 0x00 (if generating STOP) or 0xFF (if generating repeated START). Software should then wait for the next rising edge on SCL and then disable both timers. The transmit shifter should then be disabled after waiting the setup delay for a repeated START or STOP condition.

Due to synchronization delays, the data valid time for the transmit output is 2 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The I2C master data valid is delayed 2 cycles because the clock output is passed through a synchronizer before clocking the transmit/receive shifter (to guarantee some SDA hold time). Since the SCL output is synchronous with FlexIO clock, the synchronization delay is 1 cycle and then 1 cycle to generate the output.

Table 49-10. I2C Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTLn	0x0101_0082	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open drain output) on Pin 0.
SHIFTCFG(n+1)	0x0000_0020	Start bit disabled and stop bit enabled (logic 0) for ACK/NACK detection.
SHIFTCTL(n+1)	0x0180_0001	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMPn	0x0000_2501	Configure 2 word transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0102_2222	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTLn	0x01C1_0101	Configure dual 8-bit counter using Pin 1 output enable (SCL open drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_000F	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_1112	Enable when Timer 0 is enabled, disable when Timer 0 is disabled, enable start bit and stop bit at end of each word, decrement on pin input.
TIMCTL(n+1)	0x01C0_0183	Configure 16-bit counter using inverted Pin 1 input (SCL).

Table continues on the next page...

Table 49-10. I2C Master Configuration (continued)

Register	Value	Comments
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0], use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

49.5.6 I2S Master

I2S master mode can be supported using two Timers, two Shifters and four Pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FlexIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FlexIO clock frequency, and the initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure the frame sync is generated one clock cycle before the first output data.

Due to synchronization delays, the setup time for the receiver input is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 49-11. I2S Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Load transmit data on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.

Table continues on the next page...

Table 49-11. I2S Master Configuration (continued)

Register	Value	Comments
TIMCFGn	0x0000_0202	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0x0000_007F	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(n+1)	0x0000_0100	Enable when Timer 0 is enabled and never disable.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter using inverted Pin 3 output (as frame sync).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

49.5.7 I2S Slave

I2S slave mode can be supported using two Timers, two Shifters and four Pins (for single transmit and single receive, other combinations of transmit and receive are possible).

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

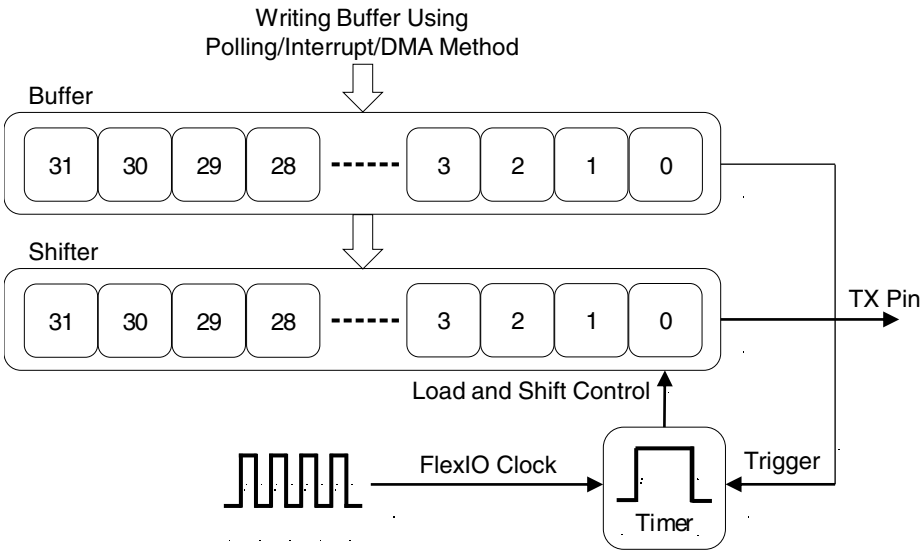
The output valid time of I2S slave is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization plus 1 cycle to output the data

Table 49-12. I2S Slave Configuration

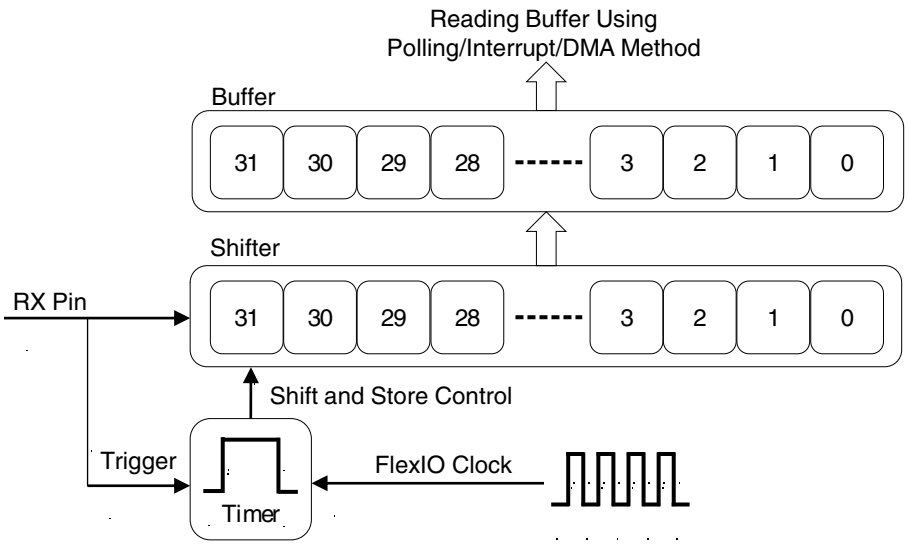
Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0103_0002	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0180_0101	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_007D	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 3.
TIMCFGn	0x0030_2400	Configure enable on pin rising edge (inverted frame sync) and disable on compare, initial clock state is logic 1 and decrement on trigger input (bit clock).
TIMCTLn	0x0440_0383	Configure 16-bit counter using inverted Pin 3 input (frame sync), with Pin 2 input (bit clock) as the trigger.
TIMCMP(n+1)	0x0000_003F	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_3500	Configure enable on pin rising edge with trigger high and disable on compare with trigger low, initial clock state is logic 0 and decrement on pin input.
TIMCTL(n+1)	0x0340_0203	Configure 16-bit counter using Pin 2 input (bit clock), with Timer 0 output as the trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

49.6 Usage Guide

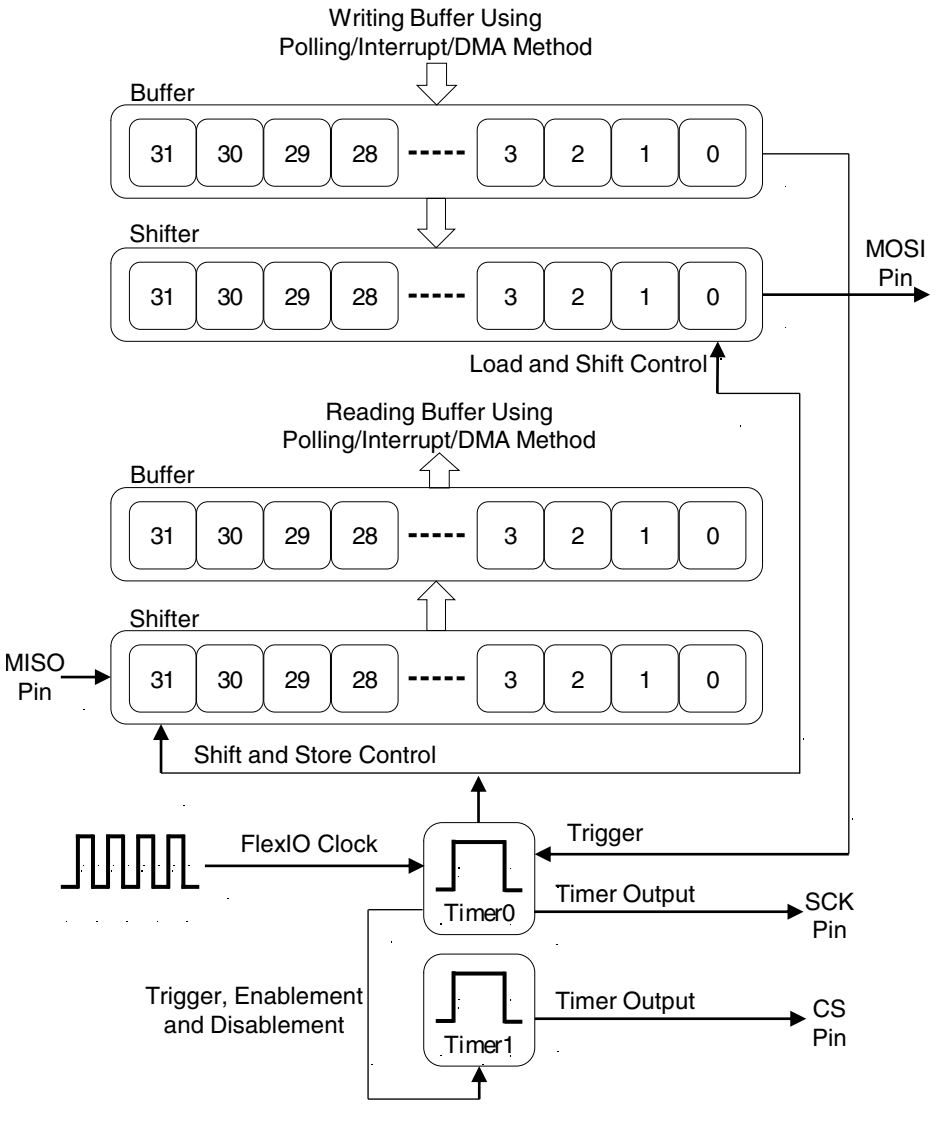
UART Transmit



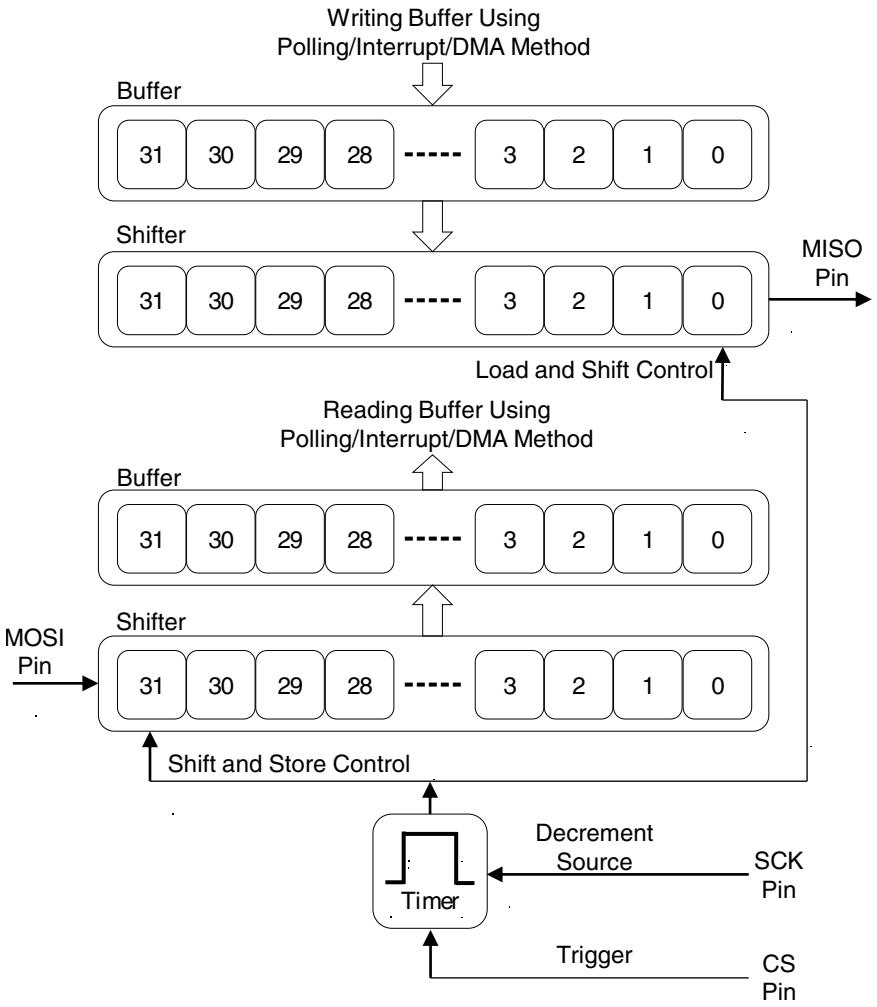
UART Receive



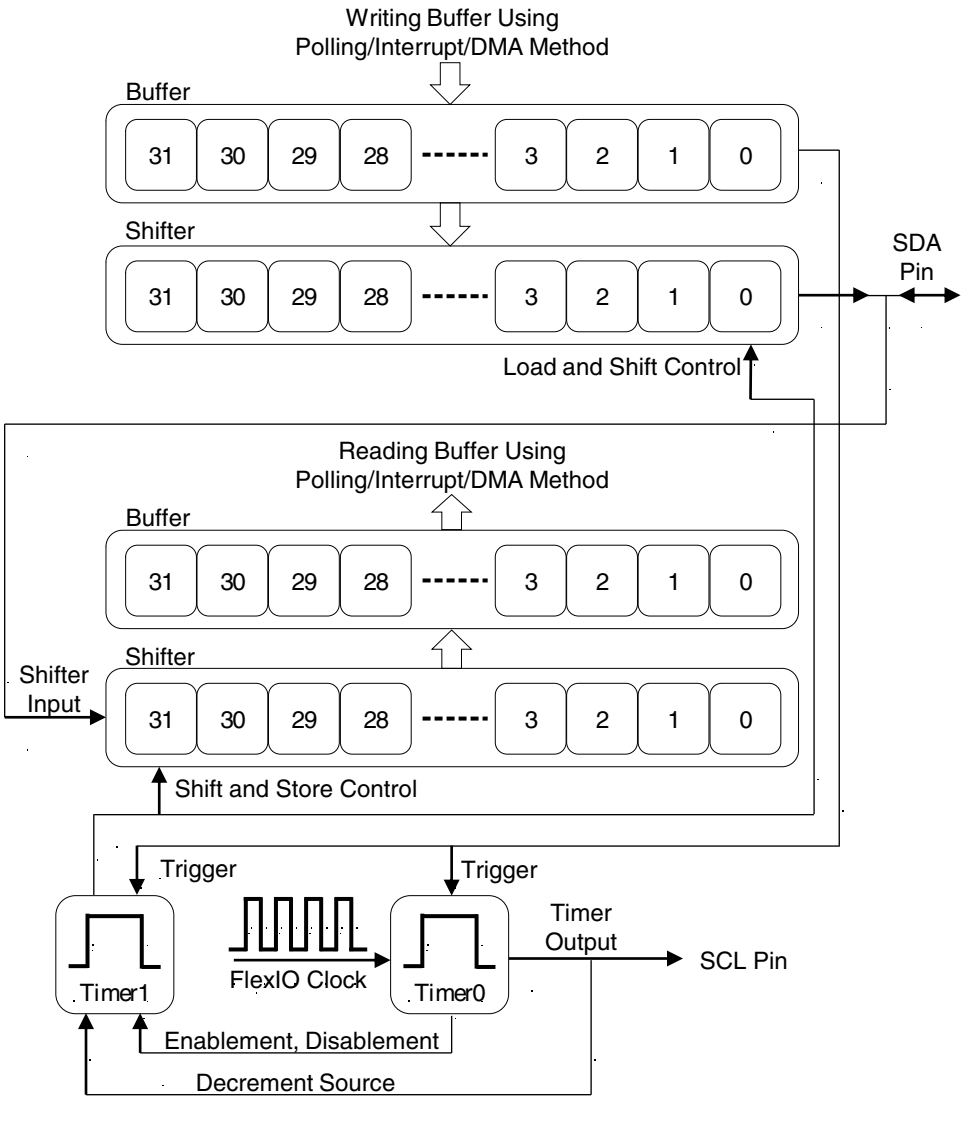
SPI Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
MOSI
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
MISO
Pin
Shift and Store Control
FlexIO Clock
Trigger
Timer0
Timer Output
SCK
Pin
Trigger, Enablement
and Disablement
Timer1
CS
Pin

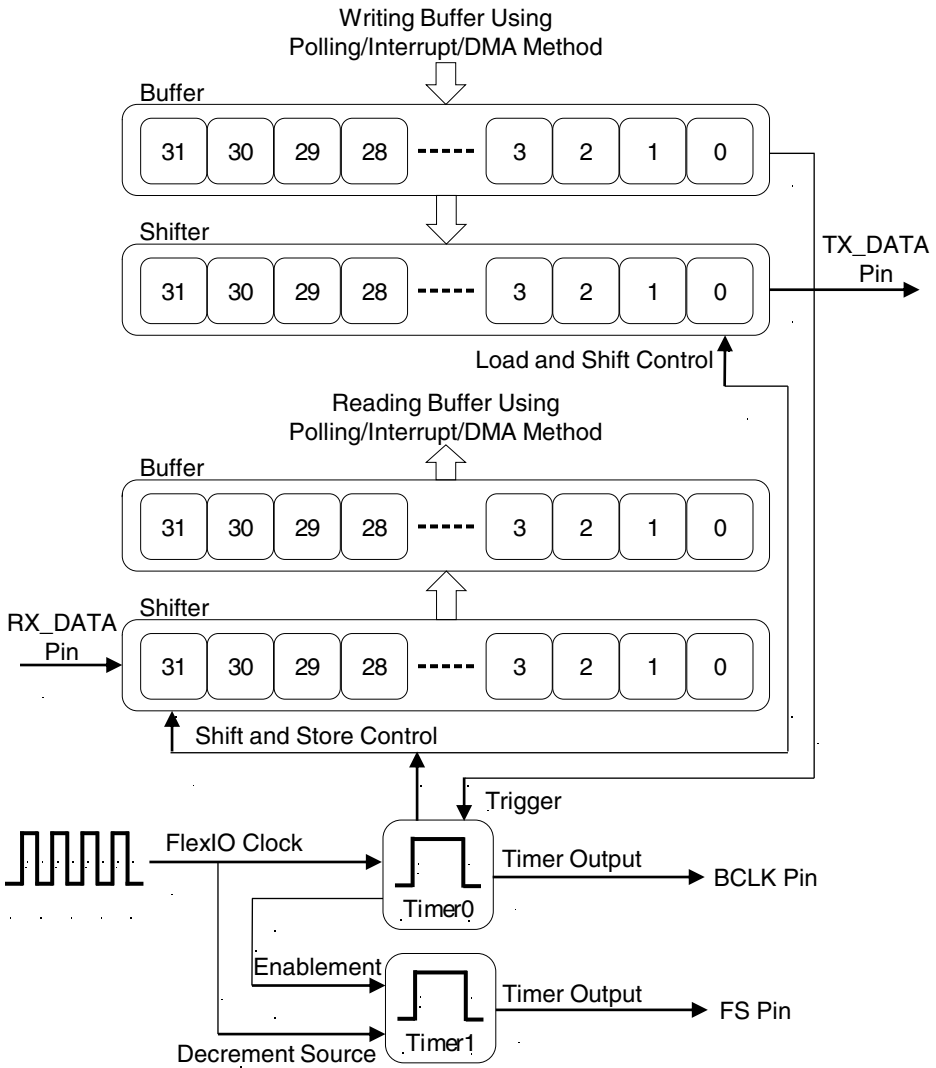
SPI Slave


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
MISO
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
MOSI
Pin
Shift and Store Control
Decrement
Source
SCK
Pin
Timer
Trigger
CS
Pin

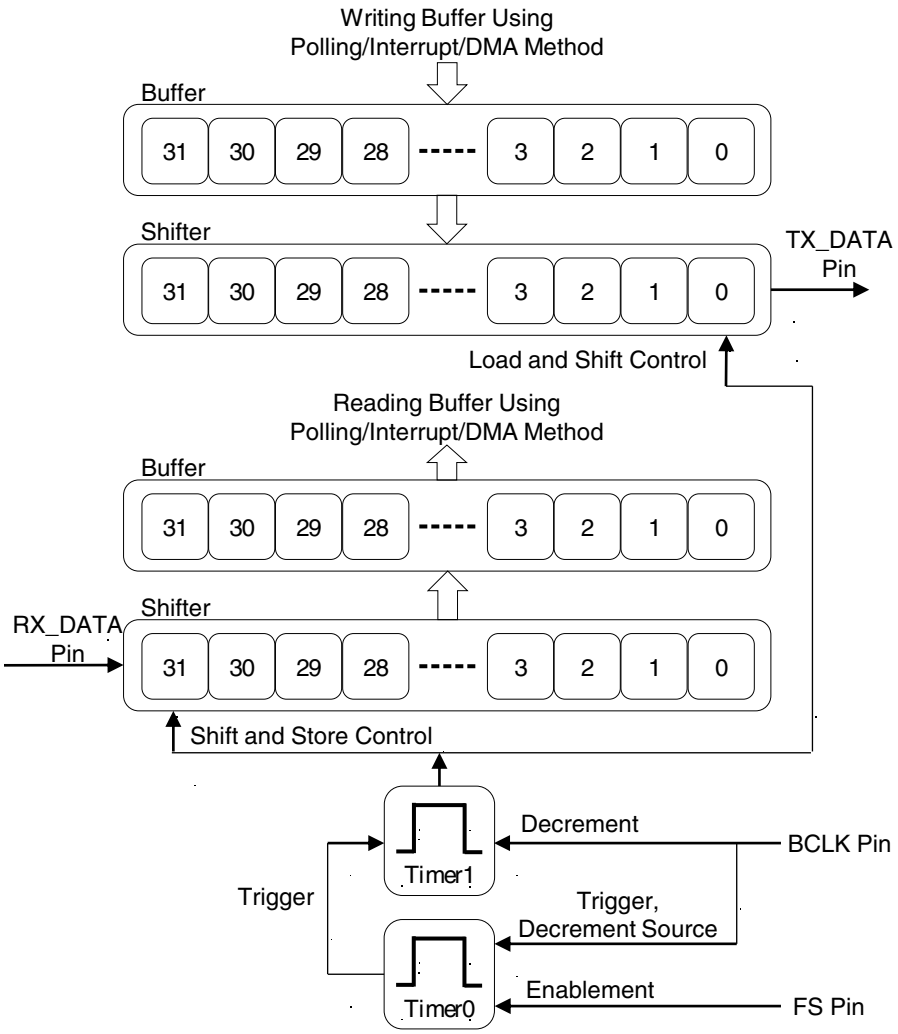
I2C Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
SDA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Shifter
Input
Shift and Store Control
Trigger
Trigger
Timer
Output
SCL Pin
Timer1
FlexIO Clock
Timer0
Enablement, Disablement
Decrement Source

I2S Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
TX_DATA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
RX_DATA
Pin
Shift and Store Control
Trigger
FlexIO Clock
Timer Output
BCLK Pin
Timer0
Enablement
FS Pin
Timer1
Decrement Source

I2S Slave


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
TX_DATA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
RX_DATA
Pin
Shift and Store Control
Decrement
BCLK Pin
Timer1
Trigger
Trigger,
Decrement Source
Timer0
Enablement
FS Pin

Chapter 50
CAN (FlexCAN)

50.1 Chip-specific information for this module

50.1.1 Instantiation Information

This device contains upto 2 FlexCAN modules.

