MC9S12ZVC-Family Reference Manual and Datasheet

S12 MagniV
Microcontrollers

MC9S12ZVCRMV2
Rev. 2.1
19-March-2019

nxp.com

The MC9S12ZVC family of microcontrollers is targeted at use in safety relevant systems and has been developed using an ISO26262 compliant development system under the NXP Safe Assure Program. For more details of the NXP Safe Assure program, refer to : NXP Safe Assure

For more details of how to use the device in safety relevant systems refer to the MC9S12ZVC Safety Manual at nxp.com

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to nxp.com/

A full list of family members and options is included in device overview section.

The following revision history table summarizes changes contained in this document.

This document contains information for all constituent modules, with the exception of the S12Z CPU. For S12ZCPU information please refer to the CPU S12Z Reference Manual.

Revision History

Date	Revision Level	Description
22-August-2016	Rev 1.6	Added item 18 and 19 Table E-1 Bandgap voltage and temperature dependency Changed item 5 Table H-2 ACMP input offset Added operating condition for C part to Table A-5
13-October-2016	Rev 1.7	Corrected Table 1-1 Two SCIs for 48pin packages. Corrected typo in table H-2 item 5
2-January-2018	Rev 1.8	Corrected Package Information for 64LQFP Exposed Pad
29-January-2018	Rev 1.9	Corrected Package Information for 64LQFP Exposed Pad
26-March-2018	Rev 2.0	Updated Appendix A MCU Electrical Specifications
19-March-2019	Rev 2.1	Updated Appendix A MCU Electrical Specifications Updated Section Chapter 1, “Device Overview MC9S12ZVC-Family

NXP reserves the right to make changes without further notice to any products herein. NXP makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the NXP product could create a situation where personal injury or death may occur. Should Buyer purchase or use NXP products for any such unintended or unauthorized application, Buyer shall indemnify and hold NXP and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that NXP was negligent regarding the design or manufacture of the part.

Chapter 1
Device Overview MC9S12ZVC-Family

Chapter 2
Port Integration Module (S12ZVCPIMV1)

Chapter 3
Background Debug Controller (S12ZBDCV2)

Chapter 4
Memory Mapping Control (S12ZMMCV1)

Chapter 5
S12Z Interrupt (S12ZINTV0)

Chapter 6
S12Z DebugLite (S12ZDBGV3) Module

Chapter 7
ECC Generation Module (SRAM_ECCV1)

Chapter 8
S12 Clock, Reset and Power Management Unit (S12CPMU_UHV_V7)

Chapter 9
Analog-to-Digital Converter (ADC12B_LBA_V1)

Chapter 10
Supply Voltage Sensor - (BATSV3)

Chapter 11
Timer Module (TIM16B8CV3) Block Description

Chapter 12
Timer Module (TIM16B4CV3) Block Description

Chapter 13
Pulse-Width Modulator (S12PWM8B8CV2)

Chapter 14
Serial Communication Interface (S12SCIV6)

Chapter 15
Serial Peripheral Interface (S12SPIV5)

Chapter 16
Inter-Integrated Circuit (IICV3) Block Description

Chapter 17
CAN Physical Layer (S12CANPHYV3)

Chapter 18
Scalable Controller Area Network (S12MSCANV3)

Chapter 19
Digital Analog Converter (DAC_8B5V_V2)

Chapter 205V Analog Comparator (ACMPV2)

Chapter 21
SENT Transmitter Module (SENTTXV1)

Chapter 22
192 KB Flash Module (S12ZFTMRZ192K2KV2)

Appendix A
MCU Electrical Specifications

Appendix B
ADC Electrical Specifications

Appendix C
MSCAN Electrical Specifications

Appendix D
SPI Electrical Specifications

Appendix E
CPMU Electrical Specifications (VREG, OSC, IRC, PLL)

Appendix F
BATS Electrical Specifications

Appendix GPIM Electrical Specifications

Appendix HACMP Electrical Specifications

Appendix I
S12CANPHY Electrical Specifications

Appendix J
DAC8B5V Electrical Specifications

Appendix K
NVM Electrical Parameters

Appendix L
Package Information

Appendix M
Ordering Information

Appendix N
Detailed Register Address Map

Chapter 1
Device Overview MC9S12ZVC-Family

Revision History

Rev. No. (Item No.)	Date (Submitted By)	Substantial Change(s)
V0.01	9-April-2013	• Initial Version
V0.02	22-August-2013	• Added Section 1.13.3 Flash IFR Mapping • Updated Section 1.14.1 ADC Calibration
V0.03	10-September-2013	• Added Table 1-3 • Added S12ZVCA and S12ZVC Table 1-1
V0.04	7-February-2014	• Added 12K RAM, new maskset and part ID, feedback from shared review
V0.05	6-March-2014	• Changed maskset N23N • Changed Package 48 LQFP without EP
V0.06	28-April-2014	• Removed VRL functionality from PAD4
V0.07	19-April-2019	• Corrected Table 1-6 Pin Summary. Control register for internal pull devices for PAD6 - PAD11 • Corrected Table 1-16 Flash IFR mapping. ADC reference is right aligned.

1.1 Introduction

The MC9S12ZVC-Family is a new member of the S12 MagniV product line integrating a battery level (12V) voltage regulator, supply voltage monitoring, high voltage inputs and a CAN physical interface. It’s primarily targeting at CAN nodes like sensors, switch panels or small actuators. It offers various low-power modes and wakeup management to address state of the art power consumption requirements.

Some members of the MC9S12ZVC-Family are also offered for high temperature applications requiring AEC-Q100 Grade 0 (-40°C to +150°C ambient operating temperature range).

The MC9S12ZVC-Family is based on the enhanced performance, linear address space S12Z core and delivers an optimized solution with the integration of several key system components into a single device, optimizing system architecture and achieving significant space savings.

1.2 Features

This section describes the key features of the MC9S12ZVC-Family. It documents the superset of features within the family. Section 1.2.1 MC9S12ZVC-Family Comparison provides information to help access the correct information for a particular part within the family.

1.2.1 MC9S12ZVC-Family Comparison

Table 1-1 provides a summary of the MC9S12ZVC-Family. This information is intended to provide an understanding of the range of functionality offered by this microcontroller family.

Table 1-1. MC9S12ZVC-Family devices

Feature	S12ZVCA				S12ZVCA				S12ZVC				S12ZVC			
	192	**128**	**96**	**64**	**192**	**128**	**96**	**64**	**192**	**128**	**96**	**64**	**192**	**128**	**96**	**64**
Package option	64-pin LQFP-EP				48-pin LQFP				64-pin LQFP-EP				48-pin LQFP			
Temperature Option (°C ambient)	-40 to 105/125/150				-40 to 85/105/125				-40 to 105/125/150				-40 to 85/105/125			
Core	S12Z				S12Z				S12Z				S12Z			
Flash memory (ECC) [KByte]	192	128	96	64	192	128	96	64	192	128	96	64	192	128	96	64
EEPROM (ECC) [KByte] (4-byte erasable)	2	2	2	1	2	2	2	1	2	2	2	1	2	2	2	1
RAM (ECC) [KByte]	12	8	8	4	12	8	8	4	12	8	8	4	12	8	8	4
High Speed CAN Physical Layer	1				1				1				1			
High Voltage Inputs	2				2				2				2			
Vreg for CAN PHY with ext. ballast (BCTLC)	yes				yes				yes				yes			
VDDX/VSSX pins	2/2				2/2				2/2				2/2			
msCAN	1				1				1				1			
SCI	2				2				2				2			
SPI	2				1				2				1			
IIC	1				1				1				1			
SENT (Transmitter)	1				1				1				1			
16-bit Timer channels	8				4				8				4			
16-bit Timer channels (20 ns resolution[1])	4				4				4				4			
16-bit PWM channels (20 ns resolution(1))	4				3				4				3			
16-bit PWM channels	4				4				4				4			
12-bit ADC channels	16				10				-				-			
10-bit ADC channels	-				-				16				10			
8-bit DAC	1				1				-				-			

Table 1-1. MC9S12ZVC-Family devices

Feature	S12ZVCA				S12ZVCA				S12ZVC				S12ZVC			
	192	128	96	64	192	128	96	64	192	128	96	64	192	128	96	64
Package option	64-pin LQFP-EP				48-pin LQFP				64-pin LQFP-EP				48-pin LQFP			
Temperature Option (°C ambient)	-40 to 105/125/150				-40 to 85/105/125				-40 to 105/125/150				-40 to 85/105/125			
ACMP 5V (with rail-to-rail inputs)	2				2				-				-			
EVDD (20 mA source)	1				1				1				1			
Open Drain (5V GPIOs with disabled PMOS)	10				5				10				5			
N-GPIO (25mA sink)	4				4				4				4			
General purpose I/O	42				28				42				28			

[1] at 25 MHz bus frequency

1.3 Chip-Level Features

On-chip modules available within the family include the following features:

- S12Z CPU core
- Up to 192 Kbyte on-chip flash with ECC
- Up to 2 Kbyte EEPROM with ECC
- Up to 12Kbyte on-chip SRAM with ECC
- Phase locked loop (IPLL) frequency multiplier with internal filter
- 1 MHz internal RC oscillator with +/-1.3% accuracy over rated temperature range
- 4-20MHz amplitude controlled pierce oscillator
- Internal COP (watchdog) module
- Analog-to-digital converter (ADC) with 12-bit resolution and up to 16 channels available on external pins
- Two analog comparators (ACMP) with rail-to-rail inputs
- One 8-bit 5V digital-to-analog converter (DAC)
- Up to two serial peripheral interface (SPI) modules
- Up to two serial communication interface (SCI) modules
- SENT Transmitter Interface
- MSCAN (1 Mbit/s, CAN 2.0 A, B software compatible) module
- One on-chip CAN physical layer module
- 8-channel timer module (TIM0) with input capture/output compare
- 4-channel timer module (TIM1) with input capture/output compare (fast max 64MHz)
- Inter-IC (IIC) module
- 4-channel 16-bit Pulse Width Modulation module (PWM0)
- 4-channel 16-bit Pulse Width Modulation module (PWM1) (fast max 64MHz)

- On-chip voltage regulator (VREG) for regulation of input supply and all internal voltages
- Autonomous periodic interrupt (API), supports cyclic wakeup from Stop mode
- Four pins to support 25 mA drive strength to VSSX
- One pin to support 20 mA drive strength from VDDX (EVDD)
- Two High Voltage Input (HVI) pins
- Supply V_{SUP} monitoring with warning
- On-chip temperature sensor, temperature value can be measured with ADC or can generate a high temperature interrupt

1.4 Module Features

The following sections provide more details of the integrated modules.

1.4.1 S12Z Central Processor Unit (CPU)

The S12Z CPU is a revolutionary high-speed core, with code size and execution efficiencies over the S12X CPU. The S12Z CPU also provides a linear memory map eliminating the inconvenience and performance impact of page swapping.

- Harvard Architecture - parallel data and code access
- 3-stage pipeline
- 32-bit wide instruction and databus
- 32-bit ALU
- 24-bit addressing (16 MB linear address space)
- Instructions and Addressing modes optimized for C-Programming and Compiler
  - MAC unit 32bit += 32bit*32bit
  - Hardware divider
  - Single cycle multi-bit shifts (Barrel shifter)
  - Special instructions for fixed point math
- Unimplemented opcode traps
- Unprogrammed byte value (0xFF) defaults to SWI instruction

1.4.1.1 Background Debug Controller (BDC)

- Single-wire communication with host development system
- SYNC command to determine communication rate
- Genuine non-intrusive handshake protocol
- Enhanced handshake protocol for error detection and stop mode recognition
- Active out of reset in special single chip mode
- Most commands not requiring active BDM, for minimal CPU intervention
- Full global memory map access without paging

- Simple flash mass erase capability

1.4.1.2 Debugger (DBG)

- Three comparators (A, B and D)
  - Comparator A compare the full address bus and full 32-bit data bus
  - Comparators B and D compare the full address bus onlyEach comparator can be configured to monitor PC addresses or addresses of data accesses
  - Each comparator can select either read or write access cycles
  - Comparator matches can force state sequencer state transitions
- Three comparator modes
  - Simple address/data comparator match mode
  - Inside address range mode, Addmin ≤ Address ≤ Addmax
  - Outside address range match mode, Address < Addmin or Address > Addmax
- State sequencer control
  - State transitions forced by comparator matches
  - State transitions forced by software write to TRIGState transitions forced by an external event
- The following types of breakpoints
  - CPU breakpoint entering active BDM on breakpoint (BDM)
  - CPU breakpoint executing SWI on breakpoint (SWI)

1.4.2 Embedded Memory

1.4.2.1 Memory Access Integrity

- Illegal address detection
- ECC support on embedded NVM and system RAM

1.4.2.2 Flash

On-chip flash memory on the MC9S12ZVC-Family on the features the following:

- Up to 192Kbytes of program flash memory
  - Automated program and erase algorithm
  - Protection scheme to prevent accidental program or erase

1.4.2.3 EEPROM

- 2 Kbytes EEPROM
  - 16 data bits plus 6 syndrome ECC (error correction code) bits allow single bit error correction and double fault detection
  - Erase sector size 4 bytes
  - Automated program and erase algorithm

  - User margin level setting for reads

1.4.2.4 SRAM

- 12 Kbytes of general-purpose RAM with ECC
  - Single bit error correction and double bit error detection code based on 16-bit data words

1.4.3 Clocks, Reset and Power Management Unit (CPMU)

- Real time interrupt (RTI)
- Clock monitor, supervising the correct function of the oscillator (CM)
- Computer operating properly (COP) watchdog
  - Configurable as window COP for enhanced failure detection
  - Can be initialized out of reset using option bits located in flash memory
- System reset generation
- Autonomous periodic interrupt (API) (combination with cyclic, watchdog)
- Low Power Operation
  - RUN mode is the main full performance operating mode with the entire device clocked.
  - WAIT mode when the internal CPU clock is switched off, so the CPU does not execute instructions.
  - Pseudo STOP - system clocks are stopped but the oscillator the RTI, the COP, and API modules can be enabled
  - STOP - the oscillator is stopped in this mode, all clocks are switched off and all counters and dividers remain frozen, with the exception of the COP and API which can optionally run from ACLK.

1.4.3.1 Internal Phase-Locked Loop (IPLL)

- Phase-locked-loop clock frequency multiplier
  - No external components required
  - Reference divider and multiplier allow large variety of clock rates
  - Automatic bandwidth control mode for low-jitter operation
  - Automatic frequency lock detector
  - Configurable option to spread spectrum for reduced EMC radiation (frequency modulation)
  - Reference clock sources:
    - Internal 1 MHz RC oscillator (IRC)
    - External 4-16MHz crystal oscillator/resonator

1.4.3.2 Internal RC Oscillator (IRC)

- 1 MHz internal RC oscillator with +/-1.3% accuracy over rated temperature range

1.4.4 Main External Oscillator (XOSCLCP)

- Amplitude controlled Pierce oscillator using 4 MHz to 20 MHz crystal
  - Current gain control on amplitude output
  - Signal with low harmonic distortion
  - Low power
  - Good noise immunity
  - Eliminates need for external current limiting resistor
  - Transconductance sized for optimum start-up margin for typical crystals
  - Oscillator pins shared with GPIO functionality

1.4.5 Timer (TIM0 and TIM1)

- two independent timer modules with own 16-bit free-running counter and with 8-bit precision prescaler
  - 8 x 16-bit channels Timer module (TIM0) for input capture or output compare, channel 7 supports pulse accumulator feature
  - 4 x 16-bit channels fast Timer module (TIM1) for input capture or output compare

1.4.6 Pulse Width Modulation Module (PWM0 and PWM1)

- Four channel x 16-bit pulse width modulator PWM0
- Four channel x 16-bit pulse width modulator PWM1 fast max 64MHz
  - Programmable period and duty cycle per channel
  - Center-aligned or left-aligned outputs
  - Programmable clock select logic with a wide range of frequencies

1.4.7 Inter-IC Module (IIC)

- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Broadcast mode support
- 10-bit address support

1.4.8 CAN Physical Layer (CANPHY)

- High speed CAN interface for baud rates of up to 1 Mbit/s
- ISO 11898-2 and ISO 11898-5 compliant for 12 V battery systems
- SPLIT pin driver for bus recessive level stabilization
- Low power mode with remote CAN wake-up handled by MSCAN module
- Over-current shutdown for CANH and CANL
- Voltage monitoring on CANH and CANL

- CPTXD-dominant timeout feature monitoring the CPTXD signal
- Fulfills the OEM “Hardware Requirements for (LIN,) CAN (and FlexRay) Interfaces in Automotive Applications” v1.3

1.4.9 Multi-Scalable Controller Area Network (MSCAN)

- Implementation of CAN protocol - Version 2.0A/B
- Five receive buffers with FIFO storage scheme
- Three transmit buffers with internal prioritization using “local priority” concept
- Flexible maskable identifier filter supports two full-size (32-bit) extended identifier filters, or four 16-bit filters, or eight 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter

1.4.10 SENT Transmitter (SENT_TX)

- Features a 13-bit prescaler to derive a SENT-protocol compatible time unit of 3 to 90 µs from bus-clock.
- Programmable number of transmitted data-nibbles (1 to 6).
- Provides hardware to support SAE J2716-2010 (SENT) Fast Channel communication.
- CRC nibble generation:
  - Supports SENT legacy method CRC generation in hardware.
  - Supports SENT recommended method CRC generation in hardware.
  - Optionally, the SENT configuration- and status-nibble can be included in the automatic calculation of the CRC nibble.
  - Automatic CRC generation hardware can be bypassed to supply the CRC nibble directly from software
- Supports optional pause-pulse generation. The optional pause pulse can have a fixed length or its length can be automatically adapted to get a fixed overall message length.
- Supports both continuous and software-triggered transmission.
- Interrupt-driven operation with five flags:
  - Transmit buffer empty
  - Transmission complete
  - Calibration start
  - Transmitter underrun
  - Pause-pulse Rising-Edge

1.4.11 Serial Communication Interface Module (SCI)

- Full-duplex or single-wire operation
- Standard mark/space non-return-to-zero (NRZ) format
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse widths
- Baud rate generator by a 16-bit divider from the bus clock
- Programmable character length
- Programmable polarity for transmitter and receiver
- Active-edge receive wakeup
- Break detect and transmit collision detect supporting LIN

1.4.12 Serial Peripheral Interface Module (SPI)

- Configurable 8- or 16-bit data size
- Full-duplex or single-wire bidirectional
- Double-buffered transmit and receive
- Master or slave mode
- MSB-first or LSB-first shifting
- Serial clock phase and polarity options

1.4.13 Analog-to-Digital Converter Module (ADC)

- 12-bit resolution
- Up to 16 external channels and 8 internal channels
- Left or right aligned result data
- Continuous conversion mode
- Programmers model with list based command and result storage architecture
- ADC directly writes results to RAM, preventing stall of further conversions
- Internal signals monitored with the ADC module
  - VRH, VRL, (VRL+VRH)/2, VSUP monitor, VBG, Temperature Sensor, Port L HVI inputs
- External pins can also be used as digital I/O
- Up to 16 pins can be used as keyboard wake-up interrupt (KWU)

1.4.14 Digital-to-Analog Converter Module (DAC)

- 8-bit resolution
- Buffered and unbuffered analog output voltage usable
- Operational amplifier stand alone usable

1.4.15 Analog Comparator Module (ACMP)

- 0V to VDDA supply rail-to-rail inputs
- Low offset
- Up to 4 inputs selectable as inverting and non-inverting comparator inputs:
  - 2 low-impedance inputs with selectable low pass filter for external pins
  - 2 high-impedance inputs with fixed filter for SoC-internal signals
- Selectable hysteresis
- Selectable interrupt on rising edge, falling edge, or rising and falling edges of comparator output
- Option to output comparator signal on an external pin with selectable polarity
- Support for triggering timer input capture events
- Operational over supply range from 3V-5% to 5V+10%

1.4.16 Supply Voltage Sensor (BATS)

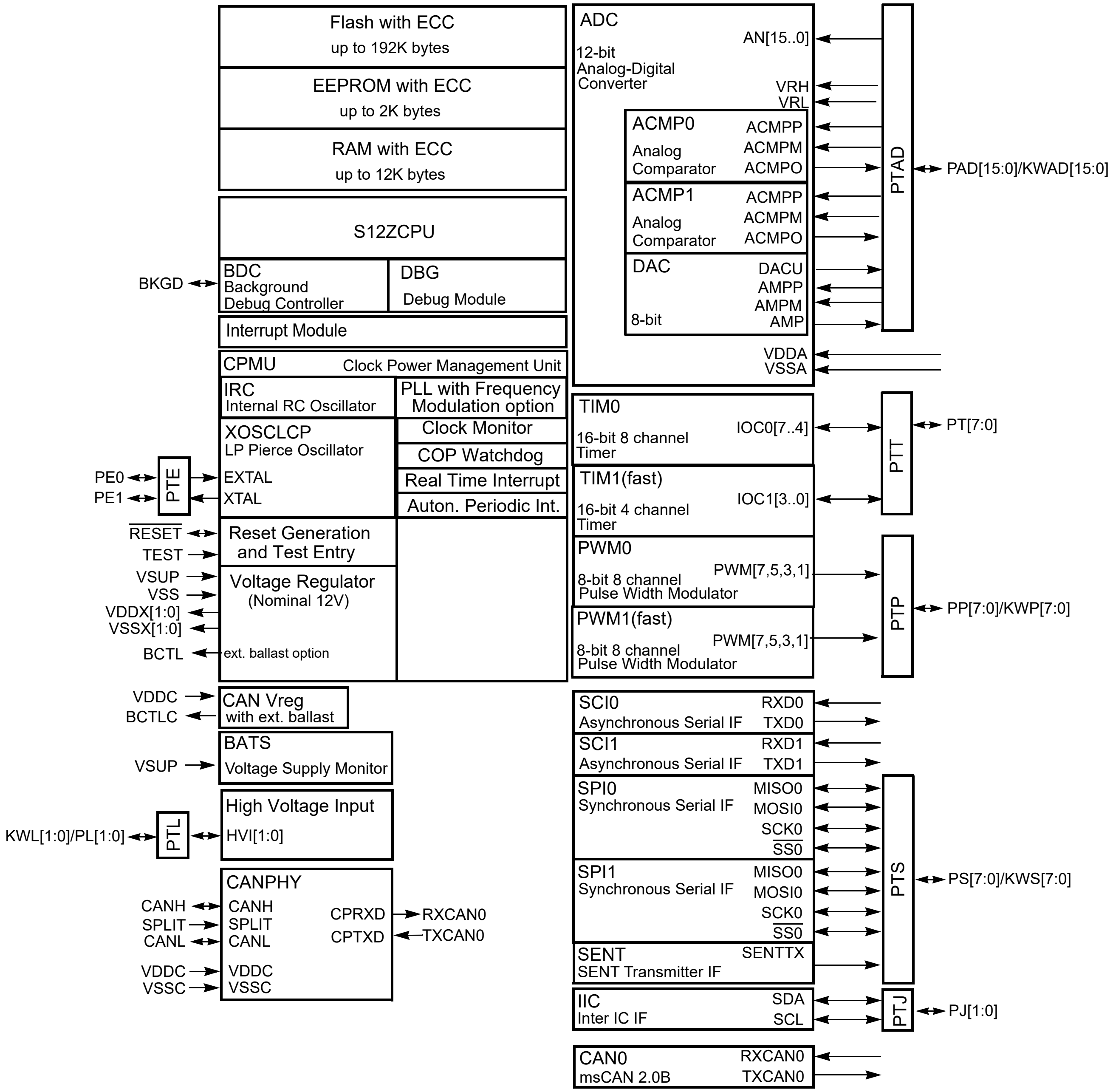
- Monitoring of supply (VSUP) voltage
- Internal ADC interface from an internal resistive divider
- Generation of low or high voltage interrupts

1.4.17 On-Chip Voltage Regulator system (VREG)

- Voltage regulator
  - Linear voltage regulator directly supplied by VSUP
  - Low-voltage detect on VSUP
  - Power-on reset (POR)
  - Low-voltage reset (LVR) for VDDX domain
  - External ballast device support to extend current capability and reduce internal power dissipation
  - Capable of supplying both the MCU internally plus external components
  - Over-temperature interrupt
- Internal voltage regulator
  - Linear voltage regulator with bandgap reference
  - Low-voltage detect on VDDA
  - Power-on reset (POR) circuit
  - Low-voltage reset for VDD, VDDF & VDDX domain

1.5 Block Diagram

Figure 1 shows a high-level block diagram of the MC9S12ZVC-Family.



Block Diagram shows the maximum configuration!
Not all pins or all peripherals are available on all devices and packages.
Rerouting options are not shown.

Figure 1. MC9S12ZVC-Family Block Diagram

1.6 Family Memory Map

Table 1-2.shows the MC9S12ZVC-Family register memory map.

Table 1-2. Device Register Memory Map

Address	Module	Size (Bytes)
0x0000–0x0003	ID Register	4
0x0004–0x000F	Reserved	12
0x0010–0x001F	INT	16
0x0020–0x006F	Reserved	80
0x0070–0x008F	MMC	32
0x0090–0x00FF	MMC Reserved	112
0x0100–0x017F	DBG	128
0x0180–0x01FF	Reserved	128
0x0200–0x037F	PIM	380
0x0380–0x039F	FTMRZ	32
0x03A0–0x03BF	Reserved	32
0x03C0–0x03CF	RAM ECC	16
0x03D0–0x03FF	Reserved	48
0x0400–0x042F	TIM1	48
0x0430–0x047F	Reserved	48
0x0480–0x04AF	PWM0	48
0x04B0–0x04FF	Reserved	80
0x0500–0x052F	PWM1	48
0x0530–0x05BF	Reserved	144
0x05C0–0x05EF	TIM0	48
0x05F0–0x05FF	Reserved	16
0x0600–0x063F	ADC0	64
0x0640–0x067F	Reserved	64
0x0680–0x0687	DAC	8
0x0688–0x068F	Reserved	128
0x0690–0x0697	ACMP0	8
0x0698–0x069F	ACMP1	8
0x06A0–0x06BF	Reserved	32
0x06C0–0x06DF	CPMU	32
0x06E0–0x06EF	Reserved	16
0x06F0–0x06F7	BATS	8
0x06F8–0x06FF	Reserved	8
0x0700–0x0707	SCI0	8
0x0708–0x070F	Reserved	8
0x0710–0x0717	SCI1	8

Address	Module	Size (Bytes)
0x0718–0x077F	Reserved	104
0x0780–0x0787	SPI0	8
0x0788–0x078F	Reserved	8
0x0790–0x0797	SPI1	8
0x0798–0x07BF	Reserved	32
0x07C0–0x07C7	IIC0	8
0x07C8–0x097F	Reserved	56
0x0800–0x083F	CAN0	64
0x0840–0x098F	Reserved	336
0x0990–0x0997	CANPHY0	8
0x0998–0x099F		8
0x09A0–0x09AF	SENTTX	16
0x09B0–0x0FFF		1616

NOTE

Reserved register space shown in Table 1-2. is not allocated to any module. This register space is reserved for future use. Writing to these locations has no effect. Read access to these locations returns zero.

Table 1-3. S12ZVC-Family Memory Address Ranges

Device	Address	Memory Block	Size (Bytes)
MC9S12ZVC64 & MC9S12ZVCA64	0x00_1000 - 0x00_1FFF	SRAM	4K
	0x10_0000 - 0x10_03FF	EEPROM	1K
	0xFF_0000 - 0xFF_FFFF	Program Flash	64K
MC9S12ZVC96 & MC9S12ZVCA96	0x00_1000 - 0x00_2FFF	SRAM	8K
	0x10_0000 - 0x10_07FF	EEPROM	2K
	0xFE_8000 - 0xFF_FFFF	Program Flash	96K
MC9S12ZVC128 & MC9S12ZVCA128	0x00_1000 - 0x00_2FFF	SRAM	8K
	0x10_0000 - 0x10_07FF	EEPROM	2K
	0xFE_0000 - 0xFF_FFFF	Program Flash	128K
MC9S12ZVC192 & MC9S12ZVCA192	0x00_1000 - 0x00_3FFF	SRAM	12K
	0x10_0000 - 0x10_07FF	EEPROM	2K
	0xFD_0000 - 0xFF_FFFF	Program Flash	192K

Figure 1-2. shows S12Z CPU global memory map as a graphical representation.

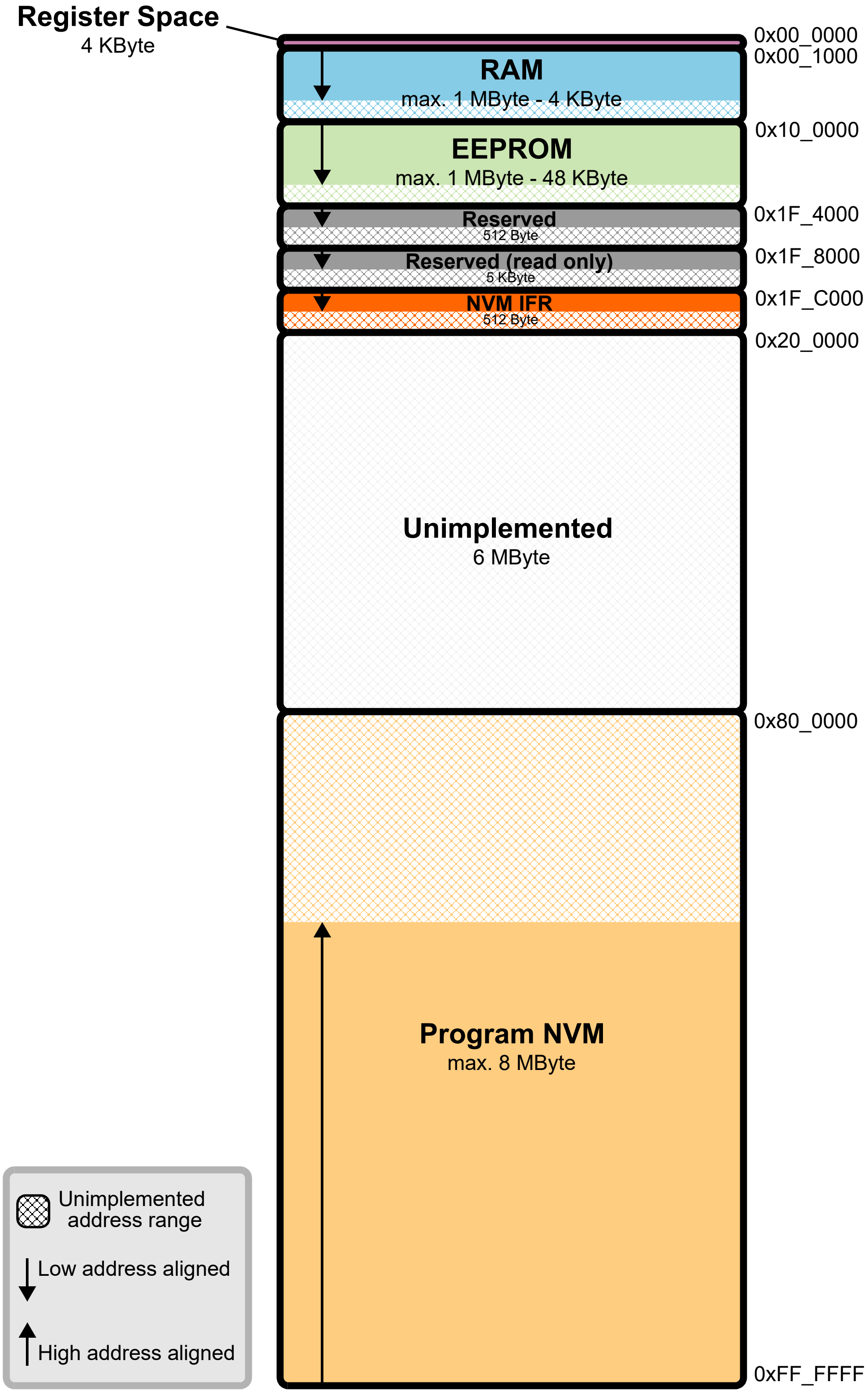


Figure 1-2. MC9S12ZVC-Family Global Memory Map

1.6.1 Part ID Assignments

The ID is located in four 8-bit registers at addresses 0x0000 to 0x0003. The read-only value is a unique part ID for each revision of the chip. Table 1-4 shows the assigned Part ID register value.

Table 1-4. Assigned ID Numbers

Device	Mask Set Number	Part ID
MC9S12ZVCA64	N23N	05.1D.10.00
MC9S12ZVCA96	N23N	05.1D.10.00
MC9S12ZVCA128	N23N	05.1D.10.00
MC9S12ZVCA192	N23N	05.1D.10.00
MC9S12ZVC64	N23N	05.1D.10.00
MC9S12ZVC96	N23N	05.1D.10.00
MC9S12ZVC128	N23N	05.1D.10.00
MC9S12ZVC192	N23N	05.1D.10.00

1.7 Signal Description and Device Pinouts

This section describes signals that connect off-chip. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals. It is built from the signal description sections of the individual IP blocks on the device.

NOTE

> To avoid current drawn from floating inputs, all non-bonded pins should be configured as output or configured as input with a pull up or pull down device enabled.

1.7.1 Pin Assignment Overview

Table 1-5. - Port Availability by package option

Port	64 LQFP-EP	48 LQFP
Port AD	PAD[15:0]	PAD[9:0]
Port E	PE[1:0]	PE[1:0]
Port L (HVI)	PL[1:0]	PL[1:0]
Port J	PJ[1:0]	—
Port P	PP[7:0]	PP[6:4,2, 0]
Port S	PS[7:0]	PS[7,3:0]
Port T	PT[7:0]	PT[7,4:0]
sum of ports	46	30

1.7.2 Detailed Signal Descriptions

This section describes the signal properties.

1.7.2.1 $\overline{\text{RESET}}$ — External Reset signal

The $\overline{\text{RESET}}$ signal is an active low bidirectional control signal. It acts as an input to initialize the MCU to a known start-up state, and an output when an internal MCU function causes a reset. The $\overline{\text{RESET}}$ pin has an internal pull-up device.

1.7.2.2 TEST — Test pin

This input only pin is reserved for factory test. This pin has an internal pull-down device.

NOTE

The TEST pin must be tied to ground in all applications.

1.7.3 MODC — Mode C signal

The MODC signal is used as a MCU operating mode select during reset. The state of this signal is latched to the MODC bit at the rising edge of $\overline{\text{RESET}}$. The signal has an internal pull-up device.

1.7.4 PAD[15:0] / KWAD[15:0] — Port AD, input pins of ADC

PAD[15:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWAD[15:0]). These signals can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

1.7.5 PE[1:0] — Port E I/O signals

PE[1:0] are general-purpose input or output signals. They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull down devices are enabled.

1.7.6 PJ[1:0] — Port J I/O signals

PJ[1:0] are general-purpose input or output signals. These signals can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull-up devices are enabled.

1.7.7 PL[1:0] / KWL[1:0] — Port L input signals

PL[1:0] are the high voltage input signal. These signals can be configured on a per signal basis as interrupt inputs with wake-up capability (KWL[1:0]). These signals can alternatively be used as analog inputs measured by the ADC.

1.7.8 PP[7:0] / KWP[7:0] — Port P I/O signals

PP[7:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWP[7:0]). They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

1.7.9 PS[7:0] / KWS[7:0] — Port S I/O signals

PS[7:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWS[7:0]). They can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull up devices are enabled.

1.7.10 PT[7:0] — Port T I/O signals

PT[7:0] are general-purpose input or output signals. These signals can have a pull-up or pull-down device selected and enabled on per signal basis. Out of reset the pull devices are disabled.

1.7.11 AN[15:0] — ADC input signals

AN[15:0] are the analog inputs of the Analog-to-Digital Converters.

1.7.12 ACMP Signals

1.7.12.1 ACMP0_0 / ACMP0_1— Analog Comparator 0 Inputs

ACMP0_0 and ACMP0_1 are the inputs of the analog comparator 0 ACMP0.

1.7.12.2 ACMP1_0 / ACMP1_1 — Analog Comparator 1 Inputs

ACMP1_0 and ACMP1_1 are the inputs of the analog comparator 1 ACMP1.

1.7.12.3 ACMPO[1:0] — Analog Comparator Output

ACMPO[1:0] are the outputs of the analog comparators.

1.7.13 DAC Signals

1.7.13.1 DACU Output Pin

This analog pin is used for the unbuffered analog output voltage from the DAC resistor network output, when the according mode is selected in DACCTL register bits DACM[2:0].

1.7.13.2 AMP Output Pin

This analog pin is used for the buffered analog output voltage from the operational amplifier outputs, when the according mode is selected in DACCTL register bits DACM[2:0].

1.7.13.3 AMPP Input Pin

This analog input pin is used as input signal for the operational amplifier positive input pins when the according mode is selected in DACCTL register bits DACM[2:0].

1.7.13.4 AMPM Input Pin

This analog input pin is used as input signal for the operational amplifiers negative input pin when the according mode is selected in DACCTL register bits DACM[2:0].

1.7.14 VRH_0, VRH_1, VRL_0, VRL_1 — ADC reference signals

VRH_0 , VRH_1, VRL_0 and VRL_1 are the reference voltage input pins for the analog-to-digital converter.

1.7.15 ETRIG0 — External ADC trigger signal

This signal inputs to the Analog-to-Digital Converter. Their purpose is to trigger ADC conversions (sequence of conversions).

1.7.16 SPI signals

1.7.16.1 $\overline{SS}$[1:0] signals

This signal is associated with the slave select $\overline{SS}$ functionality of the serial peripheral interface SPI0 and SPI1.

1.7.16.2 SCK[1:0] signals

This signal is associated with the serial clock SCK functionality of the serial peripheral interface SPI0 and SPI1.

1.7.16.3 MISO[1:0] signals

This signal is associated with the MISO functionality of the serial peripheral interface SPI0 and SPI1. This signal acts as master input during master mode or as slave output during slave mode.

1.7.16.4 MOSI[1:0] signals

This signal is associated with the MOSI functionality of the serial peripheral interface SPI0 and SPI1. This signal acts as master output during master mode or as slave input during slave mode

1.7.17 SCI signals

1.7.17.1 RXD[1:0] signals

These signals are associated with the receive functionality of the serial communication interfaces SCI[1:0].

1.7.17.2 TXD[1:0] signals

These signals are associated with the transmit functionality of the serial communication interfaces SCI[1:0].

1.7.18 Timer IOC[7:0] signals

The signals IOC0[7:0] and IOC1[3:0] are associated with the input capture or output compare functionality of the timer modules TIM0 and TIM1.

1.7.19 PWM[7:0] signals

The signals PWM0[7:0] and PWM1[7:0] are associated with the outputs of the PWM0 and PWM1 modules.

1.7.20 IIC signals

1.7.20.1 SDA signal

This signal is associated with the serial data pin of IIC.

1.7.20.2 SCL signal

This signal is associated with the serial clock pin of IIC.

1.7.21 Interrupt signals — $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$

$\overline{\text{IRQ}}$ is a maskable level or falling edge sensitive input. $\overline{\text{XIRQ}}$ is a non-maskable level-sensitive interrupt.

1.7.22 Oscillator and Clock Signals

1.7.22.1 4-16MHZ Oscillator Signal — EXTAL and XTAL

EXTAL and XTAL are the crystal driver. On reset, the OSC is not enabled, all the device clocks are derived from the internal reference clock. EXTAL is the oscillator input. XTAL is the oscillator output.

1.7.22.2 ECLK

This signal is associated with the output of the divided bus clock (ECLK).

NOTE

> This feature is only intended for debug purposes at room temperature. It must not be used for clocking external devices in an application.

1.7.23 BDC and Debug Signals

1.7.23.1 BKGD — Background Debug Signal

The BKGD signal is used as a pseudo-open-drain signal for the background debug communication. The BKGD signal has an internal pull-up device.

1.7.23.2 DBGEEV — External Event Input

This signal is the DBG external event input. It is input only. Within the DBG module, it allows an external event to force a state sequencer transition. A falling edge at the external event signal constitutes an event. Rising edges have no effect. The maximum frequency of events is half the internal core bus frequency.

1.7.24 CAN0 Signals

1.7.24.1 RXCAN0 Signal

This signal is associated with the receive functionality of the scalable controller area network controller (MSCAN0).

1.7.24.2 TXCAN0 Signal

This signal is associated with the transmit functionality of the scalable controller area network controller (MSCAN0).

1.7.25 CAN Physical Layer Signals(CANPHY0)

1.7.25.1 CANH0 — CAN Bus High Pin0

The CANH0 signal either connects directly to CAN bus high line or through an optional external common mode choke.

1.7.25.2 CANL0 — CAN Bus Low Pin0

The CANL0 signal either connects directly to CAN bus low line or through an optional external common mode choke.

1.7.25.3 SPLIT0 — CAN Bus Termination Pin0

The SPLIT0 pin can drive a 2.5 V bias for bus termination purpose (CAN bus middle point). Usage of this pin is optional and depends on bus termination strategy for a given bus network.

1.7.25.4 CPTXD0

This is the CAN physical layer transmitter input signal.

1.7.25.5 CPRXD0

This is the CAN physical layer receiver output signal.

1.7.26 BCTL

BCTL is the ballast connection for the on chip voltage regulator. It provides the base current of an external bipolar of the V_{DDX} and V_{DDA} supplies.

1.7.27 BCTLC

BCTLC provides the base current of an external bipolar that supplies an external or internal CAN physical interface.

1.7.28 VDDC

VDDC is the CANPHY supply. This is the output voltage of the external bipolar transistor, whose base current is supplied by BCTLC. It is fed back to the MCU for regulation.

1.7.29 High Current Output - EVDD

This is a high current, low voltage drop output intended for supplying external devices in a range of up to 20mA. Configuring the pin direction as output automatically enables the high current capability.

1.7.30 Power Supply Pins

The power and ground pins are described below. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible.

NOTE

All ground pins must be connected together in the application.

1.7.30.1 VDDX1, VDDX2, VSSX1, VSSX2 — Digital I/O power and ground pins

VDDX is the voltage regulator output for the digital I/O drivers. It supplies the VDDX domain pads. The VSSX1 and VSSX2 pin is the ground pin for the digital I/O drivers.

Bypass capacitor requirements on VDDX/VSSX depend on how heavily the MCU pins are loaded.

1.7.30.2 VDDA, VSSA — Power supply pins for ADC

These are the power supply and ground pins for the analog-to-digital converter and the voltage regulator. These pins must be externally connected to the voltage regulator (VDDX, VSSX). A separate bypass capacitor for the ADC supply is recommended.

1.7.30.3 VSUP — Voltage supply pin

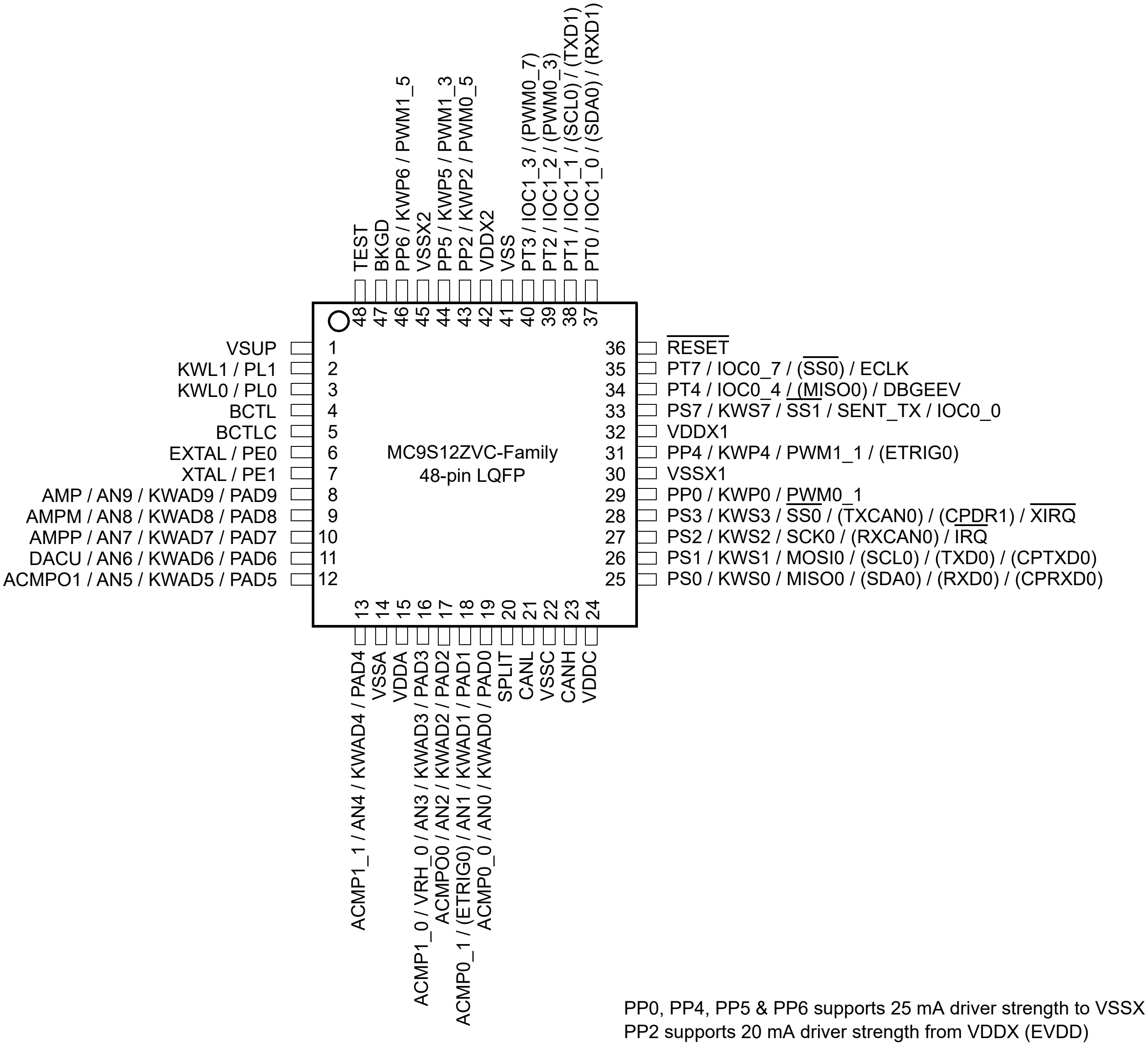
VSUP is the 12V supply voltage pin for the on chip voltage regulator. This is the voltage supply input from which the voltage regulator generates the on chip voltage supplies. It must be protected externally against a reverse battery connection.

1.8 Device Pinouts

The MC9S12ZVC-Family will be offered in 48 pin LQFP and 64 pin LQFP-EP packages. The exposed pad on the package bottom of the LQFP-EP package must be connected to a ground pad on the PCB.

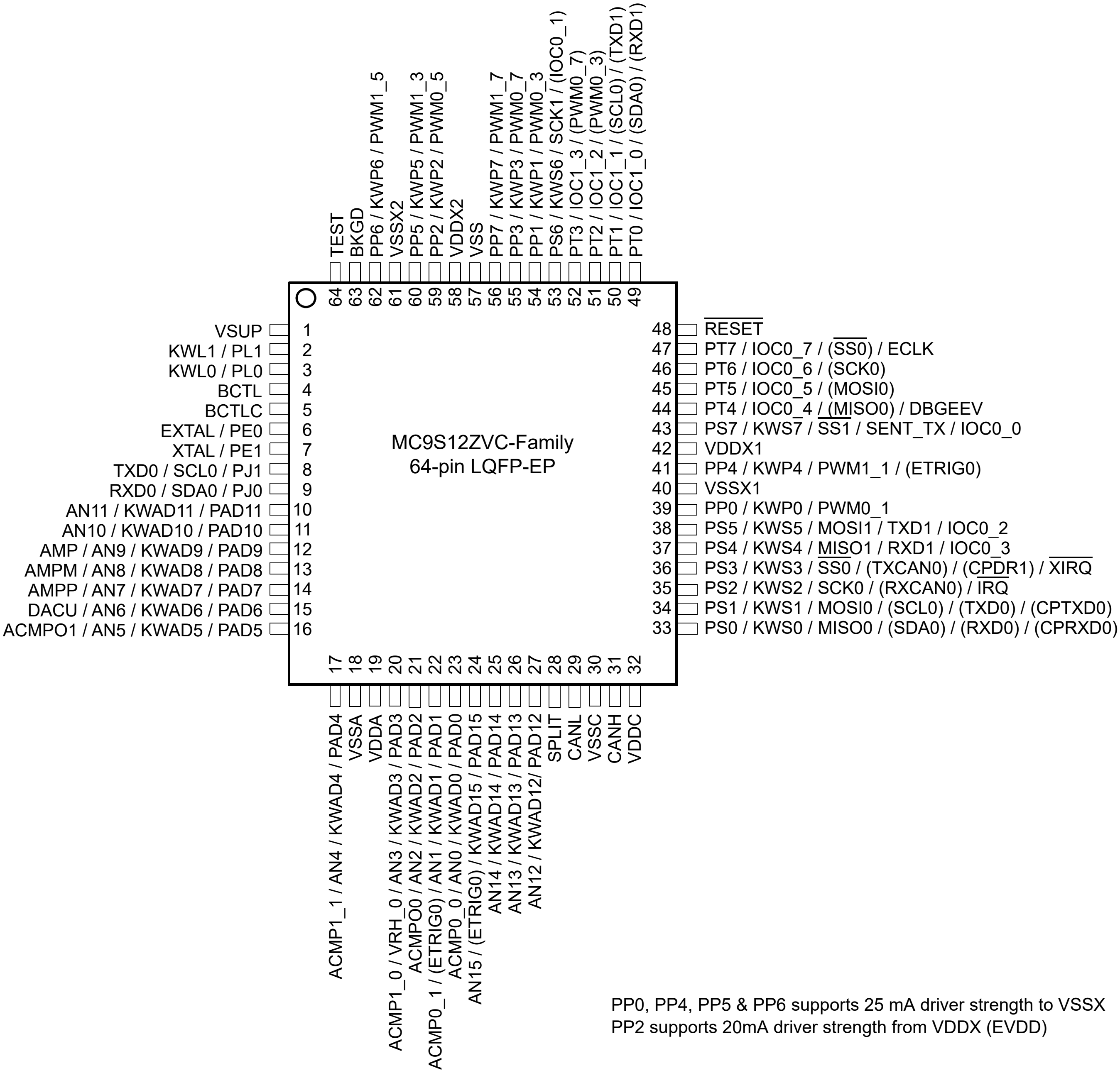
:

Figure 1-3. Pinout MC9S12ZVC-Family 48-pin LQFP



PP0, PP4, PP5 & PP6 supports 25 mA driver strength to VSSX
PP2 supports 20 mA driver strength from VDDX (EVDD)

Figure 1-4. Pinout MC9S12ZVC-Family 64-pin LQFP-EP[1]



PP0, PP4, PP5 & PP6 supports 25 mA driver strength to VSSX
PP2 supports 20mA driver strength from VDDX (EVDD)

---

[1] The exposed pad on the package bottom must be connected to a ground pad on the PCB.

Table 1-6. MC9S12ZVC-Family Pin Summary

LQFP			Function					Power Supply	Internal Pull Resistor	
64	48	Pin	1st Func.	2nd Func.	3rd Func.	4th Func.	5th Func.		CTRL	Reset State
1	1	VSUP	—	—	—	—	—	V_{SUP}	—	—
2	2	PL1	HVI1	KWL1	—	—	—	V_{DDX}	—	—
3	3	PL0	HVI0	KWL0	—	—	—	V_{DDX}	—	—
4	4	BCTL			—	—	—	V_{DDX}		
5	5	BCTLC			—	—	—	V_{DDX}		
6	6	PE0	EXTAL		—	—	—	V_{DDX}	PERE/ PPSE	Down
7	7	PE1	XTAL	—	—	—	—	V_{DDX}	PERE/ PPSE	Down
8		PJ1	SCL0	TXD0	—	—	—	V_{DDX}	PERJ/ PPSJ	Up
9		PJ0	SDA0	RXD0	—	—	—	V_{DDX}	PERJ/ PPSJ	Up
10	—	PAD11	KWAD11	AN11	—	—	—	V_{DDA}	PERADH / PPSADH	Off
11	—	PAD10	KWAD10	AN10	—	—	—	V_{DDA}	PERADH / PPSADH	Off
12	8	PAD9	KWAD9	AN9	AMP	—	—	V_{DDA}	PERADH / PPSADH	Off
13	9	PAD8	KWAD8	AN8	AMPM	—	—	V_{DDA}	PERADH / PPSADH	Off
14	10	PAD7	KWAD7	AN7	AMPP	—	—	V_{DDA}	PERADL/ PPSADL	Off
15	11	PAD6	KWAD6	AN6	DACU	—	—	V_{DDA}	PERADL/ PPSADL	Off
16	12	PAD5	KWAD5	AN5	ACMPO1	—	—	V_{DDA}	PERADL/ PPSADL	Off
17	13	PAD4	KWAD4	AN4	ACMP1_ 1	—	—	V_{DDA}	PERADL/ PPSADL	Off
18	14	VSSA					—	V_{DDA}		
19	15	VDDA					—	V_{DDA}		

Table 1-6. MC9S12ZVC-Family Pin Summary

LQFP			Function					Power Supply	Internal Pull Resistor	
64	**48**	**Pin**	**1st Func.**	**2nd Func.**	**3rd Func.**	**4th Func.**	**5th Func.**		**CTRL**	**Reset State**
20	16	PAD3	KWAD3	AN3	VRH_0	ACMP1_0	—	V_{DDA}	PERADL/ PPSADL	Off
21	17	PAD2	KWAD2	AN2	ACMPO0	—	—	V_{DDA}	PERADL/ PPSADL	Off
22	18	PAD1	KWAD1	AN1	(ERTIG0)	ACMP0_1		V_{DDA}	PERADL/ PPSADL	Off
23	19	PAD0	KWAD0	AN0	ACMP0_0	—	—	V_{DDA}	PERADL/ PPSADL	Off
24		PAD15	KWAD15	(ERTIG0)	AN15	—	—	V_{DDA}	PERADL/ PPSADL	Off
25		PAD14	KWAD14	AN14		—	—	V_{DDA}	PERADL/ PPSADL	Off
26		PAD13	KWAD13	AN13		—	—	V_{DDA}	PERADL/ PPSADL	Off
27		PAD12	KWAD12	AN12		—	—	V_{DDA}	PERADL/ PPSADL	Off
28	20	SPLIT		—	—	—	—	V_{DDC}		
29	21	CANL		—	—	—	—	V_{DDC}		
30	22	VSSC		—	—	—	—	V_{DDC}		
31	23	CANH		—	—	—	—	V_{DDC}		
32	24	VDDC		—	—	—	—	V_{DDC}		
33	25	PS0	KWS0	MISO0	(SDA0)	(RXD0)	(CPRXD0)	V_{DDX}	PERS/ PPSS	Up
34	26	PS1	KWS1	MOSI0	(SCL0)	(TXD0)	(CPTXD0)	V_{DDX}	PERS/ PPSS	Up
35	27	PS2	KWS2	SCK0	(RXCAN0)	$\overline{\text{IRQ}}$		V_{DDX}	PERS/ PPSS	Up
36	28	PS3	KWS3	$\overline{\text{SS0}}$	(TXCAN0)	(CPDR1)	$\overline{\text{XIRQ}}$	V_{DDX}	PERS/ PPSS	Up
37		PS4	KWS4	MISO1	RXD1	IOC0_3[1]		V_{DDX}	PERS/ PPSS	Up
38		PS5	KWS5	MOSI1	TXD1	IOC0_2[2]		V_{DDX}	PERS/ PPSS	Up
39	29	PP0	KWP0	PWM0_1		—	—	V_{DDX}	PERP/ PPSP	Off
40	30	VSSX1		—	—	—	—	V_{DDX}		

Table 1-6. MC9S12ZVC-Family Pin Summary

LQFP			Function					Power Supply	Internal Pull Resistor	
64	**48**	**Pin**	**1st Func.**	**2nd Func.**	**3rd Func.**	**4th Func.**	**5th Func.**		**CTRL**	**Reset State**
41	31	PP4	KWP4	PWM1_1	(ETRIG0)	—	—	V_{DDX}	PERP/ PPSP	Off
42	32	VDDX1			—	—	—	V_{DDX}		
43	33	PS7	KWS7	$\overline{\text{SS1}}$	SENT_TX	IOC0_0		V_{DDX}	PERS/ PPSS	Up
44	34	PT4	IOC0_4	(MISO0)	DBGEEV		—	V_{DDX}	PERT/ PPST	Off
45	—	PT5	IOC0_5	(MOSI0)		—	—	V_{DDX}	PERT/ PPST	Off
46	—	PT6	IOC0_6	(SCK0)	—	—	—	V_{DDX}	PERT/ PPST	Off
47	35	PT7	IOC0_7	($\overline{\text{SS0}}$)	ECLK	—	—	V_{DDX}	PERT/ PPST	Off
48	36	$\overline{\text{RESET}}$	—	—	—	—	—	V_{DDX}	TEST pin	Up
49	37	PT0	IOC1_0	(SDA0)	(RXD1)			V_{DDX}	PERT/ PPST	Off
50	38	PT1	IOC1_1	(SCL0)	(TXD1)			V_{DDX}	PERT/ PPST	Off
51	39	PT2	IOC1_2	(PWM0_3)				V_{DDX}	PERT/ PPST	Off
52	40	PT3	IOC1_3	(PWM0_7)				V_{DDX}	PERT/ PPST	Off
53		PS6	KWS6	SCK1	(IOC0_1)			V_{DDX}	PERS/ PPSS	Up
54	—	PP1	KWP1	PWM0_3	—	—	—	V_{DDX}	PERP/ PPSP	Off
55	—	PP3	KWP3	PWM0_7	—	—	—	V_{DDX}	PERP/ PPSP	Off
56	—	PP7	KWP7	PWM1_7	—	—	—	V_{DDX}	PERP/ PPSP	Off
57	41	VSS	—	—	—	—	—			
58	42	VDDX2	—	—	—	—	—	V_{DDX}		
59	43	PP2	KWP2	PWM0_5	—	—	—	V_{DDX}	PERP/ PPSP	Off
60	44	PP5	KWP5	PWM1_3	—	—	—	V_{DDX}	PERP/ PPSP	Off
61	45	VSSX2	—	—	—	—	—	V_{DDX}		

Table 1-6. MC9S12ZVC-Family Pin Summary

LQFP			Function					Power Supply	Internal Pull Resistor	
64	48	Pin	1st Func.	2nd Func.	3rd Func.	4th Func.	5th Func.		CTRL	Reset State
62	46	PP6	KWP6	PWM1_5	—	—	—	V_{DDX}	PERP/ PPSP	Off
63	47	BKGD	MODC	—	—	—	—	V_{DDX}		Up
64	48	TEST	—	—	—	—	—	—	$\overline{\text{RESET}}$	Down

[1] Input capture routed to PS7 by default. Output compare feature always on PS6.
[2] Input capture routed to PS7 by default. Output compare feature always on PS6

1.9 Internal Signal Mapping

This section specifies the mapping of inter-module signals at device level.