Table 50-1. Number of message buffers

Module	Number of Message Buffers (MB)	CAN-FD feature
FlexCAN0	16 MBs	No
FlexCAN1	16 MBs	No

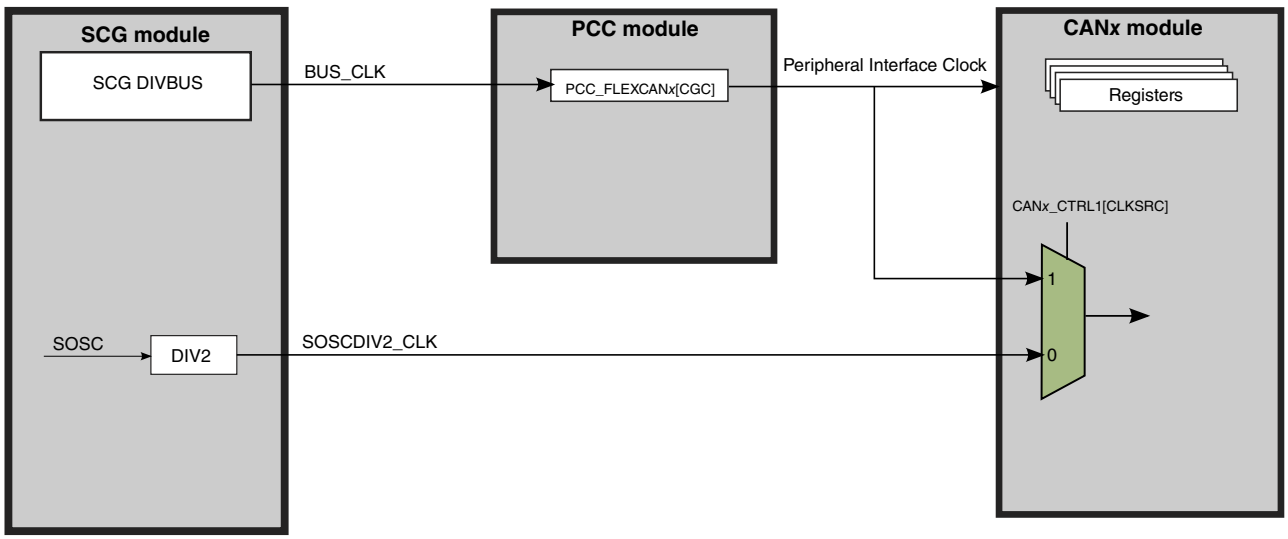
50.1.1.1 Reset value of MDIS bit

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

50.1.2 FlexCAN Clocking Information

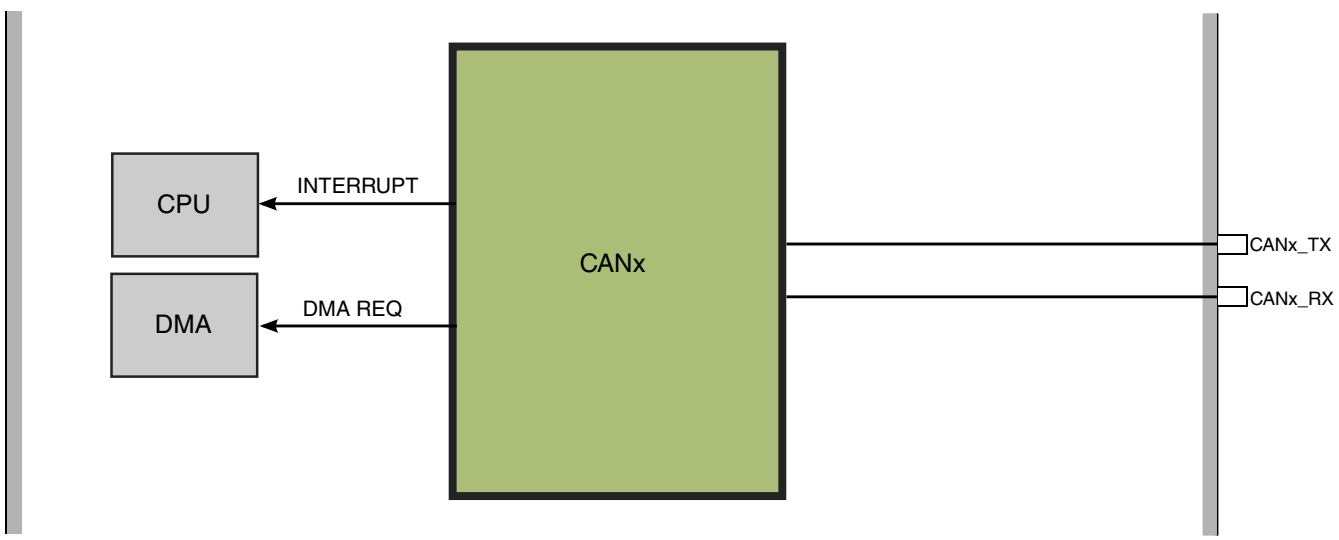
The following figure shows the input clock sources available for this module.

Peripheral Clocking - FlexCAN



50.1.3 Inter-connectivity Information

The FlexCAN inter-connectivity is shown in the following diagram.



50.2 Introduction

The FlexCAN module is a communication controller implementing the CAN protocol according to the ISO 11898-1 standard and CAN 2.0 B protocol specifications. A general block diagram is shown in the following figure, which describes the main subblocks

implemented in the FlexCAN module, including one associated memory for storing message buffers, Receive Global Mask registers, Receive Individual Mask registers, Receive FIFO filters, and Receive FIFO ID filters. The functions of the submodules are described in subsequent sections.

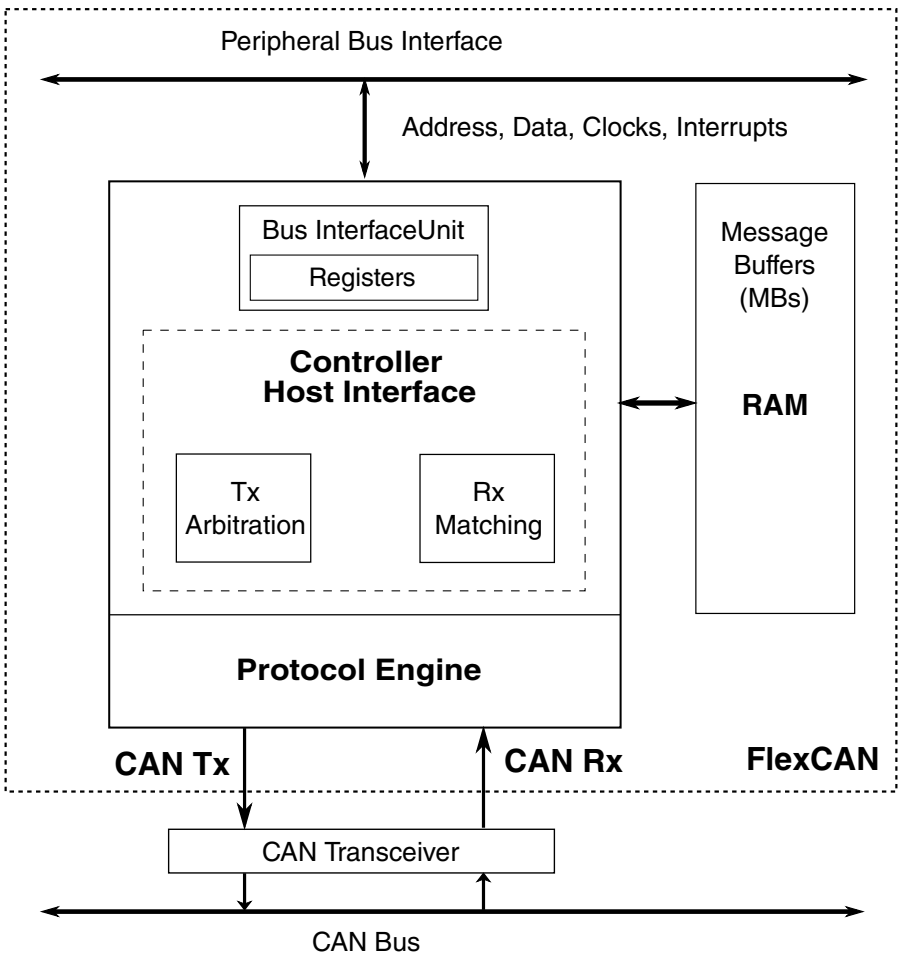


Figure 50-1. FlexCAN block diagram

50.2.1 Overview

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific requirements of this field:

- Real-time processing
- Reliable operation in the EMI environment of a vehicle
- Cost-effectiveness
- Required bandwidth

The FlexCAN module is a full implementation of the CAN protocol specification, the CAN 2.0 version B protocol, which supports both standard and extended message frames. The message buffers are stored in an embedded RAM dedicated to the FlexCAN module. See the chip configuration details for the actual number of message buffers configured in the chip.

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- Requesting RAM access for receiving and transmitting message frames
- Validating received messages
- Performing error handling

The Controller Host Interface (CHI) sub-module handles message buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms.

The Bus Interface Unit (BIU) sub-module controls the access to and from the internal interface bus, in order to establish connection to the CPU and to other blocks. Clocks, address and data buses, interrupt outputs, DMA and test signals are accessed through the BIU.

50.2.2 FlexCAN module features

The FlexCAN module includes these distinctive features:

- Full implementation of the CAN protocol specification, Version 2.0 B
    - Standard data frames
    - Extended data frames
    - Zero to eight bytes data length
    - Programmable bit rate up to 1 Mb/sec
    - Content-related addressing
- Compliant with the ISO 11898-1 standard
- Flexible mailboxes of zero to eight bytes data length
- Each mailbox configurable as receive or transmit, all supporting standard and extended messages
- Individual Rx Mask registers per mailbox

- Full-featured Rx FIFO with storage capacity for up to six frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Flexible message buffers (MBs), totaling 16 message buffers of 8 bytes data length each, configurable as Rx or Tx
- Programmable clock source to the CAN Protocol Interface, either peripheral clock or oscillator clock
- RAM not used by reception or transmission structures can be used as general purpose RAM space
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Time stamp based on 16-bit free-running timer, with an optional external time tick
- Global network time, synchronized by a specific message
- Maskable interrupts
- Independence from the transmission medium (an external transceiver is assumed)
- Short latency time due to an arbitration scheme for high-priority messages
- Low power modes, with programmable wake up on bus activity
- Remote request frames may be handled automatically or by software
- CAN bit time settings and configuration bits can only be written in Freeze mode
- Tx mailbox status (Lowest priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- SYNCH bit available in Error in Status 1 register to inform that the module is synchronous with CAN bus
- CRC status for transmitted message
- Rx FIFO Global Mask register
- Selectable priority between mailboxes and Rx FIFO during matching process

- Powerful Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard, or 512 partial (8 bit) IDs, with up to 32 individual masking capability
- 100% backward compatibility with previous FlexCAN version

50.2.3 Modes of operation

The FlexCAN module has these functional modes:

- Normal mode (User or Supervisor):

  In Normal mode, the module operates receiving and/or transmitting message frames, errors are handled normally, and all CAN Protocol functions are enabled. User and Supervisor Modes differ in the access to some restricted control registers.

- Freeze mode:

  Freeze mode is enabled when the FRZ bit in MCR is asserted. If enabled, Freeze mode is entered when MCR[HALT] is set or when Debug mode is requested at chip level and MCR[FRZ_ACK] is asserted by the FlexCAN. In this mode, no transmission or reception of frames is done and synchronicity to the CAN bus is lost. See Freeze mode for more information.

- Loop-Back mode:

  The module enters this mode when the LPB field in the Control 1 Register is asserted. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is internally fed back to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic '1'). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- Listen-Only mode:

  The module enters this mode when the LOM field in the Control 1 Register is asserted. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.

For low-power operation, the FlexCAN module has:

- Module Disable mode:

  This low-power mode is entered when the MDIS bit in the MCR Register is asserted by the CPU and the LPM_ACK is asserted by the FlexCAN. When disabled, the module requests to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Exit from this mode is done by negating the MDIS bit in the MCR register. See Module Disable mode for more information.

- Stop mode:

  This low power mode is entered when Stop mode is requested at chip level and the LPM_ACK bit in the MCR Register is asserted by the FlexCAN. When in Stop Mode, the module puts itself in an inactive state and then informs the CPU that the clocks can be shut down globally. Exit from this mode happens when the Stop mode request is removed, or when activity is detected on the CAN bus and the Self Wake Up mechanism is enabled. See Stop mode for more information.

50.3 FlexCAN signal descriptions

The FlexCAN module has two I/O signals connected to the external chip pins. These signals are summarized in the following table and described in more detail in the next subsections.

Table 50-2. FlexCAN signal descriptions

Signal	Description	I/O
CAN Rx	CAN Receive Pin	Input
CAN Tx	CAN Transmit Pin	Output

50.3.1 CAN Rx

This pin is the receive pin from the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

50.3.2 CAN Tx

This pin is the transmit pin to the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

50.4 Memory map/register definition

This section describes the registers and data structures in the FlexCAN module. The base address of the module depends on the particular memory map of the chip.

50.4.1 FlexCAN memory mapping

The memory map for the FlexCAN module is shown in the following table.

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0x0080.

Each individual register is identified by its complete name and the corresponding mnemonic. The access type can be Supervisor (S) or Unrestricted (U). Most of the registers can be configured to have either Supervisor or Unrestricted access by programming the SUPV field in the MCR register. These registers are identified as S/U in the Access column of Table 50-3.

Table 50-3. Register access and reset information

Register	Access type	Affected by hard reset	Affected by soft reset
Module Configuration Register (CAN_MCR)	S	Yes	Yes
Control 1 register (CAN_CTRL1)	S/U	Yes	No
Free Running Timer register (CAN_TIMER)	S/U	Yes	Yes
Rx Mailboxes Global Mask register (CAN_RXMGMASK)	S/U	No	No
Rx Buffer 14 Mask register (CAN_RX14MASK)	S/U	No	No
Rx Buffer 15 Mask register (CAN_RX15MASK)	S/U	No	No
Error Counter Register (CAN_ECR)	S/U	Yes	Yes
Error and Status 1 Register (CAN_ESR1)	S/U	Yes	Yes
Interrupt Masks 1 register (CAN_IMASK1)	S/U	Yes	Yes
Interrupt Flags 1 register (CAN_IFLAG1)	S/U	Yes	Yes
Control 2 Register (CAN_CTRL2)	S/U	Yes	No
Error and Status 2 Register (CAN_ESR2)	S/U	Yes	Yes
CRC Register (CAN_CRCR)	S/U	Yes	Yes
Rx FIFO Global Mask register (CAN_RXFGMASK)	S/U	No	No
Rx FIFO Information Register (CAN_RXFIR)	S/U	No	No
CAN Bit Timing Register (CAN_CBT)	S/U	Yes	No
Message buffers	S/U	No	No
Rx Individual Mask Registers	S/U	No	No

The FlexCAN module can store CAN messages for transmission and reception using mailboxes and Rx FIFO structures.

The table below shows the FlexCAN memory map.

The address range from offset 0x80 to 0x17F allocates the sixteen 128-bit Message Buffers (MBs).

CAN memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4000	Module Configuration Register (CAN0_MCR)	32	R/W	See section	50.4.2/1392
4002_4004	Control 1 register (CAN0_CTRL1)	32	R/W	0000_0000h	50.4.3/1397
4002_4008	Free Running Timer (CAN0_TIMER)	32	R/W	0000_0000h	50.4.4/1401
4002_4010	Rx Mailboxes Global Mask Register (CAN0_RXMGMASK)	32	R/W	Undefined	50.4.5/1402
4002_4014	Rx 14 Mask register (CAN0_RX14MASK)	32	R/W	Undefined	50.4.6/1403
4002_4018	Rx 15 Mask register (CAN0_RX15MASK)	32	R/W	Undefined	50.4.7/1403
4002_401C	Error Counter (CAN0_ECR)	32	R/W	0000_0000h	50.4.8/1404
4002_4020	Error and Status 1 register (CAN0_ESR1)	32	R/W	See section	50.4.9/1406
4002_4028	Interrupt Masks 1 register (CAN0_IMASK1)	32	R/W	0000_0000h	50.4.10/ 1412
4002_4030	Interrupt Flags 1 register (CAN0_IFLAG1)	32	R/W	0000_0000h	50.4.11/ 1412
4002_4034	Control 2 register (CAN0_CTRL2)	32	R/W	See section	50.4.12/ 1415
4002_4038	Error and Status 2 register (CAN0_ESR2)	32	R/W	0000_0000h	50.4.13/ 1419
4002_4044	CRC Register (CAN0_CRCR)	32	R	0000_0000h	50.4.14/ 1420
4002_4048	Rx FIFO Global Mask register (CAN0_RXFGMASK)	32	R/W	Undefined	50.4.15/ 1421
4002_404C	Rx FIFO Information Register (CAN0_RXFIR)	32	R	Undefined	50.4.16/ 1422
4002_4050	CAN Bit Timing Register (CAN0_CBT)	32	R/W	See section	50.4.17/ 1423
4002_4880	Rx Individual Mask Registers (CAN0_RXIMR0)	32	R/W	Undefined	50.4.18/ 1424
4002_4884	Rx Individual Mask Registers (CAN0_RXIMR1)	32	R/W	Undefined	50.4.18/ 1424
4002_4888	Rx Individual Mask Registers (CAN0_RXIMR2)	32	R/W	Undefined	50.4.18/ 1424
4002_488C	Rx Individual Mask Registers (CAN0_RXIMR3)	32	R/W	Undefined	50.4.18/ 1424

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4890	Rx Individual Mask Registers (CAN0_RXIMR4)	32	R/W	Undefined	50.4.18/ 1424
4002_4894	Rx Individual Mask Registers (CAN0_RXIMR5)	32	R/W	Undefined	50.4.18/ 1424
4002_4898	Rx Individual Mask Registers (CAN0_RXIMR6)	32	R/W	Undefined	50.4.18/ 1424
4002_489C	Rx Individual Mask Registers (CAN0_RXIMR7)	32	R/W	Undefined	50.4.18/ 1424
4002_48A0	Rx Individual Mask Registers (CAN0_RXIMR8)	32	R/W	Undefined	50.4.18/ 1424
4002_48A4	Rx Individual Mask Registers (CAN0_RXIMR9)	32	R/W	Undefined	50.4.18/ 1424
4002_48A8	Rx Individual Mask Registers (CAN0_RXIMR10)	32	R/W	Undefined	50.4.18/ 1424
4002_48AC	Rx Individual Mask Registers (CAN0_RXIMR11)	32	R/W	Undefined	50.4.18/ 1424
4002_48B0	Rx Individual Mask Registers (CAN0_RXIMR12)	32	R/W	Undefined	50.4.18/ 1424
4002_48B4	Rx Individual Mask Registers (CAN0_RXIMR13)	32	R/W	Undefined	50.4.18/ 1424
4002_48B8	Rx Individual Mask Registers (CAN0_RXIMR14)	32	R/W	Undefined	50.4.18/ 1424
4002_48BC	Rx Individual Mask Registers (CAN0_RXIMR15)	32	R/W	Undefined	50.4.18/ 1424
4002_5000	Module Configuration Register (CAN1_MCR)	32	R/W	See section	50.4.2/1392
4002_5004	Control 1 register (CAN1_CTRL1)	32	R/W	0000_0000h	50.4.3/1397
4002_5008	Free Running Timer (CAN1_TIMER)	32	R/W	0000_0000h	50.4.4/1401
4002_5010	Rx Mailboxes Global Mask Register (CAN1_RXMGMASK)	32	R/W	Undefined	50.4.5/1402
4002_5014	Rx 14 Mask register (CAN1_RX14MASK)	32	R/W	Undefined	50.4.6/1403
4002_5018	Rx 15 Mask register (CAN1_RX15MASK)	32	R/W	Undefined	50.4.7/1403
4002_501C	Error Counter (CAN1_ECR)	32	R/W	0000_0000h	50.4.8/1404
4002_5020	Error and Status 1 register (CAN1_ESR1)	32	R/W	See section	50.4.9/1406
4002_5028	Interrupt Masks 1 register (CAN1_IMASK1)	32	R/W	0000_0000h	50.4.10/ 1412
4002_5030	Interrupt Flags 1 register (CAN1_IFLAG1)	32	R/W	0000_0000h	50.4.11/ 1412
4002_5034	Control 2 register (CAN1_CTRL2)	32	R/W	See section	50.4.12/ 1415
4002_5038	Error and Status 2 register (CAN1_ESR2)	32	R/W	0000_0000h	50.4.13/ 1419
4002_5044	CRC Register (CAN1_CRCR)	32	R	0000_0000h	50.4.14/ 1420

Table continues on the next page...

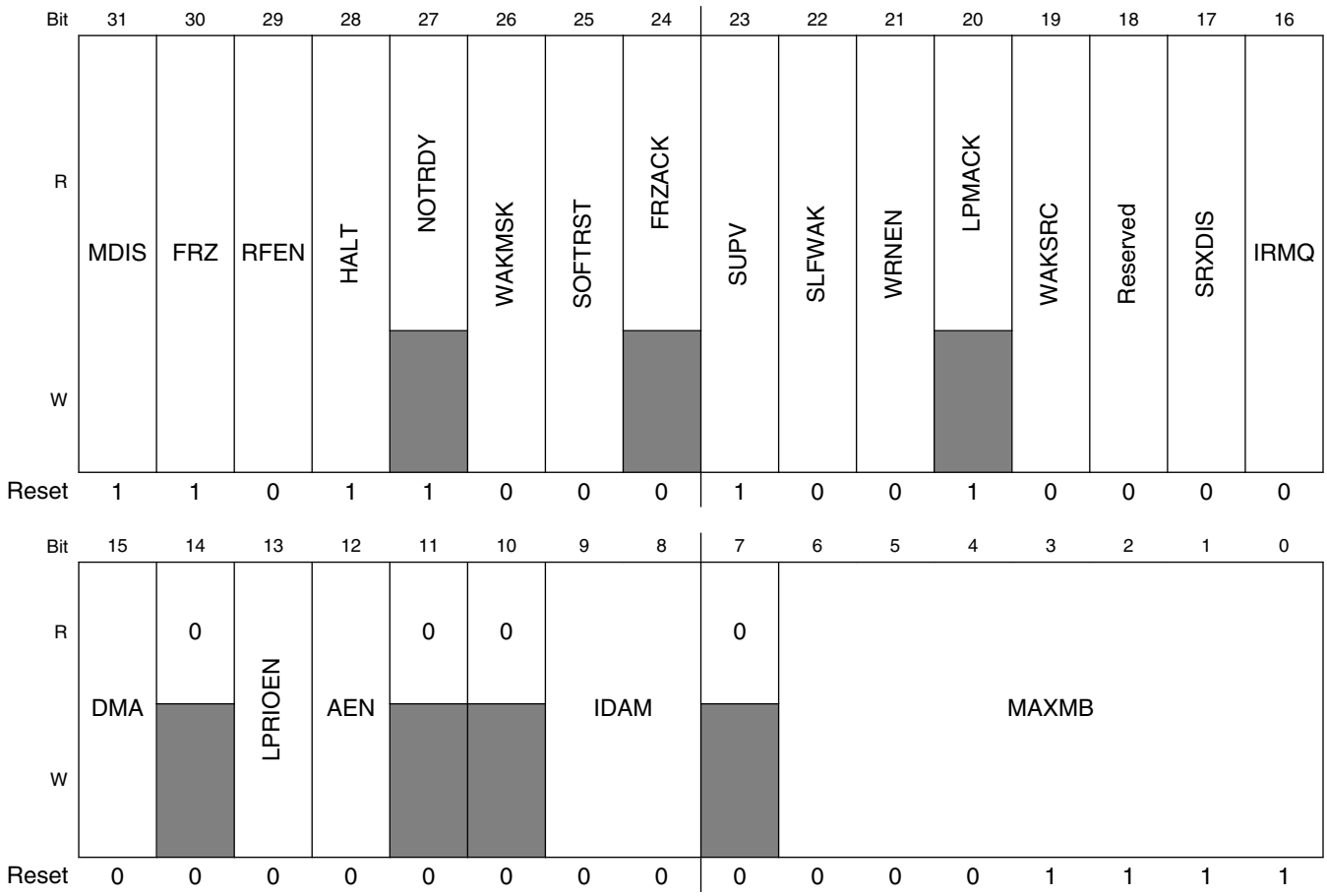
CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_5048	Rx FIFO Global Mask register (CAN1_RXFGMASK)	32	R/W	Undefined	50.4.15/ 1421
4002_504C	Rx FIFO Information Register (CAN1_RXFIR)	32	R	Undefined	50.4.16/ 1422
4002_5050	CAN Bit Timing Register (CAN1_CBT)	32	R/W	See section	50.4.17/ 1423
4002_5880	Rx Individual Mask Registers (CAN1_RXIMR0)	32	R/W	Undefined	50.4.18/ 1424
4002_5884	Rx Individual Mask Registers (CAN1_RXIMR1)	32	R/W	Undefined	50.4.18/ 1424
4002_5888	Rx Individual Mask Registers (CAN1_RXIMR2)	32	R/W	Undefined	50.4.18/ 1424
4002_588C	Rx Individual Mask Registers (CAN1_RXIMR3)	32	R/W	Undefined	50.4.18/ 1424
4002_5890	Rx Individual Mask Registers (CAN1_RXIMR4)	32	R/W	Undefined	50.4.18/ 1424
4002_5894	Rx Individual Mask Registers (CAN1_RXIMR5)	32	R/W	Undefined	50.4.18/ 1424
4002_5898	Rx Individual Mask Registers (CAN1_RXIMR6)	32	R/W	Undefined	50.4.18/ 1424
4002_589C	Rx Individual Mask Registers (CAN1_RXIMR7)	32	R/W	Undefined	50.4.18/ 1424
4002_58A0	Rx Individual Mask Registers (CAN1_RXIMR8)	32	R/W	Undefined	50.4.18/ 1424
4002_58A4	Rx Individual Mask Registers (CAN1_RXIMR9)	32	R/W	Undefined	50.4.18/ 1424
4002_58A8	Rx Individual Mask Registers (CAN1_RXIMR10)	32	R/W	Undefined	50.4.18/ 1424
4002_58AC	Rx Individual Mask Registers (CAN1_RXIMR11)	32	R/W	Undefined	50.4.18/ 1424
4002_58B0	Rx Individual Mask Registers (CAN1_RXIMR12)	32	R/W	Undefined	50.4.18/ 1424
4002_58B4	Rx Individual Mask Registers (CAN1_RXIMR13)	32	R/W	Undefined	50.4.18/ 1424
4002_58B8	Rx Individual Mask Registers (CAN1_RXIMR14)	32	R/W	Undefined	50.4.18/ 1424
4002_58BC	Rx Individual Mask Registers (CAN1_RXIMR15)	32	R/W	Undefined	50.4.18/ 1424

50.4.2 Module Configuration Register (CANx_MCR)

This register defines global system configurations, such as the module operation modes and the maximum message buffer configuration.

Address: Base address + 0h offset



CANx_MCR field descriptions

Field	Description
31 MDIS	Module Disable This bit controls whether FlexCAN is enabled or not. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface sub-modules. This bit is not affected by soft reset. 0 Enable the FlexCAN module. 1 Disable the FlexCAN module.
30 FRZ	Freeze Enable The FRZ bit specifies the FlexCAN behavior when the HALT bit in the CAN_MCR Register is set or when Debug mode is requested at chip level. When FRZ is asserted, FlexCAN is enabled to enter Freeze mode. Negation of this bit field causes FlexCAN to exit from Freeze mode.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 Not enabled to enter Freeze mode. 1 Enabled to enter Freeze mode.
29 RFEN	Rx FIFO Enable This bit controls whether the Rx FIFO feature is enabled or not. When RFEN is set, MBs 0 to 5 cannot be used for normal reception and transmission because the corresponding memory region (0x80-0xDC) is used by the FIFO engine as well as additional MBs (up to 32, depending on CAN_CTRL2[RFFN] setting) which are used as Rx FIFO ID Filter Table elements. RFEN also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in the table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (see Arbitration and matching timing). This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Rx FIFO not enabled. 1 Rx FIFO enabled.
28 HALT	Halt FlexCAN Assertion of this bit puts the FlexCAN module into Freeze mode. The CPU should clear it after initializing the Message Buffers and the Control Registers CAN_CTRL1 and CAN_CTRL2. No reception or transmission is performed by FlexCAN before this bit is cleared. Freeze mode cannot be entered while FlexCAN is in a low power mode. 0 No Freeze mode request. 1 Enters Freeze mode if the FRZ bit is asserted.
27 NOTRDY	FlexCAN Not Ready This read-only bit indicates that FlexCAN is either in Disable mode, Stop mode or Freeze mode. It is negated once FlexCAN has exited these modes. This bit is not affected by soft reset. 0 FlexCAN module is either in Normal mode, Listen-Only mode or Loop-Back mode. 1 FlexCAN module is either in Disable mode, Stop mode or Freeze mode.
26 WAKMSK	Wake Up Interrupt Mask This bit enables the Wake Up Interrupt generation under Self Wake Up mechanism. 0 Wake Up Interrupt is disabled. 1 Wake Up Interrupt is enabled.
25 SOFTRST	Soft Reset When this bit is asserted, FlexCAN resets its internal state machines and some of the memory mapped registers. The SOFTRST bit can be asserted directly by the CPU when it writes to the MCR Register. Because soft reset is synchronous and has to follow a request/acknowledge procedure across clock domains, it may take some time to fully propagate its effect. The SOFTRST bit remains asserted while reset is pending, and is automatically negated when reset completes. Therefore, software can poll this bit to know when the soft reset has completed. Soft reset cannot be applied while clocks are shut down in a low power mode. The module should be first removed from low power mode, and then soft reset can be applied. This bit is not affected by soft reset. 0 No reset request. 1 Resets the registers affected by soft reset.
24 FRZACK	Freeze Mode Acknowledge

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	This read-only bit indicates that FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore the software can poll the FRZACK bit to know when FlexCAN has actually entered Freeze mode. If Freeze Mode request is negated, then this bit is negated after the FlexCAN prescaler is running again. If Freeze mode is requested while FlexCAN is in a low power mode, then the FRZACK bit will be set only when the low-power mode is exited. See Section "Freeze Mode". This bit is not affected by soft reset. **NOTE:** FRZACK will be asserted within 178 CAN bits from the freeze mode request by the CPU, and negated within 2 CAN bits after the freeze mode request removal (see Protocol timing). 0 FlexCAN not in Freeze mode, prescaler running. 1 FlexCAN in Freeze mode, prescaler stopped.
23 SUPV	Supervisor Mode This bit configures the FlexCAN to be either in Supervisor or User mode. The registers affected by this bit are marked as S/U in the Access Type column of the module memory map. Reset value of this bit is 1, so the affected registers start with Supervisor access allowance only. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 FlexCAN is in User mode. Affected registers allow both Supervisor and Unrestricted accesses. 1 FlexCAN is in Supervisor mode. Affected registers allow only Supervisor access. Unrestricted access behaves as though the access was done to an unimplemented register location.
22 SLFWAK	Self Wake Up This bit enables the Self Wake Up feature when FlexCAN is in a low-power mode other than Disable mode. When this feature is enabled, the FlexCAN module monitors the bus for wake up event, that is, a recessive-to-dominant transition. If a wake up event is detected during Stop mode, then FlexCAN generates, if enabled to do so, a Wake Up interrupt to the CPU so that it can exit Stop mode globally and FlexCAN can request to resume the clocks. When FlexCAN is in a low-power mode other than Disable mode, this bit cannot be written as it is blocked by hardware. 0 FlexCAN Self Wake Up feature is disabled. 1 FlexCAN Self Wake Up feature is enabled.
21 WRNEN	Warning Interrupt Enable When asserted, this bit enables the generation of the TWRNINT and RWRNINT flags in the Error and Status Register 1 (ESR1). If WRNEN is negated, the TWRNINT and RWRNINT flags will always be zero, independent of the values of the error counters, and no warning interrupt will ever be generated. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 TWRNINT and RWRNINT bits are zero, independent of the values in the error counters. 1 TWRNINT and RWRNINT bits are set when the respective error counter transitions from less than 96 to greater than or equal to 96.
20 LPMACK	Low-Power Mode Acknowledge This read-only bit indicates that FlexCAN is in a low-power mode (Disable mode, Stop mode). A low-power mode cannot be entered until all current transmission or reception processes have finished, so the CPU can poll the LPMACK bit to know when FlexCAN has actually entered low power mode. This bit is not affected by soft reset.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	NOTE: LPMACK will be asserted within 180 CAN bits from the low-power mode request by the CPU, and negated within 2 CAN bits after the low-power mode request removal (see Protocol timing). 0 FlexCAN is not in a low-power mode. 1 FlexCAN is in a low-power mode.
19 WAKSRC	Wake Up Source This bit defines whether the integrated low-pass filter is applied to protect the Rx CAN input from spurious wake up. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 FlexCAN uses the unfiltered Rx input to detect recessive to dominant edges on the CAN bus. 1 FlexCAN uses the filtered Rx input to detect recessive to dominant edges on the CAN bus.
18 Reserved	This field is reserved. When writing to this field, always write the reset value.
17 SRXDIS	Self Reception Disable This bit defines whether FlexCAN is allowed to receive frames transmitted by itself. If this bit is asserted, frames transmitted by the module will not be stored in any MB, regardless if the MB is programmed with an ID that matches the transmitted frame, and no interrupt flag or interrupt signal will be generated due to the frame reception. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 Self reception enabled. 1 Self reception disabled.
16 IRMQ	Individual Rx Masking And Queue Enable This bit indicates whether Rx matching process will be based either on individual masking and queue or on masking scheme with CAN_RXMGMASK, CAN_RX14MASK, CAN_RX15MASK and CAN_RXFGMASK. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Individual Rx masking and queue feature are disabled. For backward compatibility with legacy applications, the reading of C/S word locks the MB even if it is EMPTY. 1 Individual Rx masking and queue feature are enabled.
15 DMA	DMA Enable The DMA Enable bit controls whether the DMA feature is enabled or not. The DMA feature can only be used in Rx FIFO, consequently the bit CAN_MCR[RFEN] must be asserted. When DMA and RFEN are set, the CAN_IFLAG1[BUF5I] generates the DMA request and no RX FIFO interrupt is generated. This bit can be written in Freeze mode only as it is blocked by hardware in other modes. 0 DMA feature for RX FIFO disabled. 1 DMA feature for RX FIFO enabled.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 LPRIOEN	Local Priority Enable This bit is provided for backwards compatibility with legacy applications. It controls whether the local priority feature is enabled or not. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word, but the actual transmitted ID still has 11-bit for standard frames and 29-bit for extended frames. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 Local Priority disabled. 1 Local Priority enabled.
12 AEN	Abort Enable When asserted, this bit enables the Tx abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE:** When CAN_MCR[AEN] is asserted, only the abort mechanism (see Transmission abort mechanism) must be used for updating Mailboxes configured for transmission. **CAUTION:** Writing the Abort code into Rx Mailboxes can cause unpredictable results when the CAN_MCR[AEN] is asserted. 0 Abort disabled. 1 Abort enabled.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9–8 IDAM	ID Acceptance Mode This 2-bit field identifies the format of the Rx FIFO ID Filter Table elements. Note that all elements of the table are configured at the same time by this field (they are all the same format). See Section "Rx FIFO Structure". This field can be written only in Freeze mode because it is blocked by hardware in other modes. 00 Format A: One full ID (standard and extended) per ID Filter Table element. 01 Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID Filter Table element. 10 Format C: Four partial 8-bit Standard IDs per ID Filter Table element. 11 Format D: All frames rejected.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MAXMB	Number Of The Last Message Buffer This 7-bit field defines the number of the last Message Buffers that will take part in the matching and arbitration processes. The reset value (0x0F) is equivalent to a 16 MB configuration. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Number of the last MB = MAXMB **NOTE:** MAXMB must be programmed with a value smaller than or equal to the number of available Message Buffers. Additionally, the definition of MAXMB value must take into account the region of MBs occupied by Rx FIFO and its ID filters table space defined by RFFN bit in CAN_CTRL2 register. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (see Arbitration and matching timing).

50.4.3 Control 1 register (CANx_CTRL1)

This register is defined for specific FlexCAN control features related to the CAN bus, such as bit-rate, programmable sampling point within an Rx bit, Loop Back mode, Listen-Only mode, Bus Off recovery behavior and interrupt enabling (Bus-Off, Error, Warning). It also determines the Division Factor for the clock prescaler.

The CAN bit timing variables (PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW) can also be configured in CAN_CBT register, which extends the range of all these variables. If CAN_CBT[BTF] is set, PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW fields of CAN_CTRL1 become read only.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CAN_CTRL1 and in CAN_CBT are stored in the same register.

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRESDIV								RJW		PSEG1			PSEG2		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BOFFMSK	ERRMSK	CLKSRC	LPB	TWRNMSK	RWRNMSK	Reserved	Reserved	SMP	BOFFREC	TSYN	LBUF	LOM	PROPSEG		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_CTRL1 field descriptions

Field	Description
31–24 PRESDIV	Prescaler Division Factor This 8-bit field defines the ratio between the PE clock frequency and the Serial Clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The Maximum value of this field is 0xFF, that gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing . This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (PRESDIV + 1)
23–22 RJW	Resync Jump Width This 2-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization. One time quantum is equal to the Sclock period. The valid programmable values are 0–3. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = RJW + 1.
21–19 PSEG1	Phase Segment 1 This 3-bit field defines the length of Phase Segment 1 in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time-Quanta.
18–16 PSEG2	Phase Segment 2 This 3-bit field defines the length of Phase Segment 2 in the bit time. The valid programmable values are 1–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time-Quanta.
15 BOFFMSK	Bus Off Interrupt Mask This bit provides a mask for the Bus Off Interrupt BOFFINT in CAN_ESR1 register. 0 Bus Off interrupt disabled. 1 Bus Off interrupt enabled.
14 ERRMSK	Error Interrupt Mask This bit provides a mask for the Error Interrupt ERRINT in the CAN_ESR1 register. 0 Error interrupt disabled. 1 Error interrupt enabled.
13 CLKSRC	CAN Engine Clock Source This bit selects the clock source to the CAN Protocol Engine (PE) to be either the peripheral clock or the oscillator clock. The selected clock is the one fed to the prescaler to generate the Serial Clock (Sclock). In order to guarantee reliable operation, this bit can be written only in Disable mode because it is blocked by hardware in other modes. See Protocol timing. **NOTE:** Please refer to the clock distribution chapter (module clocks table) to identify the proper clock source. **NOTE:** The user must ensure the protocol engine clock tolerance according to the CAN Protocol standard (ISO 11898-1). 0 The CAN engine clock source is the oscillator clock. Under this condition, the oscillator clock frequency must be lower than the bus clock. 1 The CAN engine clock source is the peripheral clock.

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
12 LPB	Loop Back Mode This bit configures FlexCAN to operate in Loop-Back mode. In this mode, FlexCAN performs an internal loop back that can be used for self test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field, generating an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE:** In this mode, the CAN_MCR[SRXDIS] cannot be asserted because this will impede the self reception of a transmitted message. 0 Loop Back disabled. 1 Loop Back enabled.
11 TWRNMSK	Tx Warning Interrupt Mask This bit provides a mask for the Tx Warning Interrupt associated with the TWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when CAN_MCR[WRNEN] bit is negated. This bit can be written only if CAN_MCR[WRNEN] bit is asserted. 0 Tx Warning Interrupt disabled. 1 Tx Warning Interrupt enabled.
10 RWRNMSK	Rx Warning Interrupt Mask This bit provides a mask for the Rx Warning Interrupt associated with the RWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when CAN_MCR[WRNEN] bit is negated. This bit can be written only if CAN_MCR[WRNEN] bit is asserted. 0 Rx Warning Interrupt disabled. 1 Rx Warning Interrupt enabled.
9 Reserved	This field is reserved.
8 Reserved	This field is reserved.
7 SMP	CAN Bit Sampling This bit defines the sampling mode of CAN bits at the Rx input. It can be written in Freeze mode only because it is blocked by hardware in other modes. **NOTE:** For proper operation, to assert SMP it is necessary to guarantee a minimum value of 2 TQs in CAN_CTRL1[PSEG1] (or CAN_CBT[EPSEG1]). 0 Just one sample is used to determine the bit value. 1 Three samples are used to determine the value of the received bit: the regular one (sample point) and 2 preceding samples; a majority rule is used.
6 BOFFREC	Bus Off Recovery This bit defines how FlexCAN recovers from Bus Off state. If this bit is negated, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If the bit is asserted, automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated by the user. If the

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
	negation occurs before 128 sequences of 11 recessive bits are detected on the CAN bus, then Bus Off recovery happens as if the BOFFREC bit had never been asserted. If the negation occurs after 128 sequences of 11 recessive bits occurred, then FlexCAN will re-synchronize to the bus by waiting for 11 recessive bits before joining the bus. After negation, the BOFFREC bit can be re-asserted again during Bus Off, but it will be effective only the next time the module enters Bus Off. If BOFFREC was negated when the module entered Bus Off, asserting it during Bus Off will not be effective for the current Bus Off recovery. **NOTE:** Refer to Bus off in the CAN Protocol standard (ISO 11898-1) for details. 0 Automatic recovering from Bus Off state enabled. 1 Automatic recovering from Bus Off state disabled.
5 TSYN	Timer Sync This bit enables a mechanism that resets the free-running timer each time a message is received in Message Buffer 0. This feature provides means to synchronize multiple FlexCAN stations with a special "SYNC" message, that is, global network time. If the RFEN bit in CAN_MCR is set (Rx FIFO enabled), the first available Mailbox, according to CAN_CTRL2[RFFN] setting, is used for timer synchronization instead of MB0. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Timer Sync feature disabled 1 Timer Sync feature enabled
4 LBUF	Lowest Buffer Transmitted First This bit defines the ordering mechanism for Message Buffer transmission. When asserted, the CAN_MCR[LPRIOEN] bit does not affect the priority arbitration. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Buffer with highest priority is transmitted first. 1 Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode This bit configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in CAN_ECR register are frozen and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error without changing the receive error counter (RXERRCNT) in CAN_ECR register, as if it was trying to acknowledge the message. Listen-Only mode is acknowledged by the state of CAN_ESR1[FLTCONF] field indicating Passive Error. There can be some delay between the Listen-Only mode request and acknowledge. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Listen-Only mode is deactivated. 1 FlexCAN module operates in Listen-Only mode.
PROPSEG	Propagation Segment This 3-bit field defines the length of the Propagation Segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (PROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.

50.4.4 Free Running Timer (CANx_TIMER)

This register represents a 16-bit free running counter that can be read and written by the CPU. The timer starts from 0x0 after Reset, counts linearly to 0xFFFF, and wraps around.

When the TIMER_SRC in CAN_CTRL2 register is asserted, the timer is continuously incremented by an external time tick. The time tick must be synchronous to the Peripheral Clock, with a minimum pulse width of one clock cycle.

When the TIMER_SRC bit in CAN_CTRL2 register is negated, the timer is incremented by the CAN bit clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable, Stop and Freeze modes.

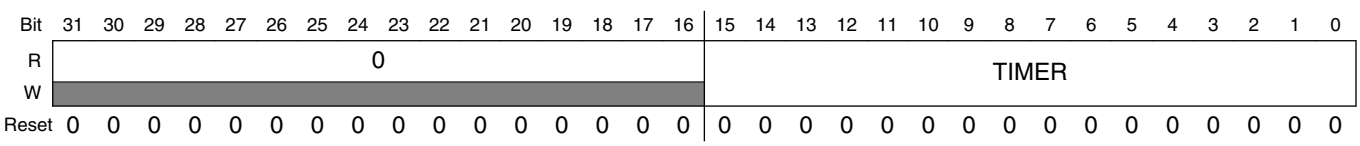
The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the Time Stamp entry in a message buffer after a successful reception or transmission of a message.

If bit CAN_CTRL1[TSYN] is asserted, the Timer is reset whenever a message is received in the first available Mailbox, according to CAN_CTRL2[RFFN] setting.

The CPU can write to this register anytime. However, if the write occurs at the same time that the Timer is being reset by a reception in the first Mailbox, then the write value is discarded.

Reading this register affects the Mailbox Unlocking procedure, see Section "Mailbox Lock Mechanism".

Address: Base address + 8h offset



CANx_TIMER field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TIMER	Timer Value Contains the free-running counter value.

50.4.5 Rx Mailboxes Global Mask Register (CANx_RXMGMASK)

This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When the CAN_MCR[IRMQ] bit is negated, RXMGMASK is always in effect (the bits in the MG field will mask the Mailbox filter bits).
- When the CAN_MCR[IRMQ] bit is asserted, RXMGMASK has no effect (the bits in the MG field will not mask the Mailbox filter bits).

RXMGMASK is used to mask the filter fields of all Rx MBs, excluding MBs 14-15, which have individual mask registers.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	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	MG																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RXMGMASK field descriptions

Field	Description
MG	Rx Mailboxes Global Mask Bits These bits mask the Mailbox filter bits. Note that the alignment with the ID word of the Mailbox is not perfect as the two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the Mailbox. The following table shows in detail which MG bits mask each Mailbox filter field.

SMB[RTR] [1]	CAN_CTRL2[RRS]	CAN_CTRL2[EACEN]	Mailbox filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	-	0	note [2]	note [3]	MG[28:0]	MG[31:29]
0	-	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	-	-	-	-	MG[31:0]
1	1	0	-	-	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

0 The corresponding bit in the filter is "don't care."
1 The corresponding bit in the filter is checked.

1. RTR bit of the Incoming Frame. It is saved into an auxiliary MB called Rx Serial Message Buffer (Rx SMB).
2. If the CTRL2[EACEN] bit is negated, the RTR bit of Mailbox is never compared with the RTR bit of the incoming frame.
3. If the CAN_CTRL2[EACEN] bit is negated, the IDE bit of Mailbox is always compared with the IDE bit of the incoming frame.

50.4.6 Rx 14 Mask register (CANx_RX14MASK)

This register is located in RAM.

RX14MASK is provided for legacy application support. When the CAN_MCR[IRMQ] bit is asserted, RX14MASK has no effect.

RX14MASK is used to mask the filter fields of Message Buffer 14.

This register can only be programmed while the module is in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RX14M																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RX14MASK field descriptions

Field	Description
RX14M	Rx Buffer 14 Mask Bits Each mask bit masks the corresponding Mailbox 14 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

50.4.7 Rx 15 Mask register (CANx_RX15MASK)

This register is located in RAM.

RX15MASK is provided for legacy application support. When the CAN_MCR[IRMQ] bit is asserted, RX15MASK has no effect.

RX15MASK is used to mask the filter fields of Message Buffer 15.

This register can be programmed only while the module is in Freeze mode because it is blocked by hardware in other modes.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	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	RX15M																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RX15MASK field descriptions

Field	Description
RX15M	Rx Buffer 15 Mask Bits Each mask bit masks the corresponding Mailbox 15 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

50.4.8 Error Counter (CANx_ECR)

This register has two 8-bit fields reflecting the value of the FlexCAN error counters:

- Transmit Error Counter (TXERRCNT field)
- Receive Error Counter (RXERRCNT field)

The Fault Confinement State (FLTCONF field in Error and Status Register 1 - CAN_ESR1) is updated based on TXERRCNT and RXERRCNT counters. TXERRCNT and RXERRCNT counters can be written in Freeze mode only. The rules for increasing and decreasing these counters are described in the CAN protocol and are completely implemented in the FlexCAN module.

The following are the basic rules for FlexCAN bus state transitions:

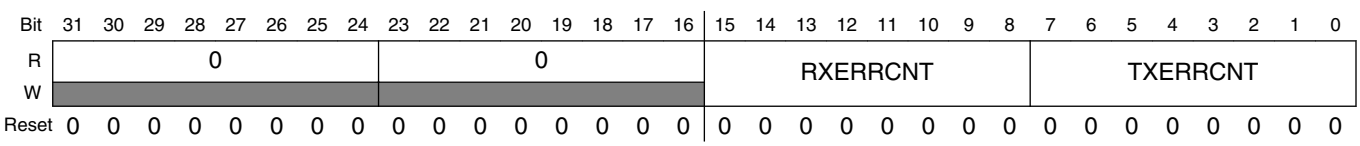
- If the value of TXERRCNT or RXERRCNT increases to be greater than or equal to 128, the FLTCONF field in the Error and Status Register is updated to reflect "Error Passive" state.
- If the FlexCAN state is "Error Passive", and either TXERRCNT or RXERRCNT decrements to a value less than or equal to 127 while the other already satisfies this condition, the FLTCONF field in the Error and Status Register is updated to reflect "Error Active" state.
- If the value of TXERRCNT increases to be greater than 255, the FLTCONF field in the Error and Status Register is updated to reflect "Bus Off" state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.