1.9.1 ACMP0 and ACMP1 Connectivity

Table 1-7. shows the connectivity for the analog comparator modules ACMP0 and ACMP1.

Table 1-7. ACMP0 and ACMP1 connectivity

Analog Comparator Inputs	ACMP0	ACMP1
acmpi_0[1]	DACI (unbuffered DAC output)	DACI (unbuffered DAC output)
acmpi_1[2]	AMP (buffered DAC output)	AMP (buffered DAC output)
ACMP_0	PAD0(AN0)	PAD3(AN3)
ACMP_1	PAD1(AN1)	PAD4(AN4)

[1] Output of DAC module DACI need to be enabled
[2] Output of the DAC module AMP need to be enabled

1.9.2 DAC Connectivity

DAC reference Voltage signal VRH is mapped to VDDA and VRL is mapped to VSSA.

1.9.3 ADC Connectivity

1.9.3.1 ADC Reference Voltages

ADC reference Voltage signal VRH_1 is mapped to VDDA and VRH_0 is mapped to PAD3. VRL_1 is mapped to VSSA and VRL_0 is mapped to VSSA.

1.9.3.2 ADC External Trigger Input

The ADC module includes a trigger input to start a sequence of conversions. The trigger signal can be routed to 1 out of 4 sources:

1. internal to the TIM0 OC2 (trigger signal generated by TIM0)
2. external to the pin PAD15 (ETRIG0)
3. external to the pin PAD1 (ETRIG0)
4. external to the pin PP4 (ETRIG0)

1.9.3.3 ADC Internal Channels

The ADC internal channel mapping is shown in Table 1-8.

Table 1-8. ADC Channel Assignment

ADCCMD_L[CH_SEL]							
[5]	[4]	[3]	[2]	[1]	[0]	**Analog Input Channel**	**Usage**
0	0	0	0	0	0	V_{RL}	
0	0	0	0	0	1	V_{RH}	
0	0	0	0	1	0	$(V_{RH}-V_{RL})/2$	
0	0	0	0	1	1	Reserved	
0	0	0	1	0	0	Reserved	
0	0	0	1	0	1	Reserved	
0	0	0	1	1	0	Reserved	
0	0	0	1	1	1	Reserved	
0	0	1	0	0	0	Internal_0	RESERVED
0	0	1	0	0	1	Internal_1	Bandgap Voltage V_{BG} or Chip temperature sensor V_{HT} see Section 8.3.2.14 High Temperature Control Register (CPMUHTCTL)
0	0	1	0	1	0	Internal_2	Flash Voltage V_{DDF}
0	0	1	0	1	1	Internal_3	RESERVED
0	0	1	1	0	0	Internal_4	V_{SUP} see Section 10.3.2.1 BATS Module Enable Register (BATE)
0	0	1	1	0	1	Internal_5	High voltage input port L0 see Section 2.3.4.10 Port L ADC Connection Enable Register (PTAENL)
0	0	1	1	1	0	Internal_6	High voltage input port L1 Section 2.3.4.10 Port L ADC Connection Enable Register (PTAENL)
0	0	1	1	1	1	Internal_7	RESERVED

1.9.4 TIM0 and TIM1 Clock Source Connectivity

The clock for TIM1 is the device core clock generated in the CPMU module. (maximum core clock is 64MHz)

The clock for TIM0 is the device bus clock generated in the CPMU module. (maximum bus clock 32MHz)

1.9.5 TIM0 and TIM1 IOC Channel Connectivity

Table 1-9 shows a summary of TIM0 and TIM1channel connections.

Table 1-9. TIM0 and TIM1 Connections

IOC Channel	TIM0	TIM1 (fast)
IOC0	SENTTX	PT0
IOC1	SENTTX	PT1
IOC2	ACLK / ADC Trigger	PT2 / ACMP0 output
IOC3	RXD0 / RXD1	PT3 / ACMP1 output
IOC4	PT4	
IOC5	PT5	
IOC6	PT6	
IOC7	PT7	

1.9.6 PWM0 and PWM1 Clock Source Connectivity

The clock for PWM1, PWM Clock , is mapped to device core clock, generated in the CPMU module. (maximum core clock is 64MHz)

The clock for PWM0 , PWM Clock , is mapped to device bus clock, generated in the CPMU module. (maximum bus clock is 32MHz)

1.9.7 BDC Clock Source Connectivity

The BDC clock, BDCCLK, is mapped to the IRCCLK generated in the CPMU module.

The BDC clock, BDCFCLK is mapped to the device bus clock, generated in the CPMU module.

1.9.8 FTMRZ Connectivity

The soc_erase_all_req input to the flash module is driven directly by a BDC erase flash request resulting from the BDC ERASE_FLASH command.

1.9.9 CPMU Connectivity

The API clock generated in the CPMU is not mapped to a device pin in the MC9S12ZVC-Family.

1.10 Modes of Operation

The MCU can operate in different modes. These are described in 1.10.1 Chip Configuration Modes.

The MCU can operate in different power modes to facilitate power saving when full system performance is not required. These are described in 1.10.3 Low Power Modes.

Some modules feature a software programmable option to freeze the module status whilst the background debug module is active to facilitate debugging. This is referred to as freeze mode at module level.

1.10.1 Chip Configuration Modes

The different modes and the security state of the MCU affect the debug features (enabled or disabled).

The operating mode out of reset is determined by the state of the MODC signal during reset (Table 1-10). The MODC bit in the MODE register shows the current operating mode and provides limited mode switching during operation. The state of the MODC signal is latched into this bit on the rising edge of $\overline{\text{RESET}}$.

Table 1-10. Chip Modes

Chip Modes	MODC
Normal single chip	1
Special single chip	0

1.10.1.1 Normal Single-Chip Mode

This mode is intended for normal device operation. The opcode from the on-chip memory is executed after reset (requires the reset vector to be programmed correctly). The processor program is executed from internal memory.

NOTE

To avoid unpredictable behaviour do not start the device in Normal Single-Chip Mode while the flash is erased.

1.10.1.2 Special Single-Chip Mode

This mode is used for debugging operation, boot-strapping, or security related operations. The background debug mode (BDM) is active on leaving reset in this mode.

1.10.2 Debugging Modes

The background debug mode (BDM) can be activated by the BDC module or directly when resetting into Special Single-Chip mode. Detailed information can be found in the BDC module section.

Writing to internal memory locations using the debugger, whilst code is running or at a breakpoint, can change the flow of application code.

The MC9S12ZVC-Family supports BDC communication throughout the device Stop mode. During Stop mode, writes to control registers can alter the operation and lead to unexpected results. It is thus recommended not to reconfigure the peripherals during STOP using the debugger.

1.10.3 Low Power Modes

The device has two dynamic-power modes (run and wait) and two static low-power modes (stop and pseudo stop). For a detailed description refer to the CPMU section.

- Dynamic power mode: Run
  - Run mode is the main full performance operating mode with the entire device clocked. The user can configure the device operating speed through selection of the clock source and the phase locked loop (PLL) frequency. To save power, unused peripherals must not be enabled.
- Dynamic power mode: Wait
  - This mode is entered when the CPU executes the WAI instruction. In this mode the CPU does not execute instructions. The internal CPU clock is switched off. All peripherals can be active in system wait mode. For further power consumption the peripherals can individually turn off their local clocks. Asserting $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, $\overline{\text{IRQ}}$, or any other interrupt that is not masked, either locally or globally by a CCR bit, ends system wait mode.
- Static power modes:
  Static power (Stop) modes are entered following the CPU STOP instruction unless an NVM command is active. When no NVM commands are active, the Stop request is acknowledged and the device enters either Stop or Pseudo Stop mode.
  - Pseudo-stop: In this mode the system clocks are stopped but the oscillator is still running and the real time interrupt (RTI), watchdog (COP) and Autonomous Periodic Interrupt (API) may be enabled. Other peripherals are turned off. This mode consumes more current than system STOP mode but, as the oscillator continues to run, the full speed wake up time from this mode is significantly shorter.
  - Stop: In this mode the oscillator is stopped and clocks are switched off and the VREG enters reduced performance mode (RPM). The counters and dividers remain frozen. The autonomous periodic interrupt (API) may remain active but has a very low power consumption. The KWx pins and the SCI module can be configured to wake the device, whereby current consumption is negligible.
    If the BDC is enabled, in Stop mode, the VREG remains in full performance mode. With BDC enabled and BDCCIS bit set, then all clocks remain active during Stop mode to allow BDC access to internal peripherals. If the BDC is enabled and BDCCIS is clear, then the BDCSI clock remains active to allow BDC register access, but other clocks (with the exception of the API) are switched off. With the BDC enabled during Stop, the VREG full performance mode and clock activity lead to higher current consumption than with BDC disabled.
    If the BDC is enabled in Stop mode, then the voltage monitoring remains enabled.

NOTE

The U-bit should be cleared and the S-bit (stop enable) should be cleared in the CPU condition code register (CCR) to execute the STOP instruction.Otherwise the STOP instruction is considered as a NOP.

1.11 Security

The MCU security mechanism prevents unauthorized access to the flash memory. It must be emphasized that part of the security must lie with the application code. An extreme example would be application code that dumps the contents of the internal memory. This would defeat the purpose of security. Also, if an application has the capability of downloading code through a serial port and then executing that code (e.g. an application containing bootloader code), then this capability could potentially be used to read the EEPROM and Flash memory contents even when the microcontroller is in the secure state. In this example, the security of the application could be enhanced by requiring a response authentication before any code can be downloaded.

Device security details are also described in the flash block description.

1.11.1 Features

The security features of the S12Z chip family are:

- Prevent external access of the non-volatile memories (Flash, EEPROM) content
- Restrict execution of NVM commands

1.11.2 Securing the Microcontroller

The chip can be secured by programming the security bits located in the options/security byte in the Flash memory array. These non-volatile bits keep the device secured through reset and power-down.

This byte can be erased and programmed like any other Flash location. Two bits of this byte are used for security (SEC[1:0]). The contents of this byte are copied into the Flash security register (FSEC) during a reset sequence.

The meaning of the security bits SEC[1:0] is shown in Table 1-11. For security reasons, the state of device security is controlled by two bits. To put the device in unsecured mode, these bits must be programmed to SEC[1:0] = ‘10’. All other combinations put the device in a secured mode. The recommended value to put the device in secured state is the inverse of the unsecured state, i.e. SEC[1:0] = ‘01’.

Table 1-11. Security Bits

SEC[1:0]	Security State
00	1 (secured)
01	1 (secured)
10	***0 (unsecured)***
11	1 (secured)

NOTE

Please refer to the Flash block description for more security byte details.

1.11.3 Operation of the Secured Microcontroller

By securing the device, unauthorized access to the EEPROM and Flash memory contents is prevented. Secured operation has the following effects on the microcontroller:

1.11.3.1 Normal Single Chip Mode (NS)

- Background debug controller (BDC) operation is completely disabled.
- Execution of Flash and EEPROM commands is restricted (described in flash block description).

1.11.3.2 Special Single Chip Mode (SS)

- Background debug controller (BDC) commands are restricted
- Execution of Flash and EEPROM commands is restricted (described in flash block description).

In special single chip mode the device is in active BDM after reset. In special single chip mode on a secure device, only the BDC mass erase and BDC control and status register commands are possible. BDC access to memory mapped resources is disabled. The BDC can only be used to erase the EEPROM and Flash memory without giving access to their contents.

1.11.4 Unsecuring the Microcontroller

Unsecuring the microcontroller can be done using three different methods:

1. Backdoor key access
2. Reprogramming the security bits
3. Complete memory erase

1.11.4.1 Unsecuring the MCU Using the Backdoor Key Access

In normal single chip mode, security can be temporarily disabled using the backdoor key access method. This method requires that:

- The backdoor key has been programmed to a valid value
- The KEYEN[1:0] bits within the Flash options/security byte select ‘enabled’.
- The application program programmed into the microcontroller has the capability to write to the backdoor key locations

The backdoor key values themselves would not normally be stored within the application data, which means the application program would have to be designed to receive the backdoor key values from an external source (e.g. through a serial port)

The backdoor key access method allows debugging of a secured microcontroller without having to erase the Flash This is particularly useful for failure analysis.

NOTE

No backdoor key word is allowed to have the value 0x0000 or 0xFFFF.

1.11.5 Reprogramming the Security Bits

Security can also be disabled by erasing and reprogramming the security bits within the flash options/security byte to the unsecured value. Since the erase operation will erase the entire sector (0x7F_FE00–0x7F_FFFF) the backdoor key and the interrupt vectors will also be erased; this method is not recommended for normal single chip mode. The application software can only erase and program the Flash options/security byte if the Flash sector containing the Flash options/security byte is not protected (see Flash protection). Thus Flash protection is a useful means of preventing this method. The microcontroller enters the unsecured state after the next reset following the programming of the security bits to the unsecured value.

This method requires that:

- The application software previously programmed into the microcontroller has been designed to have the capability to erase and program the Flash options/security byte.
- The Flash sector containing the Flash options/security byte is not protected.

1.11.6 Complete Memory Erase

The microcontroller can be unsecured by erasing the entire EEPROM and Flash memory contents. If ERASE_FLASH is successfully completed, then the Flash unsecures the device and programs the security byte automatically.

1.12 Resets and Interrupts

Table 1-12. lists all Reset sources and the vector address. Resets are explained in detail in the Chapter 8, “S12 Clock, Reset and Power Management Unit (S12CPMU_UHV_V7)””

Table 1-12. Reset Sources and Vector Locations

Vector Address	Reset Source	CCR Mask	Local Enable
0xFF_FFFC	Power-On Reset (POR)	None	None
	Low Voltage Reset (LVR)	None	None
	External pin $\overline{\text{RESET}}$	None	None
	Clock monitor reset	None	OSCE Bit in CPMUOSC register & OMRE Bit in CPMUOSC2 register
	COP watchdog reset	None	CR[2:0] in CPMUCOP register

1.12.1 Interrupt Vectors

Table 1-13. lists all interrupt sources and vectors in the default order of priority. The interrupt module description provides an interrupt vector base register (IVBR) to relocate the vectors.

Table 1-13. Interrupt Vector Locations

Vector Address	Interrupt Source	CCR Mask	Local Enable	Wake up from STOP	Wake up from WAIT
Vector base + 0x1F8	Unimplemented page1 op-code trap (SPARE)	None	None	–	–
Vector base + 0x1F4	Unimplemented page 2 op-code trap (TRAP)	None	None	–	–
Vector base + 0x1F0	Software interrupt instruction (SWI)	None	None	–	–
Vector base + 0x1EC	System call interrupt instruction (SYS)	None	None	–	–
Vector base + 0x1E8	Machine exception	None	None	–	–
Vector base + 0x1E4	Reserved				
Vector base + 0x1E0	Reserved				
Vector base + 0x1DC	Spurious interrupt	–	None	–	–
Vector base + 0x1D8	$\overline{\text{XIRQ}}$ interrupt request	X bit	None	Yes	Yes
Vector base + 0x1D4	$\overline{\text{IRQ}}$ interrupt request	I bit	IRQCR (IRQEN)	Yes	Yes
Vector base + 0x1D0	RTI timeout interrupt	I bit	CPMUINT (RTIE)	Yes	Yes
Vector base + 0x1CC	TIM0 timer channel 0	I bit	TIE (C0I)	No	Yes
Vector base + 0x1C8	TIM0 timer channel 1	I bit	TIM0TIE (C1I)	No	Yes
Vector base + 0x1C4	TIM0 timer channel 2	I bit	TIM0TIE (C2I)	No	Yes
Vector base + 0x1C0	TIM0 timer channel 3	I bit	TIM0TIE (C3I)	No	Yes
Vector base + 0x1BC	TIM0 timer channel 4	I bit	TIM0TIE (C4I)	No	Yes
Vector base + 0x1B8	TIM0 timer channel 5	I bit	TIM0TIE (C5I)	No	Yes
Vector base + 0x1B4	TIM0 timer channel 6	I bit	TIM0TIE (C6I)	No	Yes
Vector base + 0x1B0	TIM0 timer channel 7	I bit	TIM0TIE (C7I)	No	Yes
Vector base + 0x1AC	TIM0 timer overflow	I bit	TIM0TSCR2 (TOF)	No	Yes
Vector base + 0x1A8	TIM0 pulse accumulator A overflow	I bit	TIM0PACTL(PAOVI)	No	Yes
Vector base + 0x1A4	TIM0 pulse accumulator input edge	I bit	TIM0PACTL (PAI)	No	Yes
Vector base + 0x1A0	SPI0	I bit	SPI0CR1 (SPIE, SPTIE)	No	Yes
Vector base + 0x19C	SCI0	I bit	SCI0CR2 (TIE, TCIE, RIE, ILIE)	Yes	Yes
Vector base + 0x198	SCI1	I bit	SCI1CR2 (TIE, TCIE, RIE, ILIE)	Yes	Yes
Vector base + 0x194	Reserved				
Vector base + 0x190	Reserved				

Vector Address	Interrupt Source	CCR Mask	Local Enable	Wake up from STOP	Wake up from WAIT
Vector base + 0x18C	ADC0 error interrupt	I bit	ADC0EIE(IA_EIE, CMD_EIE, EOL_EIE, TRIG_EIE,RSTAR_EIE , LDOK_EIE)	No	Yes
Vector base + 0x188	ADC0 conversion sequence abort	I bit	ADCIE(SEQAR_IE, CONIF_OIE)	No	Yes
Vector base + 0x184	ADC0 conversion complete interrupt	I bit	ADCCONIE[9:0]	No	Yes
Vector base + 0x180	Oscillator status interrupt	I bit	CPMUINT(OSCIE)	No	Yes
Vector base + 0x17C	PLL lock interrupt	I bit	CPMUINT(LOCKIE)	No	Yes
Vector base + 0x178	ACMP0	I bit	ACMP0C(ACIE)	No	Yes
Vector base + 0x174	ACMP1	I bit	ACMP1C(ACIE)	No	Yes
Vector base + 0x170	RAM error	I bit	EECIE (SBEEIE)	No	Yes
Vector base + 0x16C	SPI1	I bit	SPI1CR1(SPIE,SPTIE)	No	Yes
Vector base + 0x168	Reserved				
Vector base + 0x164	FLASH error	I bit	FERCNFG (SFDIE)	No	No
Vector base + 0x160	FLASH command	I bit	FCNFG (CCIE)	No	Yes
Vector base + 0x15C	CAN wake-up	I bit	CANRIER (WUPIE)	Yes	Yes
Vector base + 0x158	CAN error	I bit	CANRIER (CSCIE, OVRIE)	No	Yes
Vector base + 0x154	CAN receive	I bit	CANRIER (RXFIE)	No	Yes
Vector base + 0x150	CAN transmit	I bit	CANRIER (TXEIE[2:0])	No	Yes
Vector base + 0x14C to Vector base + 0x144	Reserved				
Vector base + 0x140	BATS supply voltage monitor interrupt	I bit	BATIE(BVHIE, BVLIE)	No	Yes
Vector base + 0x13C to Vector base + 0x12C	Reserved				
Vector base + 0x128	CAN Physical Layer	I bit	CPIE(CPVFIE, CPOCIE)	No	Yes
Vector base + 0x124	Port S interrupt (Key Wakeup)	I bit	PIES(PIES[7..0])	Yes	Yes
Vector base + 0x120 to Vector base + 0x110	Reserved				
Vector base + 0x10C	Port P interrupt	I bit	PIEP(PIEP[7..0])	Yes	Yes
Vector base + 0x108	EVDD and NGPIO over-current interrupt	I bit	OCPEP(OCPEP[6,4,2:0])	No	Yes
Vector base + 0x104	Low-voltage interrupt (LVI)	I bit	CPMUCTRL (LVIE)	No	Yes
Vector base + 0x100	Autonomous periodical interrupt (API)	I bit	CPMUAPICTRL(APIE)	Yes	Yes
Vector base + 0x0FC	High temperature interrupt	I bit	CPMUHTCTL (HTIE)	No	Yes

Vector Address	Interrupt Source	CCR Mask	Local Enable	Wake up from STOP	Wake up from WAIT
Vector base + 0x0F8	Reserved				
Vector base + 0x0F4	Port AD interrupt (Key Wakeup)	I bit	PIEADH(PIEADH[7..0]) PIEADL(PIEADL[7..0])	Yes	Yes
Vector base + 0x0F0 to Vector base + 0x0C4	Reserved				
Vector base + 0x0C0	Port L interrupt (Key Wakeup)	I bit	PIEL(PIEL[1:0])	Yes	Yes
Vector base + 0x0BC to Vector base + 0x0B0	Reserved				
Vector base + 0x0AC	TIM1 timer channel 0	I bit	TIM1TIE (C0I)	No	Yes
Vector base + 0x0A8	TIM1 timer channel 1	I bit	TIM1TIE (C1I)	No	Yes
Vector base + 0x0A4	TIM1 timer channel 2	I bit	TIM1TIE (C2I)	No	Yes
Vector base + 0x0A0	TIM1 timer channel 3	I bit	TIM1TIE (C3I)	No	Yes
Vector base + 0x09C to Vector base + 0x090	Reserved				
Vector base + 0x08C	TIM1 timer overflow	I bit	TIM1TSCR2 (TOF)	No	Yes
Vector base + 0x088 to Vector base + 0x064	Reserved				
Vector base + 0x060	IIC0	I bit	IBCR(IBIE)	No	Yes
Vector base + 0x05C	SENTTX	I bit	INTEN(xxIE)	No	Yes
Vector base + 0x058 to Vector base + 0x000	Reserved				

1.12.2 Effects of Reset

When a reset occurs, MCU registers and control bits are initialized. Refer to the respective block sections for register reset states.

On each reset, the Flash module executes a reset sequence to load Flash configuration registers.

1.12.2.1 Flash Configuration Reset Sequence Phase

On each reset, the Flash module will hold CPU activity while loading Flash module registers from the Flash memory. If double faults are detected in the reset phase, Flash module protection and security may be active on leaving reset. This is explained in more detail in the Flash module description.

1.12.2.2 Reset While Flash Command Active

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

1.12.2.3 I/O Pins

Refer to the PIM section for reset configurations of all peripheral module ports.

1.12.2.4 RAM

The system RAM arrays, including their ECC syndromes, are initialized following a power on reset. All other RAM arrays are not initialized out of any type of reset.

With the exception of a power-on-reset the RAM content is unaltered by a reset occurrence.

1.13 Module device level dependencies

1.13.1 API External Clock Output

API external clock output API_EXTCLK mentioned in 8.2.12 and 8.3.2.16 is not available on S12ZVC-Family.

1.13.2 COP Configuration

The COP timeout rate bits CR[2:0] and the WCOP bit in the CPMUCOP register are loaded from the Flash configuration field byte at global address 0xFF_FE0E during the reset sequence. See Table 1-14 and Table 1-15 for coding.

Table 1-14. Initial COP Rate Configuration

NV[2:0] in FOPT Register	CR[2:0] in COPCTL Register
000	111
001	110
010	101
011	100
100	011
101	010
110	001
111	000

Table 1-15. Initial WCOP Configuration

NV[3] in FOPT Register	WCOP in COPCTL Register
1	0
0	1

1.13.3 Flash IFR Mapping

Table 1-16. Flash IFR Mapping

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	IFR Byte Address
				ADC0 reference conversion using VDDA/VSSA												0x1F_C040 & 0x1F_C041
				ADC0 reference conversion using PAD1/PAD0												0x1F_C042 & 0x1F_C043
Reserved																0x1F_C050 & 0x1F_C051
Reserved																0x1F_C052 & 0x1F_C053
Reserved																0x1F_C054 & 0x1F_C055
Reserved																0x1F_C056 & 0x1F_C057
Reserved																0x1F_C058 & 0x1F_C059
Reserved																0x1F_C05A & 0x1F_C05B
		ACLKTR[5:0][1] (CPMU)										HTTR[3:0][2] (CPMU)				0x1F_C0B8 & 0x1F_C0B9
	TCTRIM[4:0] (CPMU)[3]					IRCTRIM[9:0] (CPMU)[3]										0x1F_C0BA & 0x1F_C0BB

[1] see Section 8.3.2.17 Autonomous Clock Trimming Register (CPMUACLKTR)
[2] see Section 8.3.2.20 High Temperature Trimming Register (CPMUHTTR)
[3] see Section 8.3.2.21 S12CPMU_UHV_V7 IRC1M Trim Registers (CPMUIRCTRIMH / CPMUIRCTRIML)

1.14 Application Information

1.14.1 ADC Calibration

For applications that do not provide external ADC reference voltages, the VDDA/VSSA supplies can be used as sources for VRH/VRL respectively. Since the VDDA must be connected to VDDX at board level in the application, the accuracy of the VDDA reference is limited by the internal voltage regulator accuracy. In order to compensate for VDDA reference voltage variation in this case, the on chip bandgap reference voltage V_{BG} is measured during production test. V_{BG} has a narrow variation over temperature and external voltage supply. V_{BG} is connected to an internal channel of the ADC module (see Table 1-8). The12-bit left justified ADC conversion result of V_{BG} is stored in the flash IFR for reference, as listed in Table 1-16.

By measuring the voltage V_{BG} in the application environment and comparing the result to the reference value in the IFR, it is possible to determine the current ADC reference voltage V_{RH}:

$$V_{RH} = \frac{\text{StoredReference}}{\text{ConvertedReference}} \bullet 5\text{V}$$

The exact absolute value of an analog conversion can be determined as follows:

$$\text{Result} = \text{ConvertedADInput} \bullet \frac{\text{StoredReference} \bullet 5\text{V}}{\text{ConvertedReference} \bullet 2^{n}}$$

With:

Converted AD Input:	Result of the analog to digital conversion of the desired pin
Converted Reference:	Result of internal channel conversion
Stored Reference:	Value in IFR location
n:	ADC resolution (12 bit)

NOTE

> The ADC reference voltage V_{RH} must remain at a constant level throughout the conversion process.

The reference voltage V_{BG} is measured under the conditions shown in Table 1-17. The value stored in the IFR is the average of 8 consecutive conversions.

Table 1-17. V_{BG} Reference Conversion Measurement Conditions for 5V operation

Description	Symbol	Value	Unit
I/O supply voltage	V_{DDX}	5	V
Analog supply voltage	V_{DDA}	5	V
ADC reference voltage	V_{RH}	5	V
ADC clock	f_{ADCCLK}	2	MHz
ADC sample time	t_{SMP}	4	ADC clock cycles
Bus frequency	f_{bus}	25	MHz
Ambient temperature	T_A	150	°C

1.14.2 Use Case Urea Concentration Level Sensor

In this application the signal runtime of ultrasonic response is measured. Sensor output is connected to PAD1. This case uses the ACMP0 to compare the sensor output voltage with a reference voltage provided by the DAC module. The output of the ACMP0 module is connected to high resolution timer (TIM1) to measure the signal runtime. See **Figure 1-5.**

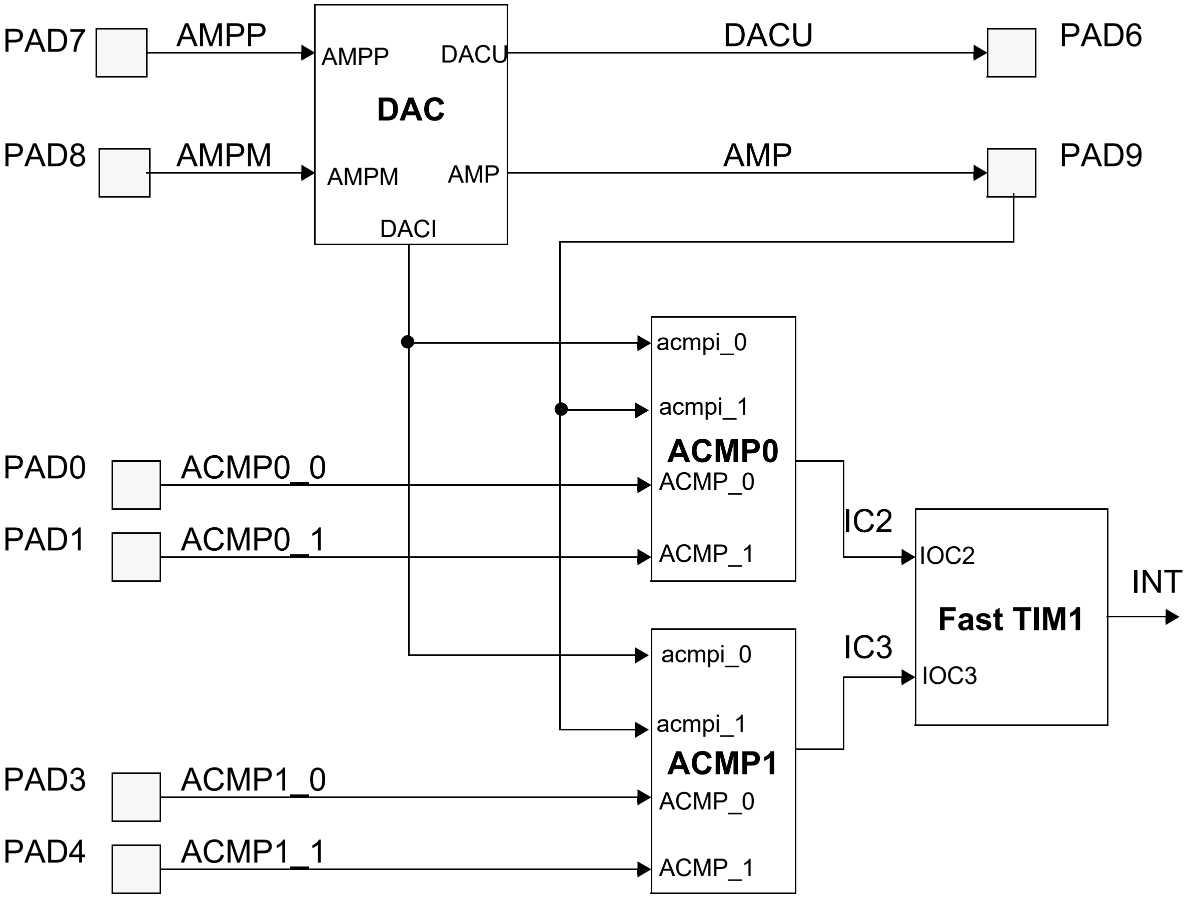
1. Setup ACMP0, DAC,and PIM:

- Select PAD1/ACMP0_1 as positive input to ACMP0
  - Set ACMP0C1.[ACPSEL1:ACPSEL0]= 0x01
- Select Unbuffered DAC output DACU as negative input to ACMP0
  - Set ACMP0C1.[ACNSEL1:ACNSEL0]= 0x2
- Select required ACMP0 hysteresis VACMP_hys
  - Set ACMP0C0.[ACHYS1:ACHYS0]= 11 (largest hysteresis)
- Select required ACMP0 input filter characteristic
  - Set ACMP0C0.[ACDLY]= 0x1 (high speed characteristic).
- Select Full Voltage Range for DAC
  - Set DACCTL[FVR]=0x1
- Select DAC output voltage Vout = DACVOL[7:0] x (VRH-VRL)/256 + VRL
  - Set DACVOL=<required reference voltage>
- Select DAC Mode unbuffered DAC and wait for DACU to settle Tsettle_DACU
  - Set DACCTL[DACM2:DACM0]=100

- Route ACMP0 output to capture input of fast timer (TIM1). Timer resolution for core clock 64MHz about 16ns.
  — Set MODRR1[T1IC2RR]=0x1.
- Enable ACMP0 and wait for the initialization delay of 127 bus clock cycles (for bus clock 32MHz about 4us)
  — ACMP0C0[ACE]=0x1

2. Configure fast TIM1 input Capture channel IC2 to measure time between two input capture events

Figure 1-5. DAC , ACMP and Pad Connectivity



1.14.3 Voltage Domain Monitoring

The BATS module monitors the voltage on the VSUP pin, providing status and flag bits, an interrupt and a connection to the ADC, for accurate measurement of the scaled VSUP level.

The POR circuit monitors the VDD and VDDA domains, ensuring a reset assertion until an adequate voltage level is attained. The LVR circuit monitors the VDD, VDDF and VDDX domains, generating a reset when the voltage in any of these domains drops below the specified assert level. The VDDX LVR monitor is disabled when the VREG is in reduced power mode. A low voltage interrupt circuit monitors the VDDA domain.

Chapter 2
Port Integration Module (S12ZVCPIMV1)

Table 2-1. Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V01.00	14 May 2014		• Initial version

2.1 Introduction

2.1.1 Overview

The S12ZVC-family port integration module establishes the interface between the peripheral modules and the I/O pins for all ports. It controls the electrical pin properties as well as the signal prioritization and multiplexing on shared pins.

This document covers:

- 2-pin port E associated with the external oscillator
- 16-pin port AD with pin interrupts and key-wakeup function; associated with 16 Analog-to-Digital Converter (ADC) channels, shared with 2 Analog Comparator (ACMP) and 1 Digital-to-Analog Converter (DAC)
- 8-pin port T associated with 4 fast and 4 standard timer (TIM) channels, shared with ECLK, 1 routed SCI, 1 routed SPI, 1 routed IIC, 2 routed standard PWM channels
- 8-pin port S with pin interrupts and key-wakeup function or IRQ, XIRQ interrupt inputs; associated with 2 SPI, shared with 1 SENT transmitter, 1 SCI, 4 routed standard TIM channels, 1 routed SCI and 1 routed IIC, 1 routed MSCAN and routed CPTXD and CPRXD interface of CANPHY (for test)
- 8-pin port P with pin interrupts and key-wakeup function; associated with 4 fast and 4 standard PWM channels
- 2-pin port J associated with 1 IIC shared with 1 SCI
- 2-pin port L with pin interrupts and key-wakeup function; associated with 2 high voltage inputs (HVI)

Most I/O pins can be configured by register bits to select data direction and to enable and select pullup or pulldown devices.

NOTE

This document assumes the availability of all features offered in the largest package option. Refer to the package and pinout section in the device overview for functions not available in lower pin count packages.

2.1.2 Features

The PIM includes these distinctive registers:

- Data registers and data direction registers for ports E, AD, T, S, P and J when used as general-purpose I/O
- Control registers to enable pull devices and select pullups/pulldowns on ports E, AD, T, S, P and J
- Control register to enable open-drain (wired-or) mode on port S and J
- Control register to enable digital input buffers on port AD and L
- Interrupt flag register for pin interrupts and key-wakeup (KWU) on port AD, S, P and L
- Control register to configure $\overline{\text{IRQ}}$ pin operation
- Control register to enable ECLK output
- Routing registers to support signal relocation on external pins and control internal routings:
  - 2 PWM1 (fast) channels to alternative pins (1 option each)
  - 4 TIM0 channels to pins (1 option each)
  - IIC0 to alternative pins (2 options)
  - SCI0 to alternative pins (1 option)
  - SCI1 to alternative pins (1 option)
  - SPI0 to alternative pins (1 option)
  - ADC0 trigger input with edge select from internal TIM output compare channel link (OC0_2) or external pins (3 options)
  - Various MSCAN0-CANPHY0 routing options for standalone use and conformance testing
  - MSCAN0 to alternative pins (1 option)
  - Internal RXD0 and RXD1 link to TIM input capture channel (IC0_3) for baud rate detection
  - Internal ACLK link to TIM input capture channel (IC0_2) for calibration and clock monitoring purposes
  - SENT_TX pin link to 2 TIM0 input capture channels (IC0_0 and IC0_1)
  - Internal ACMP0 link to TIM1 (fast) input capture channel (IC1_2)
  - Internal ACMP1 link to TIM1 (fast) input capture channel (IC1_3)

A standard port pin has the following minimum features:

- Input/output selection
- 5V output drive
- 5V digital and analog input
- Input with selectable pullup or pulldown device

Optional features supported on dedicated pins:

- Open drain for wired-or connections (ports S and J)
- Interrupt input with glitch filtering
- High current drive strength from VDDX with over-current protection
- High current drive strength to VSSX
- Selectable drive strength (port P)

2.2 External Signal Description

This section lists and describes the signals that do connect off-chip.

Table 2-2 shows all pins with the pins and functions that are controlled by the PIM. Routing options are denoted in parentheses.

NOTE

> If there is more than one function associated with a pin, the output priority is indicated by the position in the table from top (highest priority) to bottom (lowest priority).

Table 2-2. Pin Functions and Priorities

Port	Pin	Pin Function & Priority	I/O	Description	Routing Register Bit	Func. after Reset
—	BKGD	MODC[1]	I	MODC input during $\overline{\text{RESET}}$	—	BKGD
		BKGD	I/O	S12ZBDC communication	—	
E	PE1	XTAL	—	CPMU OSC signal	—	GPIO
		PTE[1]	I/O	GPIO	—	
	PE0	EXTAL	—	CPMU OSC signal	—	
		PTE[0]	I/O	GPIO	—	
AD	PAD15	AN15	I	ADC0 analog input	—	GPIO
		(ETRIG0)	I	ADC0 external trigger	TRIG0RR1-0	
		PTADH[7]/ KWADH[7]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD14-10	AN14:AN10	I	ADC0 analog input	—	
		PTADH[6:2]/ KWADH[6:2]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD9	AMP	O	DAC buffered analog output	—	
		AN9	I	ADC0 analog input	—	
		PTADH[1]/ KWADH[1]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD8	AMPM	I	DAC standalone OPAMP inverting input	—	
		AN8	I	ADC0 analog input	—	
		PTADH[0]/ KWADH[0]	I/O	GPIO with pin-interrupt and key-wakeup	—	

Port	Pin	Pin Function & Priority	I/O	Description	Routing Register Bit	Func. after Reset
	PAD7	AMPP	I	DAC standalone OPAMP non-inverting input	—	
		AN7	I	ADC0 analog input	—	
		PTADL[7]/ KWADL[7]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD6	DACU	O	DAC unbuffered analog output	—	
		AN6	I	ADC0 analog input	—	
		PTADL[6]/ KWADL[6]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD5	ACMPO1	O	ACMP1 unsynchronized output	—	
		AN5	I	ADC0 analog input	—	
		PTADL[5]/ KWADL[5]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD4	ACMP1_1	I	ACMP1 analog input 1	—	
		AN4	I	ADC0 analog input	—	
		PTADL[4]/ KWADL[4]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD3	ACMP1_0	I	ACMP1 analog input 0	—	
		VRH	I	ADC0 voltage reference high	—	
		AN3	I	ADC0 analog input	—	
		PTADL[3]/ KWADL[3]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD2	ACMPO0	O	ACMP0 unsynchronized output	—	
		AN2	I	ADC0 analog input	—	
		PTADL[2]/ KWADL[2]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD1	ACMP0_1	I	ACMP0 analog input 1	—	
		AN1	I	ADC0 analog input	—	
		(ETRIG0)	I	ADC0 external trigger	TRIG0RR1-0	
		PTADL[1]/ KWADL[1]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PAD0	ACMP0_0	I	ACMP0 analog input 0	—	
		AN0	I	ADC0 analog input	—	
		PTADL[0]/ KWADL[0]	I/O	GPIO with pin-interrupt and key-wakeup	—	

Port	Pin	Pin Function & Priority	I/O	Description	Routing Register Bit	Func. after Reset
T	PT7	ECLK	O	Free-running clock	—	GPIO
		($\overline{\text{SS0}}$)	I/O	SPI0 slave select	SPI0RR	
		IOC0_7	I/O	TIM0 channel 7	—	
		PTT[7]	I/O	GPIO	—	
	PT6	(SCK0)	I/O	SPI0 serial clock	SPI0RR	
		IOC0_6	I/O	TIM0 channel 6	—	
		PTT[6]	I/O	GPIO	—	
	PT5	(MOSI0)	I/O	SPI0 master out/slave in	SPI0RR	
		IOC0_5	I/O	TIM0 channel 5	—	
		PTT[5]	I/O	GPIO	—	
	PT4	DBGEEV	I	DBG external event	—	
		(MISO0)	I/O	SPI0 master in/slave out	SPI0RR	
		IOC0_4	I/O	TIM0 channel 4	—	
		PTT[4]	I/O	GPIO	—	
	PT3	(PWM0_7)	O	PWM0 channel 7	P0C7RR	
		IOC1_3	I/O	TIM1 channel 3	T1IC3RR	
		PTT[3]	I/O	GPIO	—	
	PT2	(PWM0_3)	O	PWM0 channel 3	P0C3RR	
		IOC1_2	I/O	TIM1 channel 2	T1IC2RR	
		PTT[2]	I/O	GPIO	—	
	PT1	(TXD1)	I/O	SCI1 transmit	SCI1RR	
		(SCL0)	I/O	IIC0 serial clock	IIC0RR	
		IOC1_1	I/O	TIM1 channel 1	—	
		PTT[1]	I/O	GPIO	—	
	PT0	(RXD1)	I	SCI1 receive	SCI1RR	
		(SDA0)	I/O	IIC0 serial data	IIC0RR	
		IOC1_0	I/O	TIM1 channel 0	—	
		PTT[0]	I/O	GPIO	—	

Port	Pin	Pin Function & Priority	I/O	Description	Routing Register Bit	Func. after Reset
S	PS7	IOC0_0	I/O	TIM0 channel 0	—	GPIO
		(IOC0_1)[2]	I	TIM0 channel 1 input capture	T0IC1RR	
		SENT_TX	I/O	SENT_TX_OUT output, SENT_TX_IN input	—	
		$\overline{\text{SS1}}$	I/O	SPI1 slave select	—	
		PTS[7]/ KWS[7]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS6	IOC0_1[3]	I/O	TIM0 channel 1	T0IC1RR	
		SCK1	I/O	SPI1 serial clock	—	
		PTS[6]/ KWS[6]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS5	IOC0_2[3]	I/O	TIM0 channel 2	T0IC2RR	
		TXD1	I/O	SCI1 transmit	SCI1RR	
		MOSI1	I/O	SPI1 master out/slave in	—	
		PTS[5]/ KWS[5]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS4	IOC0_3[3]	I/O	TIM0 channel 3	T0IC3RR1-0	
		RXD1	I	SCI1 receive	SCI1RR	
		MISO1	I/O	SPI1 master in/slave out	—	
		PTS[4]/ KWS[4]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS3	$\overline{\text{XIRQ}}$[4]	I	Non-maskable level-sensitive interrupt	—	
		(TXCAN0)/ (CPDR1)	O	MSCAN0 transmit output/ Direct control output CP0DR[CPDR1]	M0C0RR2-0	
		$\overline{\text{SS0}}$	I/O	SPI0 slave select	SPI0RR	
		PTS[3]/ KWS[3]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS2	$\overline{\text{IRQ}}$	I	Maskable level- or falling edge-sensitive interrupt	—	
		(RXCAN0)	I	MSCAN0 receive input	M0C0RR2-0	
		SCK0	I/O	SPI0 serial clock	SPI0RR	
		PTS[2]/ KWS[2]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS1	(CPTXD0)	I	CANPHY0 transmit input	M0C0RR2-0	
		(TXD0)	I/O	SCI0 transmit	SCI0RR	
		(SCL0)	I/O	IIC0 serial clock	IIC0RR	
		MOSI0	I/O	SPI0 master out/slave in	SPI0RR	
		PTS[1]/ KWS[1]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PS0	(CPRXD0)	O	CANPHY0 receive output	M0C0RR2-0	
		(RXD0)	I	SCI0 receive	SCI0RR	
		(SDA0)	I/O	IIC0 serial data	IIC0RR	
		MISO0	I/O	SPI0 master in/slave out	SPI0RR	
		PTS[0]/ KWS[0]	I/O	GPIO with pin-interrupt and key-wakeup	—	

Port	Pin	Pin Function & Priority	I/O	Description	Routing Register Bit	Func. after Reset
P	PP7	PWM1_7	O	PWM1 channel 7 (fast)	—	GPIO
		PTP[7]/ KWP[7]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PP6[6]	PWM1_5	O	PWM1 channel 5 (fast) with over-current interrupt	—	
		PTP[6]/ KWP[6]	I/O	GPIO with pin-interrupt, key-wakeup and over-current interrupt	—	
	PP5[6]	PWM1_3	O	PWM1 channel 3 (fast) with over-current interrupt	—	
		PTP[5]/ KWP[5]	I/O	GPIO with pin-interrupt, key-wakeup and over-current interrupt	—	
	PP4[6]	(ETRIG0)	I	ADC0 external trigger	TRIG0RR1-0	
		PWM1_1	O	PWM1 channel 1 (fast) with over-current interrupt	—	
		PTP[4]/ KWP[4]	I/O	GPIO with pin-interrupt, key-wakeup and over-current interrupt	—	
	PP3	PWM0_7	O	PWM0 channel 7	P0C7RR	
		PTP[3]/ KWP[3]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PP2[5]	PWM0_5	O	PWM0 channel 5with over-current interrupt	—	
		PTP[2]/ KWP[2]/ EVDD1	I/O	GPIO with pin-interrupt, key-wakeup and over-current interrupt	—	
	PP1	PWM0_3	O	PWM0 channel 3	P0C3RR	
		PTP[1]/ KWP[1]	I/O	GPIO with pin-interrupt and key-wakeup	—	
	PP0[6]	PWM0_1	O	PWM0 channel 1 with over-current interrupt	—	
		PTP[0]/ KWP[0]	I/O	GPIO with pin-interrupt, key-wakeup and over-current interrupt	—	
J	PJ1	TXD0	I/O	SCI0 transmit	SCI0RR	GPIO
		SCL0	I/O	IIC0 serial clock	IIC0RR	
		PTJ[1]	I/O	GPIO	—	
	PJ0	RXD0	I	SCI0 receive	SCI0RR	
		SDA0	I/O	IIC0 serial data	IIC0RR	
		PTJ[0]	I/O	GPIO	—	
L	PL1-0	PTIL[1:0]/ KWL[1:0]	I	High-voltage input (HVI) with pin-interrupt and key-wakeup; optional ADC link	—	HVI

[1] Function active when $\overline{\text{RESET}}$ asserted

[2] Input Capture only. Output Compare on PS6.

[3] Input Capture routed to alternative signal out of reset (refer to Module Routing Register 3 (MODRR3)). Output Compare unaffected.

[4] The interrupt is enabled by clearing the X mask bit in the CPU CCR. The pin is forced to input upon first clearing of the X bit and is held in this state until reset. A stop or wait recovery using XIRQ with the X bit set is not available.

[5] High-current capable high-side output with over-current interrupt (EVDD1)

[6] High-current capable low-side output with over-current interrupt

2.2.1 Internal Routing Options

The following table summarizes the internal routing options.

Table 2-3. Internal Routing Options

Internal Signal	Connects to	Routing Bits
ACMP0 out	TIM1 IC2	T1IC2RR
ACMP1 out	TIM1 IC3	T1IC3RR
ACLK	TIM0 IC2	T0IC2RR
RXD0, RXD1	TIM0 IC3	T0IC3RR1-0
TIM0 OC2	ADC0 Trigger	TRIG0RR1-0, TRIG0NEG

2.3 Memory Map and Register Definition

This section provides a detailed description of all port integration module registers.

2.3.1 Register Map

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0200	MODRR0	R	IIC0RR1-0		SCI1RR	SCI0RR	SPI0RR	M0C0RR2-0		
		W								
0x0201	MODRR1	R	T1IC3RR	T1IC2RR	0	0	0	TRIG0NEG	TRIG0RR1-0	
		W								
0x0202	MODRR2	R	P0C7RR	0	0	0	P0C3RR	0	0	0
		W								
0x0203	MODRR3	R	0	0	0	T0IC3RR1	T0IC3RR0	T0IC2RR	T0IC1RR	0
		W								
0x0204– 0x0207	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0208	ECLKCTL	R	NECLK	0	0	0	0	0	0	0
		W								
0x0209	IRQCR	R	IRQE	IRQEN	0	0	0	0	0	0
		W								
0x020A– 0x020D	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x020E	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x020F	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0210– 0x025F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0260	PTE	R	0	0	0	0	0	0	PTE1	PTE0
		W								
0x0261	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0262	PTIE	R	0	0	0	0	0	0	PTIE1	PTIE0
		W								
0x0263	Reserved	R	0	0	0	0	0	0	0	0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0264	DDRE	R	0	0	0	0	0	0	DDRE1	DDRE0
		W								
0x0265	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0266	PERE	R	0	0	0	0	0	0	PERE1	PERE0
		W								
0x0267	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0268	PPSE	R	0	0	0	0	0	0	PPSE1	PPSE0
		W								
0x0269– 0x027F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0280	PTADH	R	PTADH7	PTADH6	PTADH5	PTADH4	PTADH3	PTADH2	PTADH1	PTADH0
		W								
0x0281	PTADL	R	PTADL7	PTADL6	PTADL5	PTADL4	PTADL3	PTADL2	PTADL1	PTADL0
		W								
0x0282	PTIADH	R	PTIADH7	PTIADH6	PTIADH5	PTIADH4	PTIADH3	PTIADH2	PTIADH1	PTIADH0
		W								
0x0283	PTIADL	R	PTIADL7	PTIADL6	PTIADL5	PTIADL4	PTIADL3	PTIADL2	PTIADL1	PTIADL0
		W								
0x0284	DDRADH	R	DDRADH7	DDRADH6	DDRADH5	DDRADH4	DDRADH3	DDRADH2	DDRADH1	DDRADH0
		W								
0x0285	DDRADL	R	DDRADL7	DDRADL6	DDRADL5	DDRADL4	DDRADL3	DDRADL2	DDRADL1	DDRADL0
		W								
0x0286	PERADH	R	PERADH7	PERADH6	PERADH5	PERADH4	PERADH3	PERADH2	PERADH1	PERADH0
		W								
0x0287	PERADL	R	PERADL7	PERADL6	PERADL5	PERADL4	PERADL3	PERADL2	PERADL1	PERADL0
		W								
0x0288	PPSADH	R	PPSADH7	PPSADH6	PPSADH5	PPSADH4	PPSADH3	PPSADH2	PPSADH1	PPSADH0
		W								
0x0289	PPSADL	R	PPSADL7	PPSADL6	PPSADL5	PPSADL4	PPSADL3	PPSADL2	PPSADL1	PPSADL0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x028A–0x028B	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x028C	PIEADH	R	PIEADH7	PIEADH6	PIEADH5	PIEADH4	PIEADH3	PIEADH2	PIEADH1	PIEADH0
		W								
0x028D	PIEADL	R	PIEADL7	PIEADL6	PIEADL5	PIEADL4	PIEADL3	PIEADL2	PIEADL1	PIEADL0
		W								
0x028E	PIFADH	R	PIFADH7	PIFADH6	PIFADH5	PIFADH4	PIFADH3	PIFADH2	PIFADH1	PIFADH0
		W								
0x028F	PIFADL	R	PIFADL7	PIFADL6	PIFADL5	PIFADL4	PIFADL3	PIFADL2	PIFADL1	PIFADL0
		W								
0x0290–0x0297	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0298	DIENADH	R	DIENADH7	DIENADH6	DIENADH5	DIENADH4	DIENADH3	DIENADH2	DIENADH1	DIENADH0
		W								
0x0299	DIENADL	R	DIENADL7	DIENADL6	DIENADL5	DIENADL4	DIENADL3	DIENADL2	DIENADL1	DIENADL0
		W								
0x029A–0x02BF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02C0	PTT	R	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
		W								
0x02C1	PTIT	R	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
		W								
0x02C2	DDRT	R	DDRT7	DDRT6	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
		W								
0x02C3	PERT	R	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
		W								
0x02C4	PPST	R	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
		W								
0x02C5–0x02CE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02CF	Reserved	R	0	0	0	0	0	0	0	0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x02D0	PTS	R	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		W								
0x02D1	PTIS	R	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		W								
0x02D2	DDRS	R	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		W								
0x02D3	PERS	R	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		W								
0x02D4	PPSS	R	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		W								
0x02D5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02D6	PIES	R	PIES7	PIES6	PIES5	PIES4	PIES3	PIES2	PIES1	PIES0
		W								
0x02D7	PIFS	R	PIFS7	PIFS6	PIFS5	PIFS4	PIFS3	PIFS2	PIFS1	PIFS0
		W								
0x02D8–0x02DE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02DF	WOMS	R	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		W								
0x02E0–0x02EF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F0	PTP	R	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
		W								
0x02F1	PTIP	R	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
		W								
0x02F2	DDRP	R	DDRP7	DDRP6	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
		W								
0x02F3	PERP	R	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
		W								
0x02F4	PPSP	R	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSP0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x02F5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F6	PIEP	R	PIEP7	PIEP6	PIEP5	PIEP4	PIEP3	PIEP2	PIEP1	PIEP0
		W								
0x02F7	PIFP	R	PIFP7	PIFP6	PIFP5	PIFP4	PIFP3	PIFP2	PIFP1	PIFP0
		W								
0x02F8	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F9	OCPEP	R	0	OCPEP6	OCPEP5	OCPEP4	0	OCPEP2	0	OCPEP0
		W								
0x02FA	OCIEP	R	0	OCIEP6	OCIEP5	OCIEP4	0	OCIEP2	0	OCIEP0
		W								
0x02FB	OCIFP	R	0	OCIFP6	OCIFP5	OCIFP4	0	OCIFP2	0	OCIFP0
		W								
0x02FC	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02FD	RDRP	R	0	RDRP6	RDRP5	RDRP4	0	RDRP2	0	RDRP0
		W								
0x02FE–0x02FF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0300–0x030F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0310	PTJ	R	0	0	0	0	0	0	PTJ1	PTJ0
		W								
0x0311	PTIJ	R	0	0	0	0	0	0	PTIJ1	PTIJ0
		W								
0x0312	DDRJ	R	0	0	0	0	0	0	DDRJ1	DDRJ0
		W								
0x0313	PERJ	R	0	0	0	0	0	0	PERJ1	PERJ0
		W								
0x0314	PPSJ	R	0	0	0	0	0	0	PPSJ1	PPSJ0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0315– 0x031E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x031F	WOMJ	R	0	0	0	0	0	0	WOMJ1	WOMJ0
		W								
0x0320– 0x032F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0330	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0331	PTIL	R	0	0	0	0	0	0	PTIL1	PTIL0
		W								
0x0332	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0333	PTPSL	R	0	0	0	0	0	0	PTPSL1	PTPSL0
		W								
0x0334	PPSL	R	0	0	0	0	0	0	PPSL1	PPSL0
		W								
0x0335	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0336	PIEL	R	0	0	0	0	0	0	PIEL1	PIEL0
		W								
0x0337	PIFL	R	0	0	0	0	0	0	PIFL1	PIFL0
		W								
0x0338– 0x0339	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x033A	PTABYPL	R	0	0	0	0	0	0	PTABYPL1	PTABYPL0
		W								
0x033B	PTADIRL	R	0	0	0	0	0	0	PTADIRL1	PTADIRL0
		W								
0x033C	DIENL	R	0	0	0	0	0	0	DIENL1	DIENL0
		W								
0x033D	PTAENL	R	0	0	0	0	0	0	PTAENL1	PTAENL0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x033E	PIRL	R	0	0	0	0	0	0	PIRL1	PIRL0
		W								
0x033F	PTTEL	R	0	0	0	0	0	0	PTTEL1	PTTEL0
		W								
0x0340– 0x037F	Reserved	R	0	0	0	0	0	0	0	0
		W								

2.3.2 PIM Registers 0x0200-0x020F

This section details the specific purposes of register implemented in address range 0x0200-0x020F. These registers serve for specific PIM related functions not part of the generic port registers.

- If not stated differently, writing to reserved bits has no effect and read returns zero.
- All register read accesses are synchronous to internal clocks.
- Register bits can be written at any time if not stated differently.

2.3.2.1 Module Routing Register 0 (MODRR0)

Address 0x0200 Access: User read[1]

	7	6	5	4	3	2	1	0
R W	IIC0RR1-0		SCI1RR	SCI0RR	SPI0RR	M0C0RR2-0		
Routing Option	IIC0		SCI1	SCI0	SPI0	MSCAN0-CANPHY0 interface		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-1. Module Routing Register 0 (MODRR0)

[1] Read: Anytime
Write: Once in normal, anytime in special mode

Table 2-4. MODRR0 Routing Register Field Descriptions

Field	Description
7-6 IIC0RR1-0	**Module Routing Register** — IIC0 routing 11 Reserved 10 SCL0 on PT1; SDA0 on PT0 01 SCL0 on PS1; SDA0 on PS0 00 SCL0 on PJ1; SDA0 on PJ0
5 SCI1RR	**Module Routing Register** — SCI1 routing 1 TXD1 on PT1; RXD1 on PT0 0 TXD1 on PS5; RXD1 on PS4
4 SCI0RR	**Module Routing Register** — SCI0 routing 1 TXD0 on PS1; RXD0 on PS0 0 TXD0 on PJ1; RXD0 on PJ0
3 SPI0RR	**Module Routing Register** — SPI0 routing 1 MISO0 on PT4; MOSI0 on PT5; SCK0 on PT6; SS0 on PT7 0 MISO0 on PS0; MOSI0 on PS1; SCK0 on PS2; SS0 on PS3
2-0 M0C0RR2-0	**Module Routing Register** — MSCAN0-CANPHY0 routing Selection of MSCAN0-CANPHY0 interface routing options to support probing and conformance testing. Refer to Figure 2-2 for an illustration and Table 2-5 for preferred settings. MSCAN0 must be enabled for TXCAN0 routing to take effect on pin. CANPHY0 must be enabled for CPRXD0 and CP0DR[CPDR1] routings to take effect on pins.

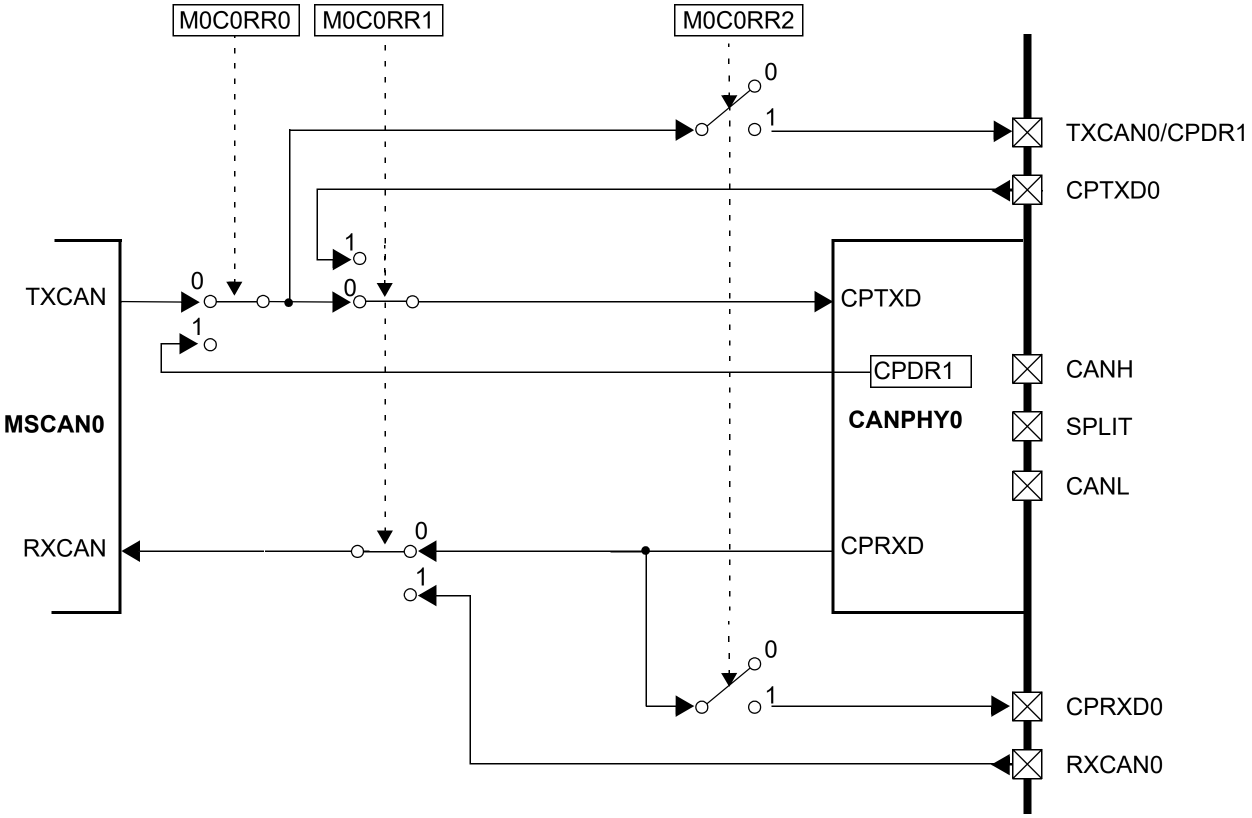


Figure 2-2. CAN Routing Options Illustration

.

Table 2-5. Preferred Interface Configurations

M0C0RR[2:0]	Description
000	Default setting: MSCAN connects to CANPHY, interface internal only
001	Direct control setting: CP0DR[CPDR1] connects to CPTXD, interface internal only
100	Probe setting: MSCAN connects to CANPHY, interface visible on 2 external pins
110	Conformance test setting: Interface opened and all 4 signals routed externally

NOTE

For standalone usage of MSCAN0 on external pins set M0C0RR[2:0]=0b110 and disable CANPHY0 (CPCR[CPE]=0). This releases the CANPHY0 associated pins to other shared functions.

2.3.2.2 Module Routing Register 1 (MODRR1)

Address 0x0201 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	T1IC3RR	T1IC2RR	0	0	0	TRIG0NEG	TRIG0RR1-0	
W								
	TIM1 IC3	TIM1 IC2	—	—	—	ADC0 trigger		
Reset	0	0	0	0	0	0	0	0

Figure 2-3. Module Routing Register 1 (MODRR1)

[1] Read: Anytime
Write: Anytime

Table 2-6. MODRR1 Routing Register Field Descriptions

Field	Description
7 T1IC3RR	**Module Routing Register** — TIM1 IC3 routing 1 TIM1 input capture channel 3 is connected to internal ACMP1 out 0 TIM1 input capture channel 3 is connected to PT3
6 T1IC2RR	**Module Routing Register** — TIM1 IC2 routing 1 TIM1 input capture channel 2 is connected to internal ACMP0 out 0 TIM1 input capture channel 2 is connected to PT2
2 TRIG0NEG	**Module Routing Register** — ADC0 trigger input inverted polarity 1 Falling edge active on ADC0 trigger input 0 Rising edge active on ADC0 trigger input
1-0 TRIG0RR	**Module Routing Register** — ADC0 trigger input routing 11 PP4 (ETRIG0) to ADC0 trigger input 10 PAD1 (ETRIG0) to ADC0 trigger input 01 PAD15 (ETRIG0) to ADC0 trigger input 00 TIM0 output compare channel 2 to ADC0 trigger input (output compare function on pin remains active unless disabled in timer config register TIM0OCPD[OCPD2]=1)

2.3.2.3 Module Routing Register 2 (MODRR2)

Address 0x0202 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	P0C7RR	0	0	0	P0C3RR	0	0	0
W								
	PWM0_7	—	—	—	PWM0_3	—	—	—
Reset	0	0	0	0	0	0	0	0

Figure 2-4. Module Routing Register 2 (MODRR2)

[1] Read: Anytime
Write: Once in normal, anytime in special mode

Table 2-7. MODRR2 Routing Register Field Descriptions

Field	Description
7 P0C7RR	**Module Routing Register** — PWM0_7 routing 1 PWM0_7 to PT3 0 PWM0_7 to PP3
3 P0C3RR	**Module Routing Register** — PWM0_3 routing 1 PWM0_3 to PT2 0 PWM0_3 to PP1

2.3.2.4 Module Routing Register 3 (MODRR3)

Address 0x0203 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	T0IC3RR1	T0IC3RR0	T0IC2RR	T0IC1RR	0
W								
	—	—	—	TIM0 IC3		TIM0 IC2	TIM0 IC1	—
Reset	0	0	0	0	0	0	0	0

Figure 2-5. Module Routing Register 3 (MODRR3)

[1] Read: Anytime
Write: Anytime

Table 2-8. MODRR3 Routing Register Field Descriptions

Field	Description
4 T0IC3RR1	**Module Routing Register** — TIM0 IC3 routing bit 1 If timer channel is not used with a pin (T0IC3RR0=0) then one out of two internal sources can be selected as input. 1 TIM0 input capture channel 3 to RXD1 0 TIM0 input capture channel 3 to RXD0
3 T0IC3RR0	**Module Routing Register** — TIM0 IC3 routing bit 0 1 TIM0 input capture channel 3 to PS4 0 TIM0 input capture channel 3 internally to RXD (see T0IC3RR1)
2 T0IC2RR	**Module Routing Register** — TIM0 IC2 routing 1 TIM0 input capture channel 2 to PS5 0 TIM0 input capture channel 2 internally to ACLK
1 T0IC1RR	**Module Routing Register** — TIM0 IC1 routing 1 TIM0 input capture channel 1 to PS6 0 TIM0 input capture channel 1 to PS7 (SENT_TX)

2.3.2.5 ECLK Control Register (ECLKCTL)

Address 0x0208 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	NECLK	0	0	0	0	0	0	0
W								
Reset:	1	0	0	0	0	0	0	0

Figure 2-6. ECLK Control Register (ECLKCTL)

[1] Read: Anytime
Write: Anytime

Table 2-9. ECLKCTL Register Field Descriptions

Field	Description
7 NECLK	**No ECLK** — Disable ECLK output This bit controls the availability of a free-running clock on the ECLK pin. This clock has a fixed rate equivalent to the internal bus clock. 1 ECLK disabled 0 ECLK enabled

2.3.2.6 IRQ Control Register (IRQCR)

Address 0x0209 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	IRQE	IRQEN	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-7. IRQ Control Register (IRQCR)

[1] Read: Anytime
Write:
IRQE: Once in normal mode, anytime in special mode
IRQEN: Anytime

Table 2-10. IRQCR Register Field Descriptions

Field	Description
7 IRQE	**$\overline{\text{IRQ}}$ select edge sensitive only** — 1 $\overline{\text{IRQ}}$ pin configured to respond only to falling edges. Falling edges on the $\overline{\text{IRQ}}$ pin are detected anytime when IRQE=1 and will be cleared only upon a reset or the servicing of the $\overline{\text{IRQ}}$ interrupt. 0 $\overline{\text{IRQ}}$ configured for low level recognition
6 IRQEN	**$\overline{\text{IRQ}}$ enable** — 1 $\overline{\text{IRQ}}$ pin is connected to interrupt logic 0 $\overline{\text{IRQ}}$ pin is disconnected from interrupt logic

2.3.2.7 Reserved Register

Address 0x020E Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 2-8. Reserved Register

[1] Read: Anytime
Write: Only in special mode

This reserved register is designed for factory test purposes only and is not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

2.3.2.8 Reserved Register

Address 0x020F Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 2-9. Reserved Register

[1] Read: Anytime
Write: Only in special mode

NOTE

This reserved register is designed for factory test purposes only and is not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

2.3.3 PIM Generic Registers

This section describes the details of all configuration registers.

- Writing to reserved bits has no effect and read returns zero.
- All register read accesses are synchronous to internal clocks.
- All registers can be written at any time, however a specific configuration might not become active. E.g. a pullup device does not become active while the port is used as a push-pull output.

- General-purpose data output availability depends on prioritization; input data registers always reflect the pin status independent of the use.
- Pull-device availability, pull-device polarity, wired-or mode, key-wake up functionality are independent of the prioritization unless noted differently.
- For availability of individual bits refer to Section 2.3.1, “Register Map” and Table 2-33.

2.3.3.1 Port Data Register

Address 0x0260 PTE
0x0280 PTADH
0x0281 PTADL
0x02C0 PTT
0x02D0 PTS
0x02F0 PTP
0x0310 PTJ

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PTx7	PTx6	PTx5	PTx4	PTx3	PTx2	PTx1	PTx0
Reset	0	0	0	0	0	0	0	0

Figure 2-10. Port Data Register

[1] Read: Anytime. The data source is depending on the data direction value.
Write: Anytime

This is a generic description of the standard port data registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-11. Port Data Register Field Descriptions

Field	Description
7-0 PTx7-0	**Port Data — General purpose input/output data** This register holds the value driven out to the pin if the pin is used as a general purpose output. When not used with the alternative function (refer to Table 2-2), these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

2.3.3.2 Port Input Register

Address 0x0262 PTIE
0x0282 PTIADH
0x0283 PTIADL
0x02C1 PTIT
0x02D1 PTIS
0x02F1 PTIP
0x0311 PTIJ
0x0331 PTIL

Access: User read only[1]

	7	6	5	4	3	2	1	0
R	PTIx7	PTIx6	PTIx5	PTIx4	PTIx3	PTIx2	PTIx1	PTIx0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

Figure 2-11. Port Input Register

[1] Read: Anytime
Write:Never

This is a generic description of the standard port input registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-12. Port Input Register Field Descriptions

Field	Description
7-0 PTIx7-0	**Port Input** — Data input A read always returns the buffered input state of the associated pin. It can be used to detect overload or short circuit conditions on output pins.

2.3.3.3 Data Direction Register

Address 0x0264 DDRE
0x0284 DDRADH
0x0285 DDRADL
0x02C2 DDRT
0x02D2 DDRS
0x02F2 DDRP
0x0312 DDRJ

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	DDRx7	DDRx6	DDRx5	DDRx4	DDRx3	DDRx2	DDRx1	DDRx0
Reset	0	0	0	0	0	0	0	0

Figure 2-12. Data Direction Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard data direction registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-13. Data Direction Register Field Descriptions

Field	Description
7-0 DDRx7-0	**Data Direction** — Select general-purpose data direction This bit determines whether the pin is a general-purpose input or output. If a peripheral module controls the pin the content of the data direction register is ignored. Independent of the pin usage with a peripheral module this register determines the source of data when reading the associated data register address. **Note:** Due to internal synchronization circuits, it can take up to two bus clock cycles until the correct value is read on port data and port input registers, when changing the data direction register. 1 Associated pin is configured as output 0 Associated pin is configured as input

2.3.3.4 Pull Device Enable Register

Address 0x0266 PERE
0x0286 PERADH
0x0287 PERADL
0x02C3 PERT
0x02D3 PERS
0x02F3 PERP
0x0313 PERJ

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PERx7	PERx6	PERx5	PERx4	PERx3	PERx2	PERx1	PERx0
Reset								
Ports E, J:	0	0	0	0	0	0	1	1
Ports S:	1	1	1	1	1	1	1	1
Others:	0	0	0	0	0	0	0	0

Figure 2-13. Pull Device Enable Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard pull device enable registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-14. Pull Device Enable Register Field Descriptions

Field	Description
7-0 PERx7-0	**Pull Enable** — Activate pull device on input pin This bit controls whether a pull device on the associated port input or open-drain output pin is active. If a pin is used as push-pull output this bit has no effect. The polarity is selected by the related polarity select register bit. On open-drain output pins only a pullup device can be enabled. 1 Pull device enabled 0 Pull device disabled

2.3.3.5 Polarity Select Register

Address 0x0268 PPSE
0x0288 PPSADH
0x0289 PPSADL
0x02C4 PPST
0x02D4 PPSS
0x02F4 PPSP
0x0314 PPSJ

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PPSx7	PPSx6	PPSx5	PPSx4	PPSx3	PPSx2	PPSx1	PPSx0
Reset								
Ports E:	0	0	0	0	0	0	1	1
Others:	0	0	0	0	0	0	0	0

Figure 2-14. Polarity Select Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard polarity select registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-15. Polarity Select Register Field Descriptions

Field	Description
7-0 PPSx7-0	**Pull Polarity Select** — Configure pull device and pin interrupt edge polarity on input pin This bit selects a pullup or a pulldown device if enabled on the associated port input pin. If a port has interrupt functionality this bit also selects the polarity of the active edge. **Note:** If MSCAN is active a pullup device can be activated on the RXCAN input; attempting to select a pulldown disables the pull-device. 1 Pulldown device selected; rising edge selected 0 Pullup device selected; falling edge selected

2.3.3.6 Port Interrupt Enable Register

Address 0x028C PIEADH
0x028D PIEADL
0x02D6 PIES
0x02F6 PIEP
0x0336 PIEL

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PIEx7	PIEx6	PIEx5	PIEx4	PIEx3	PIEx2	PIEx1	PIEx0
Reset	0	0	0	0	0	0	0	0

Figure 2-15. Port Interrupt Enable Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard port interrupt enable registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-16. Port Interrupt Enable Register Field Descriptions

Field	Description
7-0 PIEx7-0	**Port Interrupt Enable** — Activate pin interrupt (KWU) This bit enables or disables the edge sensitive pin interrupt on the associated pin. An interrupt can be generated if the pin is operating in input or output mode when in use with the general-purpose or related peripheral function. 1 Interrupt is enabled 0 Interrupt is disabled (interrupt flag masked)

2.3.3.7 Port Interrupt Flag Register

Address 0x028E PIFADH
0x028F PIFADL
0x02D7 PIFS
0x02F7 PIFP
0x0337 PIFL

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PIFx7	PIFx6	PIFx5	PIFx4	PIFx3	PIFx2	PIFx1	PIFx0
Reset	0	0	0	0	0	0	0	0

Figure 2-16. Port Interrupt Flag Register

[1] Read: Anytime
Write: Anytime, write 1 to clear

This is a generic description of the standard port interrupt flag registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-17. Port Interrupt Flag Register Field Descriptions

Field	Description
7-0 PIFx7-0	**Port Interrupt Flag** — Signal pin event (KWU) This flag asserts after a valid active edge was detected on the related pin (see Section 2.4.4.2, “Pin Interrupts and Key-Wakeup (KWU)”). This can be a rising or a falling edge based on the state of the polarity select register. An interrupt will occur if the associated interrupt enable bit is set. Writing a logic “1” to the corresponding bit field clears the flag. 1 Active edge on the associated bit has occurred 0 No active edge occurred

2.3.3.8 Digital Input Enable Register

Address 0x0298 DIENADH
0x0299 DIENADL

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	DIENx7	DIENx6	DIENx5	DIENx4	DIENx3	DIENx2	DIENx1	DIENx0
Reset	0	0	0	0	0	0	0	0

Figure 2-17. Digital Input Enable Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard digital input enable registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-18. Digital Input Enable Register Field Descriptions

Field	Description
7-0 DIENx7-0	**Digital Input Enable** — Input buffer control This bit controls the digital input function. If set to 1 the input buffers are enabled and the pin can be used with the digital function. If a peripheral module is enabled which uses the pin with a digital function the input buffer is activated and the register bit is ignored. If the pin is used with an analog function this bit shall be cleared to avoid shoot-through current. 1 Associated pin is configured as digital input 0 Associated pin digital input is disabled

2.3.3.9 Reduced Drive Register

Address 0x02FD RDRP

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	RDRx7	RDRx6	RDRx5	RDRx4	RDRx3	RDRx2	RDRx1	RDRx0
Reset	0	0	0	0	0	0	0	0

Figure 2-18. Reduced Drive Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard reduced drive registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-19. Reduced Drive Register Field Descriptions

Field	Description
7-0 RDRx7-0	**Reduced Drive Register** — Select reduced drive for output pin This bit configures the drive strength of the associated output pin as either full or reduced. If a pin is used as input this bit has no effect. The reduced drive function is independent of which function is being used on a particular pin. 1 Reduced drive selected (approx. 1/10 of the full drive strength) 0 Full drive strength enabled

2.3.3.10 Wired-Or Mode Register

Address 0x02DF WOMS
0x031F WOMJ

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	WOMx7	WOMx6	WOMx5	WOMx4	WOMx3	WOMx2	WOMx1	WOMx0
Reset	0	0	0	0	0	0	0	0

Figure 2-19. Wired-Or Mode Register

[1] Read: Anytime
Write: Anytime

This is a generic description of the standard wired-or registers. Refer to Table 2-33 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-20. Wired-Or Mode Register Field Descriptions

Field	Description
7-0 WOMx7-0	**Wired-Or Mode** — Enable open-drain output This bit configures the output buffer as wired-or. If enabled the output is driven active low only (open-drain) while the active high drive is turned off. This allows a multipoint connection of several serial modules. These bits have no influence on pins used as inputs. 1 Output buffers operate as open-drain outputs 0 Output buffers operate as push-pull outputs

2.3.3.11 PIM Reserved Register

Address (any reserved)

Access: User read[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-20. PIM Reserved Register

[1] Read: Always reads 0x00
Write: Unimplemented

2.3.4 PIM Generic Register Exceptions

This section lists registers with deviations from the generic description in one or more register bits.

2.3.4.1 Port P Over-Current Protection Enable Register (OCPEP)

Address 0x02F9 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	OCPEP6	OCPEP5	OCPEP4	0	OCPEP2	0	OCPEP0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-21. Over-Current Protection Enable Register (OCPEP)

[1] Read: Anytime
Write:Anytime

Table 2-21. OCPEP Register Field Descriptions

Field	Description
6 OCPEP6	**Over-Current Protection Enable** — Activate over-current detector on PP6 Refer to Section ”2.5.4, “Over-Current Protection on PP[6-4,0]”” 1 PP6 over-current detector enabled 0 PP6 over-current detector disabled
5 OCPEP5	**Over-Current Protection Enable** — Activate over-current detector on PP5 Refer to Section 2.5.4, “Over-Current Protection on PP[6-4,0]”” 1 PP5 over-current detector enabled 0 PP5 over-current detector disabled
4 OCPEP4	**Over-Current Protection Enable** — Activate over-current detector on PP4 Refer to Section 2.5.4, “Over-Current Protection on PP[6-4,0]”” 1 PP4 over-current detector enabled 0 PP4 over-current detector disabled
2 OCPEP2	**Over-Current Protection Enable** — Activate over-current detector on EVDD1 Refer to Section 2.5.3, “Over-Current Protection on PP2 (EVDD1)”” 1 EVDD1 over-current detector enabled 0 EVDD1 over-current detector disabled
0 OCPEP0	**Over-Current Protection Enable** — Activate over-current detector on PP0 Refer to Section 2.5.4, “Over-Current Protection on PP[6-4,0]”” 1 PP0 over-current detector enabled 0 PP0 over-current detector disabled

2.3.4.2 Port P Over-Current Interrupt Enable Register (OCIEP)

Address 0x02FA

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	OCIEP6	OCIEP5	OCIEP4	0	OCIEP2	0	OCIEP0
W	▨				▨		▨	
Reset	0	0	0	0	0	0	0	0

Figure 2-22. Port P Over-Current Interrupt Enable Register

[1] Read: Anytime
Write: Anytime

Table 2-22. Port P Over-Current Interrupt Enable Register

Field	Description
6 OCIEP6	**Over-Current Interrupt Enable** — This bit enables or disables the over-current interrupt on PP6. 1 PP6 over-current interrupt enabled 0 PP6 over-current interrupt disabled (interrupt flag masked)
5 OCIEP5	**Over-Current Interrupt Enable** — This bit enables or disables the over-current interrupt on PP5. 1 PP5 over-current interrupt enabled 0 PP5 over-current interrupt disabled (interrupt flag masked)
4 OCIEP4	**Over-Current Interrupt Enable** — This bit enables or disables the over-current interrupt on PP4. 1 PP4 over-current interrupt enabled 0 PP4 over-current interrupt disabled (interrupt flag masked)
2 OCIEP2	**Over-Current Interrupt Enable** — This bit enables or disables the over-current interrupt on EVDD1. 1 EVDD1 over-current interrupt enabled 0 EVDD1 over-current interrupt disabled (interrupt flag masked)
0 OCIEP0	**Over-Current Interrupt Enable** — This bit enables or disables the over-current interrupt on PP0. 1 PP0 over-current interrupt enabled 0 PP0 over-current interrupt disabled (interrupt flag masked)

2.3.4.3 Port P Over-Current Interrupt Flag Register (OCIFP)

Address 0x02FB

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	OCIFP6	OCIFP5	OCIFP4	0	OCIFP2	0	OCIFP0
W	▨				▨		▨	
Reset	0	0	0	0	0	0	0	0

Figure 2-23. Port P Over-Current Interrupt Flag Register

[1] Read: Anytime
Write: Anytime, write 1 to clear

Table 2-23. Port P Over-Current Interrupt Flag Register

Field	Description
6 OCIFP6	**Over-Current Interrupt Flag** — This flag asserts if an over-current condition is detected on PP6 (Section 2.4.4.3, “Over-Current Interrupt and Protection””). Writing a logic “1” to the corresponding bit field clears the flag. 1 PP6 over-current event occurred 0 No PP6 over-current event occurred
5 OCIFP5	**Over-Current Interrupt Flag** — This flag asserts if an over-current condition is detected on PP5 (Section 2.4.4.3, “Over-Current Interrupt and Protection””). Writing a logic “1” to the corresponding bit field clears the flag. 1 PP5 over-current event occurred 0 No PP5 over-current event occurred
4 OCIFP4	**Over-Current Interrupt Flag** — This flag asserts if an over-current condition is detected on PP4 (Section 2.4.4.3, “Over-Current Interrupt and Protection””). Writing a logic “1” to the corresponding bit field clears the flag. 1 PP4 over-current event occurred 0 No PP4 over-current event occurred
2 OCIFP2	**Over-Current Interrupt Flag** — This flag asserts if an over-current condition is detected on EVDD1 (Section 2.4.4.3, “Over-Current Interrupt and Protection””). Writing a logic “1” to the corresponding bit field clears the flag. 1 EVDD1 over-current event occurred 0 No EVDD1 over-current event occurred
0 OCIFP0	**Over-Current Interrupt Flag** — This flag asserts if an over-current condition is detected on PP0 (Section 2.4.4.3, “Over-Current Interrupt and Protection””). Writing a logic “1” to the corresponding bit field clears the flag. 1 PP0 over-current event occurred 0 No PP0 over-current event occurred

2.3.4.4 Port L Input Register (PTIL)

Address 0x0331 Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTIL1	PTIL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-24. Port L Input Register (PTIL)

[1] Read: Anytime
Write: No Write

Table 2-24. PTIL - Register Field Descriptions

Field	Description
1-0 PTIL1-0	**Port Input Data Register Port L** — A read returns the synchronized input state if the associated HVI pin is used in digital mode, that is the related DIENL bit is set to 1 and the pin is not used in analog mode (PTAENL=0). See Section 2.3.4.10, “Port L ADC Connection Enable Register (PTAENL)”. A one is read in any other case[1].

[1] Refer to PTTEL bit description in Section 2.3.4.12, “Port L Test Enable Register (PTTEL) for an override condition.

2.3.4.5 Port L Pull Select Register (PTPSL)

Address 0x0333 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTPSL1	PTPSL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-25. Port L Pull Select Register (PTPSL)

[1] Read: Anytime
Write: Anytime

Table 2-25. PTPSL Register Field Descriptions

Field	Description
1-0 PTPSL1-0	**Port L Pull Select** — This bit selects a pull device on the HVI pin in analog mode for open input detection. By default a pulldown device is active as part of the input voltage divider. If this bit set to 1 and PTTEL=1 and not in stop mode a pullup to a level close to V_{DDX} takes effect and overrides the weak pulldown device. Refer to Section 2.5.5, “Open Input Detection on PL[1:0] (HVI)”. 1 Pullup enabled 0 Pulldown enabled

2.3.4.6 Port L Polarity Select Register (PPSL)

Address 0x0334 PPSL

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PPSL1	PPSL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-26. Port L Polarity Select Register (PPSL)

[1] Read: Anytime
Write: Anytime

Table 2-26. PPSL Register Field Descriptions

Field	Description
1-0 PPSL1-0	**Polarity Select** — This bit selects the polarity of the active interrupt edge on the associated HVI pin. 1 Rising edge selected 0 Falling edge selected

2.3.4.7 Port L ADC Bypass Register (PTABYPL)

Address 0x033A

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTABYPL1	PTABYPL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-27. Port L ADC Bypass Register (PTABYPL)

[1] Read: Anytime
Write: Anytime

Table 2-27. PTABYPL Register Field Descriptions

Field	Description
1-0 PTABYPL 1-0	**Port L ADC Connection Bypass** — This bit bypasses and powers down the impedance converter stage in the signal path from the analog input pin to the ADC channel input. This bit takes effect only if using direct input connection to the ADC channel (PTADIRL=1). 1 Impedance converter bypassed 0 Impedance converter used

2.3.4.8 Port L ADC Direct Register (PTADIRL)

Address 0x033B | Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTADIRL1	PTADIRL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-28. Port L ADC Direct Register (PTADIRL)

[1] Read: Anytime
Write: Anytime

Table 2-28. PTADIRL Register Field Descriptions

Field	Description
1-0 PTADIRL 1-0	**Port L ADC Direct Connection** — This bit connects the analog input signal directly to the ADC channel bypassing the voltage divider. This bit takes effect only in analog mode (PTAENL=1). 1 Input pin directly connected to ADC channel 0 Input voltage divider active on analog input to ADC channel

2.3.4.9 Port L Digital Input Enable Register (DIENL)

Address 0x33C | Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DIENL1	DIENL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-29. Port L Digital Input Enable Register (DIENL)

[1] Read: Anytime
Write: Anytime

Table 2-29. DIENL Register Field Descriptions

Field	Description
1-0 DIENL1-0	**Digital Input Enable Port L** — Input buffer control This bit controls the HVI digital input function. If set to 1 the input buffer is enabled and the HVI pin can be used with the digital function. If the analog input function is enabled (PTAENL=1) the input buffer of the selected HVI pin is forced off[1] in run mode and is released to be active in stop mode[2] only if DIENL=1. 1 Associated pin digital input is enabled if not used as analog input in run mode[1] 0 Associated pin digital input is disabled[1]

[1] Refer to PTTEL bit description in Section 2.3.4.10, “Port L ADC Connection Enable Register (PTAENL) for an override condition.

[2] "Stop mode" is limited to RPM; refer to Table 2-36.

2.3.4.10 Port L ADC Connection Enable Register (PTAENL)

Address 0x033D Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTAENL1	PTAENL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-30. Port L ADC Connection Enable Register (PTAENL)

[1] Read: Anytime
Write: Anytime

Table 2-30. PTAENL Register Field Descriptions

Field	Description
1-0 PTAENL 1-0	**Port L ADC Connection Enable** — This bit enables the analog signal link to an ADC channel. If set to 1 the analog input function takes precedence over the digital input in run mode by forcing off the input buffer if not overridden by PTTEL=1. **Note:** When enabling the resistor paths to ground by setting PTAENL=1, a delay of t_{UNC_HVI} + two bus cycles must be accounted for. 1 ADC connection enabled 0 ADC connection disabled

2.3.4.11 Port L Input Divider Ratio Selection Register (PIRL)

Address 0x033E Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PIRL1	PIRL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-31. Port L Input Divider Ratio Selection Register (PIRL)

[1] Read: Anytime
Write: Anytime

Table 2-31. PIRL Register Field Descriptions

Field	Description
1-0 PIRL1-0	**Port L Input Divider Ratio Select** — This bit selects one of two voltage divider ratios for the associated HVI pin in analog mode. 1 Ratio_{L_HVI} selected 0 Ratio_{H_HVI} selected

2.3.4.12 Port L Test Enable Register (PTTEL)

Address 0x033F Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTTEL1	PTTEL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-32. Port L Test Enable Register (PTTEL)

[1] Read: Anytime
Write: Anytime

Table 2-32. PTTEL Register Field Descriptions

Field	Description
1-0 PTTEL1-0	**Port L Test Enable** — This bit forces the input buffer of the HVI pin active while using the analog function to support open input detection in run mode. Refer to Section 2.5.5, "Open Input Detection on PL[1:0] (HVI)"). In stop mode this bit has no effect. **Note:** In direct mode (PTADIRL=1) the digital input buffer is not enabled. 1 Input buffer enabled when used with analog function and not in direct mode (PTADIRL=0) 0 Input buffer disabled when used with analog function

2.4 Functional Description

2.4.1 General

Each pin except BKGD can act as general-purpose I/O. In addition each pin can act as an output or input of a peripheral module.

2.4.2 Registers

Table 2-33 lists the implemented configuration bits which are available on each port. These registers except the pin input registers can be written at any time, however a specific configuration might not become active. For example a pullup device does not become active while the port is used as a push-pull output.

Unimplemented bits read zero.

Table 2-33. Bit Indices of Implemented Register Bits per Port

	Port Data Register	Port Input Register	Data Direction Register	Pull Device Enable Register	Polarity Select Register	Port Interrupt Enable Register	Port Interrupt Flag Register	Digital Input Enable Register	Reduced Drive Register	Wired-Or Mode Register
Port	**PT**	**PTI**	**DDR**	**PER**	**PPS**	**PIE**	**PIF**	**DIE**	**RDR**	**WOM**
E	1-0	1-0	1-0	1-0	1-0	-	-	-	-	-
ADH	7-0	7-0	7-0	7-0	7-0	7-0	7-0	7-0	-	-
ADL	7-0	7-0	7-0	7-0	7-0	7-0	7-0	7-0	-	-
T	7-0	7-0	7-0	7-0	7-0	-	-	-	-	-
S	7-0	7-0	7-0	7-0	7-0	7-0	7-0	-	-	7-0
P	7-0	7-0	7-0	7-0	7-0	7-0	7-0	-	6-4,2,0	-
J	1-0	1-0	1-0	1-0	1-0	-	-	-	-	1-0
L	-	1-0	-	-	1-0	1-0	1-0	1-0	-	-

Table 2-34 shows the effect of enabled peripheral features on I/O state and enabled pull devices.

Table 2-34. Effect of Enabled Features

Enabled Feature[1]	Related Signal(s)	Effect on I/O state	Effect on enabled pull device
CPMU OSC	EXTAL, XTAL	CPMU takes control	Forced off
TIMx output compare y	IOCx_y	Forced output	Forced off, pulldown forced off if open-drain
TIMx input capture y	IOCx_y	None[2]	None[3]
SPIx	MISOx, MOSx, SCKx, $\overline{\text{SSx}}$	SPI takes control	Forced off if output, pulldown forced off if open-drain
SCIx transmitter	TXDx	Forced output	Forced off, pulldown forced off if open-drain
SCIx receiver	RXDx	Forced input	None[3]
IICx	SDAx, SCLx	Forced open-drain	Pulldown forced off
S12ZDBG	DBGEEV	None[2]	None[3]
PWMx channel y	PWMx_y	Forced output	Forced off
ADCx	ANy	None[2][4]	None[3]
	VRH		
ACMPx	ACMPx_0, ACMPx_1	None[2][4]	None[3]
	ACMPOx	Forced output	Forced off
DACx	AMPPx, AMPMx	None[2][4]	None[3]
	DACUx, AMPx	Digital output forced off	Forced off

Table 2-34. Effect of Enabled Features

Enabled Feature[1]	Related Signal(s)	Effect on I/O state	Effect on enabled pull device
SENT	SENT_TX	Forced output	Forced off if push-pull, pulldown forced off if open-drain
IRQ	$\overline{\text{IRQ}}$	Forced input	None[3]
XIRQ	$\overline{\text{XIRQ}}$	Forced input	None[3]
MSCANx	TXCANx	Forced output	Forced off
	RXCANx	Forced input	Pulldown forced off
CANPHYx	CPTXDx	Forced input	None[3]
	CPRXDx	Forced output	Forced off, pulldown forced off if open-drain

[1] If applicable the appropriate routing configuration must be set for the signals to take effect on the pins.

[2] DDR maintains control

[3] PER/PPS maintain control

[4] To use the digital input function the related bit in Digital Input Enable Register (DIENADH/L) must be set to logic level “1”.

2.4.3 Pin I/O Control

Figure 2-33 illustrates the data paths to and from an I/O pin. Input and output data can always be read via the input register (PTIx, Section 2.3.3.2, “Port Input Register”) independent if the pin is used as general-purpose I/O or with a shared peripheral function. If the pin is configured as input (DDRx=0, Section 2.3.3.3, “Data Direction Register”), the pin state can also be read through the data register (PTx, Section 2.3.3.1, “Port Data Register”).

The general-purpose data direction configuration can be overruled by an enabled peripheral function shared on the same pin (Table 2-34). If more than one peripheral function is available and enabled at the same time, the highest ranked module according the predefined priority scheme in Table 2-2 will take precedence on the pin.

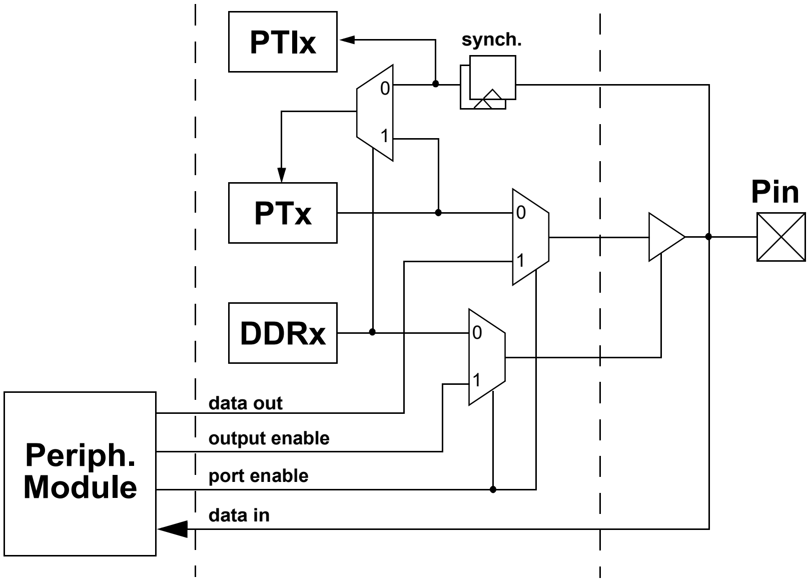


Figure 2-33. Illustration of I/O pin functionality

2.4.4 Interrupts

This section describes the interrupts generated by the PIM and their individual sources. Vector addresses and interrupt priorities are defined at MCU level.

Table 2-35. PIM Interrupt Sources

Module Interrupt Sources	Local Enable
$\overline{\text{XIRQ}}$	None
$\overline{\text{IRQ}}$	IRQCR[IRQEN]
Port AD pin interrupt	PIEADH[PIEADH] PIEADL[PIEADL]
Port S pin interrupt	PIES[PIES]
Port P pin interrupt	PIEP[PIEP]
Port L pin interrupt	PIEL[PIEL]
Port P over-current interrupt	OCIEP[OCIEP]

2.4.4.1 XIRQ, IRQ Interrupts

The $\overline{\text{XIRQ}}$ pin allows requesting non-maskable interrupts after reset initialization. During reset, the X bit in the condition code register is set and any interrupts are masked until software enables them.

The $\overline{\text{IRQ}}$ pin allows requesting asynchronous interrupts. The interrupt input is disabled out of reset. To enable the interrupt the IRQCR[IRQEN] bit must be set and the I bit cleared in the condition code register. The interrupt can be configured for level-sensitive or falling-edge-sensitive triggering. If IRQCR[IRQEN] is cleared while an interrupt is pending, the request will deassert.

Both interrupts are capable to wake-up the device from stop mode. Means for glitch filtering are not provided on these pins.

2.4.4.2 Pin Interrupts and Key-Wakeup (KWU)

Ports AD, S, P and L offer pin interrupt and key-wakeup capability. The related interrupt enable (PIE) as well as the sensitivity to rising or falling edges (PPS) can be individually configured on per-pin basis. All bits/pins in a port share the same interrupt vector. Interrupts can be used with the pins configured as inputs or outputs.

An interrupt is generated when a bit in the port interrupt flag (PIF) and its corresponding port interrupt enable (PIE) are both set. The pin interrupt feature is also capable to wake up the CPU when it is in stop or wait mode (key-wakeup).

A digital filter on each pin prevents short pulses from generating an interrupt. A valid edge on an input is detected if 4 consecutive samples of a passive level are followed by 4 consecutive samples of an active level. Else the sampling logic is restarted.

In run and wait mode the filters are continuously clocked by the bus clock. Pulses with a duration of $t_{PULSE} < n_{P\_MASK}/f_{bus}$ are assuredly filtered out while pulses with a duration of $t_{PULSE} > n_{P\_PASS}/f_{bus}$ guarantee a pin interrupt.

In stop mode the filter clock is generated by an RC-oscillator. The minimum pulse length varies over process conditions, temperature and voltage (Figure 2-34). Pulses with a duration of $t_{PULSE} < t_{P\_MASK}$ are assuredly filtered out while pulses with a duration of $t_{PULSE} > t_{P\_PASS}$ guarantee a wakeup event.

Please refer to the appendix table “Pin Interrupt Characteristics” for pulse length limits.

To maximize current saving the RC oscillator is active only if the following condition is true on any individual pin:

Sample count <= 4 (at active or passive level) and interrupt enabled (PIE[x]=1) and interrupt flag not set (PIF[x]=0).

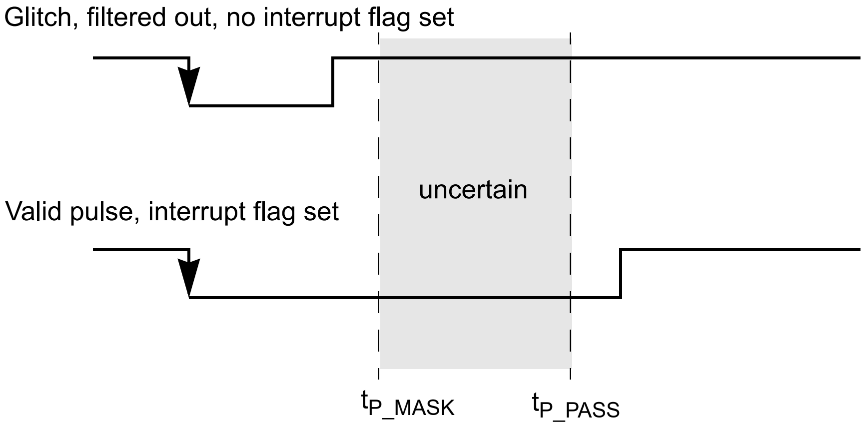


Figure 2-34. Interrupt Glitch Filter (here: active low level selected)

2.4.4.3 Over-Current Interrupt and Protection

In case of an over-current condition on PP2 (EVDD1) or PP[6-4,0] (see Section 2.5.3, “Over-Current Protection on PP2 (EVDD1)”” and 2.5.4, “Over-Current Protection on PP[6-4,0]””) the related over-current interrupt flag OCIFP[OCIFP] asserts. This flag generates an interrupt if the enable bit OCIEP[OCIEP] is set.

An asserted flag immediately forces the related output independent of its driving source (peripheral output or port register bit) to its disabled level to protect the device. The flag must be cleared to re-enable the driver.

2.4.5 High-Voltage Input

A high-voltage input (HVI) on port L has the following features:

- Input voltage proof up to V_{HVI}
- Digital input function with pin interrupt and wakeup from stop capability
- Analog input function with selectable divider ratio routable to ADC channel. Optional direct input bypassing voltage divider and impedance converter. Capable to wakeup from stop (pin interrupts in run mode not available). Open input detection.

Figure 2-35 shows a block diagram of the HVI.

NOTE

> The term stop mode (STOP) is limited to voltage regulator operating in reduced performance mode (RPM). Refer to “Low Power Modes” section in device overview.

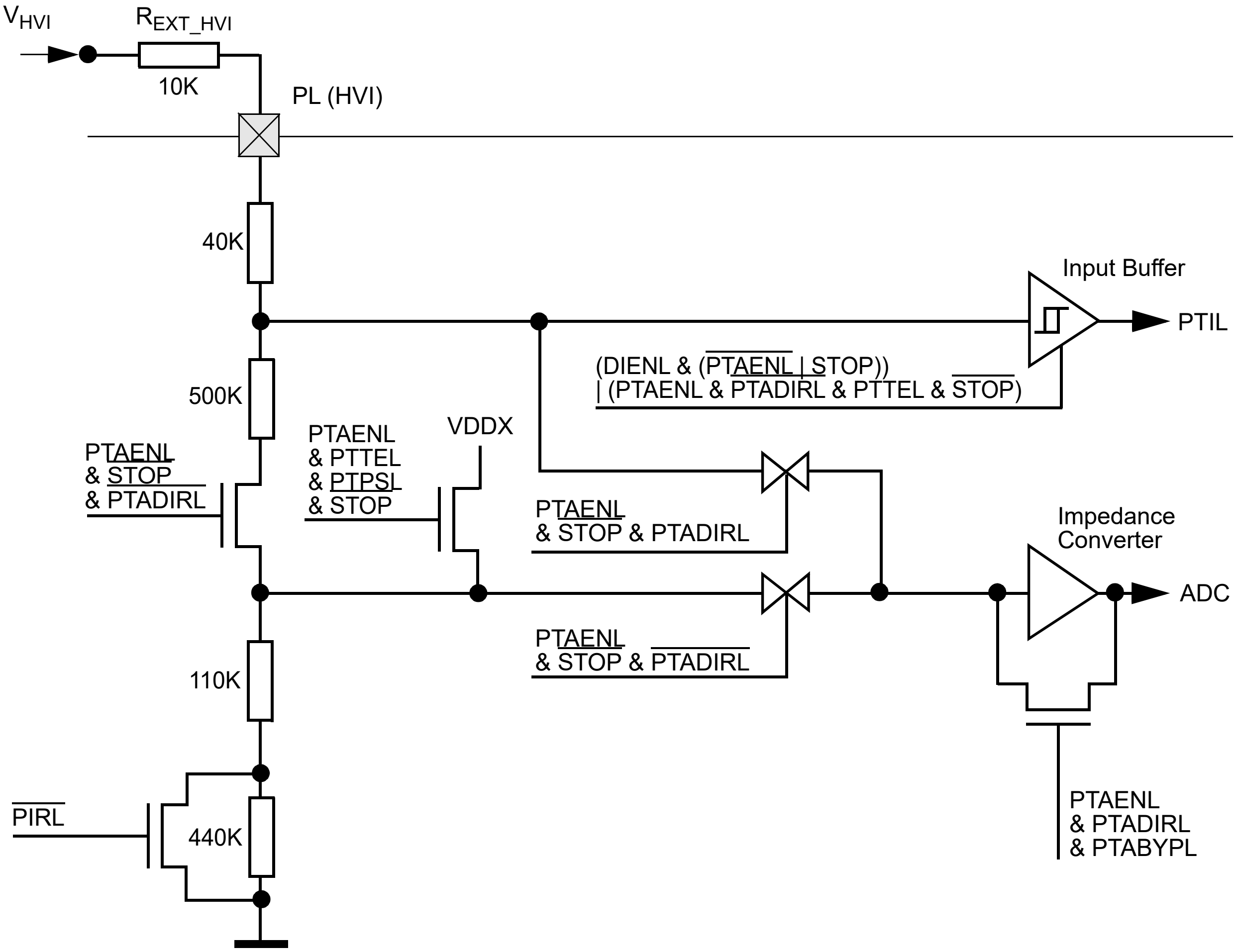


Figure 2-35. HVI Block Diagram

Voltages up to V_{HVI} can be applied to the HVI pin. Internal voltage dividers scale the input signals down to logic level. There are two modes, digital and analog, where these signals can be processed.

2.4.5.1 Digital Mode Operation

In digital mode the input buffer is enabled (DIENL=1 & PTAENL=0). The synchronized pin input state determined at threshold level V_{TH_HVI} can be read in register PTIL. An interrupt flag (PIFL) is set on input transitions of the configured edge polarity (PPSL). An interrupt (PIFL) is generated if enabled (PIEL=1) and the interrupt being set (PIFL=1). Wakeup from stop mode is supported.

2.4.5.2 Analog Mode Operation

In analog mode (PTAENL=1) the input buffer is forced off and the voltage applied to a selectable HVI pin can be measured on its related internal ADC channel (refer to device overview section for channel assignment). One of two input divider ratios ($Ratio_{H\_HVI}$, $Ratio_{L\_HVI}$) can be chosen (PIRL) on the analog input or the voltage divider can be bypassed (PTADIRL=1). Additionally in latter case the impedance converter in the ADC signal path can be used or bypassed in direct input mode (PTABYPL).

Out of reset the digital input buffer of the selected pin is disabled to avoid shoot-through current. Thus pin interrupts can only be generated if DIENL=1.

In stop mode (RPM) the digital input buffer is enabled only if DIENL=1 to support wakeup functionality.

Table 2-36 shows the HVI input configuration depending on register bits and operation mode.

Table 2-36. HVI Input Configurations

Mode	DIENL	PTAENL	Digital Input	Analog Input	Resulting Function
Run	0	0	off	off	Input disabled (Reset)
	0	1	off[1]	enabled	Analog input, interrupt not supported
	1	0	enabled	off	Digital input, interrupt supported
	1	1	off[1]	enabled	Analog input, interrupt not supported
Stop[2]	0	X	off	off	Input disabled, wakeup from stop not supported
	1	X	enabled	off	Digital input, wakeup from stop supported

[1] Enabled if PTTEL=1 & PTADIRL=0)

[2] The term “stop mode” is limited to voltage regulator operating in reduced performance mode (RPM; refer to “Low Power Modes” section in device overview). In any other case the HVI input configuration defaults to “run mode”. Therefore set PTAENL=0 before entering stop mode in order to generally support wakeup from stop.

NOTE

An external resistor R_{EXT_HVI} must always be connected to the high-voltage input to protect the device pins from fast transients and to achieve the specified pin input divider ratios when using the HVI in analog mode.

2.5 Initialization and Application Information

2.5.1 Port Data and Data Direction Register writes

It is not recommended to write PORTx/PTx and DDRx in a word access. When changing the register pins from inputs to outputs, the data may have extra transitions during the write access. Initialize the port data register before enabling the outputs.

2.5.2 SCI Baud Rate Detection

The baud rate for SCI0 and SCI1 can be determined by using a timer channel to measure the data rate on the related RXD signal.

1. Establish the link:
   - For SCI0: Set MODRR3[T0IC3RR1:T0IC3RR0]=2b00 to route TIM0 input capture channel 3 to internal RXD0 signal of SCI0.

   - For SCI1: Set MODRR3[T0IC3RR1:T0IC3RR0]=2b10 to route TIM0 input capture channel 3 to internal RXD1 signal of SCI1.
2. Determine pulse width of incoming data: Configure TIM0 input capture channel 3 to measure time between incoming signal edges.