- If FlexCAN is in "Bus Off' state, then TXERRCNT is cascaded together with another internal counter to count the 128th occurrences of 11 consecutive recessive bits on the bus. Hence, TXERRCNT is reset to zero and counts in a manner where the internal counter counts 11 such bits and then wraps around while incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, the FLTCONF field in the Error and Status Register is updated to be "Error Active" and both error counters are reset to zero. At any instance of dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value.
- If during system start-up, only one node is operating, then its TXERRCNT increases in each message it is trying to transmit, as a result of acknowledge errors (indicated by the ACKERR bit in the Error and Status Register). After the transition to "Error Passive" state, the TXERRCNT does not increment anymore by acknowledge errors. Therefore the device never goes to the "Bus Off" state.
- If the RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected while being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to resume to "Error Active" state.

NOTE

Refer to Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 1Ch offset



CANx_ECR 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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 RXERRCNT	Receive Error Counter Receive Error Counter for all errors detected in received messages. The RXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.
TXERRCNT	Transmit Error Counter Transmit Error Counter for all errors detected in transmitted messages. The TXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.

50.4.9 Error and Status 1 register (CANx_ESR1)

This register reports various error conditions detected in the reception and transmission of a CAN frame, some general status of the device and it is the source of some interrupts to the CPU.

The reported error conditions are BIT1ERR, BIT0ERR, ACKKERR, CRCERR, FRMERR and STFERR.

An error detected in a single CAN frame may be reported by one or more error flags. Also, error reporting is cummulative in case more error events happen in the next frames while the CPU does not attempt to read this register.

TXWRN, RXWRN, IDLE, TX, FLTCONF, RX and SYNCH are status bits.

BOFFINT, BOFFDONEINT, ERRINT, WAKINT, TWRNINT and RWRNINT are interrupt bits. It is recommended that CPU use the following procedure when servicing interrupt requests generated by these bits:

- Read this register to capture all error condition and status bits. This action clear the respective bits that were set since the last read access.
- Write 1 to clear the interrupt bit that has triggered the interrupt request.
- Write 1 to clear the ERR_OVR bit if it is set.

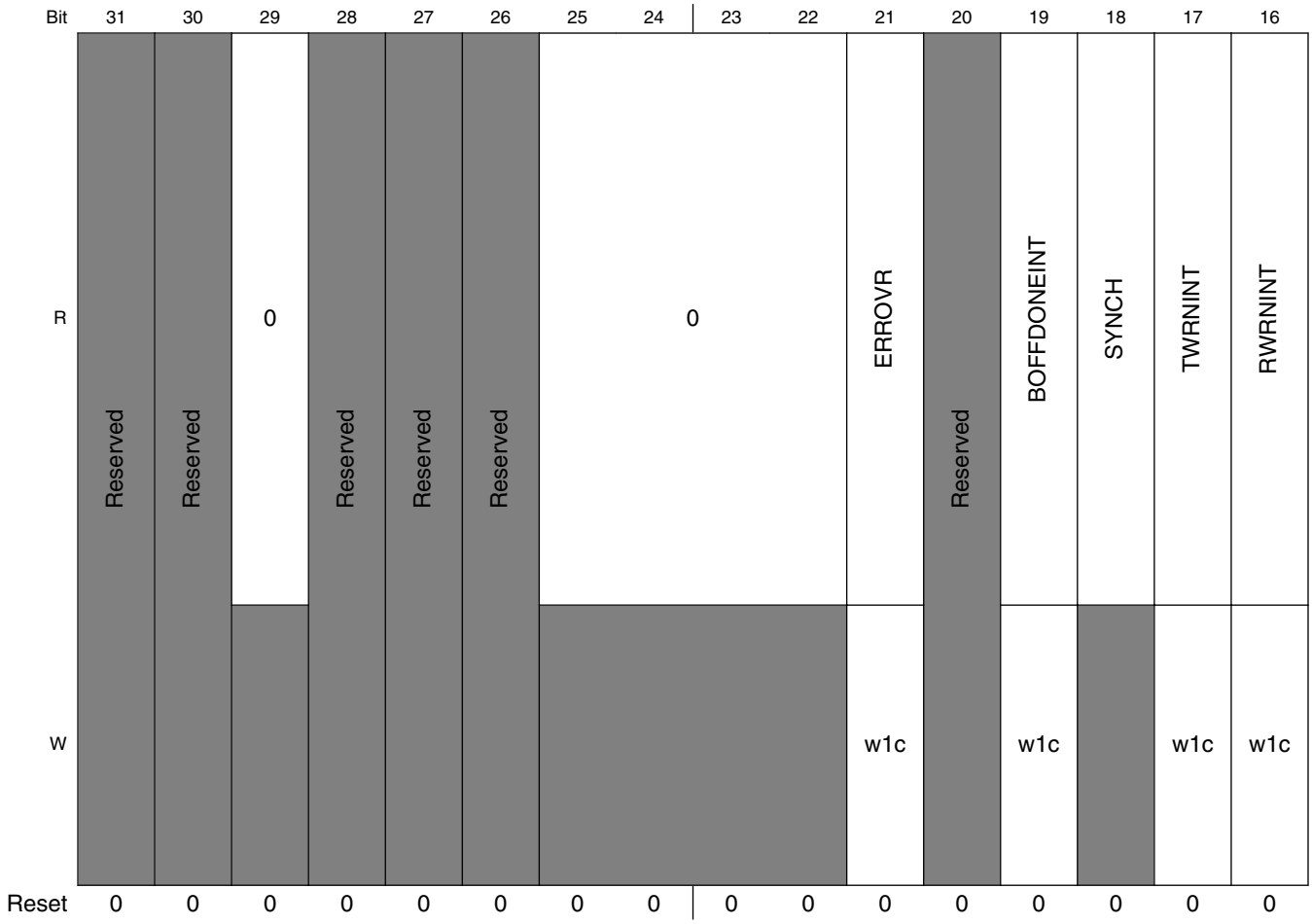
Starting from all error flags cleared, a first error event sets the ERRINT (provided the corresponding mask bit is asserted). If other error events in subsequent frames happen before the CPU to serve the interrupt request, the ERR_OVR bit is set to indicate that errors from different frames had accumulated.

SYNCH	IDLE	TX	RX	FlexCAN State
0	0	0	0	Not synchronized to CAN bus
1	1	x	x	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

Refer to Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 20h offset



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	WAKINT
W														w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_ESR1 field descriptions

Field	Description
31 Reserved	This field is reserved.
30 Reserved	This field is reserved.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 Reserved	This field is reserved.
27 Reserved	This field is reserved.
26 Reserved	This field is reserved.
25–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 ERROVR	Error Overrun bit This bit indicates that an error condition occurred when any error flag is already set. This bit is cleared by writing it to 1.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	0 Overrun has not occurred. 1 Overrun has occurred.
20 Reserved	This field is reserved.
19 BOFFDONEINT	Bus Off Done Interrupt This bit is set when the Tx Error Counter (TXERRCNT) has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If the corresponding mask bit in the Control 2 Register (BOFFDONEMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 FlexCAN module has completed Bus Off process.
18 SYNCH	CAN Synchronization Status This read-only flag indicates whether the FlexCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the FlexCAN. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not synchronized to the CAN bus. 1 FlexCAN is synchronized to the CAN bus.
17 TWRNINT	Tx Warning Interrupt Flag If the WRNEN bit in CAN_MCR is asserted, the TWRNINT bit is set when the TXWRN flag transitions from 0 to 1, meaning that the Tx error counter reached 96. If the corresponding mask bit in the Control 1 Register (CAN_CTRL1[TWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when the WRNEN is set again. Writing 0 has no effect. This flag is not generated during Bus Off state. This bit is not updated during Freeze mode. 0 No such occurrence. 1 The Tx error counter transitioned from less than 96 to greater than or equal to 96.
16 RWRNINT	Rx Warning Interrupt Flag If the WRNEN bit in CAN_MCR is asserted, the RWRNINT bit is set when the RXWRN flag transitions from 0 to 1, meaning that the Rx error counters reached 96. If the corresponding mask bit in the Control 1 Register (CAN_CTRL1[RWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when the WRNEN is set again. Writing 0 has no effect. This bit is not updated during Freeze mode. 0 No such occurrence. 1 The Rx error counter transitioned from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. **NOTE:** This bit is not set by a transmitter in case of arbitration field or ACK slot, or in case of a node sending a passive error flag that detects dominant bits. 0 No such occurrence. 1 At least one bit sent as recessive is received as dominant.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
14 BIT0ERR	Bit0 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. 0 No such occurrence. 1 At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error This bit indicates that an Acknowledge Error has been detected by the transmitter node, that is, a dominant bit has not been detected during the ACK SLOT. 0 No such occurrence. 1 An ACK error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error This bit indicates that a CRC Error has been detected by the receiver node, that is, the calculated CRC is different from the received. 0 No such occurrence. 1 A CRC error occurred since last read of this register.
11 FRMERR	Form Error This bit indicates that a Form Error has been detected by the receiver node, that is, a fixed-form bit field contains at least one illegal bit. 0 No such occurrence. 1 A Form Error occurred since last read of this register.
10 STFERR	Stuffing Error This bit indicates that a Stuffing Error has been detected by the receiver node. 0 No such occurrence. 1 A Stuffing Error occurred since last read of this register.
9 TXWRN	TX Error Warning This bit indicates when repetitive errors are occurring during message transmission and is affected by the value of TXERRCNT in CAN_ECR register only. This bit is not updated during Freeze mode. 0 No such occurrence. 1 TXERRCNT is greater than or equal to 96.
8 RXWRN	Rx Error Warning This bit indicates when repetitive errors are occurring during message reception and is affected by the value of RXERRCNT in CAN_ECR register only. This bit is not updated during Freeze mode. 0 No such occurrence. 1 RXERRCNT is greater than or equal to 96.
7 IDLE	This bit indicates when CAN bus is in IDLE state. See the table in the overall CAN_ESR1 register description. 0 No such occurrence. 1 CAN bus is now IDLE.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
6 TX	FlexCAN In Transmission This bit indicates if FlexCAN is transmitting a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not transmitting a message. 1 FlexCAN is transmitting a message.
5–4 FLTCONF	Fault Confinement State This 2-bit field indicates the Confinement State of the FlexCAN module. If the LOM bit in the Control Register 1 is asserted, after some delay that depends on the CAN bit timing the FLTCONF field will indicate “Error Passive”. Because of the very same delay, the way in which FLTCONF reflects an update to the CAN_ECR register by the CPU, also gets affected. This bit field is affected by soft reset, but if the LOM bit is asserted, its reset value lasts just one CAN bit. After this time, FLTCONF reports "Error Passive". 00 Error Active 01 Error Passive 1x Bus Off
3 RX	FlexCAN In Reception This bit indicates if FlexCAN is receiving a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not receiving a message. 1 FlexCAN is receiving a message.
2 BOFFINT	Bus Off Interrupt This bit is set when FlexCAN enters ‘Bus Off’ state. If the corresponding mask bit in the Control Register 1 (CAN_CTRL1[BOFFMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 FlexCAN module entered Bus Off state.
1 ERRINT	Error Interrupt This bit indicates that at least one of the Error Bits (BIT1ERR, BIT0ERR, ACKERR. CRCERR, FRMERR or STFERR) is set. If the corresponding mask bit CAN_CTRL1[ERRMSK] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 Indicates setting of any Error Bit in the Error and Status Register.
0 WAKINT	Wake-Up Interrupt This field applies when FlexCAN is in low-power mode under Self Wake Up mechanism: • Stop mode When a recessive-to-dominant transition is detected on the CAN bus and if the CAN_MCR[WAKMSK] bit is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When CAN_MCR[SLFWAK] is negated, this flag is masked. The CPU must clear this flag before disabling the bit. Otherwise it will be set when the SLFWAK is set again. Writing 0 has no effect.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	0 No such occurrence. 1 Indicates a recessive to dominant transition was received on the CAN bus.

50.4.10 Interrupt Masks 1 register (CANx_IMASK1)

This register allows any number of a range of the 16 Message Buffer Interrupts to be enabled or disabled for MB15 to MB0. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding CAN_IFLAG1 bit is set.

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																BUF15TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_IMASK1 field descriptions

Field	Description
31–16 Reserved	This field is reserved.
BUF15TO0M	Buffer MB $_i$ Mask Each bit enables or disables the corresponding FlexCAN Message Buffer Interrupt for MB15 to MB0. **NOTE:** Setting or clearing a bit in the CAN_IMASK1 Register can assert or negate an interrupt request, if the corresponding IFLAG1 bit is set. 0 The corresponding buffer Interrupt is disabled. 1 The corresponding buffer Interrupt is enabled.

50.4.11 Interrupt Flags 1 register (CANx_IFLAG1)

This register defines the flags for the 16 Message Buffer interrupts for MB15 to MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding CAN_IFLAG1 bit. If the corresponding CAN_IMASK1 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect. There is an exception when DMA for Rx FIFO is enabled, as described below.

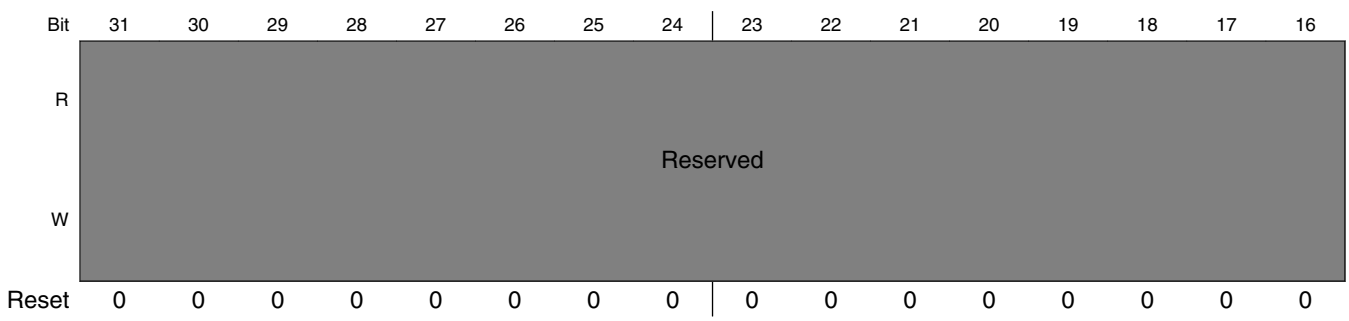
The BUF7I to BUF5I flags are also used to represent FIFO interrupts when the Rx FIFO is enabled. When the bit CAN_MCR[RFEN] is set and the bit CAN_MCR[DMA] is negated, the function of the 8 least significant interrupt flags changes: BUF7I, BUF6I and BUF5I indicate operating conditions of the FIFO, BUF0I is used to empty FIFO, and BUF4I to BUF1I bits are reserved.

Before enabling the CAN_MCR[RFEN], the CPU must service the IFLAG bits asserted in the Rx FIFO region; see Section "Rx FIFO". Otherwise, these IFLAG bits will mistakenly show the related MBs now belonging to FIFO as having contents to be serviced. When the CAN_MCR[RFEN] bit is negated, the FIFO flags must be cleared. The same care must be taken when an CAN_CTRL2[RFFN] value is selected extending Rx FIFO filters beyond MB7. For example, when RFFN is 0x8, the MB0-23 range is occupied by Rx FIFO filters and related IFLAG bits must be cleared.

When both the CAN_MCR[RFEN] and CAN_MCR[DMA] bits are asserted (DMA feature for Rx FIFO enabled), the function of the 8 least significant interrupt flags (BUF7I - BUF0I) are changed to support the DMA operation. BUF7I and BUF6I are not used, as well as, BUF4I to BUF1I. BUF5I indicates operating condition of FIFO, and BUF0I is used to empty FIFO. Moreover, BUF5I does not generate a CPU interrupt, but generates a DMA request. IMASK1 bits in Rx FIFO region are not considered when bit CAN_MCR[DMA] is enabled. In addition the CPU must not clear the flag BUF5I when DMA is enabled. Before enabling the bit CAN_MCR[DMA], the CPU must service the IFLAGs asserted in the Rx FIFO region. When the bit CAN_MCR[DMA] is negated, the FIFO must be empty.

Before updating CAN_MCR[MAXMB] field, CPU must service the CAN_IFLAG1 bits whose MB value is greater than the CAN_MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Address: Base address + 30h offset



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF15TO8I								BUF7I	BUF6I	BUF5I	BUF4TO1I				BUF0I
W	w1c								w1c	w1c	w1c	w1c				w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_IFLAG1 field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–8 BUF15TO8I	Buffer MB_i Interrupt Each bit flags the corresponding FlexCAN Message Buffer interrupt for MB15 to MB8. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception. 1 The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt Or "Rx FIFO Overflow" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), this bit flags the interrupt for MB7. **NOTE:** This flag is cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF7I flag represents "Rx FIFO Overflow" when CAN_MCR[RFEN] is set. In this case, the flag indicates that a message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. 0 No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1 MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
6 BUF6I	Buffer MB6 Interrupt Or "Rx FIFO Warning" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), this bit flags the interrupt for MB6. **NOTE:** This flag is cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF6I flag represents "Rx FIFO Warning" when CAN_MCR[RFEN] is set. In this case, the flag indicates when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. Note that if the flag is cleared while the number of unread messages is greater than 4, it does not assert again until the number of unread messages within the Rx FIFO is decreased to be equal to or less than 4. 0 No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1 MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1

Table continues on the next page...

CANx_IFLAG1 field descriptions (continued)

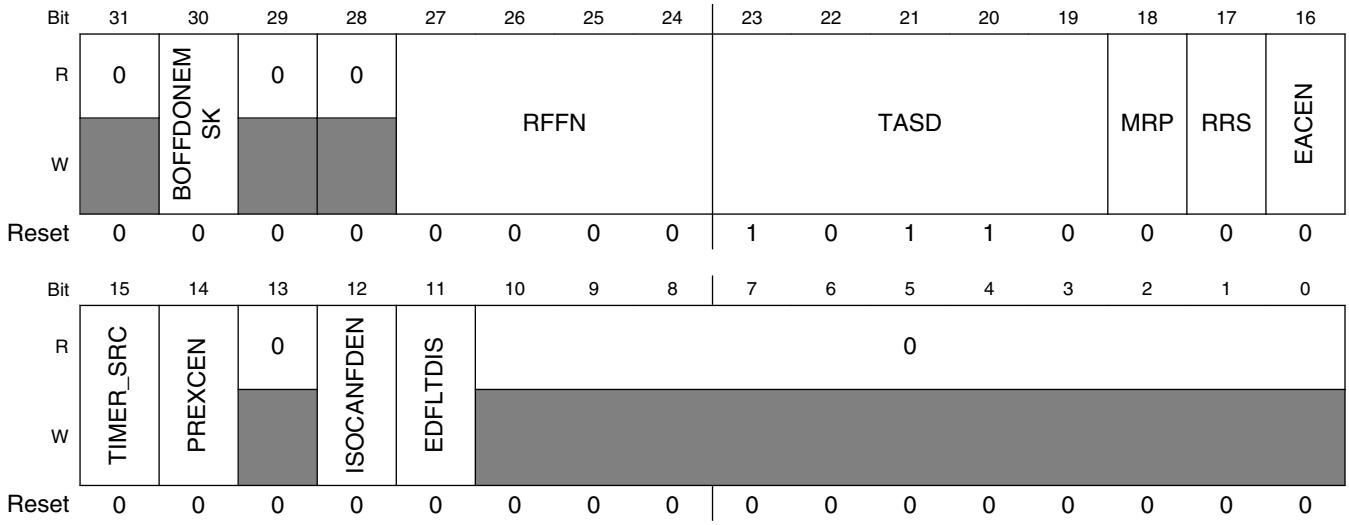
Field	Description
5 BUF5I	Buffer MB5 Interrupt Or "Frames available in Rx FIFO" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB5. **NOTE:** This flag is cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. When MCR[RFEN] is set (Rx FIFO enabled), the BUF5I flag represents "Frames available in Rx FIFO" and indicates that at least one frame is available to be read from the Rx FIFO. When the MCR[DMA] bit is enabled, this flag generates a DMA request and the CPU must not clear this bit by writing 1 in BUF5I. 0 No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1 1 MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.
4–1 BUF4TO1I	Buffer MB $_i$ Interrupt Or "reserved" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), these bits flag the interrupts for MB4 to MB1. **NOTE:** These flags are cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF4TO1I flags are reserved when CAN_MCR[RFEN] is set. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1 The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.
0 BUF0I	Buffer MB0 Interrupt Or Clear FIFO bit When the RFEN bit in MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB0. If the Rx FIFO is enabled, this bit is used to trigger the clear FIFO operation. This operation empties FIFO contents. Before performing this operation the CPU must service all FIFO related IFLAGs. When the bit MCR[DMA] is enabled this operation also clears the BUF5I flag and consequently abort the DMA request. The clear FIFO operation occurs when the CPU writes 1 in BUF0I. It is only allowed in Freeze Mode and is blocked by hardware in other conditions. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1 The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.

50.4.12 Control 2 register (CANx_CTRL2)

This register complements Control1 Register providing control bits for memory write access in Freeze Mode, for extending FIFO filter quantity, and for adjust the operation of internal FlexCAN processes like matching and arbitration.

The contents of this register are not affected by soft reset.

Address: Base address + 34h offset



CANx_CTRL2 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 BOFFDONEMSK	Bus Off Done Interrupt Mask This bit provides a mask for the Bus Off Done Interrupt in CAN_ESR1 register. 0 Bus Off Done interrupt disabled. 1 Bus Off Done interrupt enabled.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 RFFN	Number Of Rx FIFO Filters This 4-bit field defines the number of Rx FIFO filters, as shown in the following table. The maximum selectable number of filters is determined by the chip. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field must not be programmed with values that make the number of Message Buffers occupied by Rx FIFO and ID Filter exceed the number of Mailboxes present, defined by CAN_MCR[MAXMB]. **NOTE:** Each group of eight filters occupies a memory space equivalent to two Message Buffers which means that the more filters are implemented the less Mailboxes will be available. Considering that the Rx FIFO occupies the memory space originally reserved for MB0-5, RFFN should be programmed with a value corresponding to a number of filters not greater than the number of available memory words which can be calculated as follows: (SETUP_MB - 6) × 4 where SETUP_MB is the least between the parameter NUMBER_OF_MB and CAN_MCR[MAXMB]. The number of remaining Mailboxes available will be: (SETUP_MB - 8) - (RFFN × 2)

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
If the Number of Rx FIFO Filters programmed through RFFN exceeds the SETUP_MB value (memory space available) the exceeding ones will not be functional.

NOTE:
• The number of the last remaining available mailboxes is defined by the least value between the NUMBER_OF_MB minus 1 and the CAN_MCR[MAXMB] field.
• If Rx Individual Mask Registers are not enabled then all Rx FIFO filters are affected by the Rx FIFO Global Mask.

<table>
<tr><th>RFFN[3:0]</th><th>Number of Rx FIFO filter elements</th><th>Message Buffers occupied by Rx FIFO and ID Filter Table</th><th>Remaining Available Mailboxes</th><th>Rx FIFO ID Filter Table Elements Affected by Rx Individual Masks</th><th>Rx FIFO ID Filter Table Elements Affected by Rx FIFO Global Mask</th></tr>
<tr><td>0x0</td><td>8</td><td>MB 0-7</td><td>MB 8-15</td><td>Elements 0-7</td><td>none</td></tr>
<tr><td>0x1</td><td>16</td><td>MB 0-9</td><td>MB 10-15</td><td>Elements 0-9</td><td>Elements 10-15</td></tr>
<tr><td>0x2</td><td>24</td><td>MB 0-11</td><td>MB 12-15</td><td>Elements 0-11</td><td>Elements 12-23</td></tr>
<tr><td>0x3</td><td>32</td><td>MB 0-13</td><td>MB 14-15</td><td>Elements 0-13</td><td>Elements 14-31</td></tr>
<tr><td>0x4</td><td>40</td><td>MB 0-15</td><td>none</td><td>Elements 0-15</td><td>Elements 16-39</td></tr>
</table>
</td></tr>
<tr><td>23–19
TASD</td><td>Tx Arbitration Start Delay

This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See Tx Arbitration start delay for more details. This field can be written only in Freeze mode because it is blocked by hardware in other modes.</td></tr>
<tr><td>18
MRP</td><td>Mailboxes Reception Priority

If this bit is set the matching process starts from the Mailboxes and if no match occurs the matching continues on the Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

0 Matching starts from Rx FIFO and continues on Mailboxes.
1 Matching starts from Mailboxes and continues on Rx FIFO.</td></tr>
<tr><td>17
RRS</td><td>Remote Request Storing

If this bit is asserted Remote Request Frame is submitted to a matching process and stored in the corresponding Message Buffer in the same fashion of a Data Frame. No automatic Remote Response Frame will be generated.