2.5.3 Over-Current Protection on PP2 (EVDD1)

Pin PP2 can be used as general-purpose I/O or due to its increased current capability in output mode as a switchable external power supply pin (EVDD1) for external devices like Hall sensors.

EVDD1 connects the load to the digital supply VDDX.

An over-current monitor is implemented to protect the controller from short circuits or excess currents on the output which can only arise if the pin is configured for full drive. Although the full drive current is available on the high and low side, the protection is only available on the high side when sourcing current from EVDD1 to VSSX. There is also no protection to voltages higher than V_{DDX}.

To power up the over-current monitor set the related OCPEx bit.

In stop mode the over-current monitor is disabled for power saving. The increased current capability cannot be maintained to supply the external device. Therefore when using the pin as power supply the external load must be powered down prior to entering stop mode by driving the output low.

An over-current condition is detected if the output current level exceeds the threshold I_{OCD} in run mode. The output driver is immediately forced low and the over-current interrupt flag OCIFx asserts. Refer to Section 2.4.4.3, “Over-Current Interrupt and Protection””.

2.5.4 Over-Current Protection on PP[6-4,0]

Pins PP[6-4,0] can be used as general-purpose I/O or due to their increased current capability in output mode as a switchable external power ground pin for external devices like LEDs supplied by VDDX.

PP[6-4,0] connect the loads to the digital ground VSSX.

Similar protection mechanisms as for EVDD1 apply for PP[6-4,0] accordingly in an inverse way.

2.5.5 Open Input Detection on PL[1:0] (HVI)

The connection of an external pull device on a high-voltage input can be validated by using the built-in pull functionality of the HVI. Depending on the application type an external pulldown circuit can be detected with the internal pullup device whereas an external pullup circuit can be detected with the internal pulldown device which is part of the input voltage divider.

Note that the following procedures make use of a function that overrides the automatic disable mechanism of the digital input buffer when using the HVI in analog mode. Make sure to switch off the override function when using the HVI in analog mode after the check has been completed.

External pulldown device (Figure 2-36):

1. Enable analog function on HVI in non-direct mode (PTAENL=1, PTADIRL=0)

2. Select internal pullup device on HVI (PTPSL=1)
3. Enable function to force input buffer active on HVI in analog mode (PTTEL=1)
4. Verify PTIL=0 for a connected external pulldown device; read PTIL=1 for an open input

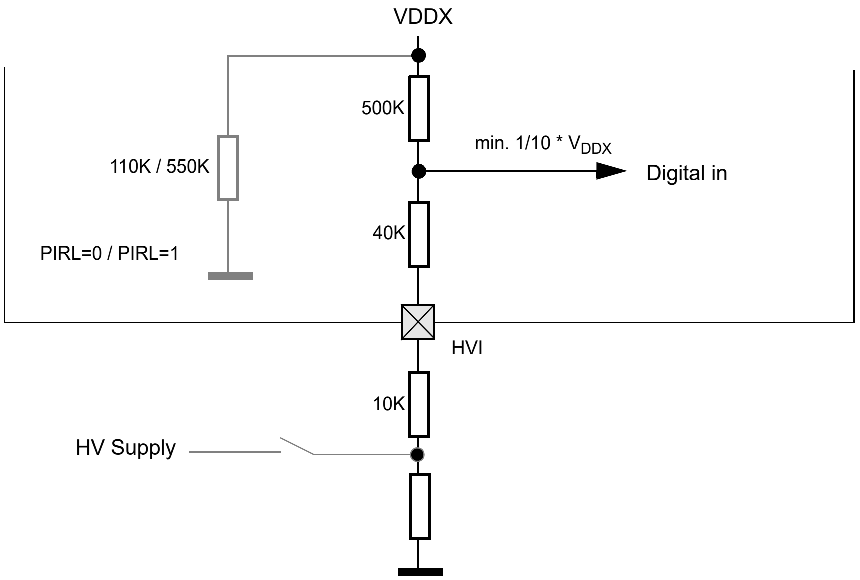


Figure 2-36. Digital Input Read with Pullup Enabled

External pullup device (Figure 2-37):

1. Enable analog function on HVI in non-direct mode (PTAENL=1, PTADIRL=0)
2. Select internal pulldown device on HVI (PTPSL=0)
3. Enable function to force input buffer active on HVI in analog mode (PTTEL=1)
4. Verify PTIL=1 for a connected external pullup device; read PTIL=0 for an open input

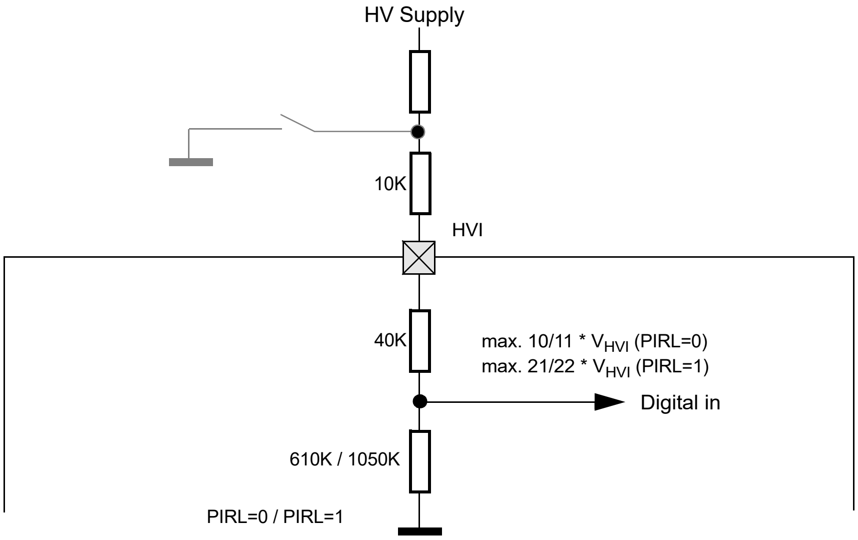


Figure 2-37. Digital Input Read with Pulldown Enabled

Chapter 3
Background Debug Controller (S12ZBDCV2)

Table 3-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V2.04	03.Dec.2012	Section 3.1.3.3, “Low-Power Modes	Included BACKGROUND/ Stop mode dependency
V2.05	22.Jan.2013	Section 3.3.2.2, “BDC Control Status Register Low (BDCCSRL)	Improved NORESP description and added STEP1/ Wait mode dependency
V2.06	22.Mar.2013	Section 3.3.2.2, “BDC Control Status Register Low (BDCCSRL)	Improved NORESP description of STEP1/ Wait mode dependency
V2.07	11.Apr.2013	Section 3.1.3.3.1, “Stop Mode	Improved STOP and BACKGROUND interdepency description
V2.08	31.May.2013	Section 3.4.4.4, “BACKGROUND Section 3.4.7.1, “Long-ACK Hardware Handshake Protocol	Removed misleading WAIT and BACKGROUND interdepency description Added subsection dedicated to Long-ACK
V2.09	29.Aug.2013	Section 3.4.4.12, “READ_DBGTB	Noted that READ_DBGTB is only available for devices featuring a trace buffer.
V2.10	21.Oct.2013	Section 3.1.3.3.2, “Wait Mode	Improved description of NORESP dependence on WAIT and BACKROUND

3.1 Introduction

The background debug controller (BDC) is a single-wire, background debug system implemented in on-chip hardware for minimal CPU intervention. The device BKGD pin interfaces directly to the BDC.

The S12ZBDC maintains the standard S12 serial interface protocol but introduces an enhanced handshake protocol and enhanced BDC command set to support the linear instruction set family of S12Z devices and offer easier, more flexible internal resource access over the BDC serial interface.

3.1.1 Glossary

Table 3-2. Glossary Of Terms

Term	Definition
DBG	On chip Debug Module
BDM	Active Background Debug Mode
CPU	S12Z CPU
SSC	Special Single Chip Mode (device operating mode
NSC	Normal Single Chip Mode (device operating mode)
BDCSI	Background Debug Controller Serial Interface. This refers to the single pin BKGD serial interface.
EWAIT	Optional S12 feature which allows external devices to delay external accesses until deassertion of EWAIT

3.1.2 Features

The BDC includes these distinctive features:

- Single-wire communication with host development system
- SYNC command to determine communication rate
- Genuine non-intrusive handshake protocol
- Enhanced handshake protocol for error detection and stop mode recognition
- Active out of reset in special single chip mode
- Most commands not requiring active BDM, for minimal CPU intervention
- Full global memory map access without paging
- Simple flash mass erase capability

3.1.3 Modes of Operation

S12 devices feature power modes (run, wait, and stop) and operating modes (normal single chip, special single chip). Furthermore, the operation of the BDC is dependent on the device security status.

3.1.3.1 BDC Modes

The BDC features module specific modes, namely disabled, enabled and active. These modes are dependent on the device security and operating mode. In active BDM the CPU ceases execution, to allow BDC system access to all internal resources including CPU internal registers.

3.1.3.2 Security and Operating mode Dependency

In device run mode the BDC dependency is as follows

- Normal modes, unsecure device

  General BDC operation available. The BDC is disabled out of reset.

- Normal modes, secure device
  BDC disabled. No BDC access possible.
- Special single chip mode, unsecure
  BDM active out of reset. All BDC commands are available.
- Special single chip mode, secure
  BDM active out of reset. Restricted command set available.

When operating in secure mode, BDC operation is restricted to allow checking and clearing security by mass erasing the on-chip flash memory. Secure operation prevents BDC access to on-chip memory other than mass erase. The BDC command set is restricted to those commands classified as Always-available.

3.1.3.3 Low-Power Modes

3.1.3.3.1 Stop Mode

The execution of the CPU STOP instruction leads to stop mode only when all bus masters (CPU, or others, depending on the device) have finished processing. The operation during stop mode depends on the ENBDC and BDCCIS bit settings as summarized in Table 3-3

Table 3-3. BDC STOP Operation Dependencies

ENBDC	BDCCIS	Description Of Operation
0	0	BDC has no effect on STOP mode.
0	1	BDC has no effect on STOP mode.
1	0	Only BDCSI clock continues
1	1	All clocks continue

A disabled BDC has no influence on stop mode operation. In this case the BDCSI clock is disabled in stop mode thus it is not possible to enable the BDC from within stop mode.

STOP Mode With BDC Enabled And BDCCIS Clear

If the BDC is enabled and BDCCIS is clear, then the BDC prevents the BDCCLK clock (Figure 3-5) from being disabled in stop mode. This allows BDC communication to continue throughout stop mode in order to access the BDCCSR register. All other device level clock signals are disabled on entering stop mode.

NOTE

This is intended for application debugging, not for fast flash programming. Thus the CLKSW bit must be clear to map the BDCSI to BDCCLK.

With the BDC enabled, an internal acknowledge delays stop mode entry and exit by 2 BDCSI clock + 2 bus clock cycles. If no other module delays stop mode entry and exit, then these additional clock cycles represent a difference between the debug and not debug cases. Furthermore if a BDC internal access is being executed when the device is entering stop mode, then the stop mode entry is delayed until the internal access is complete (typically for 1 bus clock cycle).

Accesses to the internal memory map are not possible when the internal device clocks are disabled. Thus attempted accesses to memory mapped resources are suppressed and the NORESP flag is set. Resources can be accessed again by the next command received following exit from Stop mode.

A BACKGROUND command issued whilst in stop mode remains pending internally until the device leaves stop mode. This means that subsequent active BDM commands, issued whilst BACKGROUND is pending, set the ILLCMD flag because the device is not yet in active BDM.

If ACK handshaking is enabled, then the first ACK, following a stop mode entry is long to indicate a stop exception. The BDC indicates a stop mode occurrence by setting the BDCCSR bit STOP. If the host attempts further communication before the ACK pulse generation then the OVRUN bit is set.

STOP Mode With BDC Enabled And BDCCIS Set

If the BDC is enabled and BDCCIS is set, then the BDC prevents core clocks being disabled in stop mode. This allows BDC communication, for access of internal memory mapped resources, but not CPU registers, to continue throughout stop mode.

A BACKGROUND command issued whilst in stop mode remains pending internally until the device leaves stop mode. This means that subsequent active BDM commands, issued whilst BACKGROUND is pending, set the ILLCMD flag because the device is not yet in active BDM.

If ACK handshaking is enabled, then the first ACK, following a stop mode entry is long to indicate a stop exception. The BDC indicates a stop mode occurrence by setting the BDCCSR bit STOP. If the host attempts further communication before the ACK pulse generation then the OVRUN bit is set.

3.1.3.3.2 Wait Mode

The device enters wait mode when the CPU starts to execute the WAI instruction. The second part of the WAI instruction (return from wait mode) can only be performed when an interrupt occurs. Thus on entering wait mode the CPU is in the middle of the WAI instruction and cannot permit access to CPU internal resources, nor allow entry to active BDM. Thus only commands classified as Non-Intrusive or Always-Available are possible in wait mode.

On entering wait mode, the WAIT flag in BDCCSR is set. If the ACK handshake protocol is enabled then the first ACK generated after WAIT has been set is a long-ACK pulse. Thus the host can recognize a wait mode occurrence. The WAIT flag remains set and cannot be cleared whilst the device remains in wait mode. After the device leaves wait mode the WAIT flag can be cleared by writing a “1” to it.

A BACKGROUND command issued whilst in wait mode sets the NORESP bit and the BDM active request remains pending internally until the CPU leaves wait mode due to an interrupt. The device then enters BDM with the PC pointing to the address of the first instruction of the ISR.

With ACK disabled, further Non-Intrusive or Always-Available commands are possible, in this pending state, but attempted Active-Background commands set NORESP and ILLCMD because the BDC is not in active BDM state.

With ACK enabled, if the host attempts further communication before the ACK pulse generation then the OVRUN bit is set.

Similarly the STEP1 command issued from a WAI instruction cannot be completed by the CPU until the CPU leaves wait mode due to an interrupt. The first STEP1 into wait mode sets the BDCCSR WAIT bit.

If the part is still in Wait mode and a further STEP1 is carried out then the NORESP and ILLCMD bits are set because the device is no longer in active BDM for the duration of WAI execution.

3.1.4 Block Diagram

A block diagram of the BDC is shown in Figure 3-1.

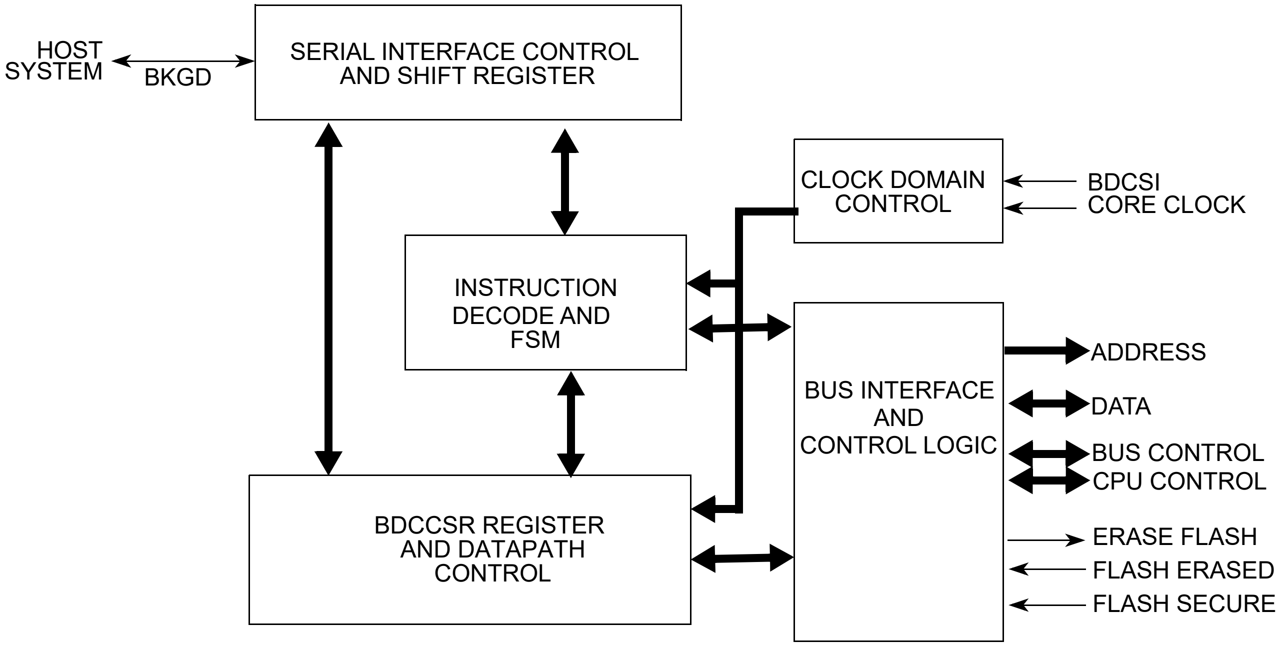


Figure 3-1. BDC Block Diagram

3.2 External Signal Description

A single-wire interface pin (BKGD) is used to communicate with the BDC system. During reset, this pin is a device mode select input. After reset, this pin becomes the dedicated serial interface pin for the BDC.

BKGD is a pseudo-open-drain pin with an on-chip pull-up. Unlike typical open-drain pins, the external RC time constant on this pin due to external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speed-up pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 3.4.6, “BDC Serial Interface” for more details.

3.3 Memory Map and Register Definition

3.3.1 Module Memory Map

Table 3-4 shows the BDC memory map.

Table 3-4. BDC Memory Map

Global Address	Module	Size (Bytes)
Not Applicable	BDC registers	2

3.3.2 Register Descriptions

The BDC registers are shown in Figure 3-2. Registers are accessed only by host-driven communications to the BDC hardware using READ_BDCCSR and WRITE_BDCCSR commands. They are not accessible in the device memory map.

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
Not Applicable	BDCCSRH	R	ENBDC	BDMACT	BDCCIS	0	STEAL	CLKSW	UNSEC	ERASE
		W		▨		▨			▨	▨
Not Applicable	BDCCSRL	R	WAIT	STOP	RAMWF	OVRUN	NORESP	RDINV	ILLACC	ILLCMD
		W								

▨ = Unimplemented, Reserved

0 = Always read zero

Figure 3-2. BDC Register Summary

3.3.2.1 BDC Control Status Register High (BDCCSRH)

Register Address: This register is not in the device memory map. It is accessible using BDC inherent addressing commands

	7	6	5	4	3	2	1	0
R	ENBDC	BDMACT	BDCCIS	0	STEAL	CLKSW	UNSEC	ERASE
W		▨		▨			▨	▨
Reset								
Secure AND SSC-Mode	1	1	0	0	0	0	0	0
Unsecure AND SSC-Mode	1	1	0	0	0	0	1	0
Secure AND NSC-Mode	0	0	0	0	0	0	0	0
Unsecure AND NSC-Mode	0	0	0	0	0	0	1	0

▨ = Unimplemented, Reserved

0 = Always read zero

Figure 3-3. BDC Control Status Register High (BDCCSRH)

Read: All modes through BDC operation only.

Write: All modes through BDC operation only, when not secured, but subject to the following:

- — Bits 7,3 and 2 can only be written by WRITE_BDCCSR commands.
- — Bit 5 can only be written by WRITE_BDCCSR commands when the device is not in stop mode.
- — Bits 6, 1 and 0 cannot be written. They can only be updated by internal hardware.

Table 3-5. BDCCSRH Field Descriptions

Field	Description
7 ENBDC	**Enable BDC** — This bit controls whether the BDC is enabled or disabled. When enabled, active BDM can be entered and non-intrusive commands can be carried out. When disabled, active BDM is not possible and the valid command set is restricted. Further information is provided in Table 3-7. 0 BDC disabled 1 BDC enabled **Note:** ENBDC is set out of reset in special single chip mode.
6 BDMACT	**BDM Active Status** — This bit becomes set upon entering active BDM. BDMACT is cleared as part of the active BDM exit sequence. 0 BDM not active 1 BDM active **Note:** BDMACT is set out of reset in special single chip mode.
5 BDCCIS	**BDC Continue In Stop** — If ENBDC is set then BDCCIS selects the type of BDC operation in stop mode (as shown in Table 3-3). If ENBDC is clear, then the BDC has no effect on stop mode and no BDC communication is possible.If ACK pulse handshaking is enabled, then the first ACK pulse following stop mode entry is a long ACK. This bit cannot be written when the device is in stop mode. 0 Only the BDCSI clock continues in stop mode 1 All clocks continue in stop mode
3 STEAL	**Steal enabled with ACK**— This bit forces immediate internal accesses with the ACK handshaking protocol enabled. If ACK handshaking is disabled then BDC accesses steal the next bus cycle. 0 If ACK is enabled then BDC accesses await a free cycle, with a timeout of 512 cycles 1 If ACK is enabled then BDC accesses are carried out in the next bus cycle
2 CLKSW	**Clock Switch** — The CLKSW bit controls the BDCSI clock source. This bit is initialized to “0” by each reset and can be written to “1”. Once it has been set, it can only be cleared by a reset. When setting CLKSW a minimum delay of 150 cycles at the initial clock speed must elapse before the next command can be sent. This guarantees that the start of the next BDC command uses the new clock for timing subsequent BDC communications. 0 BDCCLK used as BDCSI clock source 1 Device fast clock used as BDCSI clock source **Note:** Refer to the device specification to determine which clock connects to the BDCCLK and fast clock inputs.
1 UNSEC	**Unsecure** — If the device is unsecure, the UNSEC bit is set automatically. 0 Device is secure. 1 Device is unsecure. **Note:** When UNSEC is set, the device is unsecure and the state of the secure bits in the on-chip Flash EEPROM can be changed.
0 ERASE	**Erase Flash** — This bit can only be set by the dedicated ERASE_FLASH command. ERASE is unaffected by write accesses to BDCCSR. ERASE is cleared either when the mass erase sequence is completed, independent of the actual status of the flash array or by a soft reset. Reading this bit indicates the status of the requested mass erase sequence. 0 No flash mass erase sequence pending completion 1 Flash mass erase sequence pending completion.

3.3.2.2 BDC Control Status Register Low (BDCCSRL)

Register Address: This register is not in the device memory map. It is accessible using BDC inherent addressing commands

	7	6	5	4	3	2	1	0
R W	WAIT	STOP	RAMWF	OVRUN	NORESP	RDINV	ILLACC	ILLCMD
Reset	0	0	0	0	0	0	0	0

Figure 3-4. BDC Control Status Register Low (BDCCSRL)

Read: BDC access only.

Write: Bits [7:5], [3:0] BDC access only, restricted to flag clearing by writing a “1” to the bit position.
Write: Bit 4 never. It can only be cleared by a SYNC pulse.

If ACK handshaking is enabled then BDC commands with ACK causing a BDCCSRL[3:1] flag setting condition also generate a long ACK pulse. Subsequent commands that are executed correctly generate a normal ACK pulse. Subsequent commands that are not correctly executed generate a long ACK pulse. The first ACK pulse after WAIT or STOP have been set also generates a long ACK. Subsequent ACK pulses are normal, whilst STOP and WAIT remain set.

Long ACK pulses are not immediately generated if an overrun condition is caused by the host driving the BKGD pin low whilst a target ACK is pending, because this would conflict with an attempted host transmission following the BKGD edge. When a whole byte has been received following the offending BKGD edge, the OVRUN bit is still set, forcing subsequent ACK pulses to be long.

Unimplemented BDC opcodes causing the ILLCMD bit to be set do not generate a long ACK because this could conflict with further transmission from the host. If the ILLCMD is set for another reason, then a long ACK is generated for the current command if it is a BDC command with ACK.

Table 3-6. BDCCSRL Field Descriptions

Field	Description
7 WAIT	**WAIT Indicator Flag** — Indicates that the device entered wait mode. Writing a “1” to this bit whilst in wait mode has no effect. Writing a “1” after exiting wait mode, clears the bit. 0 Device did not enter wait mode 1 Device entered wait mode.
6 STOP	**STOP Indicator Flag** — Indicates that the CPU requested stop mode following a STOP instruction. Writing a “1” to this bit whilst not in stop mode clears the bit. Writing a “1” to this bit whilst in stop mode has no effect. This bit can only be set when the BDC is enabled. 0 Device did not enter stop mode 1 Device entered stop mode.
5 RAMWF	**RAM Write Fault** — Indicates an ECC double fault during a BDC write access to RAM. Writing a “1” to this bit, clears the bit. 0 No RAM write double fault detected. 1 RAM write double fault detected.

Table 3-6. BDCCSRL Field Descriptions (continued)

Field	Description
4 OVRUN	**Overrun Flag** — Indicates unexpected host activity before command completion. This occurs if a new command is received before the current command completion. With ACK enabled this also occurs if the host drives the BKGD pin low whilst a target ACK pulse is pending To protect internal resources from misinterpreted BDC accesses following an overrun, internal accesses are suppressed until a SYNC clears this bit. A SYNC clears the bit. 0 No overrun detected. 1 Overrun detected when issuing a BDC command.
3 NORESP	**No Response Flag** — Indicates that the BDC internal action or data access did not complete. This occurs in the following scenarios: a) If no free cycle for an access is found within 512 core clock cycles. This could typically happen if a code loop without free cycles is executing with ACK enabled and STEAL clear. b) With ACK disabled or STEAL set, when an internal access is not complete before the host starts data/BDCCSRL retrieval or an internal write access is not complete before the host starts the next BDC command. c) Attempted internal memory or SYNC_PC accesses during STOP mode set NORESP if BDCCIS is clear. In the above cases, on setting NORESP, the BDC aborts the access if permitted. (For devices supporting EWAIT, BDC external accesses with EWAIT assertions, prevent a command from being aborted until EWAIT is deasserted). d) If a BACKGROUND command is issued whilst the device is in wait mode the NORESP bit is set but the command is not aborted. The active BDM request is completed when the device leaves wait mode. Furthermore subsequent CPU register access commands during wait mode set the NORESP bit, should it have been cleared. e) If a command is issued whilst awaiting return from Wait mode. This can happen when using STEP1 to step over a CPU WAI instruction, if the CPU has not returned from Wait mode before the next BDC command is received. f) If STEP1 is issued with the BDC enabled as the device enters Wait mode regardless of the BDMACT state. When NORESP is set a value of 0xEE is returned for each data byte associated with the current access. Writing a “1” to this bit, clears the bit. 0 Internal action or data access completed. 1 Internal action or data access did not complete.
2 RDINV	**Read Data Invalid Flag** — Indicates invalid read data due to an ECC error during a BDC initiated read access. The access returns the actual data read from the location. Writing a “1” to this bit, clears the bit. 0 No invalid read data detected. 1 Invalid data returned during a BDC read access.
1 ILLACC	**Illegal Access Flag** — Indicates an attempted illegal access. This is set in the following cases: When the attempted access addresses unimplemented memory When the access attempts to write to the flash array When a CPU register access is attempted with an invalid CRN (Section 3.4.5.1, “BDC Access Of CPU Registers). Illegal accesses return a value of 0xEE for each data byte Writing a “1” to this bit, clears the bit. 0 No illegal access detected. 1 Illegal BDC access detected.

Table 3-6. BDCCSRL Field Descriptions (continued)

Field	Description
0 ILLCMD	**Illegal Command Flag** — Indicates an illegal BDC command. This bit is set in the following cases: When an unimplemented BDC command opcode is received. When a DUMP_MEM{_WS}, FILL_MEM{_WS} or READ_SAME{_WS} is attempted in an illegal sequence. When an active BDM command is received whilst BDM is not active When a non Always-available command is received whilst the BDC is disabled or a flash mass erase is ongoing. When a non Always-available command is received whilst the device is secure Read commands return a value of 0xEE for each data byte Writing a “1” to this bit, clears the bit. 0 No illegal command detected. 1 Illegal BDC command detected.

3.4 Functional Description

3.4.1 Security

If the device resets with the system secured, the device clears the BDCCSR UNSEC bit. In the secure state BDC access is restricted to the BDCCSR register. A mass erase can be requested using the ERASE_FLASH command. If the mass erase is completed successfully, the device programs the security bits to the unsecure state and sets the BDC UNSEC bit. If the mass erase is unsuccessful, the device remains secure and the UNSEC bit is not set.

For more information regarding security, please refer to device specific security information.

3.4.2 Enabling BDC And Entering Active BDM

BDM can be activated only after being enabled. BDC is enabled by setting the ENBDC bit in the BDCCSR register, via the single-wire interface, using the command WRITE_BDCCSR.

After being enabled, BDM is activated by one of the following[1]:

- The BDC BACKGROUND command
- A CPU BGND instruction
- The DBG Breakpoint mechanism

Alternatively BDM can be activated directly from reset when resetting into Special Single Chip Mode.

The BDC is ready for receiving the first command 10 core clock cycles after the deassertion of the internal reset signal. This is delayed relative to the external pin reset as specified in the device reset documentation. On S12Z devices an NVM initialization phase follows reset. During this phase the BDC commands classified as always available are carried out immediately, whereas other BDC commands are subject to delayed response due to the NVM initialization phase.

NOTE

> After resetting into SSC mode, the initial PC address must be supplied by the host using the WRITE_Rn command before issuing the GO command.

---

[1] BDM active immediately out of special single-chip reset.

When BDM is activated, the CPU finishes executing the current instruction. Thereafter only BDC commands can affect CPU register contents until the BDC GO command returns from active BDM to user code or a device reset occurs. When BDM is activated by a breakpoint, the type of breakpoint used determines if BDM becomes active before or after execution of the next instruction.

NOTE

Attempting to activate BDM using a BGND instruction whilst the BDC is disabled, the CPU requires clock cycles for the attempted BGND execution. However BACKGROUND commands issued whilst the BDC is disabled are ignored by the BDC and the CPU execution is not delayed.

3.4.3 Clock Source

The BDC clock source can be mapped to a constant frequency clock source or a PLL based fast clock. The clock source for the BDC is selected by the CLKSW bit as shown in Figure 3-5. The BDC internal clock is named BDCSI clock. If BDCSI clock is mapped to the BDCCLK by CLKSW then the serial interface communication is not affected by bus/core clock frequency changes. If the BDC is mapped to BDCFCLK then the clock is connected to a PLL derived source at device level (typically bus clock), thus can be subject to frequency changes in application. Debugging through frequency changes requires SYNC pulses to re-synchronize. The sources of BDCCLK and BDCFCLK are specified at device level.

BDC accesses of internal device resources always use the device core clock. Thus if the ACK handshake protocol is not enabled, the clock frequency relationship must be taken into account by the host.

When changing the clock source via the CLKSW bit a minimum delay of 150 cycles at the initial clock speed must elapse before a SYNC can be sent. This guarantees that the start of the next BDC command uses the new clock for timing subsequent BDC communications.

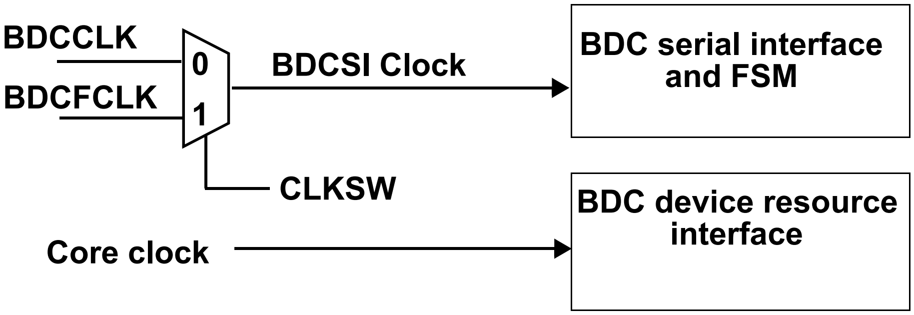


Figure 3-5. Clock Switch

3.4.4 BDC Commands

BDC commands can be classified into three types as shown in Table 3-7.

Table 3-7. BDC Command Types

Command Type	Secure Status	BDC Status	CPU Status	Command Set
Always-available	Secure or Unsecure	Enabled or Disabled	—	• Read/write access to BDCCSR • Mass erase flash memory using ERASE_FLASH • SYNC • ACK enable/disable
Non-intrusive	Unsecure	Enabled	Code execution allowed	• Read/write access to BDCCSR • Memory access • Memory access with status • Mass erase flash memory using ERASE_FLASH • Debug register access • BACKGROUND • SYNC • ACK enable/disable
Active background	Unsecure	Active	Code execution halted	• Read/write access to BDCCSR • Memory access • Memory access with status • Mass erase flash memory using ERASE_FLASH • Debug register access • Read or write CPU registers • Single-step the application • Exit active BDM to return to the application program (GO) • SYNC • ACK enable/disable

Non-intrusive commands are used to read and write target system memory locations and to enter active BDM. Target system memory includes all memory and registers within the global memory map, including external memory.

Active background commands are used to read and write all memory locations and CPU resources. Furthermore they allow single stepping through application code and to exit from active BDM.

Non-intrusive commands can only be executed when the BDC is enabled and the device unsecure. Active background commands can only be executed when the system is not secure and is in active BDM.

Non-intrusive commands do not require the system to be in active BDM for execution, although, they can still be executed in this mode. When executing a non-intrusive command with the ACK pulse handshake protocol disabled, the BDC steals the next bus cycle for the access. If an operation requires multiple cycles, then multiple cycles can be stolen. Thus if stolen cycles are not free cycles, the application code execution is delayed. The delay is negligible because the BDC serial transfer rate dictates that such accesses occur infrequently.

For data read commands, the external host must wait at least 16 BDCSI clock cycles after sending the address before attempting to obtain the read data. This is to be certain that valid data is available in the BDC shift register, ready to be shifted out. For write commands, the external host must wait 16 bdcsi cycles after sending the data to be written before attempting to send a new command. This is to avoid disturbing the BDC shift register before the write has been completed. The external host must wait at least for 16 bdcsi cycles after a control command before starting any new serial command.

If the ACK pulse handshake protocol is enabled and STEAL is cleared, then the BDC waits for the first free bus cycle to make a non-intrusive access. If no free bus cycle occurs within 512 core clock cycles then the BDC aborts the access, sets the NORESP bit and uses a long ACK pulse to indicate an error condition to the host.

Table 3-8 summarizes the BDC command set. The subsequent sections describe each command in detail and illustrate the command structure in a series of packets, each consisting of eight bit times starting with a falling edge. The bar across the top of the blocks indicates that the BKGD line idles in the high state. The time for an 8-bit command is 8 × 16 target BDCSI clock cycles.

The nomenclature below is used to describe the structure of the BDC commands. Commands begin with an 8-bit hexadecimal command code in the host-to-target direction (most significant bit first)

/ = separates parts of the command
d = delay 16 target BDCSI clock cycles (DLY)
dack = delay (16 cycles) no ACK; or delay (=> 32 cycles) then ACK.(DACK)
ad24 = 24-bit memory address in the host-to-target direction
rd8 = 8 bits of read data in the target-to-host direction
rd16 = 16 bits of read data in the target-to-host direction
rd24 = 24 bits of read data in the target-to-host direction
rd32 = 32 bits of read data in the target-to-host direction
rd64 = 64 bits of read data in the target-to-host direction
rd.sz = read data, size defined by sz, in the target-to-host direction
wd8 = 8 bits of write data in the host-to-target direction
wd16 = 16 bits of write data in the host-to-target direction
wd32 = 32 bits of write data in the host-to-target direction
wd.sz = write data, size defined by sz, in the host-to-target direction
ss = the contents of BDCCSRL in the target-to-host direction
sz = memory operand size (00 = byte, 01 = word, 10 = long)
(sz = 11 is reserved and currently defaults to long)
crn = core register number, 32-bit data width
WS = command suffix signaling the operation is with status

Table 3-8. BDC Command Summary

Command Mnemonic	Command Classification	ACK	Command Structure	Description
SYNC	Always Available	N/A	N/A[1]	Request a timed reference pulse to determine the target BDC communication speed
ACK_DISABLE	Always Available	No	0x03/d	Disable the communication handshake. This command does not issue an ACK pulse.
ACK_ENABLE	Always Available	Yes	0x02/dack	Enable the communication handshake. Issues an ACK pulse after the command is executed.
BACKGROUND	Non-Intrusive	Yes	0x04/dack	Halt the CPU if ENBDC is set. Otherwise, ignore as illegal command.

Table 3-8. BDC Command Summary (continued)

Command Mnemonic	Command Classification	ACK	Command Structure	Description
DUMP_MEM.sz	Non-Intrusive	Yes	(0x32+4 x sz)/dack/rd.sz	Dump (read) memory based on operand size (sz). Used with READ_MEM to dump large blocks of memory. An initial READ_MEM is executed to set up the starting address of the block and to retrieve the first result. Subsequent DUMP_MEM commands retrieve sequential operands.
DUMP_MEM.sz_WS	Non-Intrusive	No	(0x33+4 x sz)/d/ss/rd.sz	Dump (read) memory based on operand size (sz) and report status. Used with READ_MEM{_WS} to dump large blocks of memory. An initial READ_MEM{_WS} is executed to set up the starting address of the block and to retrieve the first result. Subsequent DUMP_MEM{_WS} commands retrieve sequential operands.
FILL_MEM.sz	Non-Intrusive	Yes	(0x12+4 x sz)/wd.sz/dack	Fill (write) memory based on operand size (sz). Used with WRITE_MEM to fill large blocks of memory. An initial WRITE_MEM is executed to set up the starting address of the block and to write the first operand. Subsequent FILL_MEM commands write sequential operands.
FILL_MEM.sz_WS	Non-Intrusive	No	(0x13+4 x sz)/wd.sz/d/ss	Fill (write) memory based on operand size (sz) and report status. Used with WRITE_MEM{_WS} to fill large blocks of memory. An initial WRITE_MEM{_WS} is executed to set up the starting address of the block and to write the first operand. Subsequent FILL_MEM{_WS} commands write sequential operands.
GO	Active Background	Yes	0x08/dack	Resume CPU user code execution
GO_UNTIL[2]	Active Background	Yes	0x0C/dack	Go to user program. ACK is driven upon returning to active background mode.
NOP	Non-Intrusive	Yes	0x00/dack	No operation
READ_Rn	Active Background	Yes	(0x60+CRN)/dack/rd32	Read the requested CPU register
READ_MEM.sz	Non-Intrusive	Yes	(0x30+4 x sz)/ad24/dack/rd.sz	Read the appropriately-sized (sz) memory value from the location specified by the 24-bit address
READ_MEM.sz_WS	Non-Intrusive	No	(0x31+4 x sz)/ad24/d/ss/rd.sz	Read the appropriately-sized (sz) memory value from the location specified by the 24-bit address and report status
READ_DBGTB	Non-Intrusive	Yes	(0x07)/dack/rd32/dack/rd32	Read 64-bits of DBG trace buffer

Table 3-8. BDC Command Summary (continued)

Command Mnemonic	Command Classification	ACK	Command Structure	Description
READ_SAME.sz	Non-Intrusive	Yes	(0x50+4 x sz)/dack/rd.sz	Read from location. An initial READ_MEM defines the address, subsequent READ_SAME reads return content of same address
READ_SAME.sz_WS	Non-Intrusive	No	(0x51+4 x sz)/d/ss/rd.sz	Read from location. An initial READ_MEM defines the address, subsequent READ_SAME reads return content of same address
READ_BDCCSR	Always Available	No	0x2D/rd16	Read the BDCCSR register
SYNC_PC	Non-Intrusive	Yes	0x01/dack/rd24	Read current PC
WRITE_MEM.sz	Non-Intrusive	Yes	(0x10+4 x sz)/ad24/wd.sz/dack	Write the appropriately-sized (sz) memory value to the location specified by the 24-bit address
WRITE_MEM.sz_WS	Non-Intrusive	No	(0x11+4 x sz)/ad24/wd.sz/d/ss	Write the appropriately-sized (sz) memory value to the location specified by the 24-bit address and report status
WRITE_Rn	Active Background	Yes	(0x40+CRN)/wd32/dack	Write the requested CPU register
WRITE_BDCCSR	Always Available	No	0x0D/wd16	Write the BDCCSR register
ERASE_FLASH	Always Available	No	0x95/d	Mass erase internal flash
STEP1 (TRACE1)	Active Background	Yes	0x09/dack	Execute one CPU command.

[1] The SYNC command is a special operation which does not have a command code.

[2] The GO_UNTIL command is identical to the GO command if ACK is not enabled.

3.4.4.1 SYNC

The SYNC command is unlike other BDC commands because the host does not necessarily know the correct speed to use for serial communications until after it has analyzed the response to the SYNC command.

To issue a SYNC command, the host:

1. Ensures that the BKGD pin is high for at least 4 cycles of the slowest possible BDCSI clock without reset asserted.
2. Drives the BKGD pin low for at least 128 cycles of the slowest possible BDCSI clock.
3. Drives BKGD high for a brief speed-up pulse to get a fast rise time. (This speedup pulse is typically one cycle of the host clock which is as fast as the maximum target BDCSI clock).
4. Removes all drive to the BKGD pin so it reverts to high impedance.
5. Listens to the BKGD pin for the sync response pulse.

Upon detecting the sync request from the host (which is a much longer low time than would ever occur during normal BDC communications), the target:

1. Discards any incomplete command
2. Waits for BKGD to return to a logic high.
3. Delays 16 cycles to allow the host to stop driving the high speed-up pulse.
4. Drives BKGD low for 128 BDCSI clock cycles.
5. Drives a 1-cycle high speed-up pulse to force a fast rise time on BKGD.
6. Removes all drive to the BKGD pin so it reverts to high impedance.
7. Clears the OVRRUN flag (if set).

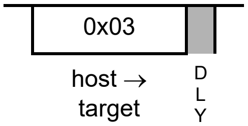
The host measures the low time of this 128-cycle SYNC response pulse and determines the correct speed for subsequent BDC communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the serial protocol can easily tolerate this speed error.

If the SYNC request is detected by the target, any partially executed command is discarded. This is referred to as a soft-reset, equivalent to a timeout in the serial communication. After the SYNC response, the target interprets the next negative edge (issued by the host) as the start of a new BDC command or the start of new SYNC request.

A SYNC command can also be used to abort a pending ACK pulse. This is explained in Section 3.4.8, “Hardware Handshake Abort Procedure.

3.4.4.2 ACK_DISABLE

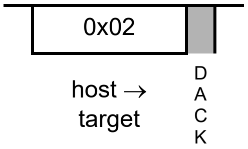
Disable host/target handshake protocol **Always Available**



Disables the serial communication handshake protocol. The subsequent commands, issued after the ACK_DISABLE command, do not execute the hardware handshake protocol. This command is not followed by an ACK pulse.

3.4.4.3 ACK_ENABLE

Enable host/target handshake protocol **Always Available**



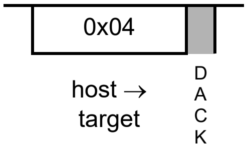
Enables the hardware handshake protocol in the serial communication. The hardware handshake is implemented by an acknowledge (ACK) pulse issued by the target MCU in response to a host command.

The ACK_ENABLE command is interpreted and executed in the BDC logic without the need to interface with the CPU. An ACK pulse is issued by the target device after this command is executed. This command can be used by the host to evaluate if the target supports the hardware handshake protocol. If the target supports the hardware handshake protocol, subsequent commands are enabled to execute the hardware handshake protocol, otherwise this command is ignored by the target. Table 3-8 indicates which commands support the ACK hardware handshake protocol.

For additional information about the hardware handshake protocol, refer to Section 3.4.7, “Serial Interface Hardware Handshake (ACK Pulse) Protocol,” and Section 3.4.8, “Hardware Handshake Abort Procedure.”

3.4.4.4 BACKGROUND

Enter active background mode (if enabled) **Non-intrusive**



Provided ENBDC is set, the BACKGROUND command causes the target MCU to enter active BDM as soon as the current CPU instruction finishes. If ENBDC is cleared, the BACKGROUND command is ignored.

A delay of 16 BDCSI clock cycles is required after the BACKGROUND command to allow the target MCU to finish its current CPU instruction and enter active background mode before a new BDC command can be accepted.

The host debugger must set ENBDC before attempting to send the BACKGROUND command the first time. Normally the host sets ENBDC once at the beginning of a debug session or after a target system reset. During debugging, the host uses GO commands to move from active BDM to application program execution and uses the BACKGROUND command or DBG breakpoints to return to active BDM.

A BACKGROUND command issued during stop or wait modes cannot immediately force active BDM because the WAI instruction does not end until an interrupt occurs. For the detailed mode dependency description refer to Section 3.1.3.3, “Low-Power Modes.

The host can recognize this pending BDM request condition because both NORESP and WAIT are set, but BDMACT is clear. Whilst in wait mode, with the pending BDM request, non-intrusive BDC commands are allowed.

3.4.4.5 DUMP_MEM.sz, DUMP_MEM.sz_WS

DUMP_MEM.sz

Read memory specified by debug address register, then increment address **Non-intrusive**

DUMP_MEM.sz

0x32		Data[7-0]
host → target	DACK	target → host

0x36		Data[15-8]	Data[7-0]
host → target	DACK	target → host	target → host

0x3A		Data[31-24]	Data[23-16]	Data[15-8]	Data[7-0]
host → target	DACK	target → host	target → host	target → host	target → host

DUMP_MEM.sz_WS

Read memory specified by debug address register with status, then increment address **Non-intrusive**

0x33		BDCCSRL	Data[7-0]
host → target	DLY	target → host	target → host

0x37		BDCCSRL	Data[15-8]	Data[7-0]
host → target	DLY	target → host	target → host	target → host

0x3B		BDCCSRL	Data[31-24]	Data23-16]	Data[15-8]	Data[7-0]
host → target	DLY	target → host	target → host	target → host	target → host	target → host

DUMP_MEM{_WS} is used with the READ_MEM{_WS} command to access large blocks of memory. An initial READ_MEM{_WS} is executed to set-up the starting address of the block and to retrieve the first result. The DUMP_MEM{_WS} command retrieves subsequent operands. The initial address is incremented by the operand size (1, 2, or 4) and saved in a temporary register. Subsequent DUMP_MEM{_WS} commands use this address, perform the memory read, increment it by the current operand size, and store the updated address in the temporary register. If the with-status option is specified, the BDCCSRL status byte is returned before the read data. This status byte reflects the state after the memory read was performed. If enabled, an ACK pulse is driven before the data bytes are transmitted. The effect of the access size and alignment on the next address to be accessed is explained in more detail in Section 3.4.5.2, “BDC Access Of Device Memory Mapped Resources”.

NOTE

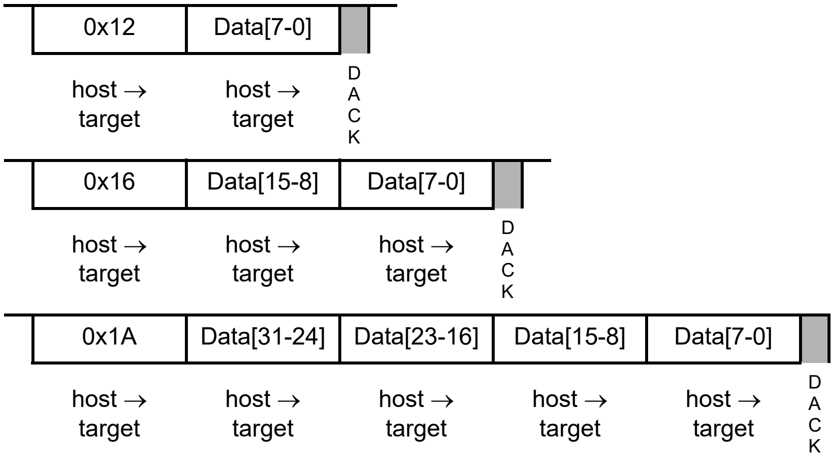
> DUMP_MEM{_WS} is a valid command only when preceded by SYNC, NOP, READ_MEM{_WS}, or another DUMP_MEM{_WS} command. Otherwise, an illegal command response is returned, setting the ILLCMD bit. NOP can be used for inter-command padding without corrupting the address pointer.

The size field (sz) is examined each time a DUMP_MEM{_WS} command is processed, allowing the operand size to be dynamically altered. The examples show the DUMP_MEM.B{_WS}, DUMP_MEM.W{_WS} and DUMP_MEM.L{_WS} commands.

3.4.4.6 FILL_MEM.sz, FILL_MEM.sz_WS

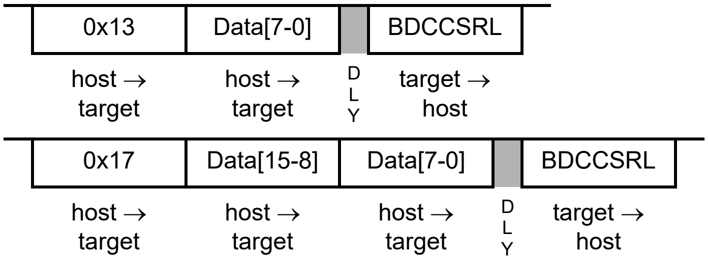
FILL_MEM.sz

Write memory specified by debug address register, then increment address **Non-intrusive**

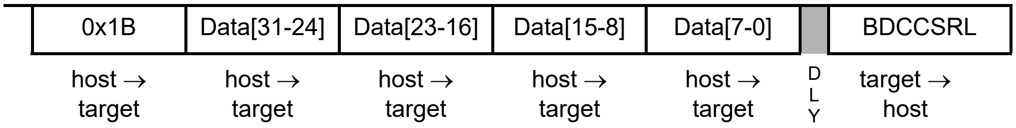


FILL_MEM.sz_WS

Write memory specified by debug address register with status, then increment address **Non-intrusive**



FILL_MEM.sz_WS



FILL_MEM{_WS} is used with the WRITE_MEM{_WS} command to access large blocks of memory. An initial WRITE_MEM{_WS} is executed to set up the starting address of the block and write the first datum. If an initial WRITE_MEM{_WS} is not executed before the first FILL_MEM{_WS}, an illegal command response is returned. The FILL_MEM{_WS} command stores subsequent operands. The initial address is incremented by the operand size (1, 2, or 4) and saved in a temporary register. Subsequent FILL_MEM{_WS} commands use this address, perform the memory write, increment it by the current operand size, and store the updated address in the temporary register. If the with-status option is specified, the BDCCSRL status byte is returned after the write data. This status byte reflects the state after the memory write was performed. If enabled an ACK pulse is generated after the internal write access has been completed or aborted. The effect of the access size and alignment on the next address to be accessed is explained in more detail in Section 3.4.5.2, "BDC Access Of Device Memory Mapped Resources"

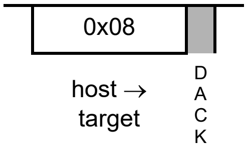
NOTE

FILL_MEM{_WS} is a valid command only when preceded by SYNC, NOP, WRITE_MEM{_WS}, or another FILL_MEM{_WS} command. Otherwise, an illegal command response is returned, setting the ILLCMD bit. NOP can be used for inter command padding without corrupting the address pointer.

The size field (sz) is examined each time a FILL_MEM{_WS} command is processed, allowing the operand size to be dynamically altered. The examples show the FILL_MEM.B{_WS}, FILL_MEM.W{_WS} and FILL_MEM.L{_WS} commands.

3.4.4.7 GO

Go **Non-intrusive**

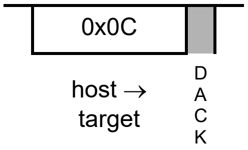


This command is used to exit active BDM and begin (or resume) execution of CPU application code. The CPU pipeline is flushed and refilled before normal instruction execution resumes. Prefetching begins at the current address in the PC. If any register (such as the PC) is altered by a BDC command whilst in BDM, the updated value is used when prefetching resumes. If enabled, an ACK is driven on exiting active BDM.

If a GO command is issued whilst the BDM is inactive, an illegal command response is returned and the ILLCMD bit is set.

3.4.4.8 GO_UNTIL

Go Until **Active Background**



This command is used to exit active BDM and begin (or resume) execution of application code. The CPU pipeline is flushed and refilled before normal instruction execution resumes. Prefetching begins at the current address in the PC. If any register (such as the PC) is altered by a BDC command whilst in BDM, the updated value is used when prefetching resumes.

After resuming application code execution, if ACK is enabled, the BDC awaits a return to active BDM before driving an ACK pulse. timeouts do not apply when awaiting a GO_UNTIL command ACK.

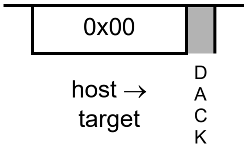
If a GO_UNTIL is not acknowledged then a SYNC command must be issued to end the pending GO_UNTIL.

If a GO_UNTIL command is issued whilst BDM is inactive, an illegal command response is returned and the ILLCMD bit is set.

If ACK handshaking is disabled, the GO_UNTIL command is identical to the GO command.

3.4.4.9 NOP

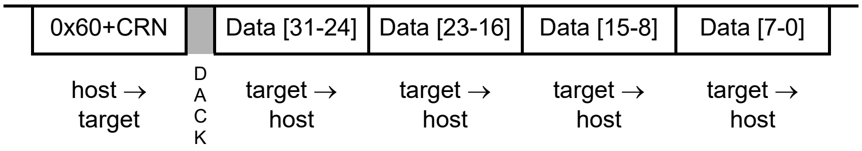
No operation **Active Background**



NOP performs no operation and may be used as a null command where required.

3.4.4.10 READ_Rn

Read CPU register **Active Background**



This command reads the selected CPU registers and returns the 32-bit result. Accesses to CPU registers are always 32-bits wide, regardless of implemented register width. Bytes that are not implemented return zero. The register is addressed through the CPU register number (CRN). See Section 3.4.5.1, “BDC

Access Of CPU Registers for the CRN address decoding. If enabled, an ACK pulse is driven before the data bytes are transmitted.

If the device is not in active BDM, this command is illegal, the ILLCMD bit is set and no access is performed.

3.4.4.11 READ_MEM.sz, READ_MEM.sz_WS

READ_MEM.sz

Read memory at the specified address **Non-intrusive**

0x30	Address[23-0]		Data[7-0]
host → target	host → target	DACK	target → host

0x34	Address[23-0]		Data[15-8]	Data[7-0]
host → target	host → target	DACK	target → host	target → host

0x38	Address[23-0]		Data[31-24]	Data[23-16]	Data[15-8]	Data[7-0]
host → target	host → target	DACK	target → host	target → host	target → host	target → host

READ_MEM.sz_WS

Read memory at the specified address with status **Non-intrusive**

0x31	Address[23-0]		BDCCSRL	Data[7-0]
host → target	host → target	DLY	target → host	target → host

0x35	Address[23-0]		BDCCSRL	Data [15-8]	Data [7-0]
host → target	host → target	DLY	target → host	target → host	target → host

0x39	Address[23-0]		BDCCSRL	Data[31-24]	Data[23-16]	Data [15-8]	Data [7-0]
host → target	host → target	DLY	target → host	target → host	target → host	target → host	target → host

Read data at the specified memory address. The address is transmitted as three 8-bit packets (msb to lsb) immediately after the command.

The hardware forces low-order address bits to zero longword accesses to ensure these accesses are on 0-modulo-size alignments. Byte alignment details are described in Section 3.4.5.2, "BDC Access Of Device Memory Mapped Resources". If the with-status option is specified, the BDCCSR status byte is returned before the read data. This status byte reflects the state after the memory read was performed. If enabled, an ACK pulse is driven before the data bytes are transmitted.

The examples show the READ_MEM.B{_WS}, READ_MEM.W{_WS} and READ_MEM.L{_WS} commands.

3.4.4.12 READ_DBGTB

Read DBG trace buffer **Non-intrusive**

0x07		TB Line [31-24]	TB Line [23-16]	TB Line [15-8]	TB Line [7-0]		TB Line [63-56]	TB Line [55-48]	TB Line [47-40]	TB Line [39-32]
host → target	DACK	target → host	target → host	target → host	target → host	DACK	target → host	target → host	target → host	target → host

This command is only available on devices, where the DBG module includes a trace buffer. Attempted use of this command on devices without a traace buffer return 0x00.

Read 64 bits from the DBG trace buffer. Refer to the DBG module description for more detailed information. If enabled an ACK pulse is generated before each 32-bit longword is ready to be read by the host. After issuing the first ACK a timeout is still possible whilst accessing the second 32-bit longword, since this requires separate internal accesses. The first 32-bit longword corresponds to trace buffer line bits[31:0]; the second to trace buffer line bits[63:32]. If ACK handshaking is disabled, the host must wait 16 clock cycles (DLY) after completing the first 32-bit read before starting the second 32-bit read.

3.4.4.13 READ_SAME.sz, READ_SAME.sz_WS

READ_SAME

Read same location specified by previous READ_MEM{_WS} **Non-intrusive**

0x54		Data[15-8]	Data[7-0]
host → target	DACK	target → host	target → host

READ_SAME_WS

Read same location specified by previous READ_MEM{_WS} **Non-intrusive**

0x55		BDCCSRL	Data [15-8]	Data [7-0]
host → target	DLY	target → host	target → host	target → host

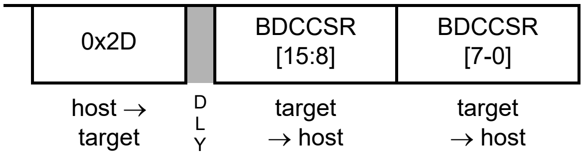
Read from location defined by the previous READ_MEM. The previous READ_MEM command defines the address, subsequent READ_SAME commands return contents of same address. The example shows the sequence for reading a 16-bit word size. Byte alignment details are described in Section 3.4.5.2, "BDC Access Of Device Memory Mapped Resources". If enabled, an ACK pulse is driven before the data bytes are transmitted.

NOTE

> READ_SAME{_WS} is a valid command only when preceded by SYNC, NOP, READ_MEM{_WS}, or another READ_SAME{_WS} command. Otherwise, an illegal command response is returned, setting the ILLCMD bit. NOP can be used for inter-command padding without corrupting the address pointer.

3.4.4.14 READ_BDCCSR

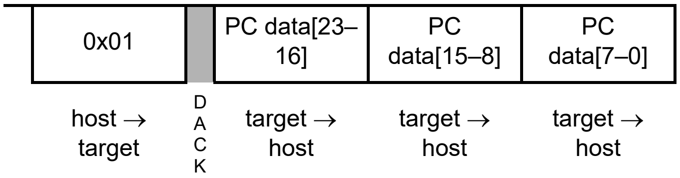
Read BDCCSR Status Register **Always Available**



Read the BDCCSR status register. This command can be executed in any mode.

3.4.4.15 SYNC_PC

Sample current PC **Non-intrusive**



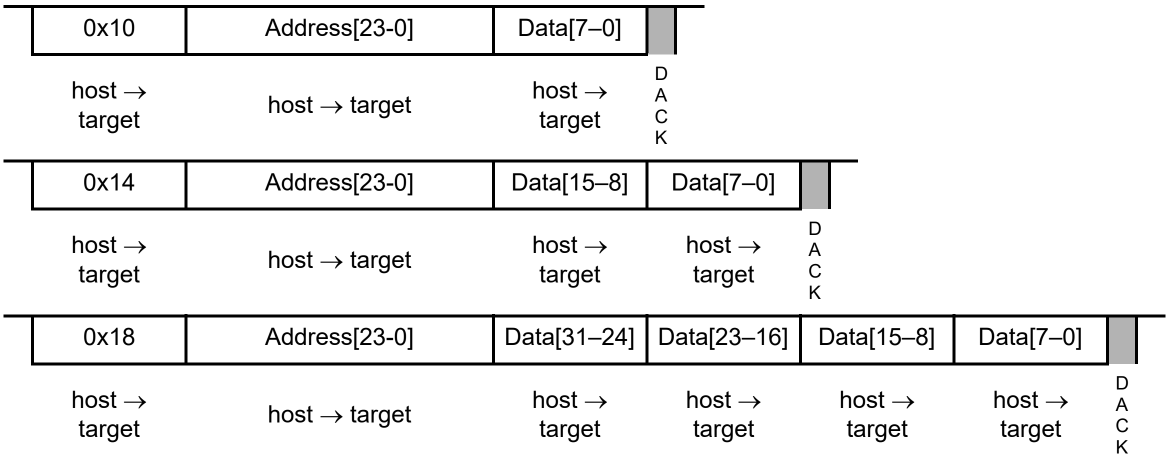
This command returns the 24-bit CPU PC value to the host. Unsuccessful SYNC_PC accesses return 0xEE for each byte. If enabled, an ACK pulse is driven before the data bytes are transmitted. The value of 0xEE is returned if a timeout occurs, whereby NORESP is set. This can occur if the CPU is executing the WAI instruction, or the STOP instruction with BDCCIS clear, or if a CPU access is delayed by EWAIT. If the CPU is executing the STOP instruction and BDCCIS is set, then SYNC_PC returns the PC address of the instruction following STOP in the code listing.

This command can be used to dynamically access the PC for performance monitoring as the execution of this command is considerably less intrusive to the real-time operation of an application than a BACKGROUND/read-PC/GO command sequence. Whilst the BDC is not in active BDM, SYNC_PC returns the PC address of the instruction currently being executed by the CPU. In active BDM, SYNC_PC returns the address of the next instruction to be executed on returning from active BDM. Thus following a write to the PC in active BDM, a SYNC_PC returns that written value.

3.4.4.16 WRITE_MEM.sz, WRITE_MEM.sz_WS

WRITE_MEM.sz

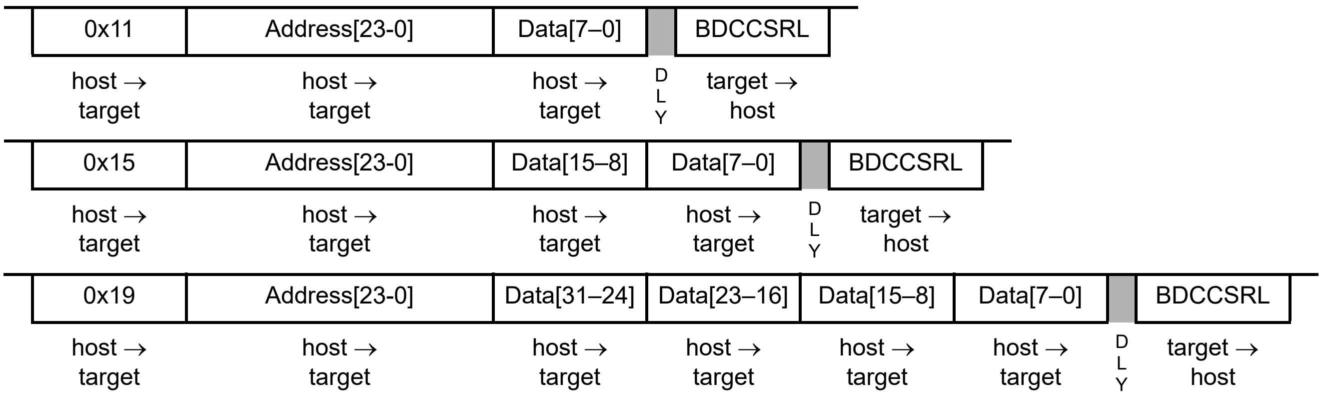
Write memory at the specified address **Non-intrusive**



0x10	Address[23-0]	Data[7–0]	▨				
host → target	host → target	host → target	DACK				
0x14	Address[23-0]	Data[15–8]	Data[7–0]	▨			
host → target	host → target	host → target	host → target	DACK			
0x18	Address[23-0]	Data[31–24]	Data[23–16]	Data[15–8]	Data[7–0]	▨	
host → target	host → target	host → target	host → target	host → target	host → target	DACK	

WRITE_MEM.sz_WS

Write memory at the specified address with status **Non-intrusive**



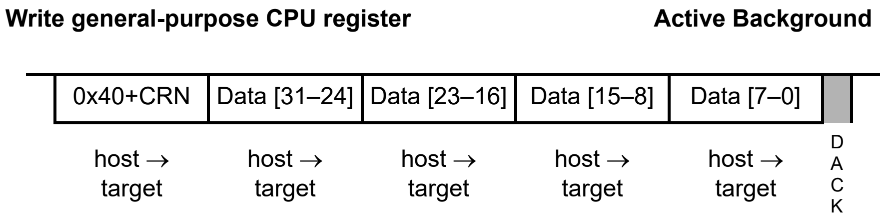
0x11	Address[23-0]	Data[7–0]	▨	BDCCSRL			
host → target	host → target	host → target	DLY	target → host			
0x15	Address[23-0]	Data[15–8]	Data[7–0]	▨	BDCCSRL		
host → target	host → target	host → target	host → target	DLY	target → host		
0x19	Address[23-0]	Data[31–24]	Data[23–16]	Data[15–8]	Data[7–0]	▨	BDCCSRL
host → target	host → target	host → target	host → target	host → target	host → target	DLY	target → host

Write data to the specified memory address. The address is transmitted as three 8-bit packets (msb to lsb) immediately after the command.

If the with-status option is specified, the status byte contained in BDCCSRL is returned after the write data. This status byte reflects the state after the memory write was performed. The examples show the WRITE_MEM.B{_WS}, WRITE_MEM.W{_WS}, and WRITE_MEM.L{_WS} commands. If enabled an ACK pulse is generated after the internal write access has been completed or aborted.

The hardware forces low-order address bits to zero longword accesses to ensure these accesses are on 0-modulo-size alignments. Byte alignment details are described in Section 3.4.5.2, “BDC Access Of Device Memory Mapped Resources”.

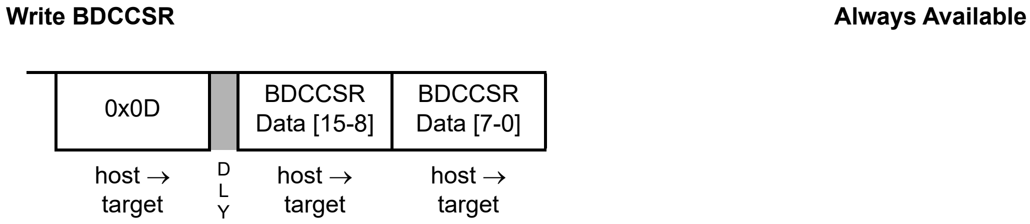
3.4.4.17 WRITE_Rn



If the device is in active BDM, this command writes the 32-bit operand to the selected CPU general-purpose register. See Section 3.4.5.1, “BDC Access Of CPU Registers for the CRN details. Accesses to CPU registers are always 32-bits wide, regardless of implemented register width. If enabled an ACK pulse is generated after the internal write access has been completed or aborted.

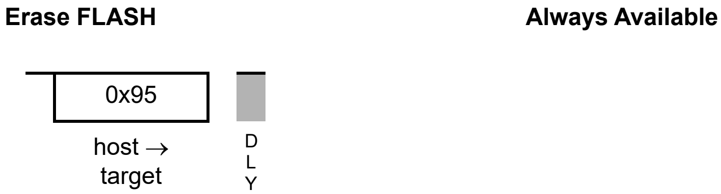
If the device is not in active BDM, this command is rejected as an illegal operation, the ILLCMD bit is set and no operation is performed.

3.4.4.18 WRITE_BDCCSR



16-bit write to the BDCCSR register. No ACK pulse is generated. Writing to this register can be used to configure control bits or clear flag bits. Refer to the register bit descriptions.

3.4.4.19 ERASE_FLASH



Mass erase the internal flash. This command can always be issued. On receiving this command twice in succession, the BDC sets the ERASE bit in BDCCSR and requests a flash mass erase. Any other BDC command following a single ERASE_FLASH initializes the sequence, such that thereafter the ERASE_FLASH must be applied twice in succession to request a mass erase. If 512 BDCSI clock cycles elapse between the consecutive ERASE_FLASH commands then a timeout occurs, which forces a soft reset and initializes the sequence. The ERASE bit is cleared when the mass erase sequence has been completed. No ACK is driven.

During the mass erase operation, which takes many clock cycles, the command status is indicated by the ERASE bit in BDCCSR. Whilst a mass erase operation is ongoing, Always-available commands can be issued. This allows the status of the erase operation to be polled by reading BDCCSR to determine when the operation is finished.

The status of the flash array can be verified by subsequently reading the flash error flags to determine if the erase completed successfully.

ERASE_FLASH can be aborted by a SYNC pulse forcing a soft reset.

NOTE: Device Bus Frequency Considerations

The ERASE_FLASH command requires the default device bus clock frequency after reset. Thus the bus clock frequency must not be changed following reset before issuing an ERASE_FLASH command.

3.4.4.20 STEP1

This command is used to step through application code. In active BDM this command executes the next CPU instruction in application code. If enabled an ACK is driven.

If a STEP1 command is issued and the CPU is not halted, the command is ignored.

Using STEP1 to step through a CPU WAI instruction is explained in Section 3.1.3.3.2, “Wait Mode.

3.4.5 BDC Access Of Internal Resources

Unsuccessful read accesses of internal resources return a value of 0xEE for each data byte. This enables a debugger to recognize a potential error, even if neither the ACK handshaking protocol nor a status command is currently being executed. The value of 0xEE is returned in the following cases.

- Illegal address access, whereby ILLACC is set
- Invalid READ_SAME or DUMP_MEM sequence
- Invalid READ_Rn command (BDM inactive or CRN incorrect)
- Internal resource read with timeout, whereby NORESP is set

3.4.5.1 BDC Access Of CPU Registers

The CRN field of the READ_Rn and WRITE_Rn commands contains a pointer to the CPU registers. The mapping of CRN to CPU registers is shown in Table 3-9. Accesses to CPU registers are always 32-bits wide, regardless of implemented register width. This means that the BDC data transmission for these commands is 32-bits long. The valid bits of the transfer are listed in the Valid Data Bits column. The other bits of the transmission are redundant.

Attempted accesses of CPU registers using a CRN of 0xD,0xE or 0xF is invalid, returning the value 0xEE for each byte and setting the ILLACC bit.

Table 3-9. CPU Register Number (CRN) Mapping

CPU Register	Valid Data Bits	Command	Opcode	Command	Opcode
D0	[7:0]	WRITE_D0	0x40	READ_D0	0x60
D1	[7:0]	WRITE_D1	0x41	READ_D1	0x61
D2	[15:0]	WRITE_D2	0x42	READ_D2	0x62
D3	[15:0]	WRITE_D3	0x43	READ_D3	0x63
D4	[15:0]	WRITE_D4	0x44	READ_D4	0x64
D5	[15:0]	WRITE_D5	0x45	READ_D5	0x65
D6	[31:0]	WRITE_D6	0x46	READ_D6	0x66
D7	[31:0]	WRITE_D7	0x47	READ_D7	0x67
X	[23:0]	WRITE_X	0x48	READ_X	0x68
Y	[23:0]	WRITE_Y	0x49	READ_Y	0x69
SP	[23:0]	WRITE_SP	0x4A	READ_SP	0x6A
PC	[23:0]	WRITE_PC	0x4B	READ_PC	0x6B
CCR	[15:0]	WRITE_CCR	0x4C	READ_CCR	0x6C

3.4.5.2 BDC Access Of Device Memory Mapped Resources

The device memory map is accessed using READ_MEM, DUMP_MEM, WRITE_MEM, FILL_MEM and READ_SAME, which support different access sizes, as explained in the command descriptions.

When an unimplemented command occurs during a DUMP_MEM, FILL_MEM or READ_SAME sequence, then that sequence is ended.

Illegal read accesses return a value of 0xEE for each byte. After an illegal access FILL_MEM and READ_SAME commands are not valid, and it is necessary to restart the internal access sequence with READ_MEM or WRITE_MEM. An illegal access does not break a DUMP_MEM sequence. After read accesses that cause the RDINV bit to be set, DUMP_MEM and READ_SAME commands are valid, it is not necessary to restart the access sequence with a READ_MEM.

The hardware forces low-order address bits to zero for longword accesses to ensure these accesses are realigned to 0-modulo-size alignments.

Word accesses map to 2-bytes from within a 4-byte field as shown in Table 3-10. Thus if address bits [1:0] are both logic “1” the access is realigned so that it does not straddle the 4-byte boundary but accesses data from within the addressed 4-byte field.

Table 3-10. Field Location to Byte Access Mapping

Address[1:0]	Access Size	00	01	10	11	Note
00	32-bit	Data[31:24]	Data[23:16]	Data [15:8]	Data [7:0]	
01	32-bit	Data[31:24]	Data[23:16]	Data [15:8]	Data [7:0]	Realigned
10	32-bit	Data[31:24]	Data[23:16]	Data [15:8]	Data [7:0]	Realigned
11	32-bit	Data[31:24]	Data[23:16]	Data [15:8]	Data [7:0]	Realigned
00	16-bit	Data [15:8]	Data [7:0]	▨	▨	▨
01	16-bit	▨	Data [15:8]	Data [7:0]	▨	▨
10	16-bit	▨	▨	Data [15:8]	Data [7:0]	
11	16-bit	▨	▨	Data [15:8]	Data [7:0]	Realigned
00	8-bit	Data [7:0]	▨	▨	▨	▨
01	8-bit	▨	Data [7:0]	▨	▨	▨
10	8-bit	▨	▨	Data [7:0]	▨	▨
11	8-bit	▨	▨	▨	Data [7:0]	
		▨	Denotes byte that is not transmitted			

3.4.5.2.1 FILL_MEM and DUMP_MEM Increments and Alignment

FILL_MEM and DUMP_MEM increment the previously accessed address by the previous access size to calculate the address of the current access. On misaligned longword accesses, the address bits [1:0] are forced to zero, therefore the following FILL_MEM or DUMP_MEM increment to the first address in the next 4-byte field. This is shown in Table 3-11, the address of the first DUMP_MEM.32 following READ_MEM.32 being calculated from 0x004000+4.

When misaligned word accesses are realigned, then the original address (not the realigned address) is incremented for the following FILL_MEM, DUMP_MEM command.

Misaligned word accesses can cause the same locations to be read twice as shown in rows 6 and 7. The hardware ensures alignment at an attempted misaligned word access across a 4-byte boundary, as shown in row 7. The following word access in row 8 continues from the realigned address of row 7.

Table 3-11. Consecutive Accesses With Variable Size

Row	Command	Address	Address[1:0]	00	01	10	11
1	READ_MEM.32	0x004003	11	Accessed	Accessed	Accessed	Accessed
2	DUMP_MEM.32	0x004004	00	Accessed	Accessed	Accessed	Accessed
3	DUMP_MEM.16	0x004008	00	Accessed	Accessed	▨	▨
4	DUMP_MEM.16	0x00400A	10	▨	▨	Accessed	Accessed
5	DUMP_MEM.08	0x00400C	00	Accessed	▨	▨	▨
6	DUMP_MEM.16	0x00400D	01	▨	Accessed	Accessed	▨
7	DUMP_MEM.16	0x00400E	10	▨	▨	Accessed	Accessed
8	DUMP_MEM.16	0x004010	01	Accessed	Accessed	▨	▨

3.4.5.2.2 READ_SAME Effects Of Variable Access Size

READ_SAME uses the unadjusted address given in the previous READ_MEM command as a base address for subsequent READ_SAME commands. When the READ_MEM and READ_SAME size parameters differ then READ_SAME uses the original base address buts aligns 32-bit and 16-bit accesses, where those accesses would otherwise cross the aligned 4-byte boundary. Table 3-12 shows some examples of this.

Table 3-12. Consecutive READ_SAME Accesses With Variable Size

Row	Command	Base Address	00	01	10	11
1	READ_MEM.32	0x004003	Accessed	Accessed	Accessed	Accessed
2	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed
3	READ_SAME.16	—			Accessed	Accessed
4	READ_SAME.08	—				Accessed
5	READ_MEM.08	0x004000	Accessed			
6	READ_SAME.08	—	Accessed			
7	READ_SAME.16	—	Accessed	Accessed		
8	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed
9	READ_MEM.08	0x004002			Accessed	
10	READ_SAME.08	—			Accessed	
11	READ_SAME.16	—			Accessed	Accessed
12	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed
13	READ_MEM.08	0x004003				Accessed
14	READ_SAME.08	—				Accessed
15	READ_SAME.16	—			Accessed	Accessed
16	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed
17	READ_MEM.16	0x004001		Accessed	Accessed	
18	READ_SAME.08	—		Accessed		
19	READ_SAME.16	—		Accessed	Accessed	
20	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed
21	READ_MEM.16	0x004003			Accessed	Accessed
22	READ_SAME.08	—				Accessed
23	READ_SAME.16	—			Accessed	Accessed
24	READ_SAME.32	—	Accessed	Accessed	Accessed	Accessed

3.4.6 BDC Serial Interface

The BDC communicates with external devices serially via the BKGD pin. During reset, this pin is a mode select input which selects between normal and special modes of operation. After reset, this pin becomes the dedicated serial interface pin for the BDC.

The BDC serial interface uses an internal clock source, selected by the CLKSW bit in the BDCCSR register. This clock is referred to as the target clock in the following explanation.

The BDC serial interface uses a clocking scheme in which the external host generates a falling edge on the BKGD pin to indicate the start of each bit time. This falling edge is sent for every bit whether data is transmitted or received. Data is transferred most significant bit (MSB) first at 16 target clock cycles per bit. The interface times out if during a command 512 clock cycles occur between falling edges from the host. The timeout forces the current command to be discarded.

The BKGD pin is a pseudo open-drain pin and has a weak on-chip active pull-up that is enabled at all times. It is assumed that there is an external pull-up and that drivers connected to BKGD do not typically drive the high level. Since R-C rise time could be unacceptably long, the target system and host provide brief drive-high (speedup) pulses to drive BKGD to a logic 1. The source of this speedup pulse is the host for transmit cases and the target for receive cases.

The timing for host-to-target is shown in Figure 3-6 and that of target-to-host in Figure 3-7 and Figure 3-8. All cases begin when the host drives the BKGD pin low to generate a falling edge. Since the host and target operate from separate clocks, it can take the target up to one full clock cycle to recognize this edge; this synchronization uncertainty is illustrated in Figure 3-6. The target measures delays from this perceived start of the bit time while the host measures delays from the point it actually drove BKGD low to start the bit up to one target clock cycle earlier. Synchronization between the host and target is established in this manner at the start of every bit time.

Figure 3-6 shows an external host transmitting a logic 1 and transmitting a logic 0 to the BKGD pin of a target system. The host is asynchronous to the target, so there is up to a one clock-cycle delay from the host-generated falling edge to where the target recognizes this edge as the beginning of the bit time. Ten target clock cycles later, the target senses the bit level on the BKGD pin. Internal glitch detect logic requires the pin be driven high no later than eight target clock cycles after the falling edge for a logic 1 transmission.

Since the host drives the high speedup pulses in these two cases, the rising edges look like digitally driven signals.

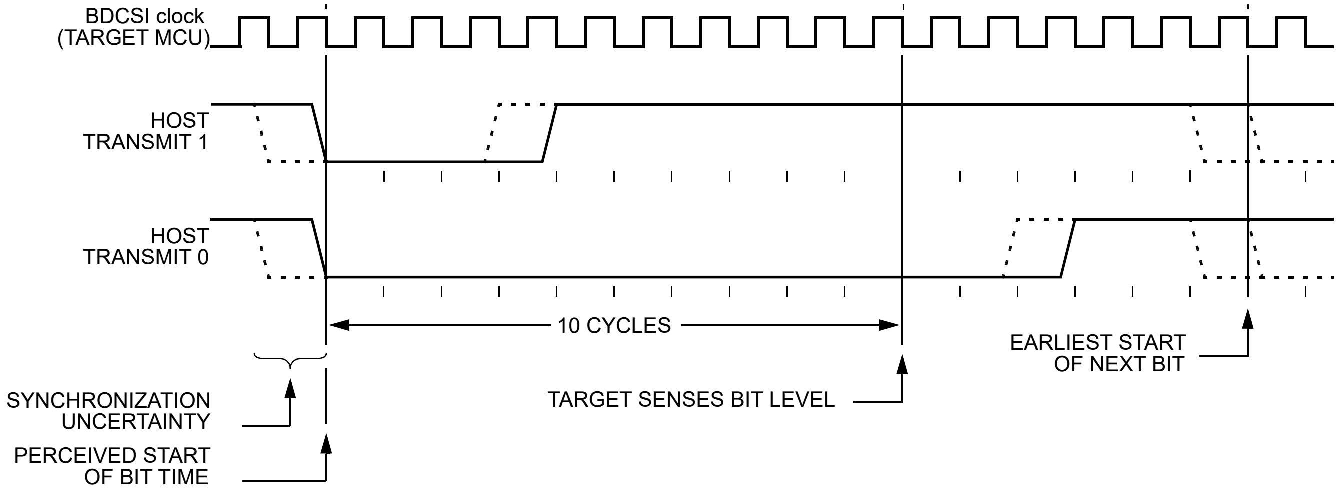


Figure 3-6. BDC Host-to-Target Serial Bit Timing

Figure 3-7 shows the host receiving a logic 1 from the target system. The host holds the BKGD pin low long enough for the target to recognize it (at least two target clock cycles). The host must release the low drive at the latest after 6 clock cycles, before the target drives a brief high speedup pulse seven target clock cycles after the perceived start of the bit time. The host should sample the bit level about 10 target clock cycles after it started the bit time.

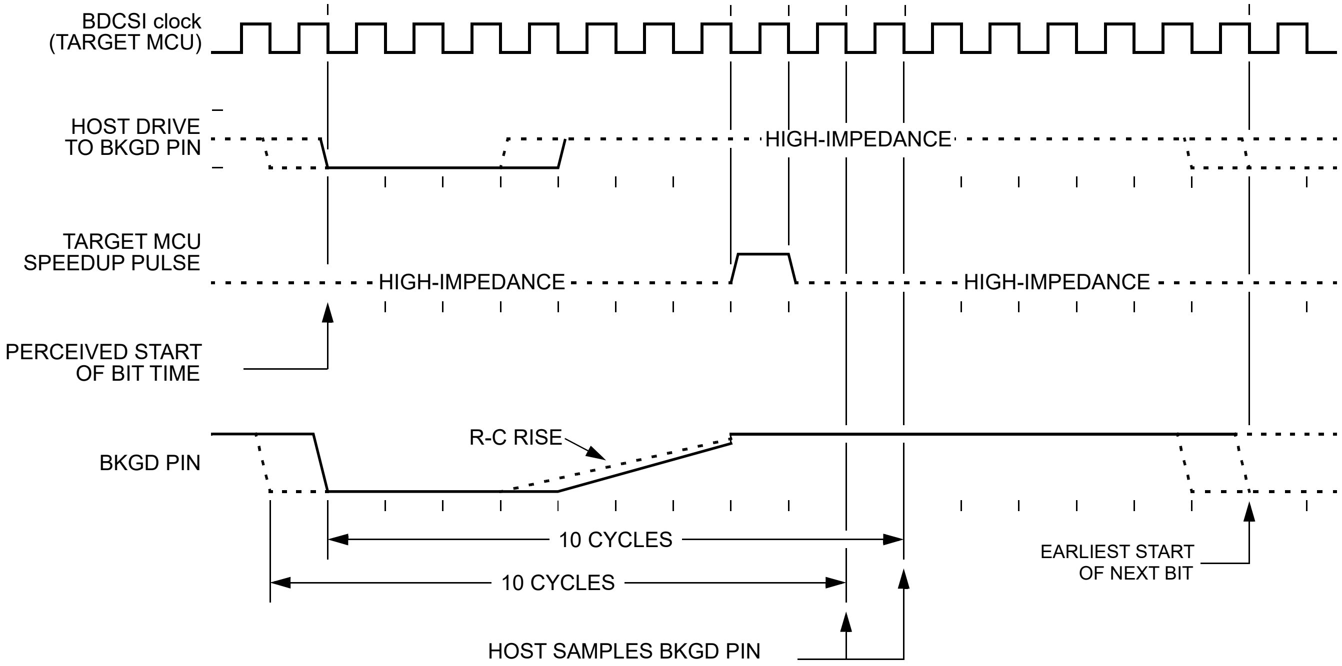


Figure 3-7. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 3-8 shows the host receiving a logic 0 from the target. The host initiates the bit time but the target finishes it. Since the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 target clock cycles then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 target clock cycles after starting the bit time.

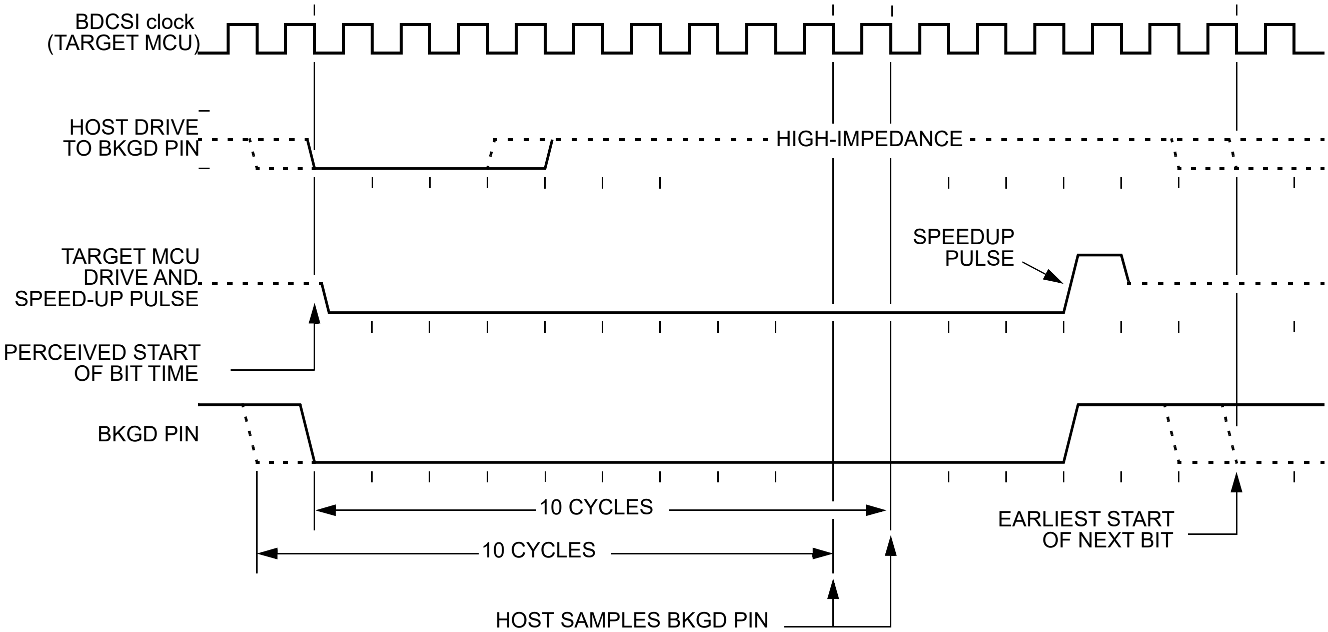


Figure 3-8. BDC Target-to-Host Serial Bit Timing (Logic 0)

3.4.7 Serial Interface Hardware Handshake (ACK Pulse) Protocol

BDC commands are processed internally at the device core clock rate. Since the BDCSI clock can be asynchronous relative to the bus frequency, a handshake protocol is provided so the host can determine when an issued command has been executed. This section describes the hardware handshake protocol.

The hardware handshake protocol signals to the host controller when a BDC command has been executed by the target. This protocol is implemented by a low pulse (16 BDCSI clock cycles) followed by a brief speedup pulse on the BKGD pin, generated by the target MCU when a command, issued by the host, has been successfully executed (see Figure 3-9). This pulse is referred to as the ACK pulse. After the ACK pulse has finished, the host can start the bit retrieval if the last issued command was a read command, or start a new command if the last command was a write command or a control command.

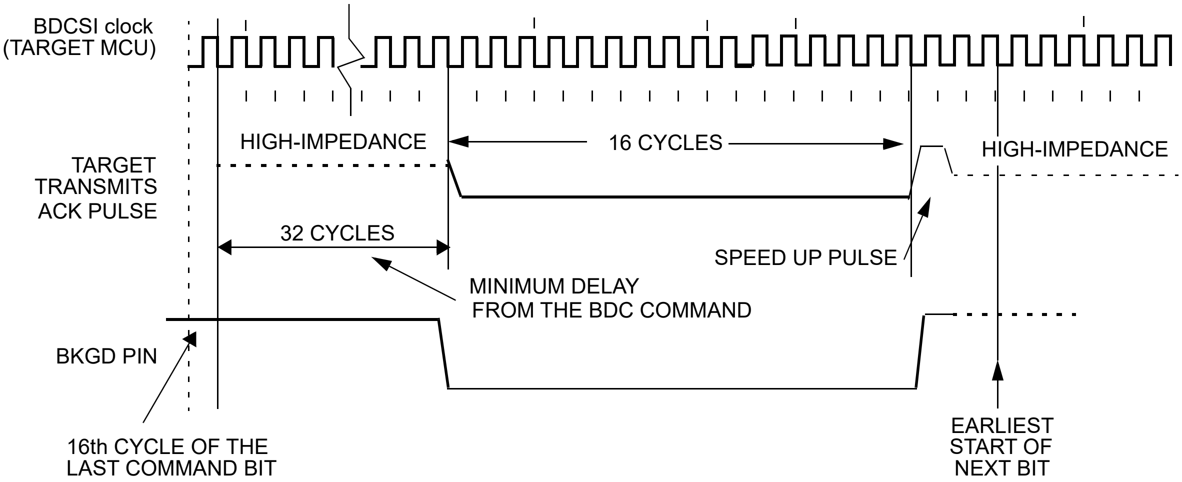


Figure 3-9. Target Acknowledge Pulse (ACK)

The handshake protocol is enabled by the ACK_ENABLE command. The BDC sends an ACK pulse when the ACK_ENABLE command has been completed. This feature can be used by the host to evaluate if the target supports the hardware handshake protocol. If an ACK pulse is issued in response to this command, the host knows that the target supports the hardware handshake protocol.

Unlike the normal bit transfer, where the host initiates the transmission by issuing a negative edge on the BKGD pin, the serial interface ACK handshake pulse is initiated by the target MCU by issuing a negative edge on the BKGD pin. Figure 3-9 specifies the timing when the BKGD pin is being driven. The host must follow this timing constraint in order to avoid the risk of an electrical conflict at the BKGD pin.

When the handshake protocol is enabled, the STEAL bit in BDCCSR selects if bus cycle stealing is used to gain immediate access. If STEAL is cleared, the BDC is configured for low priority bus access using free cycles, without stealing cycles. This guarantees that BDC accesses remain truly non-intrusive to not affect the system timing during debugging. If STEAL is set, the BDC gains immediate access, if necessary stealing an internal bus cycle.

NOTE

If bus steals are disabled then a loop with no free cycles cannot allow access. In this case the host must recognize repeated NORESP messages and then issue a BACKGROUND command to stop the target and access the data.

Figure 3-10 shows the ACK handshake protocol without steal in a command level timing diagram. The READ_MEM.B command is used as an example. First, the 8-bit command code is sent by the host, followed by the address of the memory location to be read. The target BDC decodes the command. Then an internal access is requested by the BDC. When a free bus cycle occurs the READ_MEM.B operation is carried out. If no free cycle occurs within 512 core clock cycles then the access is aborted, the NORESP flag is set and the target generates a Long-ACK pulse.

Having retrieved the data, the BDC issues an ACK pulse to the host controller, indicating that the addressed byte is ready to be retrieved. After detecting the ACK pulse, the host initiates the data read part of the command.

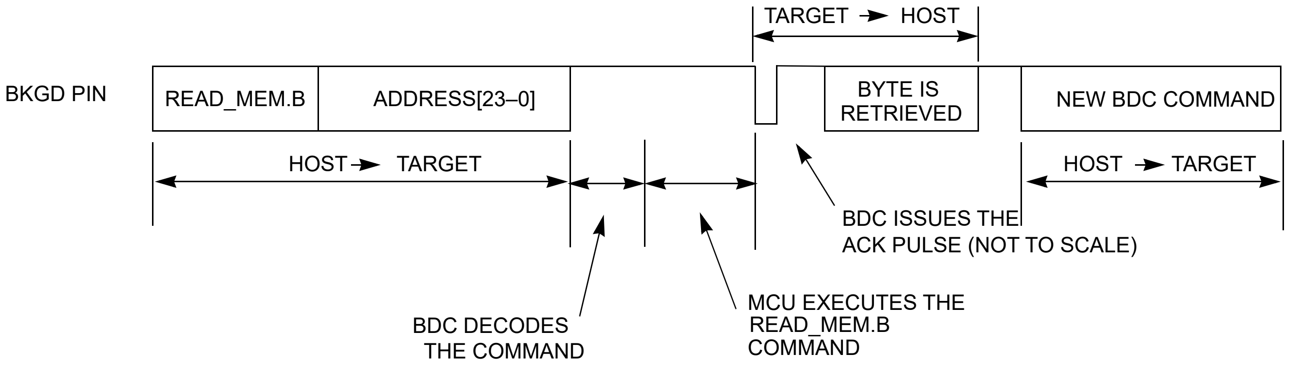


Figure 3-10. Handshake Protocol at Command Level

Alternatively, setting the STEAL bit configures the handshake protocol to make an immediate internal access, independent of free bus cycles.

The ACK handshake protocol does not support nested ACK pulses. If a BDC command is not acknowledged by an ACK pulse, the host needs to abort the pending command first in order to be able to issue a new BDC command. The host can decide to abort any possible pending ACK pulse in order to be sure a new command can be issued. Therefore, the protocol provides a mechanism in which a command, and its corresponding ACK, can be aborted.

Commands With-Status do not generate an ACK, thus if ACK is enabled and a With-Status command is issued, the host must use the 512 cycle timeout to calculate when the data is ready for retrieval.

3.4.7.1 Long-ACK Hardware Handshake Protocol

If a command results in an error condition, whereby a BDCCSRL flag is set, then the target generates a “Long-ACK” low pulse of 64 BDCSI clock cycles, followed by a brief speed pulse. This indicates to the host that an error has occurred. The host can subsequently read BDCCSR to determine the type of error. Whether normal ACK or Long-ACK, the ACK pulse is not issued earlier than 32 BDCSI clock cycles after the BDC command was issued. The end of the BDC command is assumed to be the 16th BDCSI clock cycle of the last bit. The 32 cycle minimum delay differs from the 16 cycle delay time with ACK disabled.

If a BDC access request does not gain access within 512 core clock cycles, the request is aborted, the NORESP flag is set and a Long-ACK pulse is transmitted to indicate an error case.

Following a STOP or WAI instruction, if the BDC is enabled, the first ACK, following stop or wait mode entry is a long ACK to indicate an exception.

3.4.8 Hardware Handshake Abort Procedure

The abort procedure is based on the SYNC command. To abort a command that has not responded with an ACK pulse, the host controller generates a sync request (by driving BKGD low for at least 128 BDCSI clock cycles and then driving it high for one BDCSI clock cycle as a speedup pulse). By detecting this long low pulse in the BKGD pin, the target executes the SYNC protocol, see Section 3.4.4.1, “SYNC”, and assumes that the pending command and therefore the related ACK pulse are being aborted. After the SYNC protocol has been completed the host is free to issue new BDC commands.

The host can issue a SYNC close to the 128 clock cycles length, providing a small overhead on the pulse length to assure the sync pulse is not misinterpreted by the target. See Section 3.4.4.1, “SYNC”.

Figure 3-11 shows a SYNC command being issued after a READ_MEM, which aborts the READ_MEM command. Note that, after the command is aborted a new command is issued by the host.

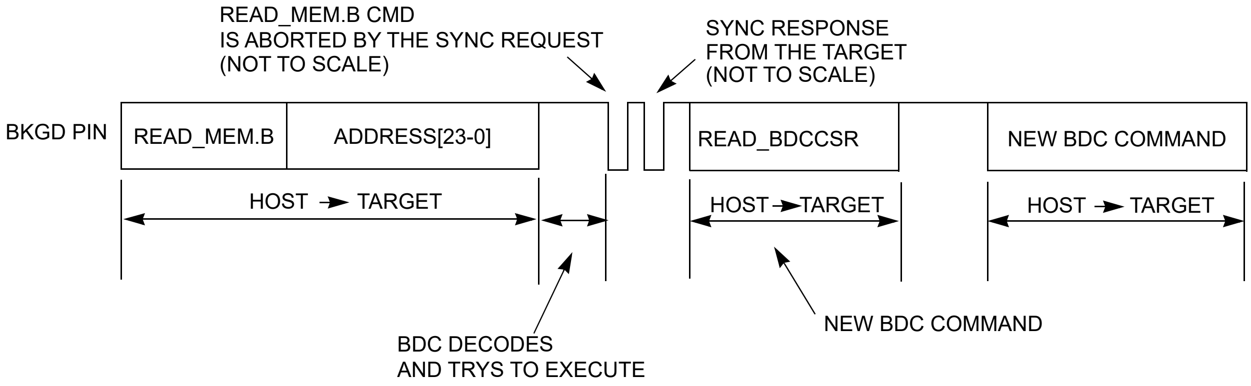


Figure 3-11. ACK Abort Procedure at the Command Level (Not To Scale)

Figure 3-12 shows a conflict between the ACK pulse and the SYNC request pulse. The target is executing a pending BDC command at the exact moment the host is being connected to the BKGD pin. In this case, an ACK pulse is issued simultaneously to the SYNC command. Thus there is an electrical conflict between the ACK speedup pulse and the SYNC pulse. As this is not a probable situation, the protocol does not prevent this conflict from happening.

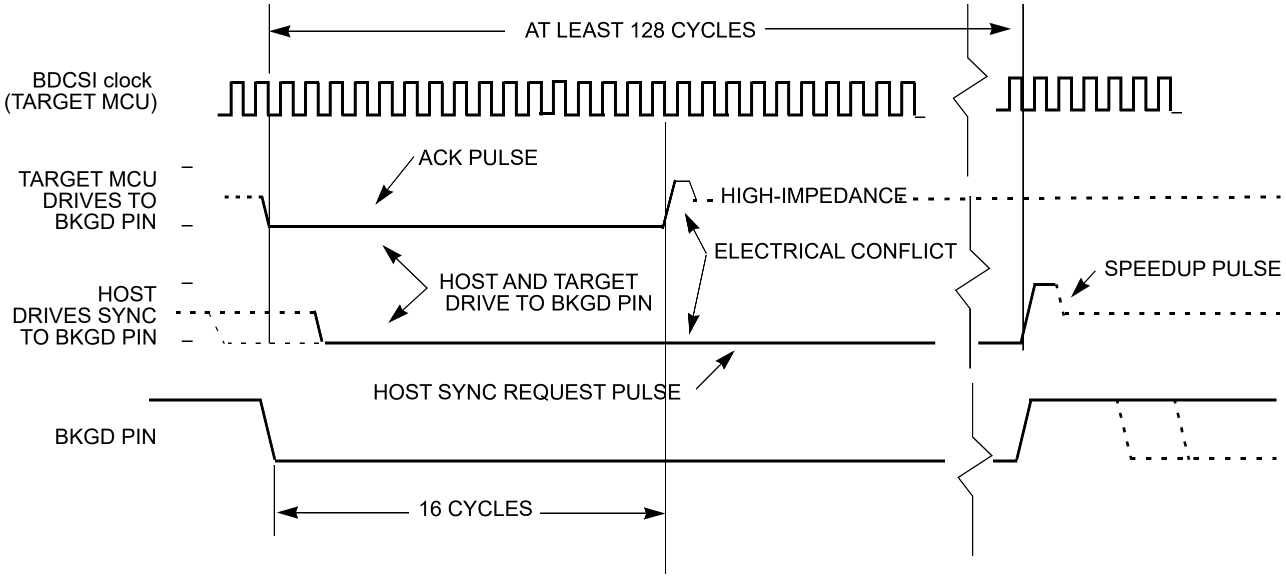


Figure 3-12. ACK Pulse and SYNC Request Conflict

3.4.9 Hardware Handshake Disabled (ACK Pulse Disabled)

The default state of the BDC after reset is hardware handshake protocol disabled. It can also be disabled by the ACK_DISABLE BDC command. This provides backwards compatibility with the existing host devices which are not able to execute the hardware handshake protocol. For host devices that support the hardware handshake protocol, true non-intrusive debugging and error flagging is offered.

If the ACK pulse protocol is disabled, the host needs to use the worst case delay time at the appropriate places in the protocol.

If the handshake protocol is disabled, the access is always independent of free cycles, whereby BDC has higher priority than CPU. Since at least 2 bytes (command byte + data byte) are transferred over BKGD the maximum intrusiveness is only once every few hundred cycles.

After decoding an internal access command, the BDC then awaits the next internal core clock cycle. The relationship between BDCSI clock and core clock must be considered. If the host retrieves the data immediately, then the BDCSI clock frequency must not be more than 4 times the core clock frequency, in order to guarantee that the BDC gains bus access within 16 the BDCSI cycle DLY period following an access command. If the BDCSI clock frequency is more than 4 times the core clock frequency, then the host must use a suitable delay time before retrieving data (see 3.5.1/3-150). Furthermore, for stretched read accesses to external resources via a device expanded bus (if implemented) the potential extra stretch cycles must be taken into consideration before attempting to obtain read data.

If the access does not succeed before the host starts data retrieval then the NORESP flag is set but the access is not aborted. The NORESP state can be used by the host to recognize an unexpected access conflict due to stretched expanded bus accesses. Although the NORESP bit is set when an access does not succeed before the start of data retrieval, the access may succeed in following bus cycles if the internal access has already been initiated.

3.4.10 Single Stepping

When a STEP1 command is issued to the BDC in active BDM, the CPU executes a single instruction in the user code and returns to active BDM. The STEP1 command can be issued repeatedly to step through the user code one instruction at a time.

If an interrupt is pending when a STEP1 command is issued, the interrupt stacking operation occurs but no user instruction is executed. In this case the stacking counts as one instruction. The device re-enters active BDM with the program counter pointing to the first instruction in the interrupt service routine.

When stepping through the user code, the execution of the user code is done step by step but peripherals are free running. Some peripheral modules include a freeze feature, whereby their clocks are halted when the device enters active BDM. Timer modules typically include the freeze feature. Serial interface modules typically do not include the freeze feature. Hence possible timing relations between CPU code execution and occurrence of events of peripherals no longer exist.

If the handshake protocol is enabled and BDCCIS is set then stepping over the STOP instruction causes the Long-ACK pulse to be generated and the BDCCSR STOP flag to be set. When stop mode is exited due to an interrupt the device enters active BDM and the PC points to the start of the corresponding interrupt service routine. Stepping can be continued.

Stepping over a WAI instruction, the STEP1 command cannot be finished because active BDM cannot be entered after CPU starts to execute the WAI instruction.

Stepping over the WAI instruction causes the BDCCSR WAIT and NORESP flags to be set and, if the handshake protocol is enabled, then the Long-ACK pulse is generated. Then the device enters wait mode, clears the BDMACT bit and awaits an interrupt to leave wait mode. In this time non-intrusive BDC commands are possible, although the STEP1 has actually not finished. When an interrupt occurs the device leaves wait mode, enters active BDM and the PC points to the start of the corresponding interrupt service routine. A further ACK related to stepping over the WAI is not generated.

3.4.11 Serial Communication Timeout

The host initiates a host-to-target serial transmission by generating a falling edge on the BKGD pin. If BKGD is kept low for more than 128 target clock cycles, the target understands that a SYNC command was issued. In this case, the target waits for a rising edge on BKGD in order to answer the SYNC request pulse. When the BDC detects the rising edge a soft reset is generated, whereby the current BDC command is discarded. If the rising edge is not detected, the target keeps waiting forever without any timeout limit.

If a falling edge is not detected by the target within 512 clock cycles since the last falling edge, a timeout occurs and the current command is discarded without affecting memory or the operating mode of the MCU. This is referred to as a soft-reset. This timeout also applies if 512 cycles elapse between 2 consecutive ERASE_FLASH commands. The soft reset is disabled whilst the internal flash mass erase operation is pending completion.

timeouts are also possible if a BDC command is partially issued, or data partially retrieved. Thus if a time greater than 512 BDCSI clock cycles is observed between two consecutive negative edges, a soft-reset occurs causing the partially received command or data retrieved to be discarded. The next negative edge at the BKGD pin, after a soft-reset has occurred, is considered by the target as the start of a new BDC command, or the start of a SYNC request pulse.

3.5 Application Information

3.5.1 Clock Frequency Considerations

Read commands without status and without ACK must consider the frequency relationship between BDCSI and the internal core clock. If the core clock is slow, then the internal access may not have been carried out within the standard 16 BDCSI cycle delay period (DLY). The host must then extend the DLY period or clock frequencies accordingly. Taking internal clock domain synchronizers into account, the minimum number of BDCSI periods required for the DLY is expressed by:

$\#DLY > 3(f_{(BDCSI\ clock)} / f_{(core\ clock)}) + 4$

and the minimum core clock frequency with respect to BDCSI clock frequency is expressed by

Minimum $f_{(core\ clock)} = (3/(\#DLY\ cycles\ -4))f_{(BDCSI\ clock)}$

For the standard 16 period DLY this yields $f_{(core\ clock)} >= (1/4)f_{(BDCSI\ clock)}$

Chapter 4
Memory Mapping Control (S12ZMMCV1)

Table 4-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.05	6 Aug 2012		Fixed wording
V01.06	12 Feb 2013	Figure 4-8 4.3.2.2/4-156	• Changed “KByte:to “KB” • Corrected the description of the MMCECH/L register •
V01.07	3 May 2013		• Fixed typos • Removed PTU references

4.1 Introduction

The S12ZMMC module controls the access to all internal memories and peripherals for the S12ZCPU, and the S12ZBDC module. It also provides dirct memory access for the ADC module. The S12ZMMC determines the address mapping of the on-chip resources, regulates access priorities and enforces memory protection. Figure 4-1 shows a block diagram of the S12ZMMC module.

4.1.1 Glossary

Table 4-2. Glossary Of Terms

Term	Definition
MCU	Microcontroller Unit
CPU	S12Z Central Processing Unit
BDC	S12Z Background Debug Controller
ADC	Analog-to-Digital Converter
unmapped address range	Address space that is not assigned to a memory
reserved address range	Address space that is reserved for future use cases
illegal access	Memory access, that is not supported or prohibited by the S12ZMMC, e.g. a data store to NVM
access violation	Either an illegal access or an uncorrectable ECC error
byte	8-bit data
word	16-bit data

4.1.2 Overview

The S12ZMMC provides access to on-chip memories and peripherals for the S12ZCPU, the S12ZBDC, and the ADC. It arbitrates memory accesses and determines all of the MCU memory maps. Furthermore, the S12ZMMC is responsible for selecting the MCUs functional mode.

4.1.3 Features

- S12ZMMC mode operation control
- Memory mapping for S12ZCPU, S12ZBDC, and ADC
  - Maps peripherals and memories into a 16 MByte address space for the S12ZCPU, the S12ZBDC, and the ADC
  - Handles simultaneous accesses to different on-chip resources (NVM, RAM, and peripherals)
- Access violation detection and logging
  - Triggers S12ZCPU machine exceptions upon detection of illegal memory accesses and uncorrectable ECC errors
  - Logs the state of the S12ZCPU and the cause of the access error

4.1.4 Modes of Operation

4.1.4.1 Chip configuration modes

The S12ZMMC determines the chip configuration mode of the device. It captures the state of the MODC pin at reset and provides the ability to switch from special-single chip mode to normal single chip-mode.

4.1.4.2 Power modes

The S12ZMMC module is only active in run and wait mode.There is no bus activity in stop mode.

4.1.5 Block Diagram

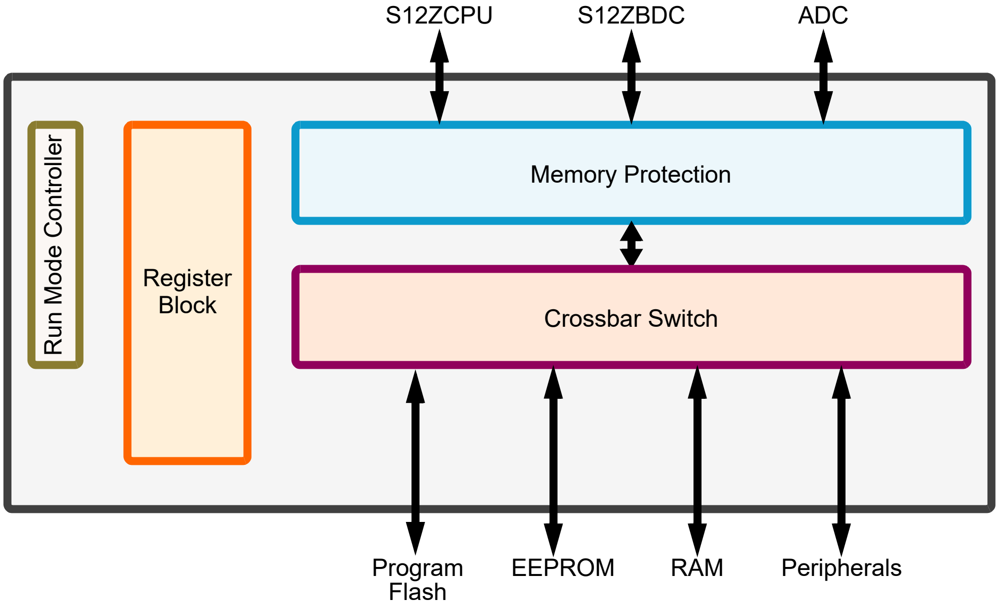


Figure 4-1. S12ZMMC Block Diagram

4.2 External Signal Description

The S12ZMMC uses two external pins to determine the devices operating mode: RESET and MODC (Table 4-3)
See device overview for the mapping of these signals to device pins.

Table 4-3. External System Pins Associated With S12ZMMC

Pin Name	Description
$\overline{\text{RESET}}$	External reset signal. The $\overline{\text{RESET}}$ signal is active low.
MODC	This input is captured in bit MODC of the MODE register when the external $\overline{\text{RESET}}$ pin deasserts.

4.3 Memory Map and Register Definition

4.3.1 Memory Map

A summary of the registers associated with the MMC block is shown in Figure 4-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0070	MODE	R	MODC	0	0	0	0	0	0	0
		W		▨	▨	▨	▨	▨	▨	▨
0x0071- 0x007F	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0080	MMCECH	R	ITR[3:0]				TGT[3:0]			
		W								
0x0081	MMCECL	R	ACC[3:0]				ERR[3:0]			
		W								
0x0082	MMCCCRH	R	CPUU	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0083	MMCCCRL	R	0	CPUX	0	CPUI	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0084	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0085	MMCPCH	R	CPUPC[23:16]							
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0086	MMCPCM	R	CPUPC[15:8]							
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0087	MMCPCL	R	CPUPC[7:0]							
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0088- 0x00FF	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 4-2. S12ZMMC Register Summary

4.3.2 Register Descriptions

This section consists of the S12ZMMC control and status register descriptions in address order.