If this bit is negated the Remote Request Frame is submitted to a matching process and an automatic Remote Response Frame is generated if a Message Buffer with CODE=0b1010 is found with the same ID.

This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

0 Remote Response Frame is generated.
1 Remote Request Frame is stored.</td></tr>
<tr><td>16
EACEN</td><td>Entire Frame Arbitration Field Comparison Enable For Rx Mailboxes

This bit controls the comparison of IDE and RTR bits within Rx Mailboxes filters with their corresponding bits in the incoming frame by the matching process. This bit does not affect matching for Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.</td></tr>
</table>

Table continues on the next page...

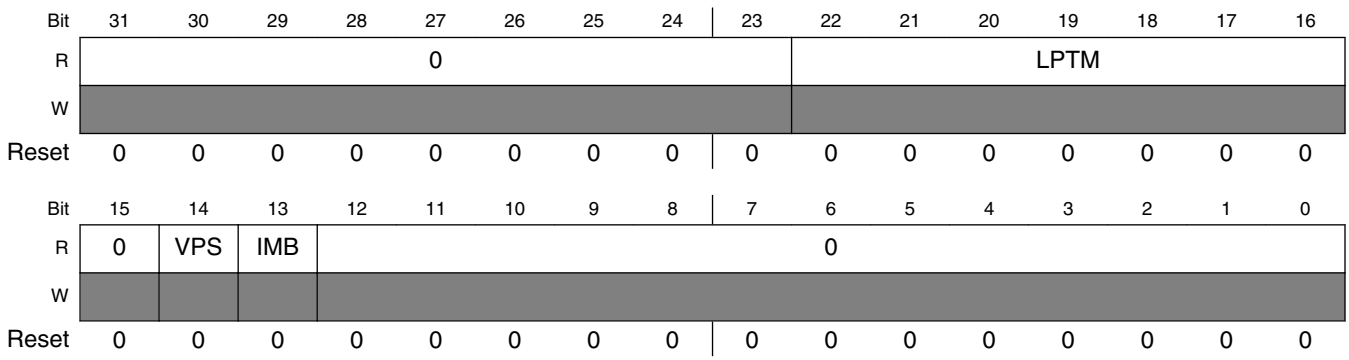
CANx_CTRL2 field descriptions (continued)

Field	Description
	0 Rx Mailbox filter's IDE bit is always compared and RTR is never compared despite mask bits. 1 Enables the comparison of both Rx Mailbox filter's IDE and RTR bit with their corresponding bits within the incoming frame. Mask bits do apply.
15 TIMER_SRC	Timer Source Selects the time tick source used for incrementing the Free Running Timer counter. This bit can be written in Freeze mode only. 0 The Free Running Timer is clocked by the CAN bit clock, which defines the baud rate on the CAN bus. 1 The Free Running Timer is clocked by an external time tick. The period can be either adjusted to be equal to the baud rate on the CAN bus, or a different value as required. See the device specific section for details about the external time tick.
14 PREXCEN	Protocol Exception Enable This bit enables the Protocol Exception feature. **NOTE:** Refer to Protocol exception event in the CAN Protocol standard (ISO 11898-1) for details. 0 Protocol Exception is disabled. 1 Protocol Exception is enabled.
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 ISOCANFDEN	ISO CAN FD Enable This field enables the CAN FD protocol according to ISO specification (ISO 11898-1) (see CAN FD ISO compliance). **NOTE:** FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO 11898-1). 0 FlexCAN operates using the non-ISO CAN FD protocol. 1 FlexCAN operates using the ISO CAN FD protocol (ISO 11898-1).
11 EDFLTDIS	Edge Filter Disable This bit disables the Edge Filter used during the bus integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus state are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of eleven consecutive recessive bits is restarted. The Edge Filter prevents the dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD Frame) from being mistaken for an idle condition. **NOTE:** Refer to Bus Integration state in the CAN Protocol standard (ISO 11898-1) for details. 0 Edge Filter is enabled. 1 Edge Filter is disabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

50.4.13 Error and Status 2 register (CANx_ESR2)

This register reports some general status information.

Address: Base address + 38h offset



CANx_ESR2 field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–16 LPTM	Lowest Priority Tx Mailbox If CAN_ESR2[VPS] is asserted, this field indicates the lowest number inactive Mailbox (see the CAN_ESR2[IMB] bit description). If there is no inactive Mailbox then the Mailbox indicated depends on CAN_CTRL1[LBUF] bit value. If CAN_CTRL1[LBUF] bit is negated then the Mailbox indicated is the one that has the greatest arbitration value (see the "Highest priority Mailbox first" section). If CAN_CTRL1[LBUF] bit is asserted then the Mailbox indicated is the highest number active Tx Mailbox. If a Tx Mailbox is being transmitted it is not considered in LPTM calculation. If CAN_ESR2[IMB] is not asserted and a frame is transmitted successfully, LPTM is updated with its Mailbox number.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 VPS	Valid Priority Status This bit indicates whether CAN_ESR2[IMB] and CAN_ESR2[LPTM] contents are currently valid or not. It is asserted upon every complete Tx arbitration process unless the CPU writes to Control and Status word of a Mailbox that has already been scanned, that is, it is behind Tx Arbitration Pointer, during the Tx arbitration process. If there is no inactive Mailbox and only one Tx Mailbox that is being transmitted then VPS is not asserted. This bit is negated upon the start of every Tx arbitration process or upon a write to Control and Status word of any Mailbox. **NOTE:** CAN_ESR2[VPS] is not affected by any CPU write into Control Status (C/S) of a MB that is blocked by abort mechanism. When CAN_MCR[AEN] is asserted, the abort code write in C/S of a MB that is being transmitted (pending abort), or any write attempt into a Tx MB with CAN_IFLAG set is blocked. 0 Contents of IMB and LPTM are invalid. 1 Contents of IMB and LPTM are valid.
13 IMB	Inactive Mailbox If ESR2[VPS] is asserted, this bit indicates whether there is any inactive Mailbox (CODE field is either 0b1000 or 0b0000). This bit is asserted in the following cases:

Table continues on the next page...

CANx_ESR2 field descriptions (continued)

Field	Description
	• During arbitration, if an CAN_ESR2[LPTM] is found and it is inactive. • If CAN_ESR2[IMB] is not asserted and a frame is transmitted successfully. This bit is cleared in all start of arbitration (see Section "Arbitration process"). **NOTE:** CAN_ESR2[LPTM] mechanism have the following behavior: if an MB is successfully transmitted and CAN_ESR2[IMB]=0 (no inactive Mailbox), then CAN_ESR2[VPS] and CAN_ESR2[IMB] are asserted and the index related to the MB just transmitted is loaded into CAN_ESR2[LPTM]. 0 If ESR2[VPS] is asserted, the ESR2[LPTM] is not an inactive Mailbox. 1 If ESR2[VPS] is asserted, there is at least one inactive Mailbox. LPTM content is the number of the first one.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

50.4.14 CRC Register (CANx_CRCR)

This register provides information about the CRC of transmitted messages. This register is updated at the same time the Tx Interrupt Flag is asserted.

NOTE

Refer to CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 44h offset

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

CANx_CRCR field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–16 MBCRC	CRC Mailbox This field indicates the number of the Mailbox corresponding to the value in CAN_CRCR[TXCRC] field.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

CANx_CRCR field descriptions (continued)

Field	Description
TXCRC	Transmitted CRC value This field indicates the CRC value of the last transmitted message.

50.4.15 Rx FIFO Global Mask register (CANx_RXFGMASK)

This register is located in RAM.

If Rx FIFO is enabled, RXFGMASK is used to mask the Rx FIFO ID Filter Table elements that do not have a corresponding RXIMR according to CAN_CTRL2[RFFN] field setting.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 48h offset

Bit	31	30	29	28	27	26	25	24	23	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	FGM																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RXFGMASK field descriptions

Field	Description
FGM	Rx FIFO Global Mask Bits These bits mask the ID Filter Table elements bits in a perfect alignment. The following table shows how the FGM bits correspond to each IDAF field.

Rx FIFO ID Filter Table Elements Format (CAN_MCR[IDAM])	Identifier Acceptance Filter Fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	-	-	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	-	FGM[29:16], FGM[13:0]		-
C	-	-		-	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

CANx_RXFGMASK field descriptions (continued)

Field	Description
	0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

1. If CAN_MCR[IDAM] field is equivalent to the format B only the fourteen most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.
2. If CAN_MCR[IDAM] field is equivalent to the format C only the eight most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.

50.4.16 Rx FIFO Information Register (CANx_RXFIR)

RXFIR provides information on Rx FIFO.

This register is the port through which the CPU accesses the output of the RXFIR FIFO located in RAM. The RXFIR FIFO is written by the FlexCAN whenever a new message is moved into the Rx FIFO as well as its output is updated whenever the output of the Rx FIFO is updated with the next message. See Section "Rx FIFO" for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case the register is read-only.

Address: Base address + 4Ch offset

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

* Notes:
- x = Undefined at reset.

CANx_RXFIR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IDHIT	Identifier Acceptance Filter Hit Indicator This field indicates which Identifier Acceptance Filter was hit by the received message that is in the output of the Rx FIFO. If multiple filters match the incoming message ID then the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only while the CAN_IFLAG1[BUF5I] is asserted.

50.4.17 CAN Bit Timing Register (CANx_CBT)

This register is an alternative way to store the CAN bit timing variables described in CAN_CTRL1 register. EPRESDIV, EPROPSEG, EPSEG1, EPSEG2 and ERJW are extended versions of PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW bit fields respectively.

The BTF bit selects the use of the timing variables defined in this register.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CAN_CTRL1 and in CAN_CBT are stored in the same register.

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

Address: Base address + 50h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BTF	EPRESDIV										ERJW				
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	EPROPSEG						EPSEG1					EPSEG2				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_CBT field descriptions

Field	Description
31 BTF	Bit Timing Format Enable Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2 and ERJW replacing the CAN bit timing variables defined in CAN_CTRL1 register. This field can be written in Freeze mode only. 0 Extended bit time definitions disabled. 1 Extended bit time definitions enabled.
30–21 EPRESDIV	Extended Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the Serial Clock (Sclock) frequency when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode because it is blocked by hardware in other modes.

Table continues on the next page...

CANx_CBT field descriptions (continued)

Field	Description
	Sclock frequency = PE clock frequency / (EPRESDIV + 1)
20–16 ERJW	Extended Resync Jump Width This 5-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[RJW] value range. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = ERJW + 1.
15–10 EPROPSEG	Extended Propagation Segment This 6-bit field defines the length of the Propagation Segment in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PROPSEG] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.
9–5 EPSEG1	Extended Phase Segment 1 This 5-bit field defines the length of Phase Segment 1 in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PSEG1] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
EPSEG2	Extended Phase Segment 2 This 5-bit field defines the length of Phase Segment 2 in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PSEG2] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

50.4.18 Rx Individual Mask Registers (CANx_RXIMR*n*)

The RX Individual Mask Registers are used to store the acceptance masks for ID filtering in Rx MBs and the Rx FIFO.

When the Rx FIFO is disabled (CAN_MCR[RFEN] bit is negated), an individual mask is provided for each available Rx Mailbox on a one-to-one correspondence. When the Rx FIFO is enabled (CAN_MCR[RFEN] bit is asserted), an individual mask is provided for each Rx FIFO ID Filter Table Element on a one-to-one correspondence depending on the setting of CAN_CTRL2[RFFN] (see Rx FIFO).

CAN_RXIMR0 stores the individual mask associated to either MB0 or ID Filter Table Element 0, CAN_RXIMR1 stores the individual mask associated to either MB1 or ID Filter Table Element 1 and so on.

CAN_RXIMR registers can only be accessed by the CPU while the module is in Freeze mode, otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general purpose memory. See Bus interface for more information.

Address: Base address + 880h offset + (4d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

CANx_RXIMR*n* field descriptions

Field	Description
MI	Individual Mask Bits Each Individual Mask Bit masks the corresponding bit in both the Mailbox filter and Rx FIFO ID Filter Table element in distinct ways. For Mailbox filters, see the RXMGMASK register description. For Rx FIFO ID Filter Table elements, see the RXFGMASK register description. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

50.4.53 Message buffer structure

The message buffer structure used by the FlexCAN module is represented in the following figure. Both Extended (29-bit identifier) and Standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual MB is formed by 16 bytes.

The memory area from 0x80 to 0x17F is used by the mailboxes.

Table 50-4. Message buffer structure

	31 30 29 28	27 24	23	22 21 20	19 18 17 16	15 8	7 0
0x0		CODE		SRR IDE RTR	DLC	TIME STAMP	
0x4	PRIO	ID (Standard/Extended)			ID (Extended)		
0x8	Data Byte 0		Data Byte 1			Data Byte 2	Data Byte 3
0xC	Data Byte 4		Data Byte 5			Data Byte 6	Data Byte 7

= Unimplemented or Reserved

CODE - Message Buffer Code

This 4-bit field can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 50-5 and Table 50-6. See Functional description for additional information.

Table 50-5. Message buffer code for Rx buffers

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
0b0000: INACTIVE - MB is not active.	INACTIVE	-	-	-	MB does not participate in the matching process.
0b0100: EMPTY - MB is active and empty.	EMPTY	-	FULL	-	When a frame is received successfully (after the Move-in) process), the CODE field is automatically updated to FULL.
0b0010: FULL - MB is full.	FULL	Yes	FULL	-	The act of reading the C/S word followed by unlocking the MB (SRV) does not make the code return to EMPTY. It remains FULL. If a new frame is moved to the MB after the MB was serviced, the code still remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	-	If the MB is FULL and a new frame is moved to this MB before the CPU services it, the CODE field is automatically updated to OVERRUN. See Matching process

Table continues on the next page...

Table 50-5. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					for details about overrun behavior.
0b0110: OVERRUN - MB is being overwritten into a full buffer.	OVERRUN	Yes	FULL	-	If the CODE field indicates OVERRUN and CPU has serviced the MB, when a new frame is moved to the MB then the code returns to FULL.
		No	OVERRUN	-	If the CODE field already indicates OVERRUN, and another new frame must be moved, the MB will be overwritten again, and the code will remain OVERRUN. See Matching process for details about overrun behavior.
0b1010: RANSWER[4] - A frame was configured to recognize a Remote Request Frame and transmit a Response Frame in return.	RANSWER	-	TANSWER(0b1110)	0	A Remote Answer was configured to recognize a remote request frame received. After that an MB is set to transmit a response frame. The code is automatically changed to TANSWER (0b1110). See Matching process for details. If CAN_CTRL2[RRS] is negated, transmit a response frame whenever a remote request frame with the same ID is received.
		-	-	1	This code is ignored during matching and arbitration process. See Matching process for details.

Table continues on the next page...

Table 50-5. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
CODE[0]=1: BUSY - FlexCAN is updating the contents of the MB. The CPU must not access the MB.	BUSY[5]	-	FULL	-	Indicates that the MB is being updated. It will be negated automatically and does not interfere with the next CODE.
		-	OVERRUN	-	

1. SRV: Serviced MB. MB was read and unlocked by reading TIMER or other MB.
2. A frame is considered a successful reception after the frame to be moved to MB (move-in process). See Move-in for details.
3. Remote Request Stored bit, see "Control 2 Register (CAN_CTRL2)" for details.
4. Code 0b1010 is not considered Tx and an MB with this code should not be aborted.
5. Note that for Tx MBs, the BUSY bit should be ignored upon read, except when AEN bit is set in the MCR register. If this bit is asserted, the corresponding MB does not participate in the matching process.

Table 50-6. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
0b1000: INACTIVE - MB is not active	INACTIVE	-	-	MB does not participate in arbitration process.
0b1001: ABORT - MB is aborted	ABORT	-	-	MB does not participate in arbitration process.
0b1100: DATA - MB is a Tx Data Frame (MB RTR must be 0)	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the MB automatically returns to the INACTIVE state.
0b1100: REMOTE - MB is a Tx Remote Request Frame (MB RTR must be 1)	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the MB automatically becomes an Rx Empty MB with the same ID.
0b1110: TANSWER - MB is a Tx Response Frame from an incoming Remote Request Frame	TANSWER	-	RANSWER	This is an intermediate code that is automatically written to the MB by the CHI as a result of a match to a remote request frame. The remote response frame will be transmitted unconditionally once, and then the code will

Table 50-6. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
				automatically return to RANSWER (0b1010). The CPU can also write this code with the same effect. The remote response frame can be either a data frame or another remote request frame depending on the RTR bit value. See Matching process and Arbitration process for details.

SRR - Substitute Remote Request

Fixed recessive bit, used only in extended format. It must be set to one by the user for transmission (Tx Buffers) and will be stored with the value received on the CAN bus for Rx receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

1 = Recessive value is compulsory for transmission in extended format frames

0 = Dominant is not a valid value for transmission in extended format frames

IDE - ID Extended Bit

This field identifies whether the frame format is standard or extended.

1 = Frame format is extended

0 = Frame format is standard

RTR - Remote Transmission Request

This bit affects the behavior of remote frames and is part of the reception filter. See Table 50-5, Table 50-6, and the description of the RRS bit in Control 2 Register (CAN_CTRL2) for additional details.

If FlexCAN transmits this bit as '1' (recessive) and receives it as '0' (dominant), it is interpreted as an arbitration loss. If this bit is transmitted as '0' (dominant), then if it is received as '1' (recessive), the FlexCAN module treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission.

1 = Indicates the current MB may have a remote request frame to be transmitted if MB is Tx. If the MB is Rx then incoming remote request frames may be stored.

0 = Indicates the current MB has a data frame to be transmitted. In Rx MB it may be considered in matching processes.

DLC - Length of Data in Bytes

This 4-bit field is the length (in bytes) of the Rx or Tx data, which is located in offset 0x8 through 0xF of the MB space (see Table 50-4) . In reception, this field is written by the FlexCAN module, copied from the DLC (Data Length Code) field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 50-7).

TIME STAMP - Free-Running Counter Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured for Tx and Rx frames at the time when the beginning of the Identifier field appears on the CAN bus.

PRIO - Local priority

This 3-bit field is used only when LPRIO_EN bit is set in CAN_MCR, and it only makes sense for Tx mailboxes. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.

ID - Frame Identifier

In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.

DATA BYTE 0 to 7 - Data Field

Up to eight bytes can be used for a data frame.

For Rx frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC as shown in the table below.

Table 50-7. DATA BYTEs validity

DLC	Valid DATA BYTEs
0	none
1	DATA BYTE 0
2	DATA BYTE 0 to 1
3	DATA BYTE 0 to 2
4	DATA BYTE 0 to 3
5	DATA BYTE 0 to 4

Table continues on the next page...

Table 50-7. DATA BYTEs validity (continued)

DLC	Valid DATA BYTEs
6	DATA BYTE 0 to 5
7	DATA BYTE 0 to 6
8 or above	DATA BYTE 0 to 7

50.4.54 Rx FIFO structure

When the CAN_MCR[RFEN] bit is set, the memory area from 0x80 to 0xDC (which is normally occupied by MBs 0–5) is used by the reception FIFO engine.

The region 0x80-0x8C contains the output of the FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The region 0x90-0xDC is reserved for internal use of the FIFO engine.

An additional memory area, which starts at 0xE0 and may extend up to 0x17C (normally occupied by MBs 6–15) depending on the CAN_CTRL2[RFFN] field setting, contains the ID filter table (configurable from 8 to 40 table elements) that specifies filtering criteria for accepting frames into the FIFO.

Out of reset, the ID filter table flexible memory area defaults to 0xE0 and extends only to 0xFC, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Rx FIFO data structure.

Table 50-8. Rx FIFO structure

	31 28 24	23 22 21 20 19 18 17 16	15 8	7 0
0x80	IDHIT	SRR IDE RTR DLC	TIME STAMP	
0x84	ID standard		ID extended	
0x88	Data byte 0	Data byte 1	Data byte 2	Data byte 3
0x8C	Data byte 4	Data byte 5	Data byte 6	Data byte 7
0x90 to 0xDC	Reserved			
0xE0	ID filter table element 0			
0xE4	ID filter table element 1			
0xE8 to 0x2D4	ID filter table elements 2 to 125			
0x2D8	ID filter table element 126			

Table continues on the next page...

Table 50-8. Rx FIFO structure (continued)

0x2DC	ID filter table element 127
	= Unimplemented or reserved

Each ID filter table element occupies an entire 32-bit word and can be compounded by one, two, or four Identifier Acceptance Filters (IDAF) depending on the CAN_MCR[IDAM] field setting. The following figures show the IDAF indexation.

The following table shows the three different formats of the ID table elements. Note that all elements of the table must have the same format. See Rx FIFO for more information.

Table 50-9. ID table structure

Format	31	30	29 ... 24	23 ... 16	15	14	13 ... 8	7 ... 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std/ext = 31–24)		RXIDC_1 (std/ext = 23–16)		RXIDC_2 (std/ext = 15–8)			RXIDC_3 (std/ext = 7–0)	
	= Unimplemented or Reserved								

RTR — Remote Frame

This bit specifies if Remote Frames are accepted into the FIFO if they match the target ID.

1 = Remote Frames can be accepted and data frames are rejected

0 = Remote Frames are rejected and data frames can be accepted

IDE — Extended Frame

Specifies whether extended or standard frames are accepted into the FIFO if they match the target ID.

1 = Extended frames can be accepted and standard frames are rejected

0 = Extended frames are rejected and standard frames can be accepted

RXIDA — Rx Frame Identifier (Format A)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.

RXIDB_0, RXIDB_1 — Rx Frame Identifier (Format B)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.

RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3 — Rx Frame Identifier (Format C)

Specifies an ID to be used as acceptance criteria for the FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.

IDHIT — Identifier Acceptance Filter Hit Indicator

This 9-bit field indicates which Identifier Acceptance Filter was hit by the received message that is in the output of the Rx FIFO. See Rx FIFO for more information.

50.5 Functional description

The FlexCAN module is a CAN protocol engine with a very flexible mailbox system for transmitting and receiving CAN frames. The mailbox system is composed by a set of Message Buffers (MB) that store configuration and control data, time stamp, message ID and data (see Message buffer structure). The memory corresponding to the first 38 MBs can be configured to support a FIFO reception scheme with a powerful ID filtering mechanism, capable of checking incoming frames against a table of IDs (up to 128 extended IDs or 256 standard IDs or 512 8-bit ID slices), with individual mask register for up to 32 ID Filter Table elements.

Simultaneous reception through FIFO and mailbox is supported. For mailbox reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed on its ID field. A masking scheme makes it possible to match the ID programmed on the MB with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of MBs to be transmitted based on the message ID (optionally augmented by 3 local priority bits) or the MB ordering.

Before proceeding with the functional description, an important concept must be explained. A Message Buffer is said to be "active" at a given time if it can participate in both the Matching and Arbitration processes. An Rx MB with a 0b0000 code is inactive (refer to Table 50-5). Similarly, a Tx MB with a 0b1000 or 0b1001 code is also inactive (refer to Table 50-6).