4.3.2.1 Mode Register (MODE)

Address: 0x0070

	7	6	5	4	3	2	1	0
R	MODC	0	0	0	0	0	0	0
W		▨	▨	▨	▨	▨	▨	▨
Reset	MODC[1]	0	0	0	0	0	0	0

[1] External signal (see Table 4-3).

▨ = Unimplemented or Reserved

Figure 4-3. Mode Register (MODE)

Read: Anytime.

Write: Only if a transition is allowed (see Figure 4-4).

The MODE register determines the operating mode of the MCU.

CAUTION

Table 4-4. MODE Field Descriptions

Field	Description
7 MODC	**Mode Select Bit** — This bit determines the current operating mode of the MCU. Its reset value is captured from the MODC pin at the rising edge of the $\overline{\text{RESET}}$ pin. Figure 4-4 illustrates the only valid mode transition from special single-chip mode to normal single chip mode.

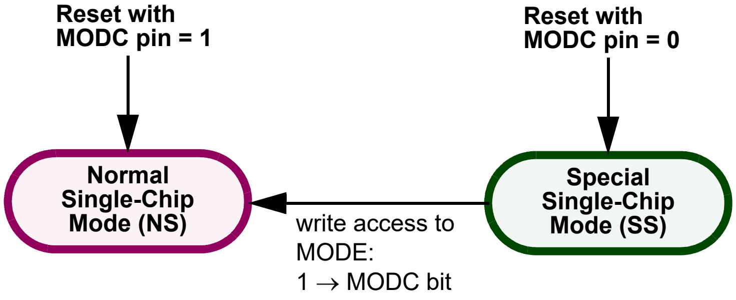


Figure 4-4. Mode Transition Diagram

4.3.2.2 Error Code Register (MMCECH, MMCECL)

Address: 0x0080 (MMCECH)

	7	6	5	4	3	2	1	0
R W	ITR[3:0]				TGT[3:0]			
Reset	0	0	0	0	0	0	0	0

Address: 0x0081 (MMCECL)

	7	6	5	4	3	2	1	0
R W	ACC[3:0]				ERR[3:0]			
Reset	0	0	0	0	0	0	0	0

Figure 4-5. Error Code Register (MMCEC)

Read: Anytime

Write: Write of 0xFFFF to MMCECH:MMCECL resets both registers to 0x0000

Table 4-5. MMCECH and MMCECL Field Descriptions

Field	Description
7-4 (MMCECH) ITR[3:0]	**Initiator Field** — The ITR[3:0] bits capture the initiator which caused the access violation. The initiator is captured in form of a 4 bit value which is assigned as follows: 0: none (no error condition detected) 1: S12ZCPU 2: reserved 3: ADC 4-15: reserved
3-0 (MMCECH) TGT[3:0]	**Target Field** — The TGT[3:0] bits capture the target of the faulty access. The target is captured in form of a 4 bit value which is assigned as follows: 0: none 1: register space 2: RAM 3: EEPROM 4: program flash 5: IFR 6-15: reserved

Field	Description
7-4 (MMCECL) ACC[3:0]	**Access Type Field** — The ACC[3:0] bits capture the type of memory access, which caused the access violation. The access type is captured in form of a 4 bit value which is assigned as follows: 0: none (no error condition detected) 1: opcode fetch 2: vector fetch 3: data load 4: data store 5-15: reserved
3-0 (MMCECL) ERR[3:0]	**Error Type Field** — The EC[3:0] bits capture the type of the access violation. The type is captured in form of a 4 bit value which is assigned as follows: 0: none (no error condition detected) 1: access to an illegal address 2: uncorrectable ECC error 3-15: reserved

The MMCEC register captures debug information about access violations. It is set to a non-zero value if a S12ZCPU access violation or an uncorrectable ECC error has occurred. At the same time this register is set to a non-zero value, access information is captured in the MMCPCn and MMCCCRn registers. The MMCECn, the MMCPCn and the MMCCCRn registers are not updated if the MMCECn registers contain a non-zero value. The MMCECn registers are cleared by writing the value 0xFFFF.

4.3.2.3 Captured S12ZCPU Condition Code Register (MMCCCRH, MMCCCRL)

Address: 0x0082 (MMCCCRH)

	7	6	5	4	3	2	1	0
R	CPUU	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Address: 0x0083 (MMCCCRL)

	7	6	5	4	3	2	1	0
R	0	CPUX	0	CPUI	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 4-6. Captured S12ZCPU Condition Code Register (MMCCCRH, MMCCCRL)

Read: Anytime

Write: Never

Table 4-6. MMCCCRH and MMCCCRL Field Descriptions

Field	Description
7 (MMCCCRH) CPUU	**S12ZCPU User State Flag** — This bit shows the state of the user/supervisor mode bit in the S12ZCPU's CCR at the time the access violation has occurred. The S12ZCPU user state flag is read-only; it will be automatically updated when the next error condition is flagged through the MMCEC register. This bit is undefined if the error code registers (MMCECn) are cleared.
6 (MMCCCRL) CPUX	**S12ZCPU X-Interrupt Mask**— This bit shows the state of the X-interrupt mask in the S12ZCPU's CCR at the time the access violation has occurred. The S12ZCPU X-interrupt mask is read-only; it will be automatically updated when the next error condition is flagged through the MMCEC register. This bit is undefined if the error code registers (MMCECn) are cleared.
4 (MMCCCRL) CPUI	**S12ZCPU I-Interrupt Mask**— This bit shows the state of the I-interrupt mask in the CPU's CCR at the time the access violation has occurred. The S12ZCPU I-interrupt mask is read-only; it will be automatically updated when the next error condition is flagged through the MMCEC register. This bit is undefined if the error code registers (MMCECn) are cleared.

4.3.2.4 Captured S12ZCPU Program Counter (MMCPCH, MMCPCM, MMCPCL)

Address: 0x0085 (MMCPCH)

	7	6	5	4	3	2	1	0
R	CPUPC[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

Address: 0x0086 (MMCPCM)

	7	6	5	4	3	2	1	0
R	CPUPC[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Address: 0x0087 (MMCPCL)

	7	6	5	4	3	2	1	0
R	CPUPC[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 4-7. Captured S12ZCPU Program Counter (MMCPCH, MMCPCM, MMCPCL)

Read: Anytime

Write: Never

Table 4-7. MMCPCH, MMCPCM, and MMCPCL Field Descriptions

Field	Description
7–0 (MMCPCH) 7–0 (MMCPCM) 7–0 (MMCPCL) CPUPC[23:0]	**S12ZCPU Program Counter Value**— The CPUPC[23:0] stores the CPU's program counter value at the time the access violation occurred. CPUPC[23:0] always points to the instruction which triggered the violation. These bits are undefined if the error code registers (MMCECn) are cleared.

4.4 Functional Description

This section provides a complete functional description of the S12ZMMC module.

4.4.1 Global Memory Map

The S12ZMMC maps all on-chip resources into an 16MB address space, the global memory map. The exact resource mapping is shown in Figure 4-8. The global address space is used by the S12ZCPU, ADC, and the S12ZBDC module.

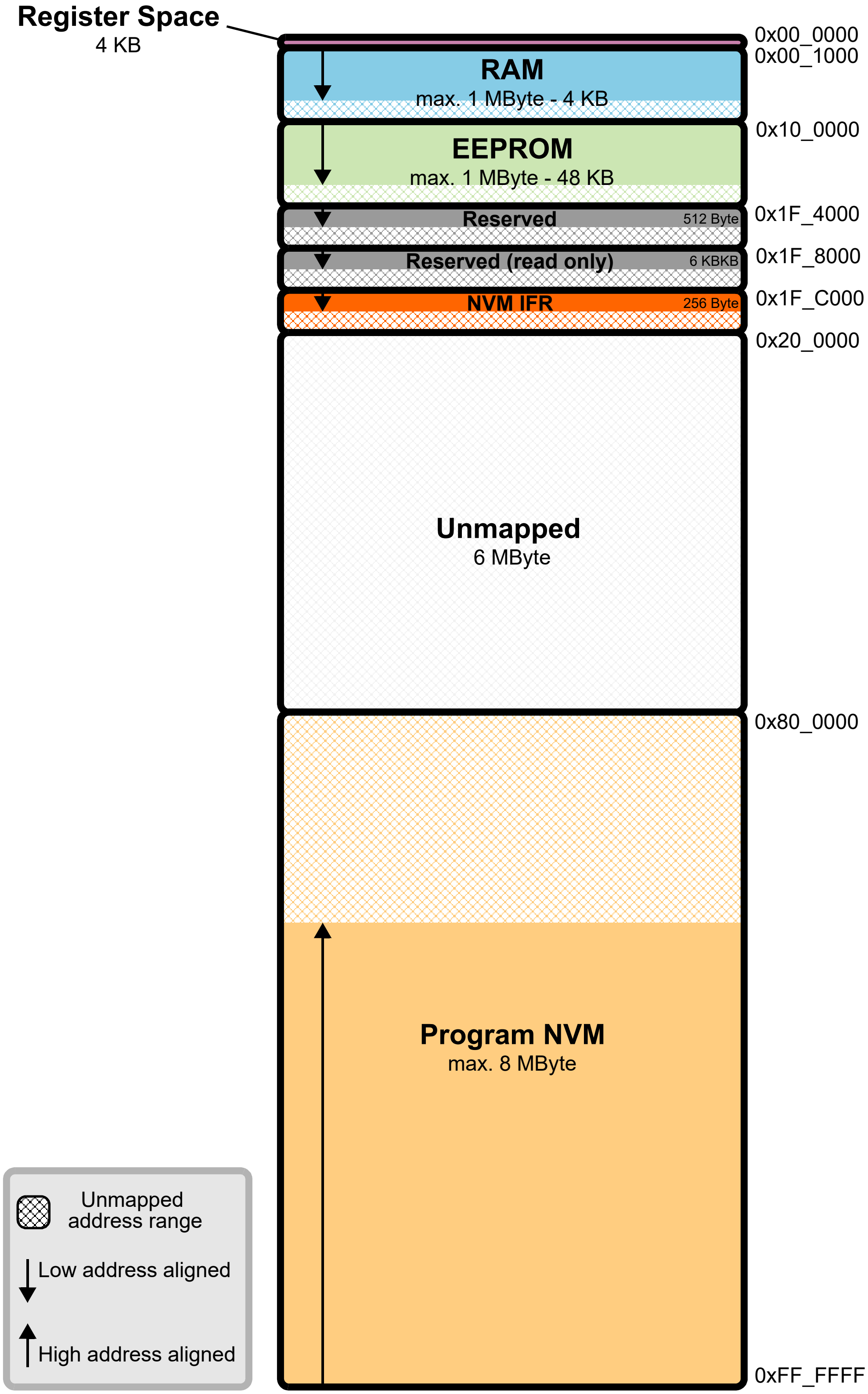


Figure 4-8. Global Memory Map

4.4.2 Illegal Accesses

The S12ZMMC module monitors all memory traffic for illegal accesses. See Table 4-8 for a complete list of all illegal accesses.

Table 4-8. Illegal memory accesses

		S12ZCPU	S12ZBDC	ADC
Register space	Read access	ok	ok	illegal access
	Write access	ok	ok	illegal access
	Code execution	illegal access		
RAM	Read access	ok	ok	ok
	Write access	ok	ok	ok
	Code execution	ok		
EEPROM	Read access	ok[1]	ok[1]	ok[1]
	Write access	illegal access	illegal access	illegal access
	Code execution	ok[1]		
Reserved Space	Read access	ok	ok	illegal access
	Write access	only permitted in SS mode	ok	illegal access
	Code execution	illegal access		
Reserved Read-only Space	Read access	ok	ok	illegal access
	Write access	illegal access	illegal access	illegal access
	Code execution	illegal access		
NVM IFR	Read access	ok[1]	ok[1]	illegal access
	Write access	illegal access	illegal access	illegal access
	Code execution	illegal access		
Program NVM	Read access	ok[1]	ok[1]	ok[1]
	Write access	illegal access	illegal access	illegal access
	Code execution	ok[1]		
Unmapped Space	Read access	illegal access	illegal access	illegal access
	Write access	illegal access	illegal access	illegal access
	Code execution	illegal access		

---

[1] Unsupported NVM accesses during NVM command execution (“collisions”), are treated as illegal accesses.

Illegal accesses are reported in several ways:

- All illegal accesses performed by the S12ZCPU trigger machine exceptions.
- All illegal accesses performed through the S12ZBDC interface, are captured in the ILLACC bit of the BDCCSRL register.

- All illegal accesses performed by the ADC module trigger error interrupts. See ADC section for details.

NOTE

Illegal accesses caused by S12ZCPU opcode prefetches will also trigger machine exceptions, even if those opcodes might not be executed in the program flow. To avoid these machine exceptions, S12ZCPU instructions must not be executed from the last (high addresses) 8 bytes of RAM, EEPROM, and Flash.

4.4.3 Uncorrectable ECC Faults

RAM and flash use error correction codes (ECC) to detect and correct memory corruption. Each uncorrectable memory corruption, which is detected during a S12ZCPU or ADC access triggers a machine exception. Uncorrectable memory corruptions which are detected during a S12ZBDC access, are captured in the RAMWF or the RDINV bit of the BDCCSRL register.

Chapter 5
S12Z Interrupt (S12ZINTV0)

Table 5-1. Revision History

Version Number	Revision Date	Effective Date	Description of Changes
V00.01	17 Apr 2009	all	Initial version based on S12XINT V2.06
V00.02	14 Jul 2009	all	Reduce RESET vectors from three to one.
V00.03	05 Oct 2009	all	Removed dedicated ECC machine exception vector and marked vector-table entry “reserved for future use”. Added a second illegal op-code vector (to distinguish between SPARE and TRAP).
V00.04	04 Jun 2010	all	Fixed remaining descriptions of RESET vectors. Split non-maskable hardware interrupts into XGATE software error and machine exception requests. Replaced mentions of CCR (old name from S12X) with CCW (new name).
V00.05	12 Jan 2011	all	Corrected wrong IRQ vector address in some descriptions.
V00.06	22 Mar 2011	all	Added vectors for RAM ECC and NVM ECC machine exceptions. And moved position to 1E0..1E8. Moved XGATE error interrupt to vector 1DC. Remaining vectors accordingly. Removed illegal address reset as a potential reset source.
V00.07	15 Apr 2011	all	Removed illegal address reset as a potential reset source from Exception vector table as well. Added the other possible reset sources to the table. Changed register addresses according to S12Z platform definition.
V00.08	02 May 2011	all	Reduced machine exception vectors to one. Removed XGATE error interrupt. Moved Spurious interrupt vector to 1DC. Moved vector base address to 010 to make room for NVM non-volatile registers.
V00.09	12 Aug 2011	all	Added: Machine exceptions can cause wake-up from STOP or WAIT
V00.10	21 Feb 2012	all	Corrected reset value for INT_CFADDR register
V00.11	02 Jul 2012	all	Removed references and functions related to XGATE
V00.12	22 May 2013	all	added footnote about availability of “Wake-up from STOP or WAIT by XIRQ with X bit set” feature

5.1 Introduction

The INT module decodes the priority of all system exception requests and provides the applicable vector for processing the exception to the CPU. The INT module supports:

- I-bit and X-bit maskable interrupt requests
- One non-maskable unimplemented page1 op-code trap

- One non-maskable unimplemented page2 op-code trap
- One non-maskable software interrupt (SWI)
- One non-maskable system call interrupt (SYS)
- One non-maskable machine exception vector request
- One spurious interrupt vector request
- One system reset vector request

Each of the I-bit maskable interrupt requests can be assigned to one of seven priority levels supporting a flexible priority scheme. The priority scheme can be used to implement nested interrupt capability where interrupts from a lower level are automatically blocked if a higher level interrupt is being processed.

5.1.1 Glossary

The following terms and abbreviations are used in the document.

Table 5-2. Terminology

Term	Meaning
CCW	Condition Code Register (in the S12Z CPU)
DMA	Direct Memory Access
INT	Interrupt
IPL	Interrupt Processing Level
ISR	Interrupt Service Routine
MCU	Micro-Controller Unit
$\overline{\text{IRQ}}$	refers to the interrupt request associated with the $\overline{\text{IRQ}}$ pin
$\overline{\text{XIRQ}}$	refers to the interrupt request associated with the $\overline{\text{XIRQ}}$ pin

5.1.2 Features

- Interrupt vector base register (IVBR)
- One system reset vector (at address 0xFFFFFC).
- One non-maskable unimplemented page1 op-code trap (SPARE) vector (at address vector base[1] + 0x0001F8).
- One non-maskable unimplemented page2 op-code trap (TRAP) vector (at address vector base[1] + 0x0001F4).
- One non-maskable software interrupt request (SWI) vector (at address vector base[1] + 0x0001F0).
- One non-maskable system call interrupt request (SYS) vector (at address vector base[1] + 0x00001EC).
- One non-maskable machine exception vector request (at address vector base[1] + 0x0001E8).
- One spurious interrupt vector (at address vector base[1] + 0x0001DC).

---

[1] The vector base is a 24-bit address which is accumulated from the contents of the interrupt vector base register (IVBR, used as the upper 15 bits of the address) and 0x000 (used as the lower 9 bits of the address).

- One X-bit maskable interrupt vector request associated with $\overline{\text{XIRQ}}$ (at address vector base[1] + 0x0001D8).
- One I-bit maskable interrupt vector request associated with $\overline{\text{IRQ}}$ (at address vector base[1] + 0x0001D4).
- up to 113 additional I-bit maskable interrupt vector requests (at addresses vector base[1] + 0x000010 .. vector base + 0x0001D0).
- Each I-bit maskable interrupt request has a configurable priority level.
- I-bit maskable interrupts can be nested, depending on their priority levels.
- Wakes up the system from stop or wait mode when an appropriate interrupt request occurs or whenever $\overline{\text{XIRQ}}$ is asserted, even if X interrupt is masked.

5.1.3 Modes of Operation

- Run mode

  This is the basic mode of operation.
- Wait mode

  In wait mode, the INT module is capable of waking up the CPU if an eligible CPU exception occurs. Please refer to Section 5.5.3, “Wake Up from Stop or Wait Mode” for details.
- Stop Mode

  In stop mode, the INT module is capable of waking up the CPU if an eligible CPU exception occurs. Please refer to Section 5.5.3, “Wake Up from Stop or Wait Mode” for details.

5.1.4 Block Diagram

Figure 5-1 shows a block diagram of the INT module.

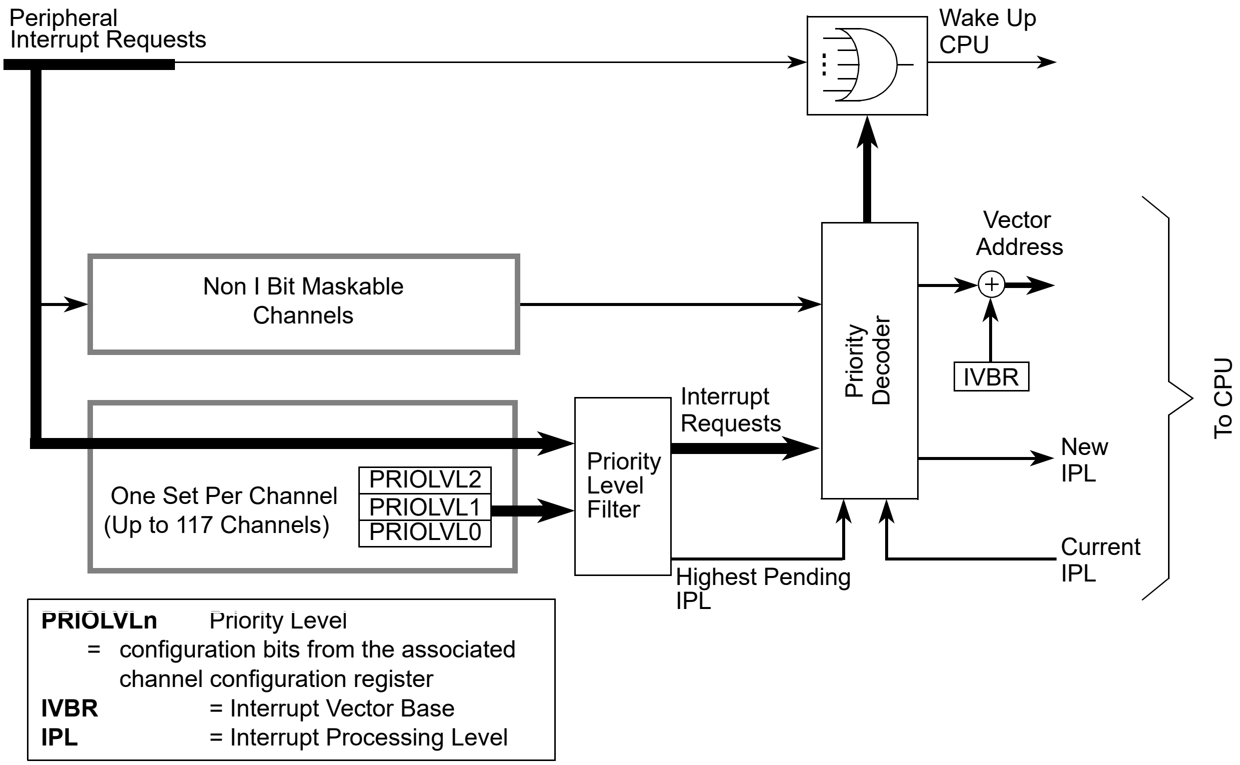


Figure 5-1. INT Block Diagram

5.2 External Signal Description

The INT module has no external signals.

5.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the INT module.

5.3.1 Module Memory Map

Table 5-3 gives an overview over all INT module registers.

Table 5-3. INT Memory Map

Address	Use	Access
0x000010–0x000011	Interrupt Vector Base Register (IVBR)	R/W
0x000012–0x000016	RESERVED	—
0x000017	Interrupt Request Configuration Address Register (INT_CFADDR)	R/W
0x000018	Interrupt Request Configuration Data Register 0 (INT_CFDATA0)	R/W

Table 5-3. INT Memory Map

0x000019	Interrupt Request Configuration Data Register 1 (INT_CFDATA1)	R/W
0x00001A	Interrupt Request Configuration Data Register 2 (INT_CFDATA2	R/W
0x00001B	Interrupt Request Configuration Data Register 3 (INT_CFDATA3)	R/W
0x00001C	Interrupt Request Configuration Data Register 4 (INT_CFDATA4)	R/W
0x00001D	Interrupt Request Configuration Data Register 5 (INT_CFDATA5)	R/W
0x00001E	Interrupt Request Configuration Data Register 6 (INT_CFDATA6)	R/W
0x00001F	Interrupt Request Configuration Data Register 7 (INT_CFDATA7)	R/W

5.3.2 Register Descriptions

This section describes in address order all the INT module registers and their individual bits.

<table>
<tr><th>Address</th><th>Register Name</th><th></th><th>Bit 7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>Bit 0</th></tr>
<tr><td>0x000010</td><td>IVBR</td><td>R</td><td colspan="8" rowspan="2">IVB_ADDR[15:8]</td></tr>
<tr><td></td><td></td><td>W</td></tr>
<tr><td>0x000011</td><td></td><td>R</td><td colspan="7" rowspan="2">IVB_ADDR[7:1]</td><td>0</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td></tr>
<tr><td>0x000017</td><td>INT_CFADDR</td><td>R</td><td>0</td><td colspan="4" rowspan="2">INT_CFADDR[6:3]</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>0x000018</td><td>INT_CFDATA0</td><td>R</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3" rowspan="2">PRIOLVL[2:0]</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>0x000019</td><td>INT_CFDATA1</td><td>R</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3" rowspan="2">PRIOLVL[2:0]</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>0x00001A</td><td>INT_CFDATA2</td><td>R</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3" rowspan="2">PRIOLVL[2:0]</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>0x00001B</td><td>INT_CFDATA3</td><td>R</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3" rowspan="2">PRIOLVL[2:0]</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>0x00001C</td><td>INT_CFDATA4</td><td>R</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3" rowspan="2">PRIOLVL[2:0]</td></tr>
<tr><td></td><td></td><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
</table>

▨ = Unimplemented or Reserved

Figure 5-2. INT Register Summary

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x00001D	INT_CFDATA5	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x00001E	INT_CFDATA6	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x00001F	INT_CFDATA7	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			

▨ = Unimplemented or Reserved

Figure 5-2. INT Register Summary

5.3.2.1 Interrupt Vector Base Register (IVBR)

Address: 0x000010

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IVB_ADDR[15:1]															0
W																▨
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Figure 5-3. Interrupt Vector Base Register (IVBR)

Read: Anytime

Write: Anytime

Table 5-4. IVBR Field Descriptions

Field	Description
15–1 IVB_ADDR [15:1]	**Interrupt Vector Base Address Bits** — These bits represent the upper 15 bits of all vector addresses. Out of reset these bits are set to 0xFFFE (i.e., vectors are located at 0xFFFE00–0xFFFFFF). **Note:** A system reset will initialize the interrupt vector base register with “0xFFFE” before it is used to determine the reset vector address. Therefore, changing the IVBR has no effect on the location of the reset vector (0xFFFFFC–0xFFFFFF).

5.3.2.2 Interrupt Request Configuration Address Register (INT_CFADDR)

Address: 0x000017

	7	6	5	4	3	2	1	0
R	0	INT_CFADDR[6:3]				0	0	0
W	▨					▨	▨	▨
Reset	0	0	0	0	1	0	0	0

▨ = Unimplemented or Reserved

Figure 5-4. Interrupt Configuration Address Register (INT_CFADDR)

Read: Anytime

Write: Anytime

Table 5-5. INT_CFADDR Field Descriptions

Field	Description
6–3 INT_CFADDR[6:3]	**Interrupt Request Configuration Data Register Select Bits** — These bits determine which of the 128 configuration data registers are accessible in the 8 register window at INT_CFDATA0–7. The hexadecimal value written to this register corresponds to the upper 4 bits of the vector number (multiply with 4 to get the vector address offset). If, for example, the value 0x70 is written to this register, the configuration data register block for the 8 interrupt vector requests starting with vector at address (vector base + (0x70*4 = 0x0001C0)) is selected and can be accessed as INT_CFDATA0–7.

5.3.2.3 Interrupt Request Configuration Data Registers (INT_CFDATA0–7)

The eight register window visible at addresses INT_CFDATA0–7 contains the configuration data for the block of eight interrupt requests (out of 128) selected by the interrupt configuration address register (INT_CFADDR) in ascending order. INT_CFDATA0 represents the interrupt configuration data register of the vector with the lowest address in this block, while INT_CFDATA7 represents the interrupt configuration data register of the vector with the highest address, respectively.

Address: 0x000018

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-5. Interrupt Request Configuration Data Register 0 (INT_CFDATA0)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x000019

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-6. Interrupt Request Configuration Data Register 1 (INT_CFDATA1)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001A

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-7. Interrupt Request Configuration Data Register 2 (INT_CFDATA2)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-8. Interrupt Request Configuration Data Register 3 (INT_CFDATA3)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001C

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-9. Interrupt Request Configuration Data Register 4 (INT_CFDATA4)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001D

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-10. Interrupt Request Configuration Data Register 5 (INT_CFDATA5)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001E

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-11. Interrupt Request Configuration Data Register 6 (INT_CFDATA6)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x00001F

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PRIOLVL[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	1[1]

▨ = Unimplemented or Reserved

Figure 5-12. Interrupt Request Configuration Data Register 7 (INT_CFDATA7)

[1] Please refer to the notes following the PRIOLVL[2:0] description below.

Read: Anytime

Write: Anytime

Table 5-6. INT_CFDATA0–7 Field Descriptions

Field	Description
2–0 PRIOLVL[2:0]	**Interrupt Request Priority Level Bits** — The PRIOLVL[2:0] bits configure the interrupt request priority level of the associated interrupt request. Out of reset all interrupt requests are enabled at the lowest active level (“1”). Please also refer to Table 5-7 for available interrupt request priority levels. **Note:** Write accesses to configuration data registers of unused interrupt channels are ignored and read accesses return all 0s. For information about what interrupt channels are used in a specific MCU, please refer to the Device Reference Manual for that MCU. **Note:** When non I-bit maskable request vectors are selected, writes to the corresponding INT_CFDATA registers are ignored and read accesses return all 0s. The corresponding vectors do not have configuration data registers associated with them. **Note:** Write accesses to the configuration register for the spurious interrupt vector request (vector base + 0x0001DC) are ignored and read accesses return 0x07 (request is handled by the CPU, PRIOLVL = 7).

Table 5-7. Interrupt Priority Levels

Priority	PRIOLVL2	PRIOLVL1	PRIOLVL0	Meaning
	0	0	0	Interrupt request is disabled
low	0	0	1	Priority level 1
	0	1	0	Priority level 2
	0	1	1	Priority level 3
	1	0	0	Priority level 4
	1	0	1	Priority level 5
	1	1	0	Priority level 6
high	1	1	1	Priority level 7

5.4 Functional Description

The INT module processes all exception requests to be serviced by the CPU module. These exceptions include interrupt vector requests and reset vector requests. Each of these exception types and their overall priority level is discussed in the subsections below.

5.4.1 S12Z Exception Requests

The CPU handles both reset requests and interrupt requests. The INT module contains registers to configure the priority level of each I-bit maskable interrupt request which can be used to implement an interrupt priority scheme. This also includes the possibility to nest interrupt requests. A priority decoder is used to evaluate the relative priority of pending interrupt requests.

5.4.2 Interrupt Prioritization

After system reset all I-bit maskable interrupt requests are configured to be enabled, are set up to be handled by the CPU and have a pre-configured priority level of 1. Exceptions to this rule are the non-maskable interrupt requests and the spurious interrupt vector request at (vector base + 0x0001DC) which cannot be disabled, are always handled by the CPU and have a fixed priority levels. A priority level of 0 effectively disables the associated I-bit maskable interrupt request.

If more than one interrupt request is configured to the same interrupt priority level the interrupt request with the higher vector address wins the prioritization.

The following conditions must be met for an I-bit maskable interrupt request to be processed.

1. The local interrupt enabled bit in the peripheral module must be set.
2. The setup in the configuration register associated with the interrupt request channel must meet the following conditions:
   a) The priority level must be set to non zero.
   b) The priority level must be greater than the current interrupt processing level in the condition code register (CCW) of the CPU (PRIOLVL[2:0] > IPL[2:0]).
3. The I-bit in the condition code register (CCW) of the CPU must be cleared.
4. There is no access violation interrupt request pending.
5. There is no SYS, SWI, SPARE, TRAP, Machine Exception or $\overline{\text{XIRQ}}$ request pending.

NOTE

All non I-bit maskable interrupt requests always have higher priority than I-bit maskable interrupt requests. If an I-bit maskable interrupt request is interrupted by a non I-bit maskable interrupt request, the currently active interrupt processing level (IPL) remains unaffected. It is possible to nest non I-bit maskable interrupt requests, e.g., by nesting SWI, SYS or TRAP calls.

5.4.2.1 Interrupt Priority Stack

The current interrupt processing level (IPL) is stored in the condition code register (CCW) of the CPU. This way the current IPL is automatically pushed to the stack by the standard interrupt stacking procedure. The new IPL is copied to the CCW from the priority level of the highest priority active interrupt request channel which is configured to be handled by the CPU. The copying takes place when the interrupt vector is fetched. The previous IPL is automatically restored from the stack by executing the RTI instruction.

5.4.3 Priority Decoder

The INT module contains a priority decoder to determine the relative priority for all interrupt requests pending for the CPU.

A CPU interrupt vector is not supplied until the CPU requests it. Therefore, it is possible that a higher priority interrupt request could override the original exception which caused the CPU to request the vector. In this case, the CPU will receive the highest priority vector and the system will process this exception first instead of the original request.

If the interrupt source is unknown (for example, in the case where an interrupt request becomes inactive after the interrupt has been recognized, but prior to the vector request), the vector address supplied to the CPU defaults to that of the spurious interrupt vector.

NOTE

Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not get processed at all or the result may be a spurious interrupt request (vector at address (vector base + 0x0001DC)).

5.4.4 Reset Exception Requests

The INT module supports one system reset exception request. The different reset types are mapped to this vector (for details please refer to the Clock and Power Management Unit module (CPMU)):

1. Pin reset
2. Power-on reset
3. Low-voltage reset
4. Clock monitor reset request
5. COP watchdog reset request

5.4.5 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT module upon request by the CPU are shown in Table 5-8. Generally, all non-maskable interrupts have higher priorities than maskable interrupts. Please note that between the four software interrupts (Unimplemented op-code trap page1/page2 requests, SWI request, SYS request) there is no real priority defined since they cannot occur simultaneously (the S12Z CPU executes one instruction at a time).

Table 5-8. Exception Vector Map and Priority

Vector Address[1]	Source
0xFFFFFC	Pin reset, power-on reset, low-voltage reset, clock monitor reset, COP watchdog reset
(Vector base + 0x0001F8)	Unimplemented page1 op-code trap (SPARE) vector request
(Vector base + 0x0001F4)	Unimplemented page2 op-code trap (TRAP) vector request
(Vector base + 0x0001F0)	Software interrupt instruction (SWI) vector request
(Vector base + 0x0001EC)	System call interrupt instruction (SYS) vector request
(Vector base + 0x0001E8)	Machine exception vector request
(Vector base + 0x0001E4)	Reserved
(Vector base + 0x0001E0)	Reserved
(Vector base + 0x0001DC)	Spurious interrupt
(Vector base + 0x0001D8)	$\overline{\text{XIRQ}}$ interrupt request
(Vector base + 0x0001D4)	$\overline{\text{IRQ}}$ interrupt request
(Vector base + 0x000010 .. Vector base + 0x0001D0)	Device specific I-bit maskable interrupt sources (priority determined by the associated configuration registers, in descending order)

[1] 24 bits vector address based

5.4.6 Interrupt Vector Table Layout

The interrupt vector table contains 128 entries, each 32 bits (4 bytes) wide. Each entry contains a 24-bit address (3 bytes) which is stored in the 3 low-significant bytes of the entry. The content of the most significant byte of a vector-table entry is ignored. Figure 5-13 illustrates the vector table entry format.

Bits	[31:24]	[23:0]
	(unused)	ISR Address

Figure 5-13. Interrupt Vector Table Entry

5.5 Initialization/Application Information

5.5.1 Initialization

After system reset, software should:

- Initialize the interrupt vector base register if the interrupt vector table is not located at the default location (0xFFFE00–0xFFFFFB).
- Initialize the interrupt processing level configuration data registers (INT_CFADDR, INT_CFDATA0–7) for all interrupt vector requests with the desired priority levels. It might be a good idea to disable unused interrupt requests.
- Enable I-bit maskable interrupts by clearing the I-bit in the CCW.
- Enable the X-bit maskable interrupt by clearing the X-bit in the CCW (if required).

5.5.2 Interrupt Nesting

The interrupt request priority level scheme makes it possible to implement priority based interrupt request nesting for the I-bit maskable interrupt requests.

- I-bit maskable interrupt requests can be interrupted by an interrupt request with a higher priority, so that there can be up to seven nested I-bit maskable interrupt requests at a time (refer to Figure 5-14 for an example using up to three nested interrupt requests).

I-bit maskable interrupt requests cannot be interrupted by other I-bit maskable interrupt requests per default. In order to make an interrupt service routine (ISR) interruptible, the ISR must explicitly clear the I-bit in the CCW (CLI). After clearing the I-bit, I-bit maskable interrupt requests with higher priority can interrupt the current ISR.

An ISR of an interruptible I-bit maskable interrupt request could basically look like this:

- Service interrupt, e.g., clear interrupt flags, copy data, etc.
- Clear I-bit in the CCW by executing the CPU instruction CLI (thus allowing interrupt requests with higher priority)
- Process data
- Return from interrupt by executing the instruction RTI

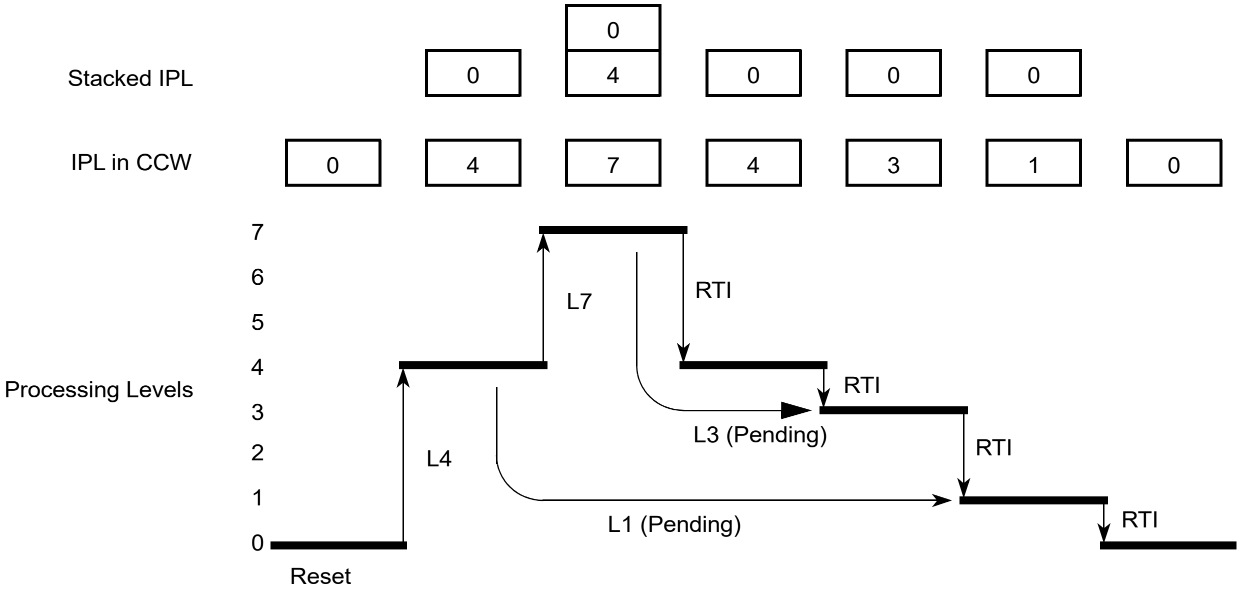


Figure 5-14. Interrupt Processing Example

5.5.3 Wake Up from Stop or Wait Mode

5.5.3.1 CPU Wake Up from Stop or Wait Mode

Every I-bit maskable interrupt request which is configured to be handled by the CPU is capable of waking the MCU from stop or wait mode. Additionally machine exceptions can wake-up the MCU from stop or wait mode.

To determine whether an I-bit maskable interrupts is qualified to wake up the CPU or not, the same settings as in normal run mode are applied during stop or wait mode:

- If the I-bit in the CCW is set, all I-bit maskable interrupts are masked from waking up the MCU.
- An I-bit maskable interrupt is ignored if it is configured to a priority level below or equal to the current IPL in CCW.

The X-bit maskable interrupt request can wake up the MCU from stop or wait mode at anytime, even if the X-bit in CCW is set[1]. If the X-bit maskable interrupt request is used to wake-up the MCU with the X-bit in the CCW set, the associated ISR is not called. The CPU then resumes program execution with the instruction following the WAI or STOP instruction. This feature works following the same rules like any interrupt request, i.e. care must be taken that the X-bit maskable interrupt request used for wake-up remains active at least until the system begins execution of the instruction following the WAI or STOP instruction; otherwise, wake-up may not occur.

---

[1] The capability of the $\overline{\text{XIRQ}}$ pin to wake-up the MCU with the X bit set may not be available if, for example, the $\overline{\text{XIRQ}}$ pin is shared with other peripheral modules on the device. Please refer to the Port Integration Module (PIM) section of the MCU reference manual for details.

Chapter 6
S12Z DebugLite (S12ZDBGV3) Module

Table 6-1. Revision History Table

Revision Number	Revision Date	Sections Affected	Description Of Changes
3.00	23.MAY.2012	General	Updated for DBGV3 using conditional text
3.01	27.JUN.2012	General	Added Lite to module name. Corrected DBGEFR register format issue
3.02	05.JUL.2012	Section 6.3.2.6, “Debug Event Flag Register (DBGEFR)	Removed ME2 flag from DBGEFR
3.03	16.NOV.2012	Section 6.5.1, “Avoiding Unintended Breakpoint Re-triggering	Modified step over breakpoint information
3.04	19.DEC.2012	General	Formatting corrections
3.05	19.APR.2013	General	Specified DBGC1[0] reserved bit as read only
3.06	15.JUL.2013	Section 6.3.2, “Register Descriptions	Added explicit names to state control register bit fields

6.1 Introduction

The DBG module provides on-chip breakpoints with flexible triggering capability to allow non-intrusive debug of application software. The DBG module is optimized for the S12Z architecture and allows debugging of CPU module operations.

Typically the DBG module is used in conjunction with the BDC module, whereby the user configures the DBG module for a debugging session over the BDC interface. Once configured the DBG module is armed and the device leaves active BDM returning control to the user program, which is then monitored by the DBG module. Alternatively the DBG module can be configured over a serial interface using SWI routines.

6.1.1 Glossary

Table 6-2. Glossary Of Terms

Term	Definition
COF	Change Of Flow. Change in the program flow due to a conditional branch, indexed jump or interrupt
PC	Program Counter
BDM	Background Debug Mode. In this mode CPU application code execution is halted. Execution of BDC “active BDM” commands is possible.
BDC	Background Debug Controller
WORD	16-bit data entity
CPU	S12Z CPU module

6.1.2 Overview

The comparators monitor the bus activity of the CPU. A single comparator match or a series of matches can generate breakpoints. A state sequencer determines if the correct series of matches occurs. Similarly an external event can generate breakpoints.

6.1.3 Features

- Three comparators (A, B, and D)
  - Comparator A compares the full address bus and full 32-bit data bus
  - Comparator A features a data bus mask register
  - Comparators B and D compare the full address bus only
  - Each comparator can be configured to monitor PC addresses or addresses of data accesses
  - Each comparator can select either read or write access cycles
  - Comparator matches can force state sequencer state transitions
- Three comparator modes
  - Simple address/data comparator match mode
  - Inside address range mode, Addmin ≤ Address ≤ Addmax
  - Outside address range match mode, Address < Addmin or Address > Addmax
- State sequencer control
  - State transitions forced by comparator matches
  - State transitions forced by software write to TRIG
  - State transitions forced by an external event
- The following types of breakpoints
  - CPU breakpoint entering active BDM on breakpoint (BDM)
  - CPU breakpoint executing SWI on breakpoint (SWI)
  -

6.1.4 Modes of Operation

The DBG module can be used in all MCU functional modes.

The DBG module can issue breakpoint requests to force the device to enter active BDM or an SWI ISR. The BDC BACKGROUND command is also handled by the DBG to force the device to enter active BDM. When the device enters active BDM through a BACKGROUND command with the DBG module armed, the DBG remains armed.

6.1.5 Block Diagram

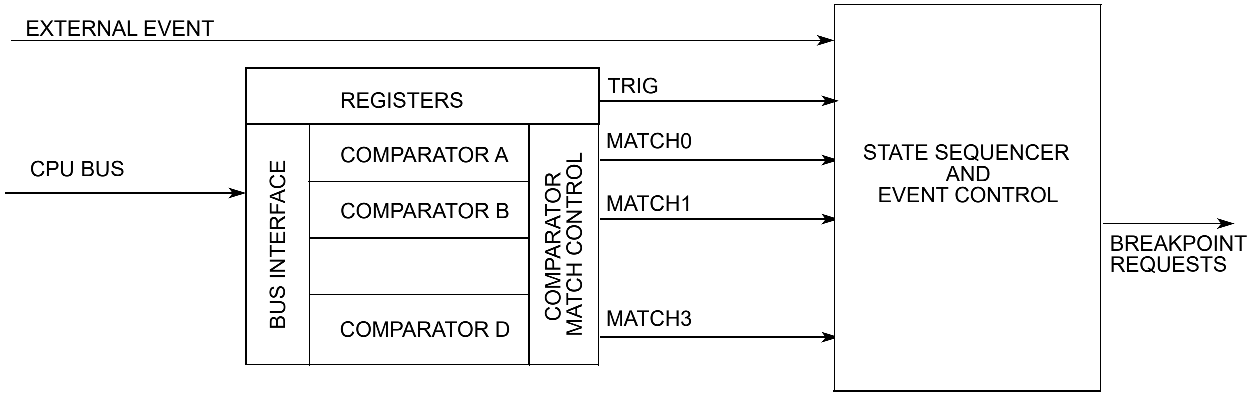


Figure 6-1. Debug Module Block Diagram

6.2 External Signal Description

6.2.1 External Event Input

The DBG module features an external event input signal, DBGEEV. The mapping of this signal to a device pin is specified in the device specific documentation. This function can be enabled and configured by the EEVE field in the DBGC1 control register. This signal is input only and allows an external event to force a state sequencer transition. With the external event function enabled, a falling edge at the external event pin constitutes an event. Rising edges have no effect. The maximum frequency of events is half the internal core bus frequency. The function is explained in the EEVE field description.

NOTE

> Due to input pin synchronization circuitry, the DBG module sees external events 2 bus cycles after they occur at the pin. Thus an external event occurring less than 2 bus cycles before arming the DBG module is perceived to occur whilst the DBG is armed.

When the device is in stop mode the synchronizer clocks are disabled and the external events are ignored.

6.3 Memory Map and Registers

6.3.1 Module Memory Map

A summary of the registers associated with the DBG module is shown in Figure 6-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0100	DBGC1	R	ARM	0	reserved	BDMBP	BRKCPU	reserved	EEVE1	0
		W		TRIG						
0x0101	DBGC2	R	0	0	0	0	0	0	ABCM	
		W								
0x0102	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0103	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0104	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0105	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0106	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0107	DBGSCR1	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0108	DBGSCR2	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0109	DBGSCR3	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x010A	DBGEFR	R	0	TRIGF	0	EEVF	ME3	0	ME1	ME0
		W								
0x010B	DBGSR	R	0	0	0	0	0	SSF2	SSF1	SSF0
		W								
0x010C- 0x010F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0110	DBGACTL	R	0	NDB	INST	0	RW	RWE	reserved	COMPE
		W								

Figure 6-2. Quick Reference to DBG Registers

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0111- 0x0114	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0115	DBGAAH	R	DBGAA[23:16]							
		W								
0x0116	DBGAAM	R	DBGAA[15:8]							
		W								
0x0117	DBGAAL	R	DBGAA[7:0]							
		W								
0x0118	DBGAD0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x0119	DBGAD1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x011A	DBGAD2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x011B	DBGAD3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x011C	DBGADM0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x011D	DBGADM1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x011E	DBGADM2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x011F	DBGADM3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0120	DBGBCTL	R	0	0	INST	0	RW	RWE	reserved	COMPE
		W								
0x0121- 0x0124	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0125	DBGBAH	R	DBGBA[23:16]							
		W								
0x0126	DBGBAM	R	DBGBA[15:8]							
		W								
0x0127	DBGBAL	R	DBGBA[7:0]							
		W								
0x0128- 0x012F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0130- 0x013F	Reserved	R	0	0	0	0	0	0	0	0
		W								

Figure 6-2. Quick Reference to DBG Registers

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0140	DBGDCTL	R	0	0	INST	0	RW	RWE	reserved	COMPE
		W								
0x0141- 0x0144	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0145	DBGDAH	R W	DBGDA[23:16]							
0x0146	DBGDAM	R W	DBGDA[15:8]							
0x0147	DBGDAL	R W	DBGDA[7:0]							
0x0148- 0x017F	Reserved	R	0	0	0	0	0	0	0	0
		W								

Figure 6-2. Quick Reference to DBG Registers

6.3.2 Register Descriptions

This section consists of the DBG register descriptions in address order. When ARM is set in DBGC1, the only bits in the DBG module registers that can be written are ARM, and TRIG

6.3.2.1 Debug Control Register 1 (DBGC1)

Address: 0x0100

	7	6	5	4	3	2	1	0
0x0100	ARM	0	reserved	BDMBP	BRKCPU	reserved	EEVE1	0
		TRIG						
Reset	0	0	0	0	0	0	0	0

Figure 6-3. Debug Control Register (DBGC1)

Read: Anytime

Write: Bit 7 Anytime . An ongoing profiling session must be finished before DBG can be armed again.
Bit 6 can be written anytime but always reads back as 0.
Bits 5:0 anytime DBG is not armed.

NOTE

On a write access to DBGC1 and simultaneous hardware disarm from an internal event, the hardware disarm has highest priority, clearing the ARM bit and generating a breakpoint, if enabled.

NOTE

When disarming the DBG by clearing ARM with software, the contents of bits[5:0] are not affected by the write, since up until the write operation, ARM = 1 preventing these bits from being written. These bits must be cleared using a second write if required.

Table 6-3. DBGC1 Field Descriptions

Field	Description
7 ARM	**Arm Bit** — The ARM bit controls whether the DBG module is armed. This bit can be set and cleared by register writes and is automatically cleared when the state sequencer returns to State0 on completing a debugging session. On setting this bit the state sequencer enters State1. 0 Debugger disarmed. No breakpoint is generated when clearing this bit by software register writes. 1 Debugger armed
6 TRIG	**Immediate Trigger Request Bit** — This bit when written to 1 requests an immediate transition to final state independent of comparator status. This bit always reads back a 0. Writing a 0 to this bit has no effect. 0 No effect. 1 Force state sequencer immediately to final state.
4 BDMBP	**Background Debug Mode Enable** — This bit determines if a CPU breakpoint causes the system to enter Background Debug Mode (BDM) or initiate a Software Interrupt (SWI). If this bit is set but the BDC is not enabled, then no breakpoints are generated. 0 Breakpoint to Software Interrupt if BDM inactive. Otherwise no breakpoint. 1 Breakpoint to BDM, if BDC enabled. Otherwise no breakpoint.
3 BRKCPU	**CPU Breakpoint Enable** — The BRKCPU bit controls whether the debugger requests a breakpoint to CPU upon transitions to State0. Please refer to Section 6.4.5, "Breakpoints for further details. 0 Breakpoints disabled 1 Breakpoints enabled
1 EEVE1	**External Event Enable** — The EEVE1 bit enables the external event function. 0 External event function disabled. 1 External event is mapped to the state sequencer, replacing comparator channel 3

6.3.2.2 Debug Control Register2 (DBGC2)

Address: 0x0101

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	ABCM	
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-4. Debug Control Register2 (DBGC2)

Read: Anytime.

Write: Anytime the module is disarmed.

This register configures the comparators for range matching.

Table 6-4. DBGC2 Field Descriptions

Field	Description
1–0 ABCM[1:0]	**A and B Comparator Match Control** — These bits determine the A and B comparator match mapping as described in Table 6-5.

Table 6-5. ABCM Encoding

ABCM	Description
00	Match0 mapped to comparator A match....... Match1 mapped to comparator B match.
01	Match0 mapped to comparator A/B inside range....... Match1 disabled.
10	Match0 mapped to comparator A/B outside range....... Match1 disabled.
11	Reserved[1]

[1] Currently defaults to Match0 mapped to inside range: Match1 disabled

6.3.2.3 Debug State Control Register 1 (DBGSCR1)

Address: 0x0107

	7	6	5	4	3	2	1	0
R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-6. Debug State Control Register 1 (DBGSCR1)

Read: Anytime.

Write: If DBG is not armed.

The state control register 1 selects the targeted next state whilst in State1. The matches refer to the outputs of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8, “Debug Comparator A Control Register (DBGACTL)”. Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 6-7. DBGSCR1 Field Descriptions

Field	Description
1–0 C0SC[1:0]	Channel 0 State Control. These bits select the targeted next state whilst in State1 following a match0.
3–2 C1SC[1:0]	Channel 1 State Control. These bits select the targeted next state whilst in State1 following a match1.
7–6 C3SC[1:0]	Channel 3 State Control. If EEVE !=10, these bits select the targeted next state whilst in State1 following a match3. If EEVE = 10, these bits select the targeted next state whilst in State1 following an external event.

Table 6-8. State1 Match State Sequencer Transitions

CxSC[1:0]	Function
00	Match has no effect
01	Match forces sequencer to State2
10	Match forces sequencer to State3
11	Match forces sequencer to Final State

In the case of simultaneous matches, the match on the higher channel number (3...0) has priority.

6.3.2.4 Debug State Control Register 2 (DBGSCR2)

Address: 0x0108

	7	6	5	4	3	2	1	0
R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-7. Debug State Control Register 2 (DBGSCR2)

Read: Anytime.

Write: If DBG is not armed

The state control register 2 selects the targeted next state whilst in State2. The matches refer to the outputs of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8, “Debug Comparator A Control Register (DBGACTL)”. Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 6-9. DBGSCR2 Field Descriptions

Field	Description
1–0 C0SC[1:0]	Channel 0 State Control. These bits select the targeted next state whilst in State2 following a match0.
3–2 C1SC[1:0]	Channel 1 State Control. These bits select the targeted next state whilst in State2 following a match1.
7–6 C3SC[1:0]	Channel 3 State Control. If EEVE !=10, these bits select the targeted next state whilst in State2 following a match3. If EEVE =10, these bits select the targeted next state whilst in State2 following an external event.

Table 6-10. State2 Match State Sequencer Transitions

CxSC[1:0]	Function
00	Match has no effect
01	Match forces sequencer to State1
10	Match forces sequencer to State3
11	Match forces sequencer to Final State

In the case of simultaneous matches, the match on the higher channel number (3...0) has priority.

6.3.2.5 Debug State Control Register 3 (DBGSCR3)

Address: 0x0109

	7	6	5	4	3	2	1	0
R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
W			▨	▨				
Reset	0	0	0	0	0	0	0	0

Figure 6-8. Debug State Control Register 3 (DBGSCR3)

Read: Anytime.

Write: If DBG is not armed.

The state control register three selects the targeted next state whilst in State3. The matches refer to the outputs of the comparator match control logic as depicted in Figure 6-1 and described in Section 6.3.2.8, “Debug Comparator A Control Register (DBGACTL)”. Comparators must be enabled by setting the comparator enable bit in the associated DBGxCTL control register.

Table 6-11. DBGSCR3 Field Descriptions

Field	Description
1–0 C0SC[1:0]	Channel 0 State Control. These bits select the targeted next state whilst in State3 following a match0.
3–2 C1SC[1:0]	Channel 1 State Control. These bits select the targeted next state whilst in State3 following a match1.
7–6 C3SC[1:0]	Channel 3 State Control. If EEVE !=10, these bits select the targeted next state whilst in State3 following a match3. If EEVE =10, these bits select the targeted next state whilst in State3 following an external event.

Table 6-12. State3 Match State Sequencer Transitions

CxSC[1:0]	Function
00	Match has no effect
01	Match forces sequencer to State1
10	Match forces sequencer to State2
11	Match forces sequencer to Final State

In the case of simultaneous matches, the match on the higher channel number (3....0) has priority.