50.5.1 Transmit process

To transmit a CAN frame, the CPU must prepare a Message Buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt bit is set and clear it.
2. If the MB is active (transmission pending), write the ABORT code (0b1001) to the CODE field of the Control and Status word to request an abort of the transmission.
3. Wait for the corresponding IFLAG bit to be asserted by polling the CAN_IFLAG register, or by the interrupt request if enabled by the respective IMASK bit.
4. Read back the CODE field to check if the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIO_EN]).
7. Write payload data bytes.
8. Configure the Control and Status word with the desired configuration.
   a. Set ID type via MB_CS[IDE].
   b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
   c. Set Data Length Code in bytes via MB_CS[DLC]. See Table 50-7 for detailed information.
   d. Activate the message buffer to transmit the CAN frame by setting MB_CS[CODE] to 0xC.

NOTE

It is strongly recommended that all the fields in MB_CS word be configured in only one 32-bit write operation to maximize software performance. If the fields are configured in separate writes, the MB_CS[CODE] must be the last write in the C/S word.

When the MB is activated, it participates in the arbitration process and is eventually transmitted according to its priority.

At the end of the successful transmission:

- The value of the Free Running Timer is written into the Time Stamp field.
- The CODE field in the Control and Status word is updated.
- A status flag is set in the Interrupt Flag register.
- An interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The new CODE field after transmission depends on the code that was used to activate the MB (see Table 50-5 and Table 50-6 in Message buffer structure).

When the Abort feature is enabled (CAN_MCR[AEN] is asserted), after the Interrupt Flag is asserted for a Mailbox configured as transmit buffer, the Mailbox is blocked. Therefore the CPU is not able to update it until the Interrupt Flag is negated by the CPU. This means that the CPU must clear the corresponding IFLAG bit before starting to prepare this MB for a new transmission or reception.

NOTE

If backwards compatibility is desired (CAN_MCR[AEN] bit is negated), write the INACTIVE code (0b1000) to the CODE field to inactivate the MB. However, in this case the pending frame may be transmitted without notification (see Mailbox inactivation).

50.5.2 Arbitration process

The arbitration process scans the Mailboxes searching the Tx one that holds the message to be sent in the next opportunity. This Mailbox is called the *arbitration winner*.

The scan starts from the lowest number Mailbox and runs toward the higher ones.

The arbitration process is triggered in the following events:

- From the CRC field of the CAN frame. The start point depends on the CAN_CTRL2[TASD] field value.
- During the Error Delimiter field of a CAN frame.
- During the Overload Delimiter field of a CAN frame.
- When the winner is inactivated and the CAN bus has still not reached the first bit of the Intermission field.
- When there is CPU write to the C/S word of a winner MB and the CAN bus has still not reached the first bit of the Intermission field.
- When CHI is in Idle state and the CPU writes to the C/S word of any MB.
- When FlexCAN exits Bus Off state.
- Upon leaving Freeze mode or Low Power mode.

If the arbitration process does not manage to evaluate all Mailboxes before the CAN bus has reached the first bit of the Intermission field the temporary arbitration winner is invalidated and the FlexCAN will not compete for the CAN bus in the next opportunity.

The arbitration process selects the winner among the active Tx Mailboxes at the end of the scan according to both CAN_CTRL1[LBUF] and CAN_MCR[LPRIOEN] bits settings.

50.5.2.1 Lowest-number Mailbox first

If CAN_CTRL1[LBUF] bit is asserted the first (lowest number) active Tx Mailbox found is the arbitration winner. CAN_MCR[LPRIOEN] bit has no effect when CAN_CTRL1[LBUF] is asserted.

50.5.2.2 Highest-priority Mailbox first

If CAN_CTRL1[LBUF] bit is negated, then the arbitration process searches the active Tx Mailbox with the highest priority, which means that this Mailbox's frame would have a higher probability to win the arbitration on CAN bus when multiple external nodes compete for the bus at the same time.

The sequence of bits considered for this arbitration is called the *arbitration value* of the Mailbox. The highest-priority Tx Mailbox is the one that has the lowest arbitration value among all Tx Mailboxes.

If two or more Mailboxes have equivalent arbitration values, the Mailbox with the lowest number is the arbitration winner.

The composition of the arbitration value depends on CAN_MCR[LPRIOEN] bit setting.

50.5.2.2.1 Local Priority disabled

If CAN_MCR[LPRIOEN] bit is negated the arbitration value is built in the exact sequence of bits as they would be transmitted in a CAN frame (see the following table) in such a way that the Local Priority is disabled.

Table 50-10. Composition of the arbitration value when Local Priority is disabled

Format	Mailbox Arbitration Value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

50.5.2.2.2 Local Priority enabled

If Local Priority is desired CAN_MCR[LPRIOEN] must be asserted. In this case the Mailbox PRIO field is included at the very left of the arbitration value (see the following table).

Table 50-11. Composition of the arbitration value when Local Priority is enabled

Format	Mailbox Arbitration Value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

As the PRIO field is the most significant part of the arbitration value Mailboxes with low PRIO values have higher priority than Mailboxes with high PRIO values regardless the rest of their arbitration values.

Note that the PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

50.5.2.3 Arbitration process (continued)

After the arbitration winner is found, its content is copied to a hidden auxiliary MB called Tx Serial Message Buffer (Tx SMB), which has the same structure as a normal MB but is not user accessible. This operation is called move-out and after it is done, write access to the C/S word of the corresponding MB is blocked (if the AEN bit in CAN_MCR register is asserted). Write access is restored in the following events:

- After the MB is transmitted and the corresponding IFLAG bit is cleared by the CPU
- FlexCAN enters in Freeze mode or Bus Off
- FlexCAN loses the bus arbitration or there is an error during the transmission

At the first opportunity window on the CAN bus, the message on the Tx SMB is transmitted according to the CAN protocol rules.

Arbitration process can be triggered in the following situations:

- During Rx and Tx frames from CAN CRC field to end of frame. CAN_CTRL2[TASD] bit value may be changed to optimize the arbitration start point.
- During CAN BusOff state from TX_ERR_CNT=124 to 128. CAN_CTRL2[TASD] bit value may be changed to optimize the arbitration start point.

- During C/S write by CPU in BusIdle. First C/S write starts arbitration process and a second C/S write during this same arbitration restarts the process. If other C/S writes are performed, Tx arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and BusIdle state starts then an arbitration process is triggered. In this case the first and second C/S write in BusIdle will not restart the arbitration process. It is possible that there is not enough time to finish arbitration in WaitForBusIdle state and the next state is Idle. In this case the scan is not interrupted, and it is completed during BusIdle state. During this arbitration C/S write does not cause arbitration restart.
- Arbitration winner deactivation during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the WaitForBusIdle state). If there is a re-synchronization during WaitForBusIdle, the arbitration process is restarted.

Arbitration process stops in the following situations:

- All Mailboxes were scanned
- A Tx active Mailbox is found in case of Lowest Buffer feature enabled
- Arbitration winner inactivation or abort during any arbitration process
- There was not enough time to finish Tx arbitration process (for instance, when a deactivation was performed near the end of frame). In this case arbitration process is pending.
- Error or Overload flag in the bus
- Low Power or Freeze mode request in Idle state

Arbitration is considered pending as described below:

- It was not possible to finish arbitration process in time
- C/S write during arbitration if write is performed in a MB whose number is lower than the Tx arbitration pointer
- Any C/S write if there is no Tx Arbitration process in progress
- Rx Match has just updated a Rx Code to Tx Code
- Entering Busoff state

C/S write during arbitration has the following effect:

- If C/S write is performed in the arbitration winner, a new process is restarted immediately.
- If C/S write is performed in a MB whose number is higher than the Tx arbitration pointer, the ongoing arbitration process will scan this MB as normal.

50.5.3 Receive process

To be able to receive CAN frames into a Mailbox, the CPU must prepare it for reception by executing the following steps:

1. If the Mailbox is active (either Tx or Rx) inactivate the Mailbox (see Mailbox inactivation), preferably with a safe inactivation (see Transmission abort mechanism).
2. Write the ID word
3. Write the EMPTY code (0b0100) to the CODE field of the Control and Status word to activate the Mailbox.

After the MB is activated, it will be able to receive frames that match the programmed filter. At the end of a successful reception, the Mailbox is updated by the *move-in* process (see Move-in) as follows:

1. The received Data field (8 bytes at most for Classical CAN message format) is stored.
2. The received Identifier field is stored.
3. The value of the Free Running Timer at the time of the second bit of frame's Identifier field is written into the Mailbox's Time Stamp field.
4. The received SRR, IDE, RTR and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 50-5 and Table 50-6 in Section Message buffer structure).
6. A status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The recommended way for CPU servicing (read) the frame received in an Mailbox is using the following procedure:

1. Read the Control and Status word of that Mailbox.
2. Check if the BUSY bit is deasserted, indicating that the Mailbox is locked. Repeat step 1) while it is asserted. See Mailbox lock mechanism.
3. Read the contents of the Mailbox. Once Mailbox is locked now, its contents won't be modified by FlexCAN Move-in processes. See Move-in.
4. Acknowledge the proper flag at IFLAG registers.

5. Read the Free Running Timer. It is optional but recommended to unlock Mailbox as soon as possible and make it available for reception.

The CPU should poll for frame reception by the status flag bit for the specific Mailbox in one of the IFLAG Registers and not by the CODE field of that Mailbox. Polling the CODE field does not work because once a frame was received and the CPU services the Mailbox (by reading the C/S word followed by unlocking the Mailbox), the CODE field will not return to EMPTY. It will remain FULL, as explained in Table 50-5 . If the CPU tries to workaround this behavior by writing to the C/S word to force an EMPTY code after reading the Mailbox without a prior *safe inactivation*, a newly received frame matching the filter of that Mailbox may be lost.

CAUTION

In summary: *never do polling by reading directly the C/S word of the Mailboxes. Instead, read the IFLAG registers.*

Note that the received frame's Identifier field is always stored in the matching Mailbox, thus the contents of the ID field in an Mailbox may change if the match was due to masking. When CAN_MCR[SRXDIS] bit is asserted, FlexCAN will not store frames transmitted by itself in any MB, even if it contains a matching Rx Mailbox, and no interrupt flag or interrupt signal will be generated. Otherwise, when CAN_MCR[SRXDIS] bit is deasserted, FlexCAN can receive frames transmitted by itself if there exists a matching Rx Mailbox.

To be able to receive CAN frames through the Rx FIFO, the CPU must enable and configure the Rx FIFO during Freeze mode (see Rx FIFO). Upon receiving the Frames Available in Rx FIFO interrupt (see the description of the BUF5I bit "Frames available in Rx FIFO" bit in the CAN_IFLAG1 register), the CPU should service the received frame using the following procedure:

1. Read the Control and Status word (optional: needed only if a mask was used for IDE and RTR bits)
2. Read the ID field (optional: needed only if a mask was used)
3. Read the Data field
4. Read the CAN_RXFIR register (optional)
5. Clear the Frames Available in Rx FIFO interrupt by writing 1 to CAN_IFLAG1[BUF5I] bit (mandatory: releases the MB and allows the CPU to read the next Rx FIFO entry)

When CAN_MCR[DMA] is asserted, upon receiving a frame in FIFO, CAN_IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Rx FIFO under DMA Operation). The CAN_IMASK1 bits in Rx FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 0x80 address, optional)
2. Read the ID field (read 0x84 address, optional)
3. Read all Data Bytes (start read at 0x88 address, optional)
4. Read the last Data Bytes (read 0x8C address is mandatory)

50.5.4 Matching process

The matching process scans the MB memory looking for Rx MBs programmed with the same ID as the one received from the CAN bus. If the FIFO is enabled, the priority of scanning can be selected between Mailboxes and FIFO filters. The matching starts from the lowest number Message Buffer toward the higher ones. If no match is found within the first structure then the other is scanned subsequently. In the event that the FIFO is full, the matching algorithm always looks for a matching MB outside the FIFO region.

As the frame is being received, it is stored in a hidden auxiliary MB called Rx Serial Message Buffer (Rx SMB).

The matching process start point depends on the following conditions:

- If the received frame is a remote frame, the start point is the CRC field of the frame
- If the received frame is a data frame with DLC field equal to zero, the start point is the CRC field of the frame
- If the received frame is a data frame with DLC field different than zero, the start point is the DATA field of the frame

If a matching ID is found in the FIFO table or in one of the Mailboxes, the contents of the Rx SMB are transferred to the FIFO or to the matched Mailbox by the move-in process. If any CAN protocol error is detected then no match results are transferred to the FIFO or to the matched Mailbox at the end of reception.

The matching process scans all matching elements of both Rx FIFO (if enabled) and the active Rx Mailboxes (CODE is EMPTY, FULL, OVERRUN or RANSWER) in search of a successful comparison with the matching elements of the Rx SMB that is receiving the

frame on the CAN bus. The Rx SMB has the same structure of a Mailbox. The reception structures (Rx FIFO or Mailboxes) associated with the matching elements that had a successful comparison are the *matched structures*. The *matching winner* is selected at the end of the scan among those matched structures and depends on conditions described ahead. See the following table.

Table 50-12. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	0	-	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY or FULL or OVERRUN
Mailbox	0	-	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	0	-	cmp	no_cmp	cmp	RANSWER
Mailbox	1	1	0	cmp	no_cmp	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
FIFO[5]	-	-	-	cmp_msk	cmp_msk	cmp_msk	-

1. For Mailbox structure, If SMB[IDE] is asserted, the ID is 29 bits (ID Standard + ID Extended). If SMB[IDE] is negated, the ID is only 11 bits (ID Standard). For FIFO structure, the ID depends on IDAM.
2. cmp: Compares the Rx SMB contents with the MB contents regardless the masks.
3. no_cmp: The Rx SMB contents are not compared with the MB contents.
4. cmp_msk: Compares the Rx SMB contents with MB contents taking into account the masks.
5. SMB[IDE] and SMB[RTR] are not taken into account when IDAM is type C.

A reception structure is *free-to-receive* when any of the following conditions is satisfied:

- The CODE field of the Mailbox is EMPTY
- The CODE field of the Mailbox is either FULL or OVERRUN and it has already been serviced (the C/S word was read by the CPU and unlocked as described in Mailbox lock mechanism)
- The CODE field of the Mailbox is either FULL or OVERRUN and an inactivation (see Mailbox inactivation) is performed
- The Rx FIFO is not full

The scan order for Mailboxes and Rx FIFO is from the matching element with lowest number to the higher ones.

The matching winner search for Mailboxes is affected by the CAN_MCR[IRMQ] bit. If it is negated, the matching winner is the first matched Mailbox regardless if it is free-to-receive or not. If it is asserted, the matching winner is selected according to the priority below:

1. the first free-to-receive matched Mailbox;
2. the last non free-to-receive matched Mailbox.

It is possible to select the priority of scan between Mailboxes and Rx FIFO by the CAN_CTRL2[MRP] bit.

If the selected priority is Rx FIFO first:

- If the Rx FIFO is a matched structure and is free-to-receive, then the Rx FIFO is the matching winner regardless of the scan for Mailboxes
- Otherwise (the Rx FIFO is not a matched structure or is not free-to-receive), then the matching winner is searched among Mailboxes as described above

If the selected priority is Mailboxes first:

- If a free-to-receive matched Mailbox is found, it is the matching winner regardless of the scan for Rx FIFO
- If no matched Mailbox is found, then the matching winner is searched in the scan for the Rx FIFO
- If both conditions above are not satisfied and a non free-to-receive matched Mailbox is found, then the matching winner determination is conditioned by the CAN_MCR[IRMQ] bit:
- - If CAN_MCR[IRMQ] bit is negated, the matching winner is the first matched Mailbox
  - If CAN_MCR[IRMQ] bit is asserted, the matching winner is the Rx FIFO if it is a free-to-receive matched structure; otherwise, the matching winner is the last non free-to-receive matched Mailbox

See the following table for a summary of matching possibilities.

Table 50-13. Matching possibilities and resulting reception structures

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
No FIFO, only MB, match is always MB first						
0	0	X[1]	None[2]	-[3]	None	Frame lost by no match
0	0	X	Free[4]	-	FirstMB	
0	1	X	None	-	None	Frame lost by no match
0	1	X	Free	-	FirstMb	
0	1	X	NotFree	-	LastMB	Overrun
FIFO enabled, no match in FIFO is as if FIFO does not exist						
1	0	X	None	None[5]	None	Frame lost by no match
1	0	X	Free	None	FirstMB	

Table continues on the next page...

Table 50-13. Matching possibilities and resulting reception structures (continued)

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
1	1	X	None	None	None	Frame lost by no match
1	1	X	Free	None	FirstMb	
1	1	X	NotFree	None	LastMB	Overrun
FIFO enabled, Queue disabled						
1	0	0	X	NotFull[6]	FIFO	
1	0	0	None	Full[7]	None	Frame lost by FIFO full (FIFO Overflow)
1	0	0	Free	Full	FirstMB	
1	0	0	NotFree	Full	FirstMB	
1	0	1	None	NotFull	FIFO	
1	0	1	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	0	1	Free	X	FirstMB	
1	0	1	NotFree	X	FirtsMb	Overrun
FIFO enabled, Queue enabled						
1	1	0	X	NotFull	FIFO	
1	1	0	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	1	0	Free	Full	FirstMB	
1	1	0	NotFree	Full	LastMb	Overrun
1	1	1	None	NotFull	FIFO	
1	1	1	Free	X	FirstMB	
1	1	1	NotFree	NotFull	FIFO	
1	1	1	NotFree	Full	LastMb	Overrun

1. This is a don't care condition.
2. Matched in MB "None" means that the frame has not matched any MB (free-to-receive or non-free-to-receive).
3. This is a forbidden condition.
4. Matched in MB "Free" means that the frame matched at least one MB free-to-receive regardless of whether it has matched MBs non-free-to-receive.
5. Matched in FIFO "None" means that the frame has not matched any filter in FIFO. It is as if the FIFO didn't exist (CAN_CTRL2[RFEN]=0).
6. Matched in FIFO "NotFull" means that the frame has matched a FIFO filter and has empty slots to receive it.
7. Matched in FIFO "Full" means that the frame has matched a FIFO filter but couldn't store it because it has no empty slots to receive it.

If a non-safe Mailbox inactivation (see Mailbox inactivation) occurs during matching process and the Mailbox inactivated is the temporary matching winner, then the temporary matching winner is invalidated. The matching elements scan is not stopped nor

restarted, it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist, therefore a message may be lost.

Suppose, for example, that the FIFO is disabled, IRMQ is enabled and there are two MBs with the same ID, and FlexCAN starts receiving messages with that ID. Let us say that these MBs are the second and the fifth in the array. When the first message arrives, the matching algorithm finds the first match in MB number 2. The code of this MB is EMPTY, so the message is stored there. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive", so it keeps looking, finds MB number 5 and stores the message there. If yet another message with the same ID arrives, the matching algorithm finds out that there are no matching MBs that are "free-to-receive", so it decides to overwrite the last matched MB, which is number 5. In doing so, it sets the CODE field of the MB to indicate OVERRUN.

The ability to match the same ID in more than one MB can be exploited to implement a reception queue (in addition to the full featured FIFO) to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the MBs. The CPU can examine the Time Stamp field of the MBs to determine the order in which the messages arrived.

Matching to a range of IDs is possible by using ID Acceptance Masks. FlexCAN supports individual masking per MB. See the description of the Rx Individual Mask Registers (CAN_RXIMRx). During the matching algorithm, if a mask bit is asserted, then the corresponding ID bit is compared. If the mask bit is negated, the corresponding ID bit is a "don't care". Note that the Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed while the module is in Freeze mode; otherwise, they are blocked by hardware.

FlexCAN also supports an alternate masking scheme with only four mask registers (CAN_RXFGMASK, CAN_RXMGMASK, CAN_RX14MASK and CAN_RX15MASK) for backwards compatibility with legacy applications. This alternate masking scheme is enabled when the IRMQ bit in the CAN_MCR Register is negated.

50.5.5 Move process

There are two types of move process: move-in and move-out.

50.5.5.1 Move-in

The move-in process is the copy of a message received by an Rx SMB to a Rx Mailbox or FIFO that has matched it. If the move destination is the Rx FIFO, attributes of the message are also copied to the CAN_RXFIR FIFO. Each Rx SMB has its own move-in process, but only one is performed at a given time as described ahead. The move-in starts only when the message held by the Rx SMB has a corresponding matching winner (see Matching process) and all of the following conditions are true:

- The CAN bus has reached or let past either:
  - The second bit of Intermission field next to the frame that carried the message that is in the Rx SMB
  - The first bit of an overload frame next to the frame that carried the message that is in the Rx SMB
- There is no ongoing matching process
- The destination Mailbox is not locked by the CPU
- There is no ongoing move-in process from another Rx SMB. If more than one move-in processes are to be started at the same time both are performed and the newest substitutes the oldest.

The term *pending move-in* is used throughout the documentation and stands for a move-to-be that still does not satisfy all of the aforementioned conditions.

The move-in is cancelled and the Rx SMB is able to receive another message if any of the following conditions is satisfied:

- The destination Mailbox is inactivated after the CAN bus has reached the first bit of Intermission field next to the frame that carried the message and its matching process has finished
- There is a previous pending move-in to the same destination Mailbox
- The Rx SMB is receiving a frame transmitted by the FlexCAN itself and the self-reception is disabled (CAN_MCR[SRXDIS] bit is asserted)
- Any CAN protocol error is detected

Note that the pending move-in is not cancelled if the module enters Freeze or Low-Power mode. It only stays on hold waiting for exiting Freeze and Low-Power mode and to be unlocked. If an MB is unlocked during Freeze mode, the move-in happens immediately.

The move-in process is the execution by the FlexCAN of the following steps:

1. Push IDHIT into the RXFIR FIFO if the message is destined to the Rx FIFO.
2. Read DATA0-3 and DATA4-7 words from the Rx SMB.
3. Write DATA0-3 and DATA4-7 words to the Rx Mailbox
4. Read the Control/Status and ID words from the Rx SMB.
5. Write Control/Status and ID words to the Rx Mailbox, and update the CODE field.

The move-in process is not atomic, in such a way that it is immediately cancelled by the inactivation of the destination Mailbox (see Mailbox inactivation) and in this case the Mailbox may be left partially updated, thus incoherent. The exception is if the move-in destination is an Rx FIFO Message Buffer, then the process cannot be cancelled.

The BUSY Bit (least significant bit of the CODE field) of the destination Message Buffer is asserted while the move-in is being performed to alert the CPU that the Message Buffer content is temporarily incoherent.

50.5.5.2 Move-out

The move-out process is the copy of the content from a Tx Mailbox to the Tx SMB when a message for transmission is available (see Section "Arbitration process"). The move-out occurs in the following conditions:

- The first bit of Intermission field
- During Bus Off state when TX Error Counter is in the 124 to 128 range
- During Bus Idle state
- During Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently out of Bus Idle state. In Bus Idle, the move-out has the lowest priority to the concurrent memory accesses.

50.5.6 Data coherence

In order to maintain data coherency and FlexCAN proper operation, the CPU must obey the rules described in Transmit process and Receive process.

50.5.6.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU if the transmission was aborted or if the frame could not be aborted and was transmitted instead.

Two primary conditions must be fulfilled in order to abort a transmission:

- CAN_MCR[AEN] bit must be asserted
- The first CPU action must be the writing of abort code (0b1001) into the CODE field of the Control and Status word.

Active MBs configured for transmission must be aborted first before they can be updated. If the abort code is written to a Mailbox that is currently being transmitted or to a Mailbox that was already loaded into the Tx SMB for transmission, the write operation is blocked and the transmission is not disturbed. However, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration
- There is an error during the transmission
- The module is put into Freeze mode
- The module enters the BusOff state
- There is an overload frame

If none of the conditions above are reached, the MB is transmitted correctly, the interrupt flag is set in the IFLAG register, and an interrupt to the CPU is generated (if enabled). The abort request is automatically cleared when the interrupt flag is set. On the other hand, if one of the above conditions is reached, the frame is not transmitted; therefore, the abort code is written into the CODE field, the interrupt flag is set in the IFLAG, and an interrupt is (optionally) generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, then the write operation is not blocked; therefore, the MB is updated and the interrupt flag is set. In this way the CPU just needs to read the abort code to make sure the active MB was *safely inactivated*. Although the AEN bit is asserted and the CPU wrote the abort code, in this case the MB is inactivated and not aborted, because the transmission did not start yet. One Mailbox is only aborted when the abort request is captured and kept pending until one of the previous conditions are satisfied.