6.3.2.6 Debug Event Flag Register (DBGEFR)

Address: 0x010A

	7	6	5	4	3	2	1	0
R	0	TRIGF	0	EEVF	ME3	0	ME1	ME0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-9. Debug Event Flag Register (DBGEFR)

Read: Anytime.

Write: Never

DBGEFR contains flag bits each mapped to events whilst armed. Should an event occur, then the corresponding flag is set. With the exception of TRIGF, the bits can only be set when the ARM bit is set. The TRIGF bit is set if a TRIG event occurs when ARM is already set, or if the TRIG event occurs simultaneous to setting the ARM bit.All other flags can only be cleared by arming the DBG module. Thus the contents are retained after a debug session for evaluation purposes.

A set flag does not inhibit the setting of other flags.

Table 6-13. DBGEFR Field Descriptions

Field	Description
6 TRIGF	**TRIG Flag** — Indicates the occurrence of a TRIG event during the debug session. 0 No TRIG event 1 TRIG event
4 EEVF	**External Event Flag** — Indicates the occurrence of an external event during the debug session. 0 No external event 1 External event
3–0 ME[3:0]	**Match Event[3:0]**— Indicates a comparator match event on the corresponding comparator channel**.**

6.3.2.7 Debug Status Register (DBGSR)

Address: 0x010B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	SSF2	SSF1	SSF0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	—	0	0	0	0	0	0	0
POR	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-10. Debug Status Register (DBGSR)

Read: Anytime.

Write: Never.

Table 6-14. DBGSR Field Descriptions

Field	Description
2–0 SSF[2:0]	**State Sequencer Flag Bits** — The SSF bits indicate the current State Sequencer state. During a debug session on each transition to a new state these bits are updated. If the debug session is ended by software clearing the ARM bit, then these bits retain their value to reflect the last state of the state sequencer before disarming. If a debug session is ended by an internal event, then the state sequencer returns to State0 and these bits are cleared to indicate that State0 was entered during the session. On arming the module the state sequencer enters State1 and these bits are forced to SSF[2:0] = 001. See Table 6-15**.**

Table 6-15. SSF[2:0] — State Sequence Flag Bit Encoding

SSF[2:0]	Current State
000	State0 (disarmed)
001	State1
010	State2
011	State3
100	Final State
101,110,111	Reserved

6.3.2.8 Debug Comparator A Control Register (DBGACTL)

Address: 0x0110

	7	6	5	4	3	2	1	0
R	0	NDB	INST	0	RW	RWE	reserved	COMPE
W	▨			▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-11. Debug Comparator A Control Register

Read: Anytime.

Write: If DBG not armed.

Table 6-16. DBGACTL Field Descriptions

Field	Description
6 NDB	**Not Data Bus** — The NDB bit controls whether the match occurs when the data bus matches the comparator register value or when the data bus differs from the register value. This bit is ignored if the INST bit in the same register is set. 0 Match on data bus equivalence to comparator register contents 1 Match on data bus difference to comparator register contents
5 INST	**Instruction Select** — This bit configures the comparator to compare PC or data access addresses. 0 Comparator compares addresses of data accesses 1 Comparator compares PC address
3 RW	**Read/Write Comparator Value Bit** — The RW bit controls whether read or write is used in compare for the associated comparator. The RW bit is ignored if RWE is clear or INST is set. 0 Write cycle is matched 1 Read cycle is matched
2 RWE	**Read/Write Enable Bit** — The RWE bit controls whether read or write comparison is enabled for the associated comparator. This bit is ignored when INST is set. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 COMPE	**Enable Bit** — Determines if comparator is enabled 0 The comparator is not enabled 1 The comparator is enabled

Table 6-17 shows the effect for RWE and RW on the comparison conditions. These bits are ignored if INST is set, because matches based on opcodes reaching the execution stage are data independent.

Table 6-17. Read or Write Comparison Logic Table

RWE Bit	RW Bit	RW Signal	Comment
0	x	0	RW not used in comparison
0	x	1	RW not used in comparison
1	0	0	Write match
1	0	1	No match
1	1	0	No match
1	1	1	Read match

6.3.2.9 Debug Comparator A Address Register (DBGAAH, DBGAAM, DBGAAL)

Address: 0x0115, DBGAAH

	23	22	21	20	19	18	17	16
R W	DBGAA[23:16]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0116, DBGAAM

	15	14	13	12	11	10	9	8
R W	DBGAA[15:8]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0117, DBGAAL

	7	6	5	4	3	2	1	0
R W	DBGAA[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 6-12. Debug Comparator A Address Register

Read: Anytime.

Write: If DBG not armed.

Table 6-18. DBGAAH, DBGAAM, DBGAAL Field Descriptions

Field	Description
23–16 DBGAA [23:16]	**Comparator Address Bits [23:16]**— These comparator address bits control whether the comparator compares the address bus bits [23:16] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one
15–0 DBGAA [15:0]	**Comparator Address Bits [15:0]**— These comparator address bits control whether the comparator compares the address bus bits [15:0] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

6.3.2.10 Debug Comparator A Data Register (DBGAD)

Address: 0x0118, 0x0119, 0x011A, 0x011B

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

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

Figure 6-13. Debug Comparator A Data Register (DBGAD)

Read: Anytime.

Write: If DBG not armed.

This register can be accessed with a byte resolution, whereby DBGAD0, DBGAD1, DBGAD2, DBGAD3 map to DBGAD[31:0] respectively.

Table 6-19. DBGAD Field Descriptions

Field	Description
31–16 Bits[31:16] (DBGAD0, DBGAD1)	**Comparator Data Bits** — These bits control whether the comparator compares the data bus bits to a logic one or logic zero. The comparator data bits are only used in comparison if the corresponding data mask bit is logic 1. 0 Compare corresponding data bit to a logic zero 1 Compare corresponding data bit to a logic one
15–0 Bits[15:0] (DBGAD2, DBGAD3)	**Comparator Data Bits** — These bits control whether the comparator compares the data bus bits to a logic one or logic zero. The comparator data bits are only used in comparison if the corresponding data mask bit is logic 1. 0 Compare corresponding data bit to a logic zero 1 Compare corresponding data bit to a logic one

6.3.2.11 Debug Comparator A Data Mask Register (DBGADM)

Address: 0x011C, 0x011D, 0x011E, 0x011F

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

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

Figure 6-14. Debug Comparator A Data Mask Register (DBGADM)

Read: Anytime.

Write: If DBG not armed.

This register can be accessed with a byte resolution, whereby DBGADM0, DBGADM1, DBGADM2, DBGADM3 map to DBGADM[31:0] respectively.

Table 6-20. DBGADM Field Descriptions

Field	Description
31–16 Bits[31:16] (DBGADM0, DBGADM1)	**Comparator Data Mask Bits** — These bits control whether the comparator compares the data bus bits to the corresponding comparator data compare bits. 0 Do not compare corresponding data bit 1 Compare corresponding data bit
15-0 Bits[15:0] (DBGADM2, DBGADM3)	**Comparator Data Mask Bits** — These bits control whether the comparator compares the data bus bits to the corresponding comparator data compare bits. 0 Do not compare corresponding data bit 1 Compare corresponding data bit

6.3.2.12 Debug Comparator B Control Register (DBGBCTL)

Address: 0x0120

	7	6	5	4	3	2	1	0
R	0	0	INST	0	RW	RWE	reserved	COMPE
W	▨	▨		▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-15. Debug Comparator B Control Register

Read: Anytime.

Write: If DBG not armed.

Table 6-21. DBGBCTL Field Descriptions

Field[1]	Description
5 INST	**Instruction Select** — This bit configures the comparator to compare PC or data access addresses. 0 Comparator compares addresses of data accesses 1 Comparator compares PC address
3 RW	**Read/Write Comparator Value Bit** — The RW bit controls whether read or write is used in compare for the associated comparator. The RW bit is ignored if RWE is clear or INST is set. 0 Write cycle is matched 1 Read cycle is matched
2 RWE	**Read/Write Enable Bit** — The RWE bit controls whether read or write comparison is enabled for the associated comparator. This bit is ignored when INST is set. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 COMPE	**Enable Bit** — Determines if comparator is enabled 0 The comparator is not enabled 1 The comparator is enabled

[1] If the ABCM field selects range mode comparisons, then DBGACTL bits configure the comparison, DBGBCTL is ignored.

Table 6-22 shows the effect for RWE and RW on the comparison conditions. These bits are ignored if INST is set, as matches based on instructions reaching the execution stage are data independent.

Table 6-22. Read or Write Comparison Logic Table

RWE Bit	RW Bit	RW Signal	Comment
0	x	0	RW not used in comparison
0	x	1	RW not used in comparison
1	0	0	Write match
1	0	1	No match
1	1	0	No match
1	1	1	Read match

6.3.2.13 Debug Comparator B Address Register (DBGBAH, DBGBAM, DBGBAL)

Address: 0x0125, DBGBAH

	23	22	21	20	19	18	17	16
R W	DBGBA[23:16]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0126, DBGBAM

	15	14	13	12	11	10	9	8
R W	DBGBA[15:8]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0127, DBGBAL

	7	6	5	4	3	2	1	0
R W	DBGBA[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 6-16. Debug Comparator B Address Register

Read: Anytime.

Write: If DBG not armed.

Table 6-23. DBGBAH, DBGBAM, DBGBAL Field Descriptions

Field	Description
23–16 DBGBA [23:16]	**Comparator Address Bits [23:16]**— These comparator address bits control whether the comparator compares the address bus bits [23:16] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one
15–0 DBGBA [15:0]	**Comparator Address Bits**[15:0]— These comparator address bits control whether the comparator compares the address bus bits [15:0] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

6.3.2.14 Debug Comparator D Control Register (DBGDCTL)

Address: 0x0140

	7	6	5	4	3	2	1	0
R	0	0	INST	0	RW	RWE	reserved	COMPE
W	▨	▨		▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-17. Debug Comparator D Control Register

Read: Anytime.

Write: If DBG not armed.

Table 6-24. DBGDCTL Field Descriptions

Field[1]	Description
5 INST	**Instruction Select** — This bit configures the comparator to compare PC or data access addresses. 0 Comparator compares addresses of data accesses 1 Comparator compares PC address
3 RW	**Read/Write Comparator Value Bit** — The RW bit controls whether read or write is used in compare for the associated comparator. The RW bit is ignored if RWE is clear or INST is set. 0 Write cycle is matched 1 Read cycle is matched
2 RWE	**Read/Write Enable Bit** — The RWE bit controls whether read or write comparison is enabled for the associated comparator. This bit is ignored if INST is set. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 COMPE	**Enable Bit** — Determines if comparator is enabled 0 The comparator is not enabled 1 The comparator is enabled

[1] If the CDCM field selects range mode comparisons, then DBGCCTL bits configure the comparison, DBGDCTL is ignored.

Table 6-25 shows the effect for RWE and RW on the comparison conditions. These bits are ignored if INST is set, because matches based on opcodes reaching the execution stage are data independent.

Table 6-25. Read or Write Comparison Logic Table

RWE Bit	RW Bit	RW Signal	Comment
0	x	0	RW not used in comparison
0	x	1	RW not used in comparison
1	0	0	Write match
1	0	1	No match
1	1	0	No match
1	1	1	Read match

6.3.2.15 Debug Comparator D Address Register (DBGDAH, DBGDAM, DBGDAL)

Address: 0x0145, DBGDAH

	23	22	21	20	19	18	17	16
R W	DBGDA[23:16]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0146, DBGDAM

	15	14	13	12	11	10	9	8
R W	DBGDA[15:8]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0147, DBGDAL

	7	6	5	4	3	2	1	0
R W	DBGDA[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 6-18. Debug Comparator D Address Register

Read: Anytime.

Write: If DBG not armed.

Table 6-26. DBGDAH, DBGDAM, DBGDAL Field Descriptions

Field	Description
23–16 DBGDA [23:16]	**Comparator Address Bits [23:16]**— These comparator address bits control whether the comparator compares the address bus bits [23:16] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one
15–0 DBGDA [15:0]	**Comparator Address Bits**[15:0]— These comparator address bits control whether the comparator compares the address bus bits [15:0] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

6.4 Functional Description

This section provides a complete functional description of the DBG module.

6.4.1 DBG Operation

The DBG module operation is enabled by setting ARM in DBGC1. When armed it can be used to generate breakpoints to the CPU. The DBG module is made up of comparators, control logic, and the state sequencer, Figure 6-1.

The comparators monitor the bus activity of the CPU. Comparators can be configured to monitor opcode addresses (effectively the PC address) or data accesses. Comparators can be configured during data accesses to mask out individual data bus bits and to use R/W access qualification in the comparison. Comparators can be configured to monitor a range of addresses.

When configured for data access comparisons, the match is generated if the address (and optionally data) of a data access matches the comparator value.

Configured for monitoring opcode addresses, the match is generated when the associated opcode reaches the execution stage of the instruction queue, but before execution of that opcode.

When a match with a comparator register value occurs, the associated control logic can force the state sequencer to another state (see Figure 6-19).

The state sequencer can transition freely between the states 1, 2 and 3. On transition to Final State, a breakpoint can be generated and the state sequencer returns to state0, disarming the DBG.

Independent of the comparators, state sequencer transitions can be forced by the external event input or by writing to the TRIG bit in the DBGC1 control register.

6.4.2 Comparator Modes

The DBG contains three comparators, A, B, and D. Each comparator compares the address stored in DBGXAH, DBGXAM, and DBGXAL with the PC (opcode addresses) or selected address bus (data accesses). Furthermore, comparator A can compare the data buses to values stored in DBGXD3-0 and allow data bit masking.

The comparators can monitor the buses for an exact address or an address range. The comparator configuration is controlled by the control register contents and the range control by the DBGC2 contents.

The comparator control register also allows the type of data access to be included in the comparison through the use of the RWE and RW bits. The RWE bit controls whether the access type is compared for the associated comparator and the RW bit selects either a read or write access for a valid match.

The INST bit in each comparator control register is used to determine the matching condition. By setting INST, the comparator matches opcode addresses, whereby the databus, data mask, RW and RWE bits are ignored. The comparator register must be loaded with the exact opcode address.

The comparator can be configured to match memory access addresses by clearing the INST bit.

Each comparator match can force a transition to another state sequencer state (see Section 6.4.3, “Events”).

Once a successful comparator match has occurred, the condition that caused the original match is not verified again on subsequent matches. Thus if a particular data value is matched at a given address, this address may not contain that data value when a subsequent match occurs.

Match[0, 1, 3] map directly to Comparators [A, B, D] respectively, except in range modes (see Section 6.3.2.2, “Debug Control Register2 (DBGC2)”). Comparator priority rules are described in the event priority section (Section 6.4.3.4, “Event Priorities”).

6.4.2.1 Exact Address Comparator Match

With range comparisons disabled, the match condition is an exact equivalence of address bus with the value stored in the comparator address registers. Qualification of the type of access (R/W) is also possible.

Code may contain various access forms of the same address, for example a 16-bit access of ADDR[n] or byte access of ADDR[n+1] both access n+1. The comparators ensure that any access of the address defined by the comparator address register generates a match, as shown in the example of Table 6-27. Thus if the comparator address register contains ADDR[n+1] any access of ADDR[n+1] matches. This means that a 16-bit access of ADDR[n] or 32-bit access of ADDR[n-1] also match because they also access ADDR[n+1]. The right hand columns show the contents of DBGxA that would match for each access.

Table 6-27. Comparator Address Bus Matches

Access	Address	ADDR[n]	ADDR[n+1]	ADDR[n+2]	ADDR[n+3]
32-bit	ADDR[n]	Match	Match	Match	Match
16-bit	ADDR[n]	Match	Match	No Match	No Match
16-bit	ADDR[n+1]	No Match	Match	Match	No Match
8-bit	ADDR[n]	Match	No Match	No Match	No Match

If the comparator INST bit is set, the comparator address register contents are compared with the PC, the data register contents and access type bits are ignored. The comparator address register must be loaded with the address of the first opcode byte.

6.4.2.2 Address and Data Comparator Match

Comparator A features data comparators, for data access comparisons. The comparators do not evaluate if accessed data is valid. Accesses across aligned 32-bit boundaries are split internally into consecutive accesses. The data comparator mapping to accessed addresses for the CPU is shown in Table 6-28, whereby the Address column refers to the lowest 2 bits of the lowest accessed address. This corresponds to the most significant data byte.

Table 6-28. Comparator Data Byte Alignment

Address[1:0]	Data Comparator
00	DBGxD0
01	DBGxD1
10	DBGxD2
11	DBGxD3

The fixed mapping of data comparator bytes to addresses within a 32-bit data field ensures data matches independent of access size. To compare a single data byte within the 32-bit field, the other bytes within that field must be masked using the corresponding data mask registers. This ensures that any access of that byte (32-bit,16-bit or 8-bit) with matching data causes a match. If no bytes are masked then the data comparator always compares all 32-bits and can only generate a match on a 32-bit access with correct 32-bit data value. In this case, 8-bit or 16-bit accesses within the 32-bit field cannot generate a match even if the contents of the addressed bytes match because all 32-bits must match. In Table 6-29 the Access Address column refers to the address bits[1:0] of the lowest accessed address (most significant data byte).

Table 6-29. Data Register Use Dependency On CPU Access Type

			Memory Address[2:0]						
Case	**Access Address**	**Access Size**	**000**	**001**	**010**	**011**	**100**	**101**	**110**
1	00	32-bit	DBGxD0	DBGxD1	DBGxD2	DBGxD3	▨	▨	▨
2	01	32-bit	▨	DBGxD1	DBGxD2	DBGxD3	DBGxD0	▨	▨
3	10	32-bit	▨	▨	DBGxD2	DBGxD3	DBGxD0	DBGxD1	▨
4	11	32-bit	▨	▨	▨	DBGxD3	DBGxD0	DBGxD1	DBGxD2
5	00	16-bit	DBGxD0	DBGxD1	▨	▨	▨	▨	▨
6	01	16-bit	▨	DBGxD1	DBGxD2	▨	▨	▨	▨
7	10	16-bit	▨	▨	DBGxD2	DBGxD3	▨	▨	▨
8	11	16-bit	▨	▨	▨	DBGxD3	DBGxD0	▨	▨
9	00	8-bit	DBGxD0	▨	▨	▨	▨	▨	▨
10	01	8-bit	▨	DBGxD1	▨	▨	▨	▨	▨
11	10	8-bit	▨	▨	DBGxD2	▨	▨	▨	▨
12	11	8-bit	▨	▨	▨	DBGxD3	▨	▨	▨
13	00	8-bit	▨	▨	▨	▨	DBGxD0	▨	▨
			▨	Denotes byte that is not accessed.					

For a match of a 32-bit access with data compare, the address comparator must be loaded with the address of the lowest accessed byte. For Case1 Table 6-29 this corresponds to 000, for Case2 it corresponds to 001. To compare all 32-bits, it is required that no bits are masked.

6.4.2.3 Data Bus Comparison NDB Dependency

The NDB control bit allows data bus comparators to be configured to either match on equivalence or on difference. This allows monitoring of a difference in the contents of an address location from an expected value.

When matching on an equivalence (NDB=0), each individual data bus bit position can be masked out by clearing the corresponding mask bit, so that it is ignored in the comparison. A match occurs when all data bus bits with corresponding mask bits set are equivalent. If all mask register bits are clear, then a match is based on the address bus only, the data bus is ignored.

When matching on a difference, mask bits can be cleared to ignore bit positions. A match occurs when any data bus bit with corresponding mask bit set is different. Clearing all mask bits, causes all bits to be ignored and prevents a match because no difference can be detected. In this case address bus equivalence does not cause a match. Bytes that are not accessed are ignored. Thus when monitoring a multi byte field for a difference, partial accesses of the field only return a match if a difference is detected in the accessed bytes.

Table 6-30. NDB and MASK bit dependency

NDB	DBGADM	Comment
0	0	Do not compare data bus bit.
0	1	Compare data bus bit. Match on equivalence.
1	0	Do not compare data bus bit.
1	1	Compare data bus bit. Match on difference.

6.4.2.4 Range Comparisons

Range comparisons are accurate to byte boundaries. Thus for data access comparisons a match occurs if at least one byte of the access is in the range (inside range) or outside the range (outside range). For opcode comparisons only the address of the first opcode byte is compared with the range.

When using the AB comparator pair for a range comparison, the data bus can be used for qualification by using the comparator A data and data mask registers. The DBGACTL RW and RWE bits can be used to qualify the range comparison on either a read or a write access. The corresponding DBGBCTL bits are ignored. The DBGACTL COMPE/INST bits are used for range comparisons. The DBGBCTL COMPE/INST bits are ignored in range modes.

6.4.2.4.1 Inside Range (CompA_Addr ≤ address ≤ CompB_Addr)

In the Inside Range comparator mode, comparator pair A and B can be configured for range comparisons by the control register (DBGC2). The match condition requires a simultaneous valid match for both comparators. A match condition on only one comparator is not valid.

6.4.2.4.2 Outside Range (address < CompA_Addr or address > CompB_Addr)

In the Outside Range comparator mode, comparator pair A and B can be configured for range comparisons. A single match condition on either of the comparators is recognized as valid. Outside range mode in combination with opcode address matches can be used to detect if opcodes are from an unexpected range.

NOTE

When configured for data access matches, an outside range match would typically occur at any interrupt vector fetch or register access. This can be avoided by setting the upper or lower range limit to $FFFFFF or $000000 respectively. Interrupt vector fetches do not cause opcode address matches.

6.4.3 Events

Events are used as qualifiers for a state sequencer change of state. The state control register for the current state determines the next state for each event. An event can immediately initiate a transition to the next state sequencer state whereby the corresponding flag in DBGSR is set.

6.4.3.1 Comparator Match Events

6.4.3.1.1 Opcode Address Comparator Match

The comparator is loaded with the address of the selected instruction and the comparator control register INST bit is set. When the opcode reaches the execution stage of the instruction queue a match occurs just before the instruction executes, allowing a breakpoint immediately before the instruction boundary. The comparator address register must contain the address of the first opcode byte for the match to occur. Opcode address matches are data independent thus the RWE and RW bits are ignored. CPU compares are disabled when BDM becomes active.

6.4.3.1.2 Data Access Comparator Match

Data access matches are generated when an access occurs at the address contained in the comparator address register. The match can be qualified by the access data and by the access type (read/write). The breakpoint occurs a maximum of 2 instructions after the access in the CPU flow. Note, if a COF occurs between access and breakpoint, the opcode address of the breakpoint can be elsewhere in the memory map.

Opcode fetches are not classed as data accesses. Thus data access matches are not possible on opcode fetches.

6.4.3.2 External Event

The DBGEEV input signal can force a state sequencer transition, independent of internal comparator matches. The DBGEEV is an input signal mapped directly to a device pin and configured by the EEVE field in DBGC1. The external events can change the state sequencer state.

If configured to change the state sequencer state, then the external match is mapped to DBGSCRx bits C3SC[1:0]. The DBGEFR bit EEVF is set when an external event occurs.

6.4.3.3 Setting The TRIG Bit

Independent of comparator matches it is possible to initiate a breakpoint by writing the TRIG bit in DBGC1 to a logic “1”. This forces the state sequencer into the Final State. the transition to Final State is followed immediately by a transition to State0.

Breakpoints, if enabled, are issued on the transition to State0.

6.4.3.4 Event Priorities

If simultaneous events occur, the priority is resolved according to Table 6-31. Lower priority events are suppressed. It is thus possible to miss a lower priority event if it occurs simultaneously with an event of a higher priority. The event priorities dictate that in the case of simultaneous matches, the match on the higher comparator channel number (3,1,0) has priority.

If a write access to DBGC1 with the ARM bit position set occurs simultaneously to a hardware disarm from an internal event, then the ARM bit is cleared due to the hardware disarm.

Table 6-31. Event Priorities

Priority	Source	Action
Highest	TRIG	Force immediately to final state
	DBGEEV	Force to next state as defined by state control registers (EEVE=2’b10)
	Match3	Force to next state as defined by state control registers
	Match1	Force to next state as defined by state control registers
Lowest	Match0	Force to next state as defined by state control registers

6.4.4 State Sequence Control

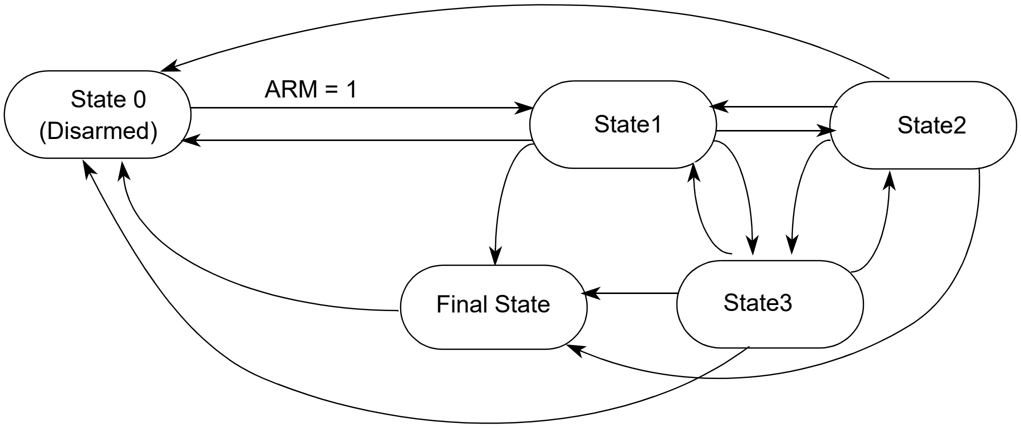


Figure 6-19. State Sequencer Diagram

The state sequencer allows a defined sequence of events to provide a breakpoint. When the DBG module is armed by setting the ARM bit in the DBGC1 register, the state sequencer enters State1. Further transitions between the states are controlled by the state control registers and depend upon event occurrences (see Section 6.4.3, “Events). From Final State the only permitted transition is back to the disarmed State0. Transition between the states 1 to 3 is not restricted. Each transition updates the SSF[2:0] flags in DBGSR accordingly to indicate the current state. If breakpoints are enabled, then an event based transition to State0 generates the breakpoint request. A transition to State0 resulting from writing “0” to the ARM bit does not generate a breakpoint request.

6.4.4.1 Final State

When the Final State is reached the state sequencer returns to State0 immediately and the debug module is disarmed.If breakpoints are enabled, a breakpoint request is generated on transitions to State0.

6.4.5 Breakpoints

Breakpoints can be generated by state sequencer transitions to State0. Transitions to State0 are forced by the following events

- Through comparator matches via Final State.
- Through software writing to the TRIG bit in the DBGC1 register via Final State.
- Through the external event input (DBGEEV) via Final State.

Breakpoints are not generated by software writes to DBGC1 that clear the ARM bit.

6.4.5.1 Breakpoints From Comparator Matches or External Events

Breakpoints can be generated when the state sequencer transitions to State0 following a comparator match or an external event.

6.4.5.2 Breakpoints Generated Via The TRIG Bit

When TRIG is written to “1”, the Final State is entered. In the next cycle TRIG breakpoints are possible even if the DBG module is disarmed.

6.4.5.3 DBG Breakpoint Priorities

6.4.5.3.1 DBG Breakpoint Priorities And BDC Interfacing

Breakpoint operation is dependent on the state of the S12ZBDC module. BDM cannot be entered from a breakpoint unless the BDC is enabled (ENBDC bit is set in the BDC). If BDM is already active, breakpoints are disabled. In addition, while executing a BDC STEP1 command, breakpoints are disabled.

When the DBG breakpoints are mapped to BDM (BDMBP set), then if a breakpoint request, either from a BDC BACKGROUND command or a DBG event, coincides with an SWI instruction in application code, (i.e. the DBG requests a breakpoint at the next instruction boundary and the next instruction is an SWI) then the CPU gives priority to the BDM request over the SWI request.

On returning from BDM, the SWI from user code gets executed. Breakpoint generation control is summarized in Table 6-32.

Table 6-32. Breakpoint Mapping Summary

BRKCPU	BDMBP Bit (DBGC1[4])	BDC Enabled	BDM Active	Breakpoint Mapping
0	X	X	X	No Breakpoint
1	0	X	0	Breakpoint to SWI
1	0	1	1	No Breakpoint
1	1	0	X	No Breakpoint
1	1	1	0	Breakpoint to BDM
1	1	1	1	No Breakpoint

6.5 Application Information

6.5.1 Avoiding Unintended Breakpoint Re-triggering

Returning from an instruction address breakpoint using an RTI or BDC GO command without PC modification, returns to the instruction that generated the breakpoint. If an active breakpoint or trigger still exists at that address, this can re-trigger, disarming the DBG. If configured for BDM breakpoints, the user must apply the BDC STEP1 command to increment the PC past the current instruction.

If configured for SWI breakpoints, the DBG can be re configured in the SWI routine. If a comparator match occurs at an SWI vector address then a code SWI and DBG breakpoint SWI could occur simultaneously. In this case the SWI routine is executed twice before returning.

6.5.2 Breakpoints from other S12Z sources

The DBG is neither affected by CPU BGND instructions, nor by BDC BACKGROUND commands.

Chapter 7
ECC Generation Module (SRAM_ECCV1)

7.1 Introduction

The purpose of ECC logic is to detect and correct as much as possible memory data bit errors. These soft errors, mainly generated by alpha radiation, can occur randomly during operation. "Soft error" means that only the information inside the memory cell is corrupt; the memory cell itself is not damaged. A write access with correct data solves the issue. If the ECC algorithm is able to correct the data, then the system can use this corrected data without any issues. If the ECC algorithm is able to detect, but not correct the error, then the system is able to ignore the memory read data to avoid system malfunction.

The ECC value is calculated based on an aligned 2 byte memory data word. The ECC algorithm is able to detect and correct single bit ECC errors. Double bit ECC errors will be detected but the system is not able to correct these errors. This kind of ECC code is called SECDED code. This ECC code requires 6 additional parity bits for each 2 byte data word.

7.1.1 Features

The SRAM_ECC module provides the ECC logic for the system memory based on a SECDED algorithm. The SRAM_ECC module includes the following features:

- SECDED ECC code
  - Single bit error detection and correction per 2 byte data word
  - Double bit error detection per 2 byte data word
- Memory initialization function
- Byte wide system memory write access
- Automatic single bit ECC error correction for read and write accesses
- Debug logic to read and write raw use data and ECC values

7.2 Memory Map and Register Definition

This section provides a detailed description of all memory and registers for the SRAM_ECC module.

7.2.1 Register Summary

Figure 7-1 shows the summary of all implemented registers inside the SRAM_ECC module.

NOTE

Register Address = Module Base Address + Address Offset, where the Module Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 ECCSTAT	R	0	0	0	0	0	0	0	RDY
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0001 ECCIE	R	0	0	0	0	0	0	0	SBEEIE
	W	▨	▨	▨	▨	▨	▨	▨	
0x0002 ECCIF	R	0	0	0	0	0	0	0	SBEEIF
	W	▨	▨	▨	▨	▨	▨	▨	
0x0003 - 0x0006 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0007 ECCDPTRH	R	DPTR[23:16]							
	W								
0x0008 ECCDPTRM	R	DPTR[15:8]							
	W								
0x0009 ECCDPTRL	R	DPTR[7:1]							0
	W								▨
0x000A - 0x000B Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000C ECCDDH	R	DDATA[15:8]							
	W								
0x000D ECCDDL	R	DDATA[7:0]							
	W								
0x000E ECCDE	R	0	0	DECC[5:0]					
	W	▨	▨						
0x000F ECCDCMD	R	ECCDRR	0	0	0	0	0	ECCDW	ECCDR
	W		▨	▨	▨	▨	▨		

▨ = Unimplemented, Reserved, Read as zero

Figure 7-1. SRAM_ECC Register Summary

7.2.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field functions follow the register diagrams, in bit order.

7.2.2.1 ECC Status Register (ECCSTAT)

Module Base + 0x00000

Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	RDY
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime
Write: Never

Figure 7-2. ECC Status Register (ECCSTAT)

Table 7-2. ECCSTAT Field Description

Field	Description
0 RDY	**ECC Ready**— Shows the status of the ECC module. 0 Internal SRAM initialization is ongoing, access to the SRAM is disabled 1 Internal SRAM initialization is done, access to the SRAM is enabled

7.2.2.2 ECC Interrupt Enable Register (ECCIE)

Module Base + 0x00001

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	SBEEIE
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime
Write: Anytime

Figure 7-3. ECC Interrupt Enable Register (ECCIE)

Table 7-3. ECCIE Field Description

Field	Description
0 SBEEIE	**Single bit ECC Error Interrupt Enable** — Enables Single ECC Error interrupt. 0 Interrupt request is disabled 1 Interrupt will be requested whenever SBEEIF is set

7.2.2.3 ECC Interrupt Flag Register (ECCIF)

Module Base + 0x0002

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	SBEEIF
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime
Write: Anytime, write 1 to clear

Figure 7-4. ECC Interrupt Flag Register (ECCIF)

Table 7-4. ECCIF Field Description

Field	Description
0 SBEEIF	**Single bit ECC Error Interrupt Flag** — The flag is set to 1 when a single bit ECC error occurs. 0 No occurrences of single bit ECC error since the last clearing of the flag 1 Single bit ECC error has occured since the last clearing of the flag

7.2.2.4 ECC Debug Pointer Register (ECCDPTRH, ECCDPTRM, ECCDPTRL)

Module Base + 0x0007 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	DPTR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x0008 Access: User read/write

	7	6	5	4	3	2	1	0
R	DPTR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x0009 Access: User read/write

	7	6	5	4	3	2	1	0
R	DPTR[7:1]							0
W								▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 7-5. ECC Debug Pointer Register (ECCDPTRH, ECCDPTRM, ECCDPTRL)

[1] Read: Anytime
Write: Anytime

Table 7-5. ECCDPTR Register Field Descriptions

Field	Description
DPTR [23:0]	**ECC Debug Pointer** — This register contains the system memory address which will be used for a debug access. Address bits not relevant for SRAM address space are not writeable, so the software should read back the pointer value to make sure the register contains the intended memory address. It is possible to write an address value to this register which points outside the system memory. There is no additional monitoring of the register content; therefore, the software must make sure that the address value points to the system memory space.

7.2.2.5 ECC Debug Data (ECCDDH, ECCDDL)

Module Base + 0x000C Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	DDATA[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x000D Access: User read/write

	7	6	5	4	3	2	1	0
R	DDATA[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 7-6. ECC Debug Data (ECCDDH, ECCDDL)

[1] Read: Anytime
Write: Anytime

Table 7-6. ECCDD Register Field Descriptions

Field	Description
DDATA [23:0]	**ECC Debug Raw Data** — This register contains the raw data which will be written into the system memory during a debug write command or the read data from the debug read command.

7.2.2.6 ECC Debug ECC (ECCDE)

Module Base + 0x000E Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	DECC[5:0]					
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime
Write: Anytime

Figure 7-7. ECC Debug ECC (ECCDE)

Table 7-7. ECCDE Field Description

Field	Description
5:0 DECC[5:0]	**ECC Debug ECC** — This register contains the raw ECC value which will be written into the system memory during a debug write command or the ECC read value from the debug read command.

7.2.2.7 ECC Debug Command (ECCDCMD)

Module Base + 0x000F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	ECCDRR	0	0	0	0	0	ECCDW	ECCDR
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime
Write: Anytime, in special mode only

Figure 7-8. ECC Debug Command (ECCDCMD)

Table 7-8. ECCDCMD Field Description

Field	Description
7 ECCDRR	**ECC Disable Read Repair Function**— Writing one to this register bit will disable the automatic single bit ECC error repair function during read access; see also chapter 7.3.7, “ECC Debug Behavior””. 0 Automatic single ECC error repair function is enabled 1 Automatic single ECC error repair function is disabled
1 ECCDW	**ECC Debug Write Command** — Writing one to this register bit will perform a debug write access, to the system memory. During this access the debug data word (DDATA) and the debug ECC value (DECC) will be written to the system memory address defined by DPTR. If the debug write access is done, this bit is cleared. Writing 0 has no effect. It is not possible to set this bit if the previous debug access is ongoing (ECCDW or ECCDR bit set).
0 ECCDR	**ECC Debug Read Command** — Writing one to this register bit will perform a debug read access from the system memory address defined by DPTR. If the debug read access is done, this bit is cleared and the raw memory read data are available in register DDATA and the raw ECC value is available in register DECC. Writing 0 has no effect. If the ECCDW and ECCDR bit are set at the same time, then only the ECCDW bit is set and the Debug Write Command is performed. It is not possible to set this bit if the previous debug access is ongoing (ECCDW or ECCDR bit set).

7.3 Functional Description

The bus system allows 1, 2, 3 and 4 byte write access to a 4 byte aligned memory address, but the ECC value is generated based on an aligned 2 byte data word. Depending on the access type, the access is separated into different access cycles. Table 7-9 shows the different access types with the expected number of access cycles and the performed internal operations.

Table 7-9. Memory access cycles

Access type	ECC error	access cycle	Internal operation	Memory content	Error indication
2 and 4 byte aligned write access	—	1	write to memory	new data	—

Table 7-9. Memory access cycles

Access type	ECC error	access cycle	Internal operation	Memory content	Error indication
1 or 3 byte write, non-aligned 2 byte write	no	2	read data from the memory	old + new data	—
			write old + new data to the memory		
	single bit	2	read data from the memory	corrected + new data	SBEEIF
			write corrected + new data to the memory		
	double bit	2	read data from the memory	unchanged	initiator module is informed
			ignore write data		
read access	no	1	read from memory	unchanged	-
	single bit	1[1]	read data from the memory	corrected data	SBEEIF
			write corrected data back to memory		
	double bit	1	read from memory	unchanged	data mark as invalid

[1] The next back to back read access to the memory will be delayed by one clock cycle

The single bit ECC error generates an interrupt when enabled. The double bit ECC errors are reported by the SRAM_ECC module, but handled at MCU level. For more information, see the MMC description.

7.3.1 Aligned 2 and 4 Byte Memory Write Access

During an aligned 2 or 4 byte memory write access, no ECC check is performed. The internal ECC logic generates the new ECC value based on the write data and writes the data words together with the generated ECC values into the memory.

7.3.2 Other Memory Write Access

Other types of write accesses are separated into a read-modify-write operation. During the first cycle, the logic reads the data from the memory and performs an ECC check. If no ECC errors were detected then the logic generates the new ECC value based on the read and write data and writes the new data word together with the new ECC value into the memory. If required both 2 byte data words are updated.

If the module detects a single bit ECC error during the read cycle, then the logic generates the new ECC value based on the corrected read and new write read. In the next cycle, the new data word and the new ECC value are written into the memory. If required both 2 byte data words are updated. The SBEEIF bit is set. Hence, the single bit ECC error was corrected by the write access. Figure 7-9 shows an example of a 2 byte non-aligned memory write access.

If the module detects a double bit ECC error during the read cycle, then the write access to the memory is blocked and the initiator module is informed about the error.

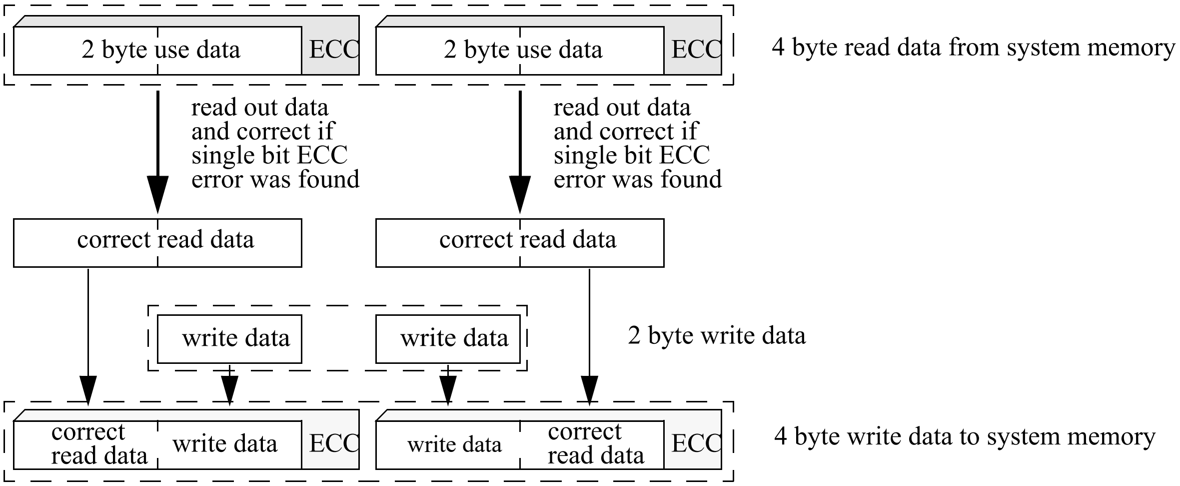


Figure 7-9. 2 byte non-aligned write access

7.3.3 Memory Read Access

During each memory read access an ECC check is performed. If the logic detects a single bit ECC error, then the module corrects the data, so that the access initiator module receives correct data. In parallel, the logic writes the corrected data back to the memory, so that this read access repairs the single bit ECC error. This automatic ECC read repair function is disabled by setting the ECCDRR bit.

If a single bit ECC error was detected, then the SBEEIF flag is set.

If the logic detects a double bit ECC error, then the data word is flagged as invalid, so that the access initiator module can ignore the data.

7.3.4 Memory Initialization

To avoid spurious ECC error reporting, memory operations that allow a read before a first write (like the read-modify-write operation of the unaligned access) require that the memory contains valid ECC values before the first read-modify-write access is performed. The ECC module provides logic to initialize the complete memory content with zero during the power up phase. During the initialization process the access to the SRAM is disabled and the RDY status bit is cleared. If the initialization process is done, SRAM access is possible and the RDY status bit is set.

7.3.5 Interrupt Handling

This section describes the interrupts generated by the SRAM_ECC module and their individual sources. Vector addresses and interrupt priority are defined at the MCU level.

Table 7-10. SRAM_ECC Interrupt Sources

Module Interrupt Sources	Local Enable
Single bit ECC error	ECCIE[SBEEIE]

7.3.6 ECC Algorithm

The table below shows the equation for each ECC bit based on the 16 bit data word.

Table 7-11. ECC Calculation

ECC bit	Use data
ECC[0]	~ (^ (data[15:0] & 0x443F))
ECC[1]	~ (^ (data[15:0] & 0x13C7))
ECC[2]	~ (^ (data[15:0] & 0xE1D1))
ECC[3]	~ (^ (data[15:0] & 0xEE60))
ECC[4]	~ (^ (data[15:0] & 0x3E8A))
ECC[5]	~ (^ (data[15:0] & 0x993C))

7.3.7 ECC Debug Behavior

For debug purposes, it is possible to read and write the uncorrected use data and the raw ECC value directly from the memory. For these debug accesses a register interface is available. The debug access is performed with the lowest priority; other memory accesses must be done before the debug access starts. If a debug access is requested during an ongoing memory initialization process, then the debug access is performed if the memory initialization process is done.

If the ECCDRR bit is set, then the automatic single bit ECC error repair function for all read accesses is disabled. In this case a read access from a system memory location with single bit ECC error will produce correct data and the single bit ECC error is flagged by the SBEEIF, but the data inside the system memory are unchanged.

By writing wrong ECC values into the system memory the debug access can be used to force single and double bit ECC errors to check the software error handling.

It is not possible to set the ECCDW or ECCDR bit if the previous debug access is ongoing (ECCDW or ECCDR bit active). This ensures that the ECCDD and ECCDE registers contains consistent data. The software should read out the status of the ECCDW and ECCDR register bit before a new debug access is requested.

7.3.7.1 ECC Debug Memory Write Access

Writing one to the ECCDW bit performs a debug write access to the memory address defined by register DPTR. During this access, the raw data DDATA and the ECC value DECC are written directly into the system memory. If the debug write access is done, the ECCDW register bit is cleared. The debug write access is always a 2 byte aligned memory access, so that no ECC check is performed and no single or double bit ECC error indication is activated.

7.3.7.2 ECC Debug Memory Read Access

Writing one to the ECCDR bit performs a debug read access from the memory address defined by register DPTR. If the ECCDR bit is cleared then the register DDATA contains the uncorrected read data from the memory. The register DECC contains the ECC value read from the memory. Independent of the ECCDRR register bit setting, the debug read access will not perform an automatic ECC repair during read access. During the debug read access no ECC check is performed, so that no single or double bit ECC error indication is activated.

If the ECCDW and the ECCDR bits are set at the same time, then only the debug write access is performed.

Chapter 8
S12 Clock, Reset and Power Management Unit (S12CPMU_UHV_V7)

Revision History

Rev. No. (Item No)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V07.00	6 March 2013		• copied from V5 • adapted for Hearst: added VDDC, added EXTCON Bit
V07.01	13 June 2013		• EXTCON register Bit: correct reset value to 1 • PMRF register Bit: corrected description
V07.02	21 Aug. 2013		• correct bit numbering for CSAD Bit • f_{PLLRST} changed to f_{VCORST} • changed frequency upper limit of external Pierce Oscillator (XOSCLCP) from 16MHz to 20MHz • corrected typo in heading of CPMUOSC2 Field Description • Memory Map, CPMUAPIRH register: corrected address typo

8.1 Introduction

This specification describes the function of the Clock, Reset and Power Management Unit (S12CPMU_UHV_V7).

- The Pierce oscillator (XOSCLCP) provides a robust, low-noise and low-power external clock source. It is designed for optimal start-up margin with typical crystal oscillators.
- The Voltage regulator (VREGAUTO) operates from the range 6V to 18V. It provides all the required chip internal voltages and voltage monitors.
- The Phase Locked Loop (PLL) provides a highly accurate frequency multiplier with internal filter.
- The Internal Reference Clock (IRC1M) provides a 1MHz internal clock.

8.1.1 Features

The Pierce Oscillator (XOSCLCP) contains circuitry to dynamically control current gain in the output amplitude. This ensures a signal with low harmonic distortion, low power and good noise immunity.

- Supports crystals or resonators from 4MHz to 20MHz.
- High noise immunity due to input hysteresis and spike filtering.
- Low RF emissions with peak-to-peak swing limited dynamically
- Transconductance (gm) sized for optimum start-up margin for typical crystals
- Dynamic gain control eliminates the need for external current limiting resistor
- Integrated resistor eliminates the need for external bias resistor
- Low power consumption: Operates from internal 1.8V (nominal) supply, Amplitude control limits power
- Optional oscillator clock monitor reset
- Optional full swing mode for higher immunity against noise injection on the cost of higher power consumption and increased emission

The Voltage Regulator (VREGAUTO) has the following features:

- Input voltage range from 6 to 18V (nominal operating range)
- Low-voltage detect (LVD) with low-voltage interrupt (LVI)
- Power-on reset (POR)
- Low-voltage reset (LVR)
- On Chip Temperature Sensor and Bandgap Voltage measurement via internal ADC channel.
- Voltage Regulator providing Full Performance Mode (FPM) and Reduced Performance Mode (RPM)
- External ballast device support to reduce internal power dissipation
- Capable of supplying both the MCU internally plus external components
- Over-temperature interrupt

The Phase Locked Loop (PLL) has the following features:

- Highly accurate and phase locked frequency multiplier
- Configurable internal filter for best stability and lock time
- Frequency modulation for defined jitter and reduced emission
- Automatic frequency lock detector
- Interrupt request on entry or exit from locked condition
- PLL clock monitor reset
- Reference clock either external (crystal) or internal square wave (1MHz IRC1M) based.
- PLL stability is sufficient for LIN communication in slave mode, even if using IRC1M as reference clock

The Internal Reference Clock (IRC1M) has the following features:

- Frequency trimming
  (A factory trim value for 1MHz is loaded from Flash Memory into the IRCTRIM register after reset, which can be overwritten by application if required)
- Temperature Coefficient (TC) trimming.
  (A factory trim value is loaded from Flash Memory into the IRCTRIM register to turn off TC trimming after reset. Application can trim the TC if required by overwriting the IRCTRIM register).

Other features of the S12CPMU_UHV_V7 include

- Oscillator clock monitor to detect loss of crystal
- Autonomous periodical interrupt (API)
- Bus Clock Generator
  - Clock switch to select either PLLCLK or external crystal/resonator as source of the Bus Clock
  - PLLCLK divider to adjust system speed
- System Reset generation from the following possible sources:
  - Power-on reset (POR)
  - Low-voltage reset (LVR)
  - COP system watchdog, COP reset on time-out, windowed COP
  - Loss of oscillation (Oscillator clock monitor fail)
  - Loss of PLL clock (PLL clock monitor fail)
  - External pin $\overline{\text{RESET}}$

8.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the S12CPMU_UHV_V7.

8.1.2.1 Run Mode

The voltage regulator is in Full Performance Mode (FPM).

NOTE

> The voltage regulator is active, providing the nominal supply voltages with full current sourcing capability (see also Appendix for VREG electrical parameters). The features ACLK clock source, Low Voltage Interrupt (LVI), Low Voltage Reset (LVR) and Power-On Reset (POR) are available.

The Phase Locked Loop (PLL) is on.

The Internal Reference Clock (IRC1M) is on.

The API is available.

- **PLL Engaged Internal (PEI)**
  - This is the default mode after System Reset and Power-On Reset.
  - The Bus Clock is based on the PLLCLK.
  - After reset the PLL is configured for 50MHz VCOCLK operation.
    Post divider is 0x03, so PLLCLK is VCOCLK divided by 4, that is 12.5MHz and Bus Clock is 6.25MHz.
    The PLL can be re-configured for other bus frequencies.
  - The reference clock for the PLL (REFCLK) is based on internal reference clock IRC1M.
- **PLL Engaged External (PEE)**
  - The Bus Clock is based on the PLLCLK.
  - This mode can be entered from default mode PEI by performing the following steps:
    - Configure the PLL for desired bus frequency.
    - Program the reference divider (REFDIV[3:0] bits) to divide down oscillator frequency if necessary.
    - Enable the external oscillator (OSCE bit).
    - Wait for oscillator to start up (UPOSC=1) and PLL to lock (LOCK=1).
- **PLL Bypassed External (PBE)**
  - The Bus Clock is based on the Oscillator Clock (OSCCLK).
  - The PLLCLK is always on to qualify the external oscillator clock. Therefore it is necessary to make sure a valid PLL configuration is used for the selected oscillator frequency.
  - This mode can be entered from default mode PEI by performing the following steps:
    - Make sure the PLL configuration is valid for the selected oscillator frequency.

  - Enable the external oscillator (OSCE bit).
  - Wait for oscillator to start up (UPOSC=1).
  - Select the Oscillator Clock (OSCCLK) as source of the Bus Clock (PLLSEL=0).
- The PLLCLK is on and used to qualify the external oscillator clock.

8.1.2.2 Wait Mode

For S12CPMU_UHV_V7 Wait Mode is the same as Run Mode.

8.1.2.3 Stop Mode

This mode is entered by executing the CPU STOP instruction.

The voltage regulator is in Reduced Performance Mode (RPM).

NOTE

> The voltage regulator output voltage may degrade to a lower value than in Full Performance Mode (FPM), additionally the current sourcing capability is substantially reduced (see also Appendix for VREG electrical parameters). Only clock source ACLK is available and the Power On Reset (POR) circuitry is functional. The Low Voltage Interrupt (LVI) and Low Voltage Reset (LVR) are disabled.

The API is available.

The Phase Locked Loop (PLL) is off.

The Internal Reference Clock (IRC1M) is off.

Core Clock and Bus Clock are stopped.

Depending on the setting of the PSTP and the OSCE bit, Stop Mode can be differentiated between Full Stop Mode (PSTP = 0 or OSCE=0) and Pseudo Stop Mode (PSTP = 1 and OSCE=1). In addition, the behavior of the COP in each mode will change based on the clocking method selected by COPOSCSEL[1:0].

- **Full Stop Mode (PSTP = 0 or OSCE=0)**
  External oscillator (XOSCLCP) is disabled.
  - If COPOSCSEL1=0:
    The COP and RTI counters halt during Full Stop Mode.
    After wake-up from Full Stop Mode the Core Clock and Bus Clock are running on PLLCLK (PLLSEL=1). COP and RTI are running on IRCCLK (COPOSCSEL0=0, RTIOSCSEL=0).
  - If COPOSCSEL1=1:
    The clock for the COP is derived from ACLK (trimmable internal RC-Oscillator clock). During Full Stop Mode the ACLK for the COP can be stopped (COP static) or running (COP active) depending on the setting of bit CSAD. When bit CSAD is set the ACLK clock source for the COP is stopped during Full Stop Mode and COP continues to operate after exit from Full Stop

Mode. For this COP configuration (ACLK clock source, CSAD set) a latency time occurs when entering or exiting (Full, Pseudo) Stop Mode. When bit CSAD is clear the ACLK clock source is on for the COP during Full Stop Mode and COP is operating.
During Full Stop Mode the RTI counter halts.
After wake-up from Full Stop Mode the Core Clock and Bus Clock are running on PLLCLK (PLLSEL=1). The COP runs on ACLK and RTI is running on IRCCLK (COPOSCSEL0=0, RTIOSCSEL=0).

- **Pseudo Stop Mode (PSTP = 1 and OSCE=1)**
  External oscillator (XOSCLCP) continues to run.
  - If COPOSCSEL1=0:
    If the respective enable bits are set (PCE=1 and PRE=1) the COP and RTI will continue to run with a clock derived from the oscillator clock.
    The clock configuration bits PLLSEL, COPOSCSEL0, RTIOSCSEL are unchanged.
  - If COPOSCSEL1=1:
    If the respective enable bit for the RTI is set (PRE=1) the RTI will continue to run with a clock derived from the oscillator clock.
    The clock for the COP is derived from ACLK (trimmable internal RC-Oscillator clock). During Pseudo Stop Mode the ACLK for the COP can be stopped (COP static) or running (COP active) depending on the setting of bit CSAD. When bit CSAD is set the ACLK for the COP is stopped during Pseudo Stop Mode and COP continues to operate after exit from Pseudo Stop Mode.
    For this COP configuration (ACLK clock source, CSAD set) a latency time occurs when entering or exiting (Pseudo, Full) Stop Mode. When bit CSAD is clear the ACLK clock source is on for the COP during Pseudo Stop Mode and COP is operating.
    The clock configuration bits PLLSEL, COPOSCSEL0, RTIOSCSEL are unchanged.

NOTE

When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator t_{UPOSC} before entering Pseudo Stop Mode.

8.1.2.4 Freeze Mode (BDM active)

For S12CPMU_UHV_V7 Freeze Mode is the same as Run Mode except for RTI and COP which can be frozen in Active BDM Mode with the RSBCK bit in the CPMUCOP register. After exiting BDM Mode RTI and COP will resume its operations starting from this frozen status.