50.5.6.2 Mailbox inactivation

Inactivation is a mechanism provided to protect the Mailbox against updates by the FlexCAN internal processes, thus allowing the CPU to rely on Mailbox data coherence after having updated it, even in Normal mode.

Inactivation of transmission Mailboxes must be performed just when MCR[AEN] bit is deasserted.

If a Mailbox is inactivated, it participates in neither the arbitration process nor the matching process until it is reactivated. See Transmit process and Receive process for more detailed instructions on how to inactivate and reactivate a Mailbox.

To inactivate a Mailbox, the CPU must update its CODE field to INACTIVE (either 0b0000 or 0b1000).

Because the user is not able to synchronize the CODE field update with the FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of the inactivated Rx Mailbox may be lost without notice, even if there are other Mailboxes with the same filter
- A frame containing the message within the inactivated Tx Mailbox may be transmitted without setting the respective IFLAG

In order to perform a *safe inactivation* and avoid the above consequences for Tx Mailboxes, the CPU must use the Transmission Abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the Mailbox (see Mailbox lock mechanism).

NOTE

Message Buffers that are part of the Rx FIFO cannot be inactivated. There is no write protection on the FIFO region by FlexCAN. CPU must maintain data coherency in the FIFO region when RFEN is asserted.

50.5.6.3 Mailbox lock mechanism

Other than Mailbox inactivation, FlexCAN has another data coherence mechanism for the receive process. When the CPU reads the Control and Status word of an Rx MB with codes FULL or OVERRUN, FlexCAN assumes that the CPU wants to read the whole MB in an atomic operation, and therefore it sets an internal lock flag for that MB. The lock is released when the CPU reads the Free Running Timer (global unlock operation), or when it reads the Control and Status word of another MB regardless of its code. A CPU write into the C/S word also unlocks the MB, but this procedure is not recommended for normal unlock use because it cancels a pending-move and potentially may lose a received message. The MB locking prevents a new frame from being written into the MB while the CPU is reading it.

NOTE

The locking mechanism applies only to Rx MBs that are not part of the FIFO and have a code different than INACTIVE (0b0000) or EMPTY[1] (0b0100). Also, Tx MBs can not be locked.

1. In previous FlexCAN versions, reading the C/S word locked the MB even if it was EMPTY. This behavior is maintained when the IRMQ bit is negated.

Suppose, for example, that the FIFO is disabled and the second and the fifth MBs of the array are programmed with the same ID, and FlexCAN has already received and stored messages into these two MBs. Suppose now that the CPU decides to read MB number 5 and at the same time another message with the same ID is arriving. When the CPU reads the Control and Status word of MB number 5, this MB is locked. The new message arrives and the matching algorithm finds out that there are no "free-to-receive" MBs, so it decides to override MB number 5. However, this MB is locked, so the new message can not be written there. It will remain in the Rx SMB waiting for the MB to be unlocked, and only then will be written to the MB.

If the MB is not unlocked in time and yet another new message with the same ID arrives, then the new message overwrites the one on the Rx SMB and there will be no indication of lost messages either in the CODE field of the MB or in the Error and Status Register.

While the message is being moved-in from the Rx SMB to the MB, the BUSY bit on the CODE field is asserted. If the CPU reads the Control and Status word and finds out that the BUSY bit is set, it should defer accessing the MB until the BUSY bit is negated.

Note

If the BUSY bit is asserted or if the MB is empty, then reading the Control and Status word does not lock the MB.

Inactivation takes precedence over locking. If the CPU inactivates a locked Rx MB, then its lock status is negated and the MB is marked as invalid for the current matching round. Any pending message on the Rx SMB will not be transferred anymore to the MB. An MB is unlocked when the CPU reads the Free Running Timer Register (see Section "Free Running Timer Register (CAN_TIMER)"), or the C/S word of another MB.

Lock and unlock mechanisms have the same functionality in both Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. However, the move-in is postponed if an unlock occurs during a low power mode (see Modes of operation), and it takes place only when the module resumes to Normal or Freeze modes.

50.5.7 Rx FIFO

The Rx FIFO is receive-only and is enabled by asserting the CAN_MCR[RFEN] bit. The reset value of this bit is zero to maintain software backward compatibility with previous versions of the module that did not have the FIFO feature.

The FIFO is 6-message deep. The memory region occupied by the FIFO structure (both Message Buffers and FIFO engine) is described in Rx FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a Message Buffer structure at the output of the FIFO.

The CAN_IFLAG1[BUF5I] (Frames available in Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO. An interrupt is generated if it is enabled by the corresponding mask bit. Upon receiving the interrupt, the CPU can read the message (accessing the output of the FIFO as a Message Buffer) and the CAN_RXFIR register and then clear the interrupt. If there are more messages in the FIFO the act of clearing the interrupt updates the output of the FIFO with the next message and update the CAN_RXFIR with the attributes of that message, reissuing the interrupt to the CPU. Otherwise, the flag remains negated. The output of the FIFO is only valid whilst the CAN_IFLAG1[BUF5I] is asserted.

The CAN_IFLAG1[BUF6I] (Rx FIFO Warning) is asserted when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. The flag remains asserted until the CPU clears it.

The CAN_IFLAG1[BUF7I] (Rx FIFO Overflow) is asserted when an incoming message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. The flag remains asserted until the CPU clears it.

Clearing one of those three flags does not affect the state of the other two.

An interrupt is generated if an IFLAG bit is asserted and the corresponding mask bit is asserted too.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing work load. The filtering criteria is specified by programming a table of up to 128 32-bit registers, according to CAN_CTRL2[RFFN] setting, that can be configured to one of the following formats (see also Rx FIFO structure):

- Format A: 128 IDAFs (extended or standard IDs including IDE and RTR)
- Format B: 256 IDAFs (standard IDs or extended 14-bit ID slices including IDE and RTR)
- Format C: 512 IDAFs (standard or extended 8-bit ID slices)

Note

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the FIFO has a corresponding IDHIT (Identifier Acceptance Filter Hit Indicator) that can read in the IDHIT field from C/S word, as shown in the Rx FIFO Structure description. Another way the CPU can obtain this information is by accessing the CAN_RXFIR register. The CAN_RXFIR[IDHIT] field refers to the message at the output of the FIFO and is valid while the CAN_IFLAG1[BUF5I] flag is asserted. The CAN_RXFIR register must be read only before clearing the flag, which guarantees that the information refers to the correct frame within the FIFO.

Up to 16 elements of the filter table are individually affected by the Individual Mask Registers (CAN_RXIMRx), according to the setting of CAN_CTRL2[RFFN], allowing very powerful filtering criteria to be defined. If the CAN_MCR[IRMQ] bit is negated, then the FIFO filter table is affected by CAN_RXFGMASK.

50.5.7.1 Rx FIFO under DMA Operation

The receive-only FIFO can support DMA, this feature is enabled by asserting both the CAN_MCR[RFEN] and CAN_MCR[DMA] bits. The reset value of CAN_MCR[DMA] bit is zero to maintain backward compatibility with previous versions of the module that did not have the DMA feature.

The DMA controller can read the received message by reading a Message Buffer structure at the FIFO output port at the 0x80-0x8C address range.

When CAN_MCR[DMA] is asserted the CPU must not access the FIFO output port address range. Before enabling the CAN_MCR[DMA], the CPU must service the IFLAGs asserted in the Rx FIFO region. Otherwise, these IFLAGs may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling the CAN_MCR[DMA], the CPU must perform a clear FIFO operation.

The CAN_IFLAG1[BUF5I] (Frames available in Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO, consequently a DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the FIFO as a Message Buffer). The DMA reading process must end by reading address 0x8C, which clears the CAN_IFLAG1[BUF5I] and updates both the FIFO output with the next message (if FIFO is not empty) and the CAN_RXFIR register with the attributes of the new message. If there are more messages stored in the FIFO, the CAN_IFLAG1[BUF5I] will be re-asserted and another DMA request is issued. Otherwise, the flag remains negated.

NOTE

CAN_RXFIR register contents cannot be read after DMA completes the FIFO read. The IDHIT information is also available in the C/S word at address 0x080 (see Rx FIFO structure.

The CAN_IFLAG1[BUF6I] and CAN_IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Rx FIFO interruption and must not clear the related IFLAGs. In addition, the related IMASKs are not used to mask the generation of DMA requests.

50.5.7.2 Clear FIFO Operation

When CAN_MCR[RFEN] is asserted, the clear FIFO operation is a feature used to empty FIFO contents. With CAN_MCR[RFEN] asserted the Clear FIFO occurs when the CPU writes 1 in CAN_IFLAG1[BUF0I]. This operation can only be performed in Freeze Mode and is blocked by hardware in other modes. This operation does not clear the FIFO IFLAGs, consequently the CPU must service all FIFO IFLAGs before execute the clear FIFO task.

When Rx FIFO is working with DMA, the clear FIFO operation clears the CAN_IFLAG1[BUF5I] and the DMA request is canceled.

CAUTION

Clear FIFO operation does not clear IFLAGs, except when CAN_MCR[DMA] is asserted, in this case only the CAN_IFLAG1[BUF5I] is cleared.

50.5.8 CAN protocol related features

This section describes the CAN protocol related features.

50.5.8.1 Remote frames

Remote frame is a special kind of frame. The user can program a mailbox to be a Remote Request Frame by configuring the mailbox as Transmit with the RTR bit set to '1'. After the remote request frame is transmitted successfully, the mailbox becomes a Receive Message Buffer, with the same ID as before.

When a remote request frame is received by FlexCAN, it can be treated in three ways, depending on Remote Request Storing (CTRL2[RRS]) and Rx FIFO Enable (MCR[RFEN]) bits:

- If RRS is negated the frame's ID is compared to the IDs of the Transmit Message Buffers with the CODE field 0b1010. If there is a matching ID, then this mailbox frame will be transmitted. Note that if the matching mailbox has the RTR bit set, then FlexCAN will transmit a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response. The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. In the case that a remote request frame is received and matches a mailbox, this message buffer immediately enters the internal arbitration process, but is considered as a normal Tx mailbox, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.
- If RRS is asserted the frame's ID is compared to the IDs of the receive mailboxes with the CODE field 0b0100, 0b0010 or 0b0110. If there is a matching ID, then this mailbox will store the remote frame in the same fashion of a data frame. No automatic remote response frame will be generated. The mask registers are used in the matching process.
- If RFEN is asserted FlexCAN will not generate an automatic response for remote request frames that match the FIFO filtering criteria. If the remote frame matches one of the target IDs, it will be stored in the FIFO and presented to the CPU. Note that for filtering formats A and B, it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted (if they match the ID). Remote Request Frames are considered as normal frames, and generate a FIFO overflow when a successful reception occurs and the FIFO is already full.

50.5.8.2 Overload frames

FlexCAN does transmit overload frames due to detection of following conditions on CAN bus:

- Detection of a dominant bit in the first/second bit of Intermission
- Detection of a dominant bit at the 7th bit (last) of End of Frame field (Rx frames)
- Detection of a dominant bit at the 8th bit (last) of Error Frame Delimiter or Overload Frame Delimiter

50.5.8.3 Time stamp

The value of the Free Running Timer is sampled at the beginning of the Identifier field on the CAN bus, and is stored at the end of "move-in" in the TIME STAMP field, providing network behavior with respect to time.

When the TIMER_SRC bit in CAN_CTRL2 register is asserted, the Free Running Timer is continuously clocked by an external time tick.

When the TIMER_SRC bit in CAN_CTRL2 register is negated, the Free Running Timer is clocked by the FlexCAN bit-clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The Free Running Timer is not incremented during Disable, Stop, and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See the TSYN description in Control 1 Register (CAN_CTRL1).

50.5.8.4 Protocol timing

The following figure shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule. The clock source bit CLKSRC in the CAN_CTRL1 Register defines whether the internal clock is connected to the output of a crystal oscillator (Oscillator Clock) or to the Peripheral Clock. In order to guarantee reliable operation, the clock source should be selected while the module is in Disable Mode (MDIS bit set in the Module Configuration Register).

NOTE

Please refer to the clock distribution chapter (module clocks table) to identify the proper clock source.

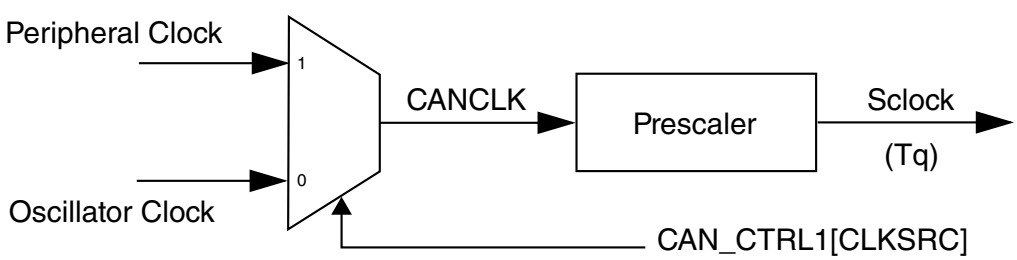


Figure 50-2. CAN engine clocking scheme

The oscillator clock should be selected whenever a tight tolerance (up to 0.1%) is required in the CAN bus timing. The crystal oscillator clock has better jitter performance than the peripheral clock.

The FlexCAN module supports a variety of means to setup bit timing parameters that are required by the CAN protocol. The Control 1 Register (CAN_CTRL1) has various fields used to control bit timing parameters: PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW.

The CAN Bit Timing register (CAN_CBT) extends the range of the CAN bit timing variables in CAN_CTRL1.

The PRESDIV field (as well as its extended range EPRESDIV) defines the Prescaler Value (see the equation below) that generates the Serial Clock (Sclock), whose period defines the 'time quantum' used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time handled by the CAN engine.

$$Tq = \frac{(\text{PRESDIV} + 1)}{f_{\text{CANCLK}}}$$

The bit rate, which defines the rate the CAN message is either received or transmitted, is given by the formula:

$$\text{CAN Bit Time} = (\text{Number of Time Quanta in 1 bit time}) * Tq$$

$$\text{Bit Rate} = \frac{1}{\text{CAN Bit Time}}$$

A bit time is subdivided into three segments[1] (see Figure 50-3 and Table 50-14):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section
- Time Segment 1: This segment includes the Propagation Segment and the Phase Segment 1 of the CAN standard. It can be programmed by setting the PROPSEG and the PSEG1 fields of the CAN_CTRL1 Register so that their sum (plus 2) is in the

1. For further explanation of the underlying concepts, see ISO 11898-1. See also the CAN 2.0A/B protocol specification for bit timing.

range of 2 to 16 time quanta. When CAN_CBT[BTF] bit is asserted, FlexCAN uses EPROPSEG and EPSEG1 fields from CAN_CBT register so that their sum (plus 2) is in the range of 2 to 96 time quanta.

- Time Segment 2: This segment represents the Phase Segment 2 of the CAN standard. It can be programmed by setting the PSEG2 field of the CAN_CTRL1 Register (plus 1) to be 2 to 8 time quanta long. When CAN_CBT[BTF] bit is asserted, FlexCAN uses EPSEG2 fields of CAN_CBT register so that its value (plus 1) is in the range of 2 to 32 time quanta. The Time Segment 2 cannot be smaller than the Information Processing Time (IPT), which value is 2 time quanta in FlexCAN.

NOTE

The bit time defined by the above time segments must not be smaller than 5 time quanta. For bit time calculations, use an Information Processing Time (IPT) of 2, which is the value implemented in the FlexCAN module.

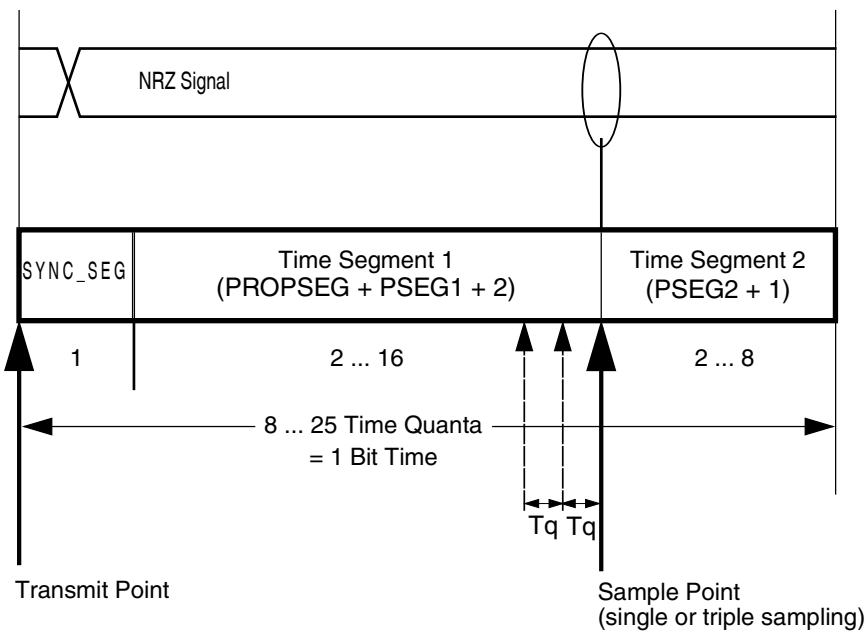


Figure 50-3. Segments within the bit time (example using CAN_CTRL1 bit timing variables for Classical CAN format)

Table 50-14. Time segment syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
TSEG1	Corresponds to the sum of PROPSEG and PSEG1.

Table continues on the next page...

Table 50-14. Time segment syntax (continued)

Syntax	Description
TSEG2	Corresponds to the PSEG2 value.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The following table gives some examples of the CAN compliant segment settings for Classical CAN format (Bosch CAN 2.0B) messages.

Table 50-15. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Re-synchronization jump width
5 .. 10	2	1 .. 2
4 .. 11	3	1 .. 3
5 .. 12	4	1 .. 4
6 .. 13	5	1 .. 4
7 .. 14	6	1 .. 4
8 .. 15	7	1 .. 4
9 .. 16	8	1 .. 4

Note

The user must ensure the bit time settings are in compliance with the CAN Protocol standard (ISO 11898-1).

Whenever CAN bit is used as a measure of time duration (e.g. estimating the occurrence of a CAN bit event in a message), the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$\text{NumClkBit} = \frac{f_{SYS}}{f_{CANCLK}} \text{ X } (\text{PRESDIV} + 1) \text{ X } (\text{PROPSEG} + \text{PSEG1} + \text{PSEG2} + 4)$$

where:

- NumClkBit is the number of peripheral clocks in one CAN bit;
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure "CAN Engine Clocking Scheme"), in Hz;
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz;
- PSEG1 is the value in CAN_CTRL1[PSEG1] field;
- PSEG2 is the value in CAN_CTRL1[PSEG2] field;

- PROPSEG is the value in CAN_CTRL1[PROPSEG] field;
- PRESDIV is the value in CAN_CTRL1[PRESDIV] field.

The formula above is also applicable to the alternative CAN bit timing variables described in the CAN Bit Timing Register (CAN_CBT).

For example, 180 CAN bits = (180 x NumClkBit) peripheral clock periods.

50.5.8.5 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in and move-out processes are executed during certain time windows inside the CAN frame, as shown in the following figures.

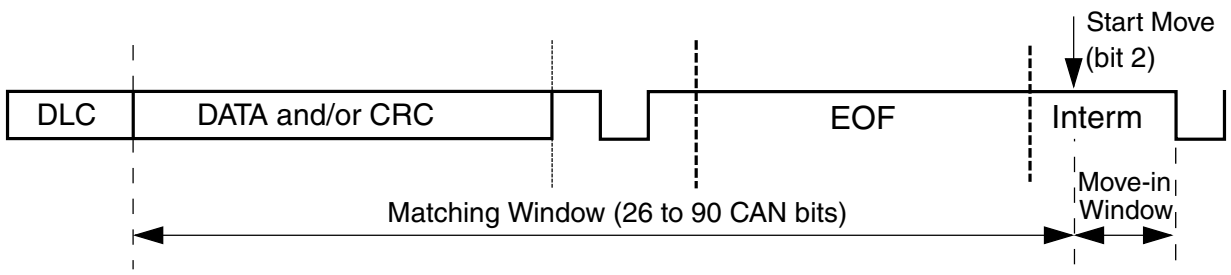


Figure 50-4. Matching and move-in time windows

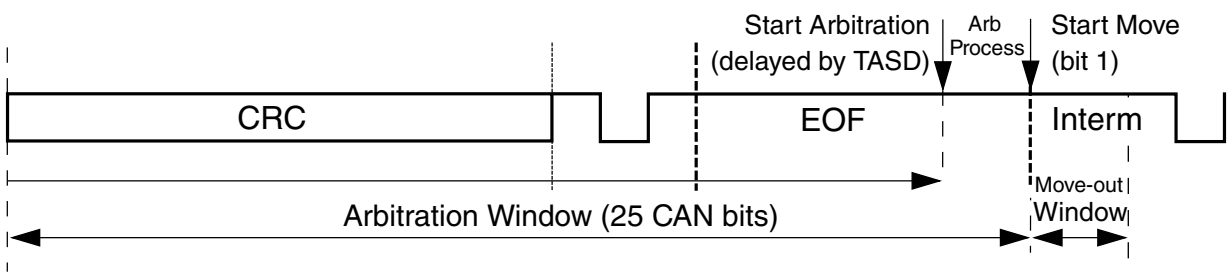


Figure 50-5. Arbitration and move-out time windows

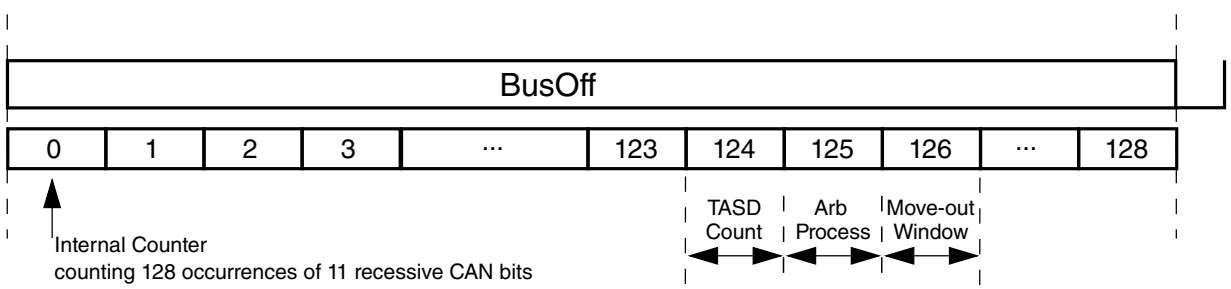


Figure 50-6. Arbitration at the end of bus off and move-out time windows

NOTE

In the preceding figures, the matching and arbitration timing does not take into account the delay caused by the concurrent memory access due to the CPU or other internal FlexCAN sub-blocks.

50.5.8.6 Tx Arbitration start delay

The Tx Arbitration Start Delay (TASD) bit field in Control 2 register (CAN_CTRL2[TASD]) is a variable that indicates the number of CAN bits used by FlexCAN to delay the Tx Arbitration process start point from the first bit of CRC field of the current frame. This variable can be written only in Freeze mode because it is blocked by hardware in other modes.

The transmission performance is impacted by the ability of the CPU to reconfigure Message Buffers (MBs) for transmission after the end of the internal Arbitration process, where FlexCAN finds the winner MB for transmission (see Arbitration process). If the Arbitration ends too early before the first bit of Intermission field, then there is a chance that the CPU reconfigures some Tx MBs and the winner MB is no longer the best candidate to be transmitted.

TASD is useful to optimize the transmission performance by defining the Arbitration start point, as shown in the next figure, based on factors such as:

- The peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- The number of Message Buffers (MBs) in use by the Matching and Arbitration processes.

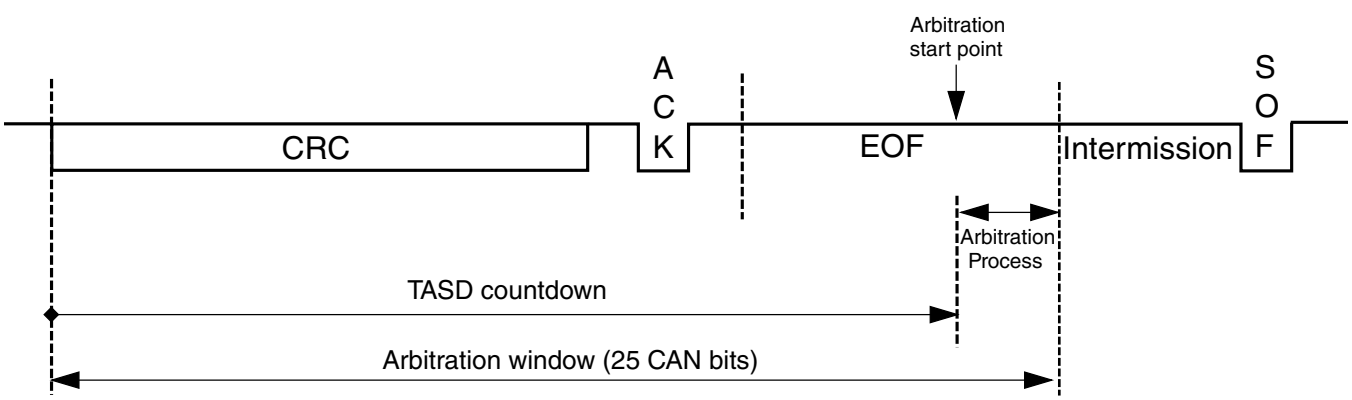


Figure 50-7. Optimal Tx Arbitration start point

The duration of an Arbitration process, in terms of CAN bits, is directly proportional to the number of available MBs and to the CAN bit rate, and inversely proportional to the peripheral clock frequency.

The optimal Arbitration timing is that in which the last MB is scanned right before the first bit of the Intermission field of a CAN frame. For instance, if there are few MBs and the peripheral/oscillator clock ratio is high and the CAN baud rate is low, then the Arbitration can be placed closer to the frame's end, adding more delay to its start point, and vice-versa.

If TASD is set to 0 then the Arbitration start is not delayed and more time is reserved for Arbitration. On the other hand, if TASD is close to 24 then the CPU can configure a Tx MB later and less time is reserved for Arbitration. If too little time is reserved for Arbitration the FlexCAN may be not be able to find a winner MB in time to be transmitted with the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

$$TASD = 25 - \left(\frac{f_{CANCLK} \times [\, MAXMB + 3 - (RFEN \times 8) - (RFEN \times RFFN \times 2)\,] \times 2}{f_{SYS} \times [1 + (PSEG1 + 1) + (PSEG2 + 1) + (PROPSEG + 1)\,] \times (PRESDIV + 1)} \right)$$

where:

- MAXMB is the value in CAN_CTRL1[MAXMB] field
- f_{CANCLK} is the oscillator clock, in Hz
- f_{SYS} is the peripheral clock, in Hz
- RFEN is the value in CAN_CTRL1[RFEN] bit
- RFFN is the value in CAN_CTRL2[RFFN] field
- PSEG1 is the value in CAN_CTRL1[PSEG1] field
- PSEG2 is the value in CAN_CTRL1[PSEG2] field
- PROPSEG is the value in CAN_CTRL1[PROPSEG] field
- PRESDIV is the value in CAN_CTRL1[PRESDIV] field

See also Protocol timing for more details.

50.5.9 Clock domains and restrictions

The FlexCAN module has two clock domains asynchronous to each other:

- The Bus Domain feeds the Control Host Interface (CHI) submodule and is derived from the peripheral clock.
- The Oscillator Domain feeds the CAN Protocol Engine (PE) submodule and is derived directly from a crystal oscillator clock, so that very low jitter performance can be achieved on the CAN bus.

When CAN_CTRL1[CLKSRC] bit is set, synchronous operation occurs because both domains are connected to the peripheral clock (creating a 1:1 ratio between the peripheral and oscillator clocks).

When the two domains are connected to clocks with different frequencies and/or phases, there are restrictions on the frequency relationship between the two clock domains. In the case of asynchronous operation, the Bus Domain clock frequency must always be greater than the Oscillator Domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When doing matching and arbitration, FlexCAN needs to scan the whole Message Buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. In order to have sufficient time to do that, the following requirements must be observed:

- The peripheral clock frequency can not be smaller than the oscillator clock frequency
- For 16 Mailboxes, the minimum number of peripheral clocks per CAN bit is 16

The minimum number of peripheral clocks per CAN bit determines the minimum peripheral clock frequency for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit, that can be defined by adjusting one or more of the bit timing values contained in either the Control 1 Register (CAN_CTRL1) or CAN Bit Time register (CAN_CBT). The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be 8; therefore, the oscillator clock frequency should be at least 8 times the CAN bit rate.

50.5.10 Modes of operation details

The FlexCAN module has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following sub-sections contain functional details on Freeze mode and the low-power modes.

CAUTION

"Permanent Dominant" failure on CAN Bus line is not supported by FlexCAN. If a Low-Power request or Freeze

mode request is done during a "Permanent Dominant", the corresponding acknowledge can never be asserted.

50.5.10.1 Freeze mode

This mode is requested either by the CPU through the assertion of the HALT bit in the CAN_MCR Register or when the chip is put into Debug mode. In both cases it is also necessary that the FRZ bit is asserted in the CAN_MCR Register and the module is not in a low-power mode.

The acknowledgement is obtained through the assertion by the FlexCAN of FRZ_ACK bit in the same register. The CPU must only consider the FlexCAN in Freeze mode when both request and acknowledgement conditions are satisfied.

When Freeze mode is requested, FlexCAN does the following:

- Waits to be in either Intermission, Passive Error, Bus Off or Idle state
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in does not prevent going to Freeze mode.
- Ignores the Rx input pin and drives the Tx pin as recessive
- Stops the prescaler, thus halting all CAN protocol activities
- Grants write access to the Error Counters Register, which is read-only in other modes
- Sets the NOT_RDY and FRZ_ACK bits in CAN_MCR

After requesting Freeze mode, the user must wait for the FRZ_ACK bit to be asserted in CAN_MCR before executing any other action, otherwise FlexCAN may operate in an unpredictable way. In Freeze mode, all memory mapped registers are accessible, except for CAN_CTRL1[CLKSRC] bit that can be read but cannot be written.

Exiting Freeze mode is done in one of the following ways:

- CPU negates the FRZ bit in the CAN_MCR Register
- The chip is removed from Debug Mode and/or the HALT bit is negated

The FRZ_ACK bit is negated after the protocol engine recognizes the negation of the freeze request. When out of Freeze mode, FlexCAN tries to re-synchronize to the CAN bus by waiting for 11 consecutive recessive bits.

50.5.10.2 Module Disable mode

This low power mode is normally used to temporarily disable a complete FlexCAN block, with no power consumption. It is requested by the CPU through the assertion of the CAN_MCR[MDIS] bit, and the acknowledgement is obtained through the assertion by the FlexCAN of the CAN_MCR[LPMACK] bit. The CPU must only consider the FlexCAN in Disable mode when both request and acknowledgement conditions are satisfied.

If the module is disabled during Freeze mode, it requests to disable the clocks to the PE and CHI sub-modules, sets the LPMACK bit and negates the FRZACK bit.

If the module is disabled during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and then checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Shuts down the clocks to the PE and CHI sub-modules
- Sets the NOTRDY and LPMACK bits in CAN_MCR

The Bus Interface Unit continues to operate, enabling the CPU to access memory mapped registers, except the Rx Mailboxes Global Mask Registers, the Rx Buffer 14 Mask Register, the Rx Buffer 15 Mask Register, the Rx FIFO Global Mask Register. The Rx FIFO Information Register, the Message Buffers, the Rx Individual Mask Registers, and the reserved words within RAM may not be accessed when the module is in Disable Mode. Exiting from this mode is done by negating the MDIS bit by the CPU, which causes the FlexCAN to request to resume the clocks and negate the LPMACK bit after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

50.5.10.3 Stop mode

This is a system low-power mode in which all chip clocks can be stopped for maximum power savings. The Stop mode is globally requested by the CPU and the acknowledgement is obtained through the assertion by the FlexCAN of a Stop Acknowledgement signal. The CPU must only consider the FlexCAN in Stop mode when both request and acknowledgement conditions are satisfied.

If FlexCAN receives the global Stop mode request during Freeze mode, it sets the LPMACK bit, negates the FRZACK bit and then sends the Stop Acknowledge signal to the CPU, in order to shut down the clocks globally.

If Stop mode is requested during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Sets the NOTRDY and LPMACK bits in CAN_MCR
- Sends a Stop Acknowledge signal to the CPU, so that it can shut down the clocks globally

Stop mode is exited when the CPU resumes the clocks and removes the Stop Mode request. This can be as a result of the Self Wake mechanism.

In the Self Wake mechanism, if the SLFWAK bit in CAN_MCR Register was set at the time FlexCAN entered Stop mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN sets the WAKINT bit in the CAN_ESR Register and, if enabled by the WAKMSK bit in CAN_MCR, generates a Wake Up interrupt to the CPU. Upon receiving the interrupt, the CPU should resume the clocks and remove the Stop mode request. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLFWAK and WAKMSK upon wake-up from Stop mode. Note that wake-up from Stop mode only works when both bits are asserted.

After the CAN protocol engine recognizes the negation of the Stop mode request, the FlexCAN negates the LPMACK bit. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up.

Table 50-16. Wake-up from Stop Mode

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No

Table continues on the next page...

Table 50-16. Wake-up from Stop Mode (continued)

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
1	0	1	No	No
1	1	0	No	No
1	1	1	Yes	Yes

The sensitivity to CAN bus activity can be modified by applying a low-pass filter function to the Rx CAN input line while in Stop mode. See the WAKSRC bit in in the description of the Module Configuration Register (CAN_MCR). This feature can be used to protect FlexCAN from waking up due to short glitches on the CAN bus lines. Such glitches can result from electromagnetic interference within noisy environments.

50.5.11 Interrupts

The module has many interrupt sources: interrupts due to message buffers and interrupts due to the ORed interrupts from MBs, Bus Off, Bus Off Done, Error, Wake Up, Tx Warning, and Rx Warning.

Each one of the message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between Tx and Rx interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each of the buffers has an assigned flag bit in the CAN_IFLAG registers. The bit is set when the corresponding buffer completes a successful transfer and is cleared when the CPU writes it to 1 (unless another interrupt is generated at the same time).

Note

It must be guaranteed that the CPU clears only the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

If the Rx FIFO is enabled (CAN_MCR[RFEN] = 1) and DMA is disabled (CAN_MCR[DMA] = 0), the interrupts corresponding to MBs 0 to 7 have different meanings. Bit 7 of the CAN_IFLAG1 register becomes the "FIFO Overflow" flag; bit 6 becomes the "FIFO Warning" flag, bit 5 becomes the "Frames Available in FIFO" flag and bits 4-0 are unused. See the description of the Interrupt Flags 1 Register (CAN_IFLAG1) for more information.

If both Rx FIFO and DMA are enabled (CAN_MCR[RFEN] and CAN_MCR[DMA] = 1) the FlexCAN does not generate any FIFO interrupt. Bit 5 of the CAN_IFLAG1 register still indicates "Frames Available in FIFO" and generates a DMA request. Bits 7, 6, 4-0 are unused.

For a combined interrupt where multiple MB interrupt sources are OR'd together, the interrupt is generated when any of the associated MBs (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the CAN_IFLAG registers to determine which MB or FIFO source caused the interrupt.

The interrupt sources for Bus Off, Bus Off Done, Error, Wake Up, Tx Warning and Rx Warning generate interrupts like the MB interrupt sources, and can be read from CAN_ESR1 register. The Bus Off, Error, Tx Warning, and Rx Warning interrupt mask bits are located in the CAN_CTRL1 Register; the Wake-Up interrupt mask bit is located in the CAN_MCR.

50.5.12 Bus interface

The CPU access to FlexCAN registers are subject to the following rules:

- Unrestricted read and write access to supervisor registers (registers identified with S/U in Table "Module Memory Map" in Supervisor Mode or with S only) results in access error.
- Read and write access to implemented reserved address space results in access error.
- Write access to positions whose bits are all currently read-only results in access error. If at least one of the bits is not read-only then no access error is issued. Write permission to positions or some of their bits can change depending on the mode of operation or transitory state. Refer to register and bit descriptions for details.
- Read and write access to unimplemented address space results in access error.
- Read and write access to RAM located positions during Low Power Mode results in access error.
- It is possible for the RXIMR memory region to be considered as general purpose memory and available for access. There are two ways of doing this:
  a. If CAN_MCR[IRMQ] is cleared, the individual masks (RXIMR) are disabled. In this case the RXIMR memory region is considered as general purpose memory.
  b. If CAN_MCR[MAXMB] is programmed with a value smaller than the available number of MBs, then the unused memory space can be used as general purpose RAM space. Note that reserved words within RAM cannot be used. As an example, suppose FlexCAN's RAM can support up to 16 MBs, CAN_CTRL2[RFFN] is 0x0, and CAN_MCR[MAXMB] is programmed with zero. The maximum number of MBs in this case becomes one. The RAM starts

at 0x0080, and the space from 0x0080 to 0x008F is used by the one MB. The memory space from 0x0090 to 0x017F is available. The space between 0x0180 and 0x087F is reserved. The space from 0x0880 to 0x0883 is used by the one Individual Mask and the available memory in the Mask Registers space would be from 0x0884 to 0x08BF. From 0x08C0 through 0x09DF there are reserved words for internal use which cannot be used as general purpose RAM. As a general rule, free memory space for general purpose depends only on MAXMB.

50.6 Initialization/application information

This section provide instructions for initializing the FlexCAN module.

50.6.1 FlexCAN initialization sequence

The FlexCAN module may be reset in three ways:

- Chip level hard reset, which resets all memory mapped registers asynchronously
- SOFTRST bit in MCR, which resets some of the memory mapped registers synchronously. See Table 50-3 to see what registers are affected by soft reset.

Soft reset is synchronous and has to follow an internal request/acknowledge procedure across clock domains. Therefore, it may take some time to fully propagate its effects. The CAN_MCR[SOFTRST] bit remains asserted while soft reset is pending, so software can poll this bit to know when the reset has completed. Also, soft reset can not be applied while clocks are shut down in a low power mode. The low power mode should be exited and the clocks resumed before applying soft reset.

The clock source should be selected while the module is in Disable mode (see CAN_CTRL1[CLKSRC] bit). After the clock source is selected and the module is enabled (CAN_MCR[MDIS] bit negated), FlexCAN automatically goes to Freeze mode. In Freeze mode, FlexCAN is un-synchronized to the CAN bus, the HALT and FRZ bits in CAN_MCR Register are set, the internal state machines are disabled and the FRZACK and NOTRDY bits in the CAN_MCR Register are set. The Tx pin is in recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Note that the Message Buffers and the Rx Individual Mask Registers are not affected by reset, so they are not automatically initialized.

For any configuration change/initialization it is required that FlexCAN is put into Freeze mode (see Freeze mode). The following is a generic initialization sequence applicable to the FlexCAN module:

- Initialize the Module Configuration Register (CAN_MCR)
  - Enable the individual filtering per MB and reception queue features by setting the IRMQ bit
  - Enable the warning interrupts by setting the WRNEN bit
  - If required, disable frame self reception by setting the SRXDIS bit
  - Enable the Rx FIFO by setting the RFEN bit
  - If Rx FIFO is enabled and DMA is required, set DMA bit
  - Enable the abort mechanism by setting the AEN bit
  - Enable the local priority feature by setting the LPRIOEN bit
- Initialize the Control 1 Register (CAN_CTRL1) and optionally the CAN Bit Timing Register (CAN_CBT).
  - Determine the bit timing parameters: PROPSEG, PSEG1, PSEG2, RJW
  - Optionally determine the bit timing parameters: EPROPSEG, EPSEG1, EPSEG2, ERJW
  - Determine the bit rate by programming the PRESDIV field and optionally the EPRESDIV field
  - Determine the internal arbitration mode (LBUF bit)
- Initialize the Message Buffers
  - The Control and Status word of all Message Buffers must be initialized
  - If Rx FIFO was enabled, the ID filter table must be initialized
  - Other entries in each Message Buffer should be initialized as required
- Initialize the Rx Individual Mask Registers (CAN_RXIMRn)
- Set required interrupt mask bits in the CAN_IMASK Registers (for all MB interrupts), in CAN_MCR Register for Wake-Up interrupt and in CAN_CTRL1 / CAN_CTRL2 Registers (for Bus Off and Error interrupts)
- Negate the HALT bit in CAN_MCR

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

50.7 Usage Guide

50.7.1 FlexCAN Interrupts

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

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

50.7.2 FlexCAN Operation in Low Power Modes

The FlexCAN module is operational in VLPR and VLPW modes. With the 2 MHz bus clock, the fastest supported FlexCAN transfer rate is 250 kbps. The bit timing parameters in the module must be adjusted for the new frequency, but full functionality is possible.

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

50.7.3 FlexCAN Doze Mode

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

Appendix A
Revision History

The following table provides a revision history for this document.

Table A-1. Revision History

Rev. No.	Date	Substantial Changes
2	09/2016	Initial public release.
3	06/2017	Some major updates after the market launch version (rev2).
3.1	07/2017	Some minor fixes in the Appendix B: Change Summary (for rev3).
3.2	07/2018	Some major updates after the rev3.
4	06/2019	Some major updates after the rev3.2, see the Appendix "Change Summary for This Revision" for more details.

Appendix B
Change Summary for This Revision

B.1 About This Manual chapter changes

- No substantial content changes

B.2 Introduction chapter changes

- No substantial content changes

B.3 Core Overview chapter changes

- No substantial content changes

B.4 Interrupts chapter changes

- No substantial content changes

B.5 SIM chapter changes

- No substantial content changes

B.6 MCM changes

- No substantial content changes

B.7 Crossbar switch module changes

- No substantial content changes

B.8 Memory Protection Unit (MPU) chapter changes

B.8.1 MPU chip-specific changes

- No substantial content changes

B.8.2 MPU module changes

- No substantial content changes

B.9 AIPS-Lite chapter changes

B.9.1 AIPS-Lite chip-specific changes

- No substantial content changes

B.9.2 AIPS-Lite module changes

- "Memory map/register definition" section restored in the AIPS chapter (the register part did not show up correctly in the former version).

B.10 TRGMUX chapter changes

- No substantial content changes

B.11 DMAMUX module changes

- No substantial content changes

B.12 eDMA module changes

- No substantial content changes

B.13 Memory and Memory Map chapter changes

- Minor updates in the table "Peripheral bridge slot assignments", in the section Peripheral Bridge (AIPS-Lite) Memory Map.

B.14 LMEM changes

- No substantial content changes

B.15 MSCM changes

- No substantial content changes

B.16 FAU chapter changes

- No substantial content changes

B.17 FTFE changes

- Replace FACNFG with FACSS and FACSN in Reset Sequence section
- Remove repeated section "Unsecuring the MCU Using Backdoor Key Access"

B.18 Clock Distribution chapter changes

- Minor update: FIRC is trimmed to 48 MHz only in this device, in the section Introduction.
- Minor fix in Figure 18-1.

B.19 SCG chapter changes

B.19.1 SCG chip-specific changes

- Note added in the section Information of SCG on this device : FIRC is trimmed to 48 MHz only in this device.
- Description added in the section Information of SCG on this device : For this device, low frequency range: 32 kHz - 40 kHz; medium frequency range: 4 MHz - 8 MHz; high frequency range: 8 MHz - 40 MHz.

B.19.2 SCG changes

- In Fast IRC Configuration Register (SCG_FIRCCFG), updated the bit field description of RANGE.

B.20 RTC Oscillator (OSC32K) changes

- Added a new note in the bit CR[ROSCEN]: It is necessary to set this bit even when the external 32k clock mode (ROSCEREFS=0) is in use, in the section OSC32 (which did not show up correctly in the Change Summary for KE1xF Reference Manual revision 3.2).

B.21 PCC chapter changes

- No substantial content changes

B.22 Reset and Boot chapter changes

- LVR statements clarified in the section Power-on reset (POR) and the section Boot sequence.

B.23 Kinetis ROM Bootloader changes

- In FlashEraseRegion command, for the "Protocol Sequence for FlashEraseRegion Command" figure, correction: from "0x5a a4 0c 00 f9 a6 02 00 00 00 00 00 00 00 04 00 00" to "0x5A A4 0C 00 F9 A6 02 00 00 02 00 00 00 00 00 04 00 00"
- In FlashEraseAllUnsecure command, for the "Protocol Sequence for FlashEraseAllUnsecure Command" figure, correction: from "0x5a a4 04 00 f6 61 0d 00 cc 00" to "0x5A A4 04 00 F6 61 0D 00 00 00 "
- In FlashSecurityDisable command, for the "Protocol Sequence for FlashSecurityDisable Command" figure, correction: from "0x5a a4 0c 00 43 7b 06 00 00 04 03 02 01 08 07 06 05" to "0x5A A4 0C 00 43 7B 06 00 00 02 04 03 02 01 08 07 06 05"
- In WriteMemory command, for the "Protocol Sequence for WriteMemory Command" figure, correction: from "0x5a a4 10 00 97 dd 04 01 00 03 00 04 00 20 64 00 00 00" to "0x5A A4 0C 00 06 5A 04 00 00 02 00 04 00 20 64 00 00 00"
- In Bootloader Command API section, added color to all protocol diagrams.
- In Ping packet section, added color to "Ping Packet Protocol Sequence" diagram.

- Updated figure "Protocol Sequence for FlashEraseRegion Command" and table "FlashEraseRegion Command Packet Format" in FlashEraseRegion command.

- Minor update in the section Read memory command : clarified as "Flash memory, SRAM_L and SRAM_U memory".
- Minor update in the section Kinetis Bootloader Status Error Codes : values not applicable to this device are removed from the table.

- Minor updates in the section The Kinetis Bootloader Configuration Area (BCA).

B.24 RCM changes

- No substantial content changes

B.25 Power Management chapter changes

- Updated the figure "Power Supply Supervisor", also with new note added, in the section Power supply supervisor.

B.26 System Mode Controller changes (SMC)

- No substantial content changes

B.27 PMC changes

- Minor updates about LVR and LVD statements, in sections Introduction, Features and Low Voltage Detect (LVD) System.

B.28 Security chapter changes

- No substantial content changes

B.29 EWM changes

- No substantial content changes

B.30 WDOG changes

- No substantial content changes

B.31 CRC changes

- No substantial content changes

B.32 Debug chapter changes

- No substantial content changes

B.33 JTAGC module changes

- No substantial content changes

B.34 Signal Multiplexing and Pin Assignment chapter changes

- No substantial content changes

B.35 Port Control and Interrupts (PORT) changes

- No substantial content changes

B.36 GPIO chapter changes

B.36.1 GPIO chip-specific changes

- No substantial content changes

B.36.2 GPIO changes

- No substantial content changes

B.37 ADC chapter changes

B.37.1 ADC chip-specific changes

- No substantial content changes

B.37.2 ADC changes

- No substantial content changes

B.38 CMP chapter changes

B.38.1 CMP chip-specific changes

- Corrected Vin1 and Vin2 connections in the chip-specific description, in the section CMP external references .

B.38.2 CMP changes

- No substantial content changes

B.39 DAC changes

- No substantial content changes

B.40 PDB changes

- No substantial content changes

B.41 FTM changes

- No substantial content changes

B.42 LPIT changes

- Note added in the register Current Timer Value (LPIT_CVAL*n*).

B.43 PWT changes

- No substantial content changes

B.44 LPTMR changes

- No substantial content changes

B.45 RTC changes

- No substantial content changes

B.46 LPSPI changes

- No substantial content changes

B.47 LPI2C changes

- No substantial content changes

B.48 LPUART changes

- No substantial content changes

B.49 FlexIO changes

- No substantial content changes

B.50 FlexCAN module changes

- Modified Transmit process.
- Added note "It is strongly recommended that all the fields in MB_CS word..." to Transmit process.
- Fixed issue in Message buffer structure. Bits MB[31], MB[30] and MB[29] are reserved for classical CAN instances.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number KE1xFP100M168SF0RM
Revision 4, 06/2019