Additionally the COP can be forced to the maximum time-out period in Active BDM Mode. For details please see also the RSBCK and CR[2:0] bit description field of Table 8-13 in Section 8.3.2.10, “S12CPMU_UHV_V7 COP Control Register (CPMUCOP)

8.1.3 S12CPMU_UHV_V7 Block Diagram

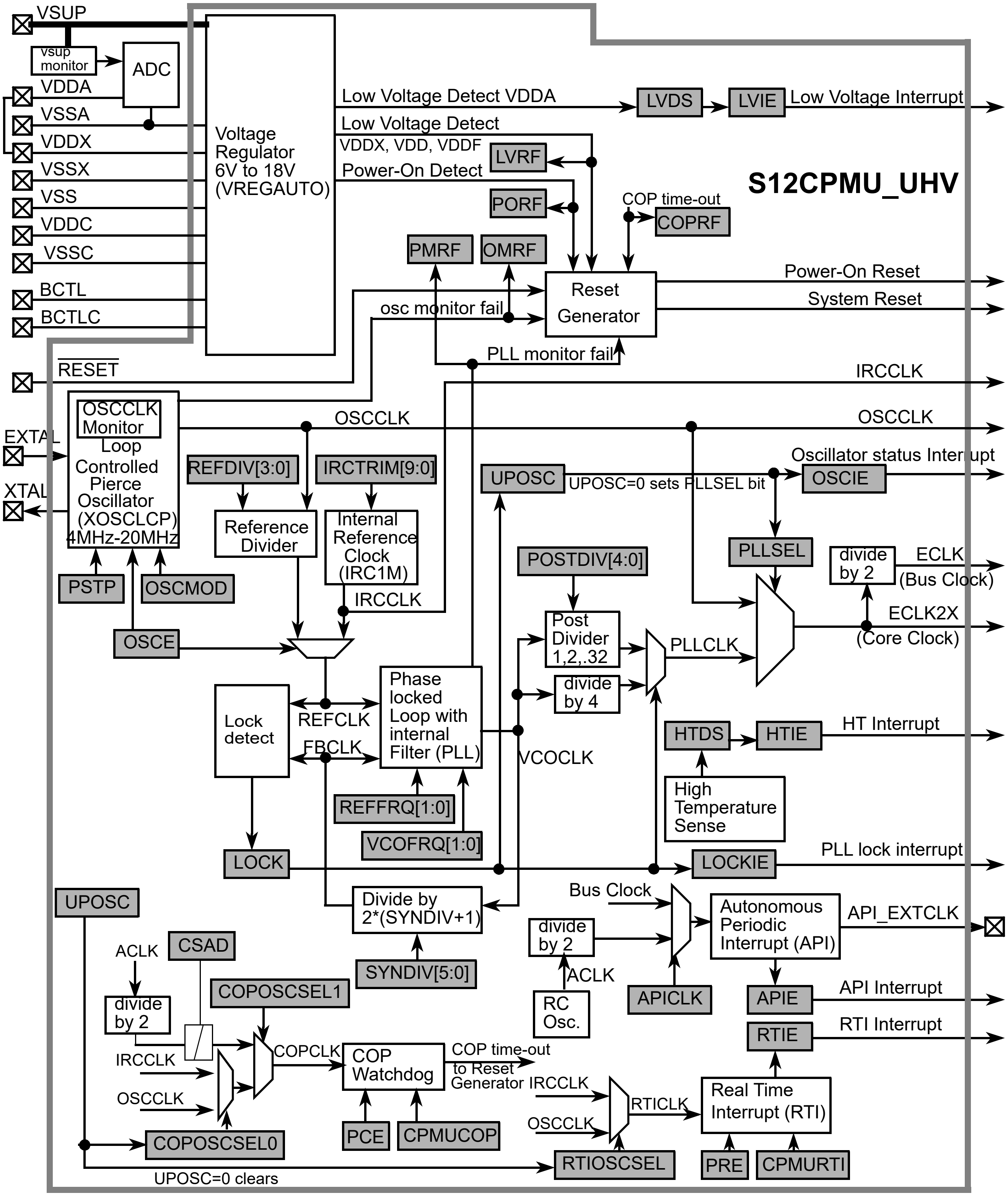


Figure 8-1. Block diagram of S12CPMU_UHV_V7

Figure 8-2 shows a block diagram of the XOSCLCP.

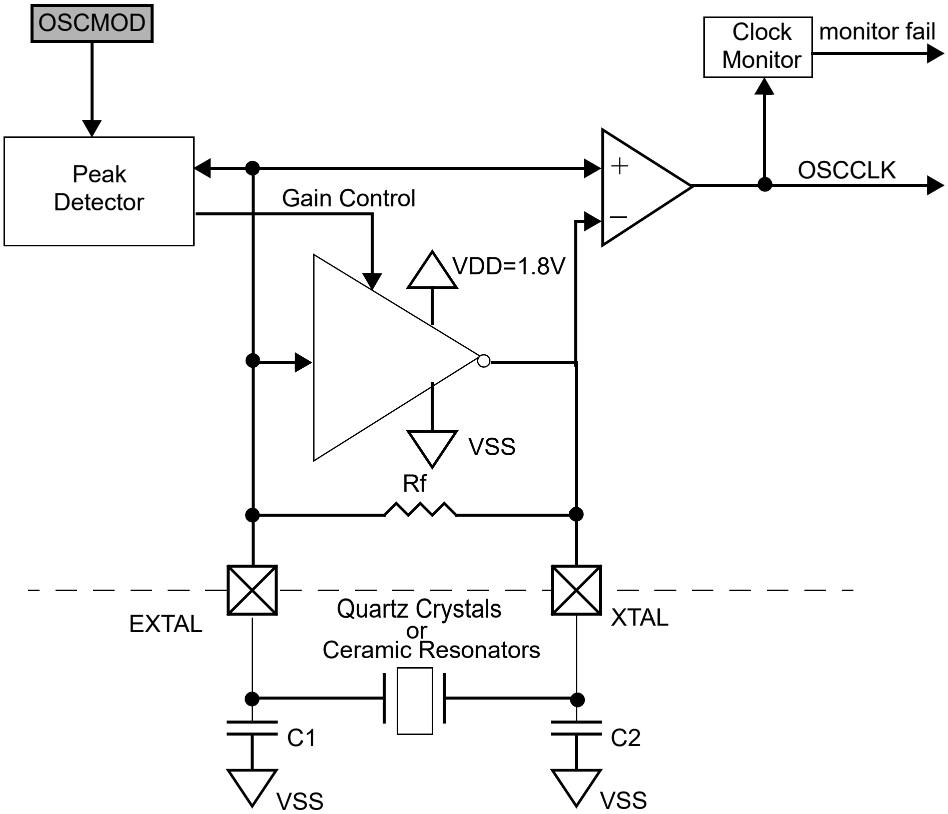


Figure 8-2. XOSCLCP Block Diagram

8.2 Signal Description

This section lists and describes the signals that connect off chip as well as internal supply nodes and special signals.

8.2.1 $\overline{\text{RESET}}$

Pin $\overline{\text{RESET}}$ is an active-low bidirectional pin. As an input it initializes the MCU asynchronously to a known start-up state. As an open-drain output it indicates that an MCU-internal reset has been triggered.

8.2.2 EXTAL and XTAL

These pins provide the interface for a crystal to control the internal clock generator circuitry. EXTAL is the input to the crystal oscillator amplifier. XTAL is the output of the crystal oscillator amplifier. If XOSCLCP is enabled, the MCU internal OSCCLK_LCP is derived from the EXTAL input frequency. If OSCE=0, the EXTAL pin is pulled down by an internal resistor of approximately 200 kΩ and the XTAL pin is pulled down by an internal resistor of approximately 700 kΩ.

NOTE

NXP recommends an evaluation of the application board and chosen resonator or crystal by the resonator or crystal supplier.
The loop controlled circuit (XOSCLCP) is not suited for overtone resonators and crystals.

8.2.3 VSUP — Regulator Power Input Pin

Pin VSUP is the power input of VREGAUTO. All currents sourced into the regulator loads flow through this pin.

A suitable reverse battery protection network can be used to connect VSUP to the car battery supply network.

8.2.4 VDDA, VSSA — Regulator Reference Supply Pins

Pins VDDA and VSSA, are used to supply the analog parts of the regulator. Internal precision reference circuits are supplied from these signals.

An off-chip decoupling capacitor (220 nF(X7R ceramic)) between VDDA and VSSA is required and can improve the quality of this supply.

VDDA has to be connected externally to VDDX.

8.2.5 VDDX, VSSX — Pad Supply Pins

VDDX is the supply domain for the digital Pads. VDDX has to be connected externally to VDDA.

An off-chip decoupling capacitor (10µF plus 220 nF(X7R ceramic)) between VDDX and VSSX is required.

This supply domain is monitored by the Low Voltage Reset circuit.

8.2.6 .VDDC, VSSC — CANPHY Supply Pin

VDDC is the supply domain for the CANPHY.

An off-chip decoupling capacitor (10µF plus 220 nF(X7R ceramic)) between VDDC and VSSC is required.

This supply domain is monitored by the Low Voltage Reset circuit.

8.2.7 BCTL — Base Control Pin for external PNP

BCTL is the ballast connection for the on chip voltage regulator. It provides the base current of an external

BJT (PNP) of the VDDX and VDDA supplies. An additional 1KΩ resistor between emitter and base of the BJT is required.

8.2.8 BCTLC — Base Control Pin for external PNP for VDDC

BCTLC is the ballast connection for the on chip voltage regulator. It provides the base current of an external BJT (PNP) of the VDDC supply. An additional 1KΩ resistor between emitter and base of the BJT is required.

8.2.9 VSS — Core Logic Ground Pin

VSS is the core logic supply return pins. It must be grounded.

8.2.10 VDD — Internal Regulator Output Supply (Core Logic)

Node VDD is a device internal supply output of the voltage regulator that provides the power supply for the internal core logic.

This supply domain is monitored by the Low Voltage Reset circuit and The Power On Reset circuit.

8.2.11 VDDF — Internal Regulator Output Supply (NVM Logic)

Node VDDF is a device internal supply output of the voltage regulator that provides the power supply for the NVM logic.

This supply domain is monitored by the Low Voltage Reset circuit.

8.2.12 API_EXTCLK — API external clock output pin

This pin provides the signal selected via APIES and is enabled with APIEA bit. See the device specification if this clock output is available on this device and to which pin it might be connected.

8.2.13 TEMPSENSE — Internal Temperature Sensor Output Voltage

Depending on the VSEL setting either the voltage level generated by the temperature sensor or the VREG bandgap voltage is driven to a special channel input of the ADC Converter. See device level specification for connectivity of ADC special channels.

8.3 Memory Map and Registers

This section provides a detailed description of all registers accessible in the S12CPMU_UHV_V7.

8.3.1 Module Memory Map

The S12CPMU_UHV_V7 registers are shown in Figure 8-3.

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	CPMU RESERVED00	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0001	CPMU RESERVED01	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0002	CPMU RESERVED02	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0003	CPMURFLG	R	0	PORF	LVRF	0	COPRF	0	OMRF	PMRF
		W	▨			▨		▨		
0x0004	CPMU SYNR	R	VCOFRQ[1:0]		SYNDIV[5:0]					
		W								
0x0005	CPMU REFDIV	R	REFFRQ[1:0]		0	0	REFDIV[3:0]			
		W			▨	▨				
0x0006	CPMU POSTDIV	R	0	0	0	POSTDIV[4:0]				
		W	▨	▨	▨					
0x0007	CPMUIFLG	R	RTIF	0	0	LOCKIF	LOCK	0	OSCIF	UPOSC
		W		▨	▨		▨	▨		▨
0x0008	CPMUINT	R	RTIE	0	0	LOCKIE	0	0	OSCIE	0
		W		▨	▨		▨	▨		▨
0x0009	CPMUCLKS	R	PLLSEL	PSTP	CSAD	COP OSCSEL1	PRE	PCE	RTI OSCSEL	COP OSCSEL0
		W								
0x000A	CPMUPLL	R	0	0	FM1	FM0	0	0	0	0
		W	▨	▨			▨	▨	▨	▨
0x000B	CPMURTI	R	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
		W								
0x000C	CPMUCOP	R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
		W			WRTMASK	▨	▨			
0x000D	RESERVED CPMUTEST0	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x000E	RESERVED CPMUTEST1	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 8-3. CPMU Register Summary

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000F	CPMU ARMCOP	R	0	0	0	0	0	0	0	0
		W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0010	CPMU HTCTL	R	0	0	VSEL	0	HTE	HTDS	HTIE	HTIF
		W	▨	▨		▨		▨		
0x0011	CPMU LVCTL	R	0	0	0	0	0	LVDS	LVIE	LVIF
		W	▨	▨	▨	▨	▨	▨		
0x0012	CPMU APICTL	R	APICLK	0	0	APIES	APIEA	APIFE	APIE	APIF
		W		▨	▨					
0x0013	CPMUACLKTR	R	ACLKTR5	ACLKTR4	ACLKTR3	ACLKTR2	ACLKTR1	ACLKTR0	0	0
		W							▨	▨
0x0014	CPMUAPIRH	R	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
		W								
0x0015	CPMUAPIRL	R	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
		W								
0x0016	RESERVED CPMUTEST3	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0017	CPMUHTTR	R	HTOE	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
		W		▨	▨	▨				
0x0018	CPMU IRCTRIMH	R	TCTRIM[4:0]					0	IRCTRIM[9:8]	
		W						▨		
0x0019	CPMU IRCTRIML	R	IRCTRIM[7:0]							
		W								
0x001A	CPMUOSC	R	OSCE	0	Reserved	0	0	0	0	0
		W		▨		▨	▨	▨	▨	▨
0x001B	CPMUPROT	R	0	0	0	0	0	0	0	PROT
		W	▨	▨	▨	▨	▨	▨	▨	
0x001C	RESERVED CPMUTEST2	R	0	0	0	0	0	0	0	0
		W			▨	▨	▨	▨	▨	
0x001D	CPMU VREGCTL	R	0	0	0	0	0	EXTCON	EXTXON	INTXON
		W	▨	▨	▨	▨	▨			
0x001E	CPMUOSC2	R	0	0	0	0	0	0	OMRE	OSCMOD
		W	▨	▨	▨	▨	▨	▨		
0x001F	CPMU RESERVED1F	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 8-3. CPMU Register Summary

8.3.2 Register Descriptions

This section describes all the S12CPMU_UHV_V7 registers and their individual bits.

Address order is as listed in Figure 8-3

8.3.2.1 S12CPMU_UHV_V7 Reset Flags Register (CPMURFLG)

This register provides S12CPMU_UHV_V7 reset flags.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	PORF	LVRF	0	COPRF	0	OMRF	PMRF
W	▨			▨		▨		
Reset	0	Note 1	Note 2	0	Note 3	0	Note 4	Note 5

1. PORF is set to 1 when a power on reset occurs. Unaffected by System Reset.
2. LVRF is set to 1 when a low voltage reset occurs. Unaffected by System Reset. Set by power on reset.
3. COPRF is set to 1 when COP reset occurs. Unaffected by System Reset. Cleared by power on reset.
4. OMRF is set to 1 when an oscillator clock monitor reset occurs. Unaffected by System Reset. Cleared by power on reset.
5. PMRF is set to 1 when a PLL clock monitor reset occurs. Unaffected by System Reset. Cleared by power on reset.

▨ = Unimplemented or Reserved

Figure 8-4. S12CPMU_UHV_V7 Flags Register (CPMURFLG)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 8-1. CPMURFLG Field Descriptions

Field	Description
6 PORF	**Power on Reset Flag** — PORF is set to 1 when a power on reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Power on reset has not occurred. 1 Power on reset has occurred.
5 LVRF	**Low Voltage Reset Flag** — LVRF is set to 1 when a low voltage reset occurs on the VDD, VDDF or VDDX domain. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Low voltage reset has not occurred. 1 Low voltage reset has occurred.
3 COPRF	**COP Reset Flag** — COPRF is set to 1 when a COP (Computer Operating Properly) reset occurs. Refer to 8.5.5, “Computer Operating Properly Watchdog (COP) Reset” and 8.3.2.10, “S12CPMU_UHV_V7 COP Control Register (CPMUCOP)” for details.This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 COP reset has not occurred. 1 COP reset has occurred.

Table 8-1. CPMURFLG Field Descriptions (continued)

Field	Description
1 OMRF	**Oscillator Clock Monitor Reset Flag** — OMRF is set to 1 when a loss of oscillator (crystal) clock occurs. Refer to8.5.3, “Oscillator Clock Monitor Reset” for details.This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Loss of oscillator clock reset has not occurred. 1 Loss of oscillator clock reset has occurred.
0 PMRF	**PLL Clock Monitor Reset Flag** — PMRF is set to 1 when a loss of PLL clock occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Loss of PLL clock reset has not occurred. 1 Loss of PLL clock reset has occurred.

8.3.2.2 S12CPMU_UHV_V7 Synthesizer Register (CPMUSYNR)

The CPMUSYNR register controls the multiplication factor of the PLL and selects the VCO frequency range.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	VCOFRQ[1:0]		SYNDIV[5:0]					
Reset	0	1	0	1	1	0	0	0

Figure 8-5. S12CPMU_UHV_V7 Synthesizer Register (CPMUSYNR)

Read: Anytime

Write: If PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register), then write anytime. Else write has no effect.

NOTE

Writing to this register clears the LOCK and UPOSC status bits.

If PLL has locked (LOCK=1) $$f_{VCO} = 2 \times f_{REF} \times (SYNDIV + 1)$$

NOTE

f_{VCO} must be within the specified VCO frequency lock range. Bus frequency f_{bus} must not exceed the specified maximum.

The VCOFRQ[1:0] bits are used to configure the VCO gain for optimal stability and lock time. For correct PLL operation the VCOFRQ[1:0] bits have to be selected according to the actual target VCOCLK frequency as shown in Table 8-2. Setting the VCOFRQ[1:0] bits incorrectly can result in a non functional PLL (no locking and/or insufficient stability).

Table 8-2. VCO Clock Frequency Selection

VCOCLK Frequency Ranges	VCOFRQ[1:0]
32MHz <= f_{VCO} <= 48MHz	00
48MHz < f_{VCO} <= 64MHz	01
Reserved	10
Reserved	11

8.3.2.3 S12CPMU_UHV_V7 Reference Divider Register (CPMUREFDIV)

The CPMUREFDIV register provides a finer granularity for the PLL multiplier steps when using the external oscillator as reference.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	REFFRQ[1:0]		0	0	REFDIV[3:0]			
W								
Reset	0	0	0	0	1	1	1	1

Figure 8-6. S12CPMU_UHV_V7 Reference Divider Register (CPMUREFDIV)

Read: Anytime

Write: If PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register), then write anytime. Else write has no effect.

NOTE

Write to this register clears the LOCK and UPOSC status bits.

If XOSCLCP is enabled (OSCE=1) $$f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$$

If XOSCLCP is disabled (OSCE=0) $$f_{REF} = f_{IRC1M}$$

The REFFRQ[1:0] bits are used to configure the internal PLL filter for optimal stability and lock time. For correct PLL operation the REFFRQ[1:0] bits have to be selected according to the actual REFCLK frequency as shown in Table 8-3.

If IRC1M is selected as REFCLK (OSCE=0) the PLL filter is fixed configured for the 1MHz <= f_{REF} <= 2MHz range. The bits can still be written but will have no effect on the PLL filter configuration.

For OSCE=1, setting the REFFRQ[1:0] bits incorrectly can result in a non functional PLL (no locking and/or insufficient stability).

Table 8-3. Reference Clock Frequency Selection if OSC_LCP is enabled

REFCLK Frequency Ranges (OSCE=1)	REFFRQ[1:0]
1MHz <= f_{REF} <= 2MHz	00
2MHz < f_{REF} <= 6MHz	01
6MHz < f_{REF} <= 12MHz	10
f_{REF} >12MHz	11

8.3.2.4 S12CPMU_UHV_V7 Post Divider Register (CPMUPOSTDIV)

The POSTDIV register controls the frequency ratio between the VCOCLK and the PLLCLK.

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	0	0	0	POSTDIV[4:0]				
W	▨	▨	▨					
Reset	0	0	0	0	0	0	1	1

▨ = Unimplemented or Reserved

Figure 8-7. S12CPMU_UHV_V7 Post Divider Register (CPMUPOSTDIV)

Read: Anytime

Write: If PLLSEL=1 write anytime, else write has no effect

If PLL is locked (LOCK=1) $$f_{PLL} = \frac{f_{VCO}}{(POSTDIV + 1)}$$

If PLL is not locked (LOCK=0) $$f_{PLL} = \frac{f_{VCO}}{4}$$

If PLL is selected (PLLSEL=1) $$f_{bus} = \frac{f_{PLL}}{2}$$

When changing the POSTDIV[4:0] value or PLL transitions to locked stated (lock=1), it takes up to 32 Bus Clock cycles until f_{PLL} is at the desired target frequency. This is because the post divider gradually changes (increases or decreases) f_{PLL} in order to avoid sudden load changes for the on-chip voltage regulator.

8.3.2.5 S12CPMU_UHV_V7 Interrupt Flags Register (CPMUIFLG)

This register provides S12CPMU_UHV_V7 status bits and interrupt flags.

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	RTIF	0	0	LOCKIF	LOCK	0	OSCIF	UPOSC
W		▨	▨		▨	▨		▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-8. S12CPMU_UHV_V7 Flags Register (CPMUIFLG)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 8-4. CPMUIFLG Field Descriptions

Field	Description
7 RTIF	**Real Time Interrupt Flag** — RTIF is set to 1 at the end of the RTI period. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (RTIE=1), RTIF causes an interrupt request. 0 RTI time-out has not yet occurred. 1 RTI time-out has occurred.
4 LOCKIF	**PLL Lock Interrupt Flag** — LOCKIF is set to 1 when LOCK status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (LOCKIE=1), LOCKIF causes an interrupt request. 0 No change in LOCK bit. 1 LOCK bit has changed.
3 LOCK	**Lock Status Bit** — LOCK reflects the current state of PLL lock condition. Writes have no effect. While PLL is unlocked (LOCK=0) f_{PLL} is f_{VCO} / 4 to protect the system from high core clock frequencies during the PLL stabilization time t_{lock}. 0 VCOCLK is not within the desired tolerance of the target frequency. $f_{PLL} = f_{VCO}/4$. 1 VCOCLK is within the desired tolerance of the target frequency. $f_{PLL} = f_{VCO}/(POSTDIV+1)$.
1 OSCIF	**Oscillator Interrupt Flag** — OSCIF is set to 1 when UPOSC status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (OSCIE=1), OSCIF causes an interrupt request. 0 No change in UPOSC bit. 1 UPOSC bit has changed.
0 UPOSC	**Oscillator Status Bit** — UPOSC reflects the status of the oscillator. Writes have no effect. Entering Full Stop Mode UPOSC is cleared. 0 The oscillator is off or oscillation is not qualified by the PLL. 1 The oscillator is qualified by the PLL.

8.3.2.6 S12CPMU_UHV_V7 Interrupt Enable Register (CPMUINT)

This register enables S12CPMU_UHV_V7 interrupt requests.

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	RTIE	0	0	LOCKIE	0	0	OSCIE	0
W		▨	▨		▨	▨		▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-9. S12CPMU_UHV_V7 Interrupt Enable Register (CPMUINT)

Read: Anytime

Write: Anytime

Table 8-5. CPMUINT Field Descriptions

Field	Description
7 RTIE	**Real Time Interrupt Enable Bit** 0 Interrupt requests from RTI are disabled. 1 Interrupt will be requested whenever RTIF is set.
4 LOCKIE	**PLL Lock Interrupt Enable Bit** 0 PLL LOCK interrupt requests are disabled. 1 Interrupt will be requested whenever LOCKIF is set.
1 OSCIE	**Oscillator Corrupt Interrupt Enable Bit** 0 Oscillator Corrupt interrupt requests are disabled. 1 Interrupt will be requested whenever OSCIF is set.

8.3.2.7 S12CPMU_UHV_V7 Clock Select Register (CPMUCLKS)

This register controls S12CPMU_UHV_V7 clock selection.

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R W	PLLSEL	PSTP	CSAD	COP OSCSEL1	PRE	PCE	RTI OSCSEL	COP OSCSEL0
Reset	1	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-10. S12CPMU_UHV_V7 Clock Select Register (CPMUCLKS)

Read: Anytime

Write:

- Only possible if PROT=0 (CPMUPROT register) in all MCU Modes (Normal and Special Mode).
- All bits in Special Mode (if PROT=0).
- PLLSEL, PSTP, PRE, PCE, RTIOSCSEL: In Normal Mode (if PROT=0).
- CSAD: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place.
- COPOSCSEL0: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place. If COPOSCSEL0 was cleared by UPOSC=0 (entering Full Stop Mode with COPOSCSEL0=1 or insufficient OSCCLK quality), then COPOSCSEL0 can be set once again.
- COPOSCSEL1: In Normal Mode (if PROT=0) until CPMUCOP write once has taken place. COPOSCSEL1 will not be cleared by UPOSC=0 (entering Full Stop Mode with COPOSCSEL1=1 or insufficient OSCCLK quality if OSCCLK is used as clock source for other clock domains: for instance core clock etc.).

NOTE

After writing CPMUCLKS register, it is strongly recommended to read back CPMUCLKS register to make sure that write of PLLSEL, RTIOSCSEL and COPOSCSEL was successful. This is because under certain circumstances writes have no effect or bits are automatically changed (see CPMUCLKS register and bit descriptions).

NOTE

When using the oscillator clock as system clock (write PLLSEL = 0) it is highly recommended to enable the oscillator clock monitor reset feature (write OMRE = 1 in CPMUOSC2 register). If the oscillator monitor reset feature is disabled (OMRE = 0) and the oscillator clock is used as system clock, the system will stall in case of loss of oscillation.

Table 8-6. CPMUCLKS Descriptions

Field	Description
7 PLLSEL	**PLL Select Bit** This bit selects the PLLCLK as source of the System Clocks (Core Clock and Bus Clock). PLLSEL can only be set to 0, if UPOSC=1. UPOSC= 0 sets the PLLSEL bit. Entering Full Stop Mode sets the PLLSEL bit. 0 System clocks are derived from OSCCLK if oscillator is up (UPOSC=1, $f_{bus} = f_{osc} / 2$). 1 System clocks are derived from PLLCLK, $f_{bus} = f_{PLL} / 2$.
6 PSTP	**Pseudo Stop Bit** This bit controls the functionality of the oscillator during Stop Mode. 0 Oscillator is disabled in Stop Mode (Full Stop Mode). 1 Oscillator continues to run in Stop Mode (Pseudo Stop Mode), option to run RTI and COP. **Note:** Pseudo Stop Mode allows for faster STOP recovery and reduces the mechanical stress and aging of the resonator in case of frequent STOP conditions at the expense of a slightly increased power consumption. **Note:** When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator t_{UPOSC} before entering Pseudo Stop Mode.
5 CSAD	**COP in Stop Mode ACLK Disable** — This bit disables the ACLK for the COP in Stop Mode. Hence the COP is static while in Stop Mode and continues to operate after exit from Stop Mode. Due to clock domain crossing synchronization there is a latency time to enter and exit Stop Mode if COP clock source is ACLK and this clock is stopped in Stop Mode. This maximum latency time is 4 ACLK cycles which must be added to the Stop Mode recovery time tSTP_REC from exit of current Stop Mode to entry of next Stop Mode. This latency time occurs no matter which Stop Mode (Full, Pseudo) is currently exited or entered next. After exit from Stop Mode (Pseudo, Full) for 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter increments at each Stop Mode exit. This bit does not influence the ACLK for the API. 0 COP running in Stop Mode (ACLK for COP enabled in Stop Mode). 1 COP stopped in Stop Mode (ACLK for COP disabled in Stop Mode)
4 COP OSCSEL1	**COP Clock Select 1** — COPOSCSEL0 and COPOSCSEL1 combined determine the clock source to the COP (see also Table 8-7). If COPOSCSEL1 = 1, COPOSCSEL0 has no effect regarding clock select and changing the COPOSCSEL0 bit does not re-start the COP time-out period. COPOSCSEL1 selects the clock source to the COP to be either ACLK (derived from trimmable internal RC-Oscillator) or clock selected via COPOSCSEL0 (IRCCLK or OSCCLK). Changing the COPOSCSEL1 bit re-starts the COP time-out period. COPOSCSEL1 can be set independent from value of UPOSC. UPOSC= 0 does not clear the COPOSCSEL1 bit. 0 COP clock source defined by COPOSCSEL0 1 COP clock source is ACLK derived from a trimmable internal RC-Oscillator
3 PRE	**RTI Enable During Pseudo Stop Bit** — PRE enables the RTI during Pseudo Stop Mode. 0 RTI stops running during Pseudo Stop Mode. 1 RTI continues running during Pseudo Stop Mode if RTIOSCSEL=1. **Note:** If PRE=0 or RTIOSCSEL=0 then the RTI will go static while Stop Mode is active. The RTI counter will not be reset.
2 PCE	**COP Enable During Pseudo Stop Bit** — PCE enables the COP during Pseudo Stop Mode. 0 COP stops running during Pseudo Stop Mode 1 COP continues running during Pseudo Stop Mode if COPOSCSEL=1 **Note:** If PCE=0 or COPOSCSEL=0 then the COP will go static while Stop Mode is active. The COP counter will not be reset.

Table 8-6. CPMUCLKS Descriptions (continued)

Field	Description
1 RTIOSCSEL	**RTI Clock Select**— RTIOSCSEL selects the clock source to the RTI. Either IRCCLK or OSCCLK. Changing the RTIOSCSEL bit re-starts the RTI time-out period. RTIOSCSEL can only be set to 1, if UPOSC=1. UPOSC= 0 clears the RTIOSCSEL bit. 0 RTI clock source is IRCCLK. 1 RTI clock source is OSCCLK.
0 COP OSCSEL0	**COP Clock Select 0** — COPOSCSEL0 and COPOSCSEL1 combined determine the clock source to the COP (see also Table 8-7) If COPOSCSEL1 = 1, COPOSCSEL0 has no effect regarding clock select and changing the COPOSCSEL0 bit does not re-start the COP time-out period. When COPOSCSEL1=0,COPOSCSEL0 selects the clock source to the COP to be either IRCCLK or OSCCLK. Changing the COPOSCSEL0 bit re-starts the COP time-out period. COPOSCSEL0 can only be set to 1, if UPOSC=1. UPOSC= 0 clears the COPOSCSEL0 bit. 0 COP clock source is IRCCLK. 1 COP clock source is OSCCLK

Table 8-7. COPOSCSEL1, COPOSCSEL0 clock source select description

COPOSCSEL1	COPOSCSEL0	COP clock source
0	0	IRCCLK
0	1	OSCCLK
1	x	ACLK

8.3.2.8 S12CPMU_UHV_V7 PLL Control Register (CPMUPLL)

This register controls the PLL functionality.

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R	0	0	FM1	FM0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 8-11. S12CPMU_UHV_V7 PLL Control Register (CPMUPLL)

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register). Else write has no effect.

NOTE

Write to this register clears the LOCK and UPOSC status bits.

NOTE

Care should be taken to ensure that the bus frequency does not exceed the specified maximum when frequency modulation is enabled.

Table 8-8. CPMUPLL Field Descriptions

Field	Description
5, 4 FM1, FM0	PLL **Frequency Modulation Enable Bits** — FM1 and FM0 enable frequency modulation on the VCOCLK. This is to reduce noise emission. The modulation frequency is f_{ref} divided by 16. See Table 8-9 for coding.

Table 8-9. FM Amplitude selection

FM1	FM0	FM Amplitude / f_{VCO} Variation
0	0	FM off
0	1	±1%
1	0	±2%
1	1	±4%

8.3.2.9 S12CPMU_UHV_V7 RTI Control Register (CPMURTI)

This register selects the time-out period for the Real Time Interrupt.

The clock source for the RTI is either IRCCLK or OSCCLK depending on the setting of the RTIOSCSEL bit. In Stop Mode with PSTP=1 (Pseudo Stop Mode) and RTIOSCSEL=1 the RTI continues to run, else the RTI counter halts in Stop Mode.

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
Reset	0	0	0	0	0	0	0	0

Figure 8-12. S12CPMU_UHV_V7 RTI Control Register (CPMURTI)

Read: Anytime

Write: Anytime

NOTE

A write to this register starts the RTI time-out period. A change of the RTIOSCSEL bit (writing a different value or loosing UPOSC status) re-starts the RTI time-out period.

Table 8-10. CPMURTI Field Descriptions

Field	Description
7 RTDEC	**Decimal or Binary Divider Select Bit** — RTDEC selects decimal or binary based prescaler values. 0 Binary based divider value. See Table 8-11 1 Decimal based divider value. See Table 8-12
6–4 RTR[6:4]	**Real Time Interrupt Prescale Rate Select Bits** — These bits select the prescale rate for the RTI.See Table 8-11 and Table 8-12.
3–0 RTR[3:0]	**Real Time Interrupt Modulus Counter Select Bits** — These bits select the modulus counter target value to provide additional granularity.Table 8-11 and Table 8-12 show all possible divide values selectable by the CPMURTI register.

Table 8-11. RTI Frequency Divide Rates for RTDEC = 0

RTR[3:0]	**RTR[6:4] =**							
	000 (OFF)	**001 (2^{10})**	**010 (2^{11})**	**011 (2^{12})**	**100 (2^{13})**	**101 (2^{14})**	**110 (2^{15})**	**111 (2^{16})**
0000 (÷1)	OFF[1]	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
0001 (÷2)	OFF	2x2^{10}	2x2^{11}	2x2^{12}	2x2^{13}	2x2^{14}	2x2^{15}	2x2^{16}
0010 (÷3)	OFF	3x2^{10}	3x2^{11}	3x2^{12}	3x2^{13}	3x2^{14}	3x2^{15}	3x2^{16}
0011 (÷4)	OFF	4x2^{10}	4x2^{11}	4x2^{12}	4x2^{13}	4x2^{14}	4x2^{15}	4x2^{16}
0100 (÷5)	OFF	5x2^{10}	5x2^{11}	5x2^{12}	5x2^{13}	5x2^{14}	5x2^{15}	5x2^{16}
0101 (÷6)	OFF	6x2^{10}	6x2^{11}	6x2^{12}	6x2^{13}	6x2^{14}	6x2^{15}	6x2^{16}
0110 (÷7)	OFF	7x2^{10}	7x2^{11}	7x2^{12}	7x2^{13}	7x2^{14}	7x2^{15}	7x2^{16}
0111 (÷8)	OFF	8x2^{10}	8x2^{11}	8x2^{12}	8x2^{13}	8x2^{14}	8x2^{15}	8x2^{16}
1000 (÷9)	OFF	9x2^{10}	9x2^{11}	9x2^{12}	9x2^{13}	9x2^{14}	9x2^{15}	9x2^{16}
1001 (÷10)	OFF	10x2^{10}	10x2^{11}	10x2^{12}	10x2^{13}	10x2^{14}	10x2^{15}	10x2^{16}
1010 (÷11)	OFF	11x2^{10}	11x2^{11}	11x2^{12}	11x2^{13}	11x2^{14}	11x2^{15}	11x2^{16}
1011 (÷12)	OFF	12x2^{10}	12x2^{11}	12x2^{12}	12x2^{13}	12x2^{14}	12x2^{15}	12x2^{16}
1100 (÷13)	OFF	13x2^{10}	13x2^{11}	13x2^{12}	13x2^{13}	13x2^{14}	13x2^{15}	13x2^{16}
1101 (÷14)	OFF	14x2^{10}	14x2^{11}	14x2^{12}	14x2^{13}	14x2^{14}	14x2^{15}	14x2^{16}
1110 (÷15)	OFF	15x2^{10}	15x2^{11}	15x2^{12}	15x2^{13}	15x2^{14}	15x2^{15}	15x2^{16}
1111 (÷16)	OFF	16x2^{10}	16x2^{11}	16x2^{12}	16x2^{13}	16x2^{14}	16x2^{15}	16x2^{16}

[1] Denotes the default value out of reset.This value should be used to disable the RTI to ensure future backwards compatibility.

Table 8-12. RTI Frequency Divide Rates for RTDEC=1

RTR[3:0]	RTR[6:4] =							
	000 ($1x10^3$)	**001 ($2x10^3$)**	**010 ($5x10^3$)**	**011 ($10x10^3$)**	**100 ($20x10^3$)**	**101 ($50x10^3$)**	**110 ($100x10^3$)**	**111 ($200x10^3$)**
0000 (÷1)	$1x10^3$	$2x10^3$	$5x10^3$	$10x10^3$	$20x10^3$	$50x10^3$	$100x10^3$	$200x10^3$
0001 (÷2)	$2x10^3$	$4x10^3$	$10x10^3$	$20x10^3$	$40x10^3$	$100x10^3$	$200x10^3$	$400x10^3$
0010 (÷3)	$3x10^3$	$6x10^3$	$15x10^3$	$30x10^3$	$60x10^3$	$150x10^3$	$300x10^3$	$600x10^3$
0011 (÷4)	$4x10^3$	$8x10^3$	$20x10^3$	$40x10^3$	$80x10^3$	$200x10^3$	$400x10^3$	$800x10^3$
0100 (÷5)	$5x10^3$	$10x10^3$	$25x10^3$	$50x10^3$	$100x10^3$	$250x10^3$	$500x10^3$	$1x10^6$
0101 (÷6)	$6x10^3$	$12x10^3$	$30x10^3$	$60x10^3$	$120x10^3$	$300x10^3$	$600x10^3$	$1.2x10^6$
0110 (÷7)	$7x10^3$	$14x10^3$	$35x10^3$	$70x10^3$	$140x10^3$	$350x10^3$	$700x10^3$	$1.4x10^6$
0111 (÷8)	$8x10^3$	$16x10^3$	$40x10^3$	$80x10^3$	$160x10^3$	$400x10^3$	$800x10^3$	$1.6x10^6$
1000 (÷9)	$9x10^3$	$18x10^3$	$45x10^3$	$90x10^3$	$180x10^3$	$450x10^3$	$900x10^3$	$1.8x10^6$
1001 (÷10)	$10 x10^3$	$20x10^3$	$50x10^3$	$100x10^3$	$200x10^3$	$500x10^3$	$1x10^6$	$2x10^6$
1010 (÷11)	$11 x10^3$	$22x10^3$	$55x10^3$	$110x10^3$	$220x10^3$	$550x10^3$	$1.1x10^6$	$2.2x10^6$
1011 (÷12)	$12x10^3$	$24x10^3$	$60x10^3$	$120x10^3$	$240x10^3$	$600x10^3$	$1.2x10^6$	$2.4x10^6$
1100 (÷13)	$13x10^3$	$26x10^3$	$65x10^3$	$130x10^3$	$260x10^3$	$650x10^3$	$1.3x10^6$	$2.6x10^6$
1101 (÷14)	$14x10^3$	$28x10^3$	$70x10^3$	$140x10^3$	$280x10^3$	$700x10^3$	$1.4x10^6$	$2.8x10^6$
1110 (÷15)	$15x10^3$	$30x10^3$	$75x10^3$	$150x10^3$	$300x10^3$	$750x10^3$	$1.5x10^6$	$3x10^6$
1111 (÷16)	$16x10^3$	$32x10^3$	$80x10^3$	$160x10^3$	$320x10^3$	$800x10^3$	$1.6x10^6$	$3.2x10^6$

8.3.2.10 S12CPMU_UHV_V7 COP Control Register (CPMUCOP)

This register controls the COP (Computer Operating Properly) watchdog.

The clock source for the COP is either ACLK, IRCCLK or OSCCLK depending on the setting of the COPOSCSEL0 and COPOSCSEL1 bit (see also Table 8-7).
In Stop Mode with PSTP=1 (Pseudo Stop Mode), COPOSCSEL0=1 and COPOSCEL1=0 and PCE=1 the COP continues to run, else the COP counter halts in Stop Mode with COPOSCSEL1 =0.
In Full Stop Mode and Pseudo Stop Mode with COPOSCSEL1=1 the COP continues to run.

Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
W			WRTMASK	▨	▨			
Reset	F	0	0	0	0	F	F	F

After de-assert of System Reset the values are automatically loaded from the Flash memory. See Device specification for details.

▨ = Unimplemented or Reserved

Figure 8-13. S12CPMU_UHV_V7 COP Control Register (CPMUCOP)

Read: Anytime

Write:

1. RSBCK: Anytime in Special Mode; write to “1” but not to “0” in Normal Mode
2. WCOP, CR2, CR1, CR0:
   - Anytime in Special Mode, when WRTMASK is 0, otherwise it has no effect
   - Write once in Normal Mode, when WRTMASK is 0, otherwise it has no effect.
     - Writing CR[2:0] to “000” has no effect, but counts for the “write once” condition.
     - Writing WCOP to “0” has no effect, but counts for the “write once” condition.

When a non-zero value is loaded from Flash to CR[2:0] the COP time-out period is started.

A change of the COPOSCSEL0 or COPOSCSEL1 bit (writing a different value) or loosing UPOSC status while COPOSCSEL1 is clear and COPOSCSEL0 is set, re-starts the COP time-out period.

In Normal Mode the COP time-out period is restarted if either of these conditions is true:

1. Writing a non-zero value to CR[2:0] (anytime in special mode, once in normal mode) with WRTMASK = 0.
2. Writing WCOP bit (anytime in Special Mode, once in Normal Mode) with WRTMASK = 0.
3. Changing RSBCK bit from “0” to “1”.

In Special Mode, any write access to CPMUCOP register restarts the COP time-out period.

Table 8-13. CPMUCOP Field Descriptions

Field	Description
7 WCOP	**Window COP Mode Bit** — When set, a write to the CPMUARMCOP register must occur in the last 25% of the selected period. A write during the first 75% of the selected period generates a COP reset. As long as all writes occur during this window, \$55 can be written as often as desired. Once \$AA is written after the \$55, the time-out logic restarts and the user must wait until the next window before writing to CPMUARMCOP. Table 8-14 shows the duration of this window for the seven available COP rates. 0 Normal COP operation 1 Window COP operation
6 RSBCK	**COP and RTI Stop in Active BDM Mode Bit** 0 Allows the COP and RTI to keep running in Active BDM mode. 1 Stops the COP and RTI counters whenever the part is in Active BDM mode.
5 WRTMASK	**Write Mask for WCOP and CR[2:0] Bit** — This write-only bit serves as a mask for the WCOP and CR[2:0] bits while writing the CPMUCOP register. It is intended for BDM writing the RSBCK without changing the content of WCOP and CR[2:0]. 0 Write of WCOP and CR[2:0] has an effect with this write of CPMUCOP 1 Write of WCOP and CR[2:0] has no effect with this write of CPMUCOP. (Does not count for “write once”.)
2–0 CR[2:0]	**COP Watchdog Timer Rate Select** — These bits select the COP time-out rate (see Table 8-14 and Table 8-15). Writing a nonzero value to CR[2:0] enables the COP counter and starts the time-out period. A COP counter time-out causes a System Reset. This can be avoided by periodically (before time-out) initializing the COP counter via the CPMUARMCOP register. While all of the following four conditions are true the CR[2:0], WCOP bits are ignored and the COP operates at highest time-out period (2^{24} cycles) in normal COP mode (Window COP mode disabled): 1) COP is enabled (CR[2:0] is not 000) 2) BDM mode active 3) RSBCK = 0 4) Operation in Special Mode

Table 8-14. COP Watchdog Rates if COPOSCSEL1=0.
(default out of reset)

CR2	CR1	CR0	COPCLK Cycles to time-out (COPCLK is either IRCCLK or OSCCLK depending on the COPOSCSEL0 bit)
0	0	0	COP disabled
0	0	1	2^{14}
0	1	0	2^{16}
0	1	1	2^{18}
1	0	0	2^{20}
1	0	1	2^{22}
1	1	0	2^{23}
1	1	1	2^{24}

Table 8-15. COP Watchdog Rates if COPOSCSEL1=1.

CR2	CR1	CR0	COPCLK Cycles to time-out (COPCLK is ACLK divided by 2)
0	0	0	COP disabled
0	0	1	2^{7}
0	1	0	2^{9}
0	1	1	2^{11}
1	0	0	2^{13}
1	0	1	2^{15}
1	1	0	2^{16}
1	1	1	2^{17}

8.3.2.11 Reserved Register CPMUTEST0

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in Special Mode can alter the S12CPMU_UHV_V7's functionality.

Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-14. Reserved Register (CPMUTEST0)

Read: Anytime

Write: Only in Special Mode

8.3.2.12 Reserved Register CPMUTEST1

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in Special Mode can alter the S12CPMU_UHV_V7's functionality.

Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-15. Reserved Register (CPMUTEST1)

Read: Anytime

Write: Only in Special Mode

8.3.2.13 S12CPMU_UHV_V7 COP Timer Arm/Reset Register (CPMUARMCOP)

This register is used to restart the COP time-out period.

Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	ARMCOP-Bit 7	ARMCOP-Bit 6	ARMCOP-Bit 5	ARMCOP-Bit 4	ARMCOP-Bit 3	ARMCOP-Bit 2	ARMCOP-Bit 1	ARMCOP-Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 8-16. S12CPMU_UHV_V7 CPMUARMCOP Register

Read: Always reads $00

Write: Anytime

When the COP is disabled (CR[2:0] = “000”) writing to this register has no effect.

When the COP is enabled by setting CR[2:0] nonzero, the following applies:

> Writing any value other than $55 or $AA causes a COP reset. To restart the COP time-out period write $55 followed by a write of $AA. These writes do not need to occur back-to-back, but the sequence ($55, $AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of $55 writes are allowed. When the WCOP bit is set, $55 and $AA writes must be done in the last 25% of the selected time-out period; writing any value in the first 75% of the selected period will cause a COP reset.

8.3.2.14 High Temperature Control Register (CPMUHTCTL)

The CPMUHTCTL register configures the temperature sense features.

Module Base + 0x0010

	7	6	5	4	3	2	1	0
R	0	0	VSEL	0	HTE	HTDS	HTIE	HTIF
W	▨	▨		▨		▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-17. High Temperature Control Register (CPMUHTCTL)

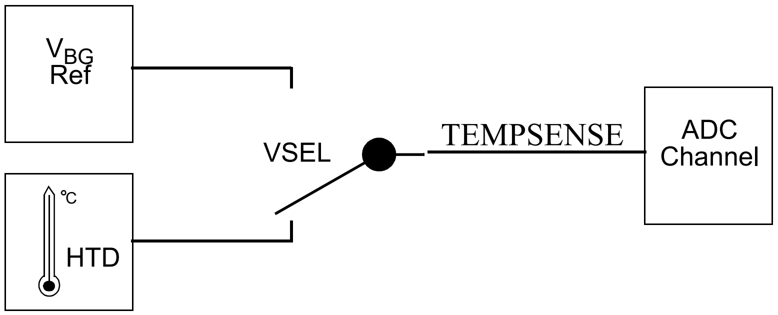
Read: Anytime

Write: VSEL, HTE, HTIE and HTIF are write anytime, HTDS is read only

Table 8-16. CPMUHTCTL Field Descriptions

Field	Description
5 VSEL	**Voltage Access Select Bit** — If set, the bandgap reference voltage V_{BG} can be accessed internally (i.e. multiplexed to an internal Analog to Digital Converter channel). If not set, the die temperature proportional voltage V_{HT} of the temperature sensor can be accessed internally. See device level specification for connectivity. For any of these access the HTE bit must be set. 0 An internal temperature proportional voltage V_{HT} can be accessed internally. 1 Bandgap reference voltage V_{BG} can be accessed internally.
3 HTE	**High Temperature Sensor/Bandgap Voltage Enable Bit** — This bit enables the high temperature sensor and bandgap voltage amplifier. 0 The temperature sensor and bandgap voltage amplifier is disabled. 1 The temperature sensor and bandgap voltage amplifier is enabled.
2 HTDS	**High Temperature Detect Status Bit** — This read-only status bit reflects the temperature status. Writes have no effect. 0 Junction Temperature is below level T_{HTID} or RPM. 1 Junction Temperature is above level T_{HTIA} and FPM.
1 HTIE	**High Temperature Interrupt Enable Bit** 0 Interrupt request is disabled. 1 Interrupt will be requested whenever HTIF is set.
0 HTIF	**High Temperature Interrupt Flag** — HTIF is set to 1 when HTDS status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (HTIE=1), HTIF causes an interrupt request. 0 No change in HTDS bit. 1 HTDS bit has changed.

Figure 8-18. Voltage Access Select



8.3.2.15 Low Voltage Control Register (CPMULVCTL)

The CPMULVCTL register allows the configuration of the low-voltage detect features.

Module Base + 0x0011

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	LVDS	LVIE	LVIF
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	U	0	U

The Reset state of LVDS and LVIF depends on the external supplied VDDA level

▨ = Unimplemented or Reserved

Figure 8-19. Low Voltage Control Register (CPMULVCTL)

Read: Anytime

Write: LVIE and LVIF are write anytime, LVDS is read only

Table 8-17. CPMULVCTL Field Descriptions

Field	Description
2 LVDS	**Low-Voltage Detect Status Bit** — This read-only status bit reflects the voltage level on VDDA. Writes have no effect. 0 Input voltage VDDA is above level V_{LVID} or RPM. 1 Input voltage VDDA is below level V_{LVIA} and FPM.
1 LVIE	**Low-Voltage Interrupt Enable Bit** 0 Interrupt request is disabled. 1 Interrupt will be requested whenever LVIF is set.
0 LVIF	**Low-Voltage Interrupt Flag** — LVIF is set to 1 when LVDS status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (LVIE = 1), LVIF causes an interrupt request. 0 No change in LVDS bit. 1 LVDS bit has changed.

8.3.2.16 Autonomous Periodical Interrupt Control Register (CPMUAPICTL)

The CPMUAPICTL register allows the configuration of the autonomous periodical interrupt features.

Module Base + 0x0012

	7	6	5	4	3	2	1	0
R	APICLK	0	0	APIES	APIEA	APIFE	APIE	APIF
W		▨	▨					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-20. Autonomous Periodical Interrupt Control Register (CPMUAPICTL)

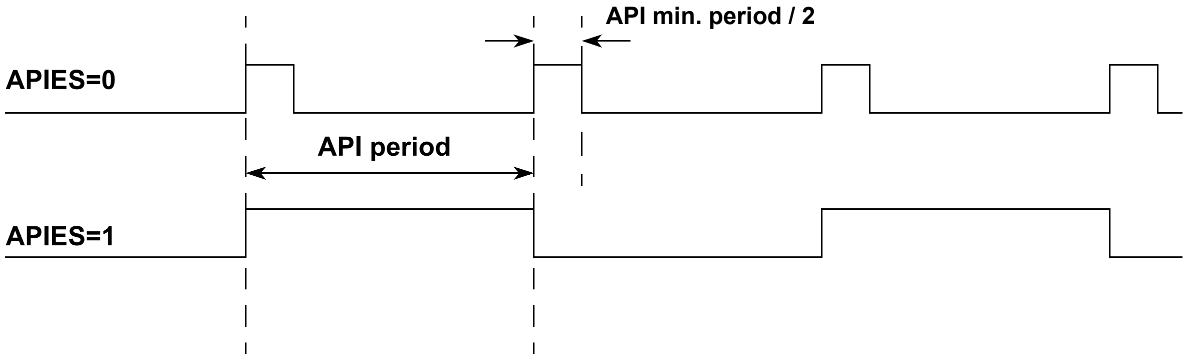
Read: Anytime

Write: Anytime

Table 8-18. CPMUAPICTL Field Descriptions

Field	Description
7 APICLK	**Autonomous Periodical Interrupt Clock Select Bit** — Selects the clock source for the API. Writable only if APIFE = 0. APICLK cannot be changed if APIFE is set by the same write operation. 0 Autonomous Clock (ACLK) used as source. 1 Bus Clock used as source.
4 APIES	**Autonomous Periodical Interrupt External Select Bit** — Selects the waveform at the external pin API_EXTCLK as shown in Figure 8-21. See device level specification for connectivity of API_EXTCLK pin. 0 If APIEA and APIFE are set, at the external pin API_EXTCLK periodic high pulses are visible at the end of every selected period with the size of half of the minimum period (APIR=0x0000 in Table 8-22). 1 If APIEA and APIFE are set, at the external pin API_EXTCLK a clock is visible with 2 times the selected API Period.
3 APIEA	**Autonomous Periodical Interrupt External Access Enable Bit** — If set, the waveform selected by bit APIES can be accessed externally. See device level specification for connectivity. 0 Waveform selected by APIES can not be accessed externally. 1 Waveform selected by APIES can be accessed externally, if APIFE is set.
2 APIFE	**Autonomous Periodical Interrupt Feature Enable Bit** — Enables the API feature and starts the API timer when set. 0 Autonomous periodical interrupt is disabled. 1 Autonomous periodical interrupt is enabled and timer starts running.
1 APIE	**Autonomous Periodical Interrupt Enable Bit** 0 API interrupt request is disabled. 1 API interrupt will be requested whenever APIF is set.
0 APIF	**Autonomous Periodical Interrupt Flag** — APIF is set to 1 when the in the API configured time has elapsed. This flag can only be cleared by writing a 1.Writing a 0 has no effect. If enabled (APIE = 1), APIF causes an interrupt request. 0 API time-out has not yet occurred. 1 API time-out has occurred.

Figure 8-21. Waveform selected on API_EXTCLK pin (APIEA=1, APIFE=1)



8.3.2.17 Autonomous Clock Trimming Register (CPMUACLKTR)

The CPMUACLKTR register configures the trimming of the Autonomous Clock (ACLK - trimmable internal RC-Oscillator) which can be selected as clock source for some CPMU features.

Module Base + 0x0013

	7	6	5	4	3	2	1	0
R	ACLKTR5	ACLKTR4	ACLKTR3	ACLKTR2	ACLKTR1	ACLKTR0	0	0
W								
Reset	F	F	F	F	F	F	0	0

After de-assert of System Reset a value is automatically loaded from the Flash memory.

Figure 8-22. Autonomous Clock Trimming Register (CPMUACLKTR)

Read: Anytime

Write: Anytime

Table 8-19. CPMUACLKTR Field Descriptions

Field	Description
7–2 ACLKTR[5:0]	**Autonomous Clock Period Trimming Bits** — See Table 8-20 for trimming effects. The ACLKTR[5:0] value represents a signed number influencing the ACLK period time.

Table 8-20. Trimming Effect of ACLKTR[5:0]

Bit	Trimming Effect
ACLKTR[5]	Increases period
ACLKTR[4]	Decreases period less than ACLKTR[5] increased it
ACLKTR[3]	Decreases period less than ACLKTR[4]
ACLKTR[2]	Decreases period less than ACLKTR[3]
ACLKTR[1]	Decreases period less than ACLKTR[2]
ACLKTR[0]	Decreases period less than ACLKTR[1]

8.3.2.18 Autonomous Periodical Interrupt Rate High and Low Register (CPMUAPIRH / CPMUAPIRL)

The CPMUAPIRH and CPMUAPIRL registers allow the configuration of the autonomous periodical interrupt rate.

Module Base + 0x0014

	7	6	5	4	3	2	1	0
R W	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
Reset	0	0	0	0	0	0	0	0

░ = Unimplemented or Reserved

Figure 8-23. Autonomous Periodical Interrupt Rate High Register (CPMUAPIRH)

Module Base + 0x0015

	7	6	5	4	3	2	1	0
R W	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
Reset	0	0	0	0	0	0	0	0

Figure 8-24. Autonomous Periodical Interrupt Rate Low Register (CPMUAPIRL)

Read: Anytime

Write: Anytime if APIFE=0, Else writes have no effect.

Table 8-21. CPMUAPIRH / CPMUAPIRL Field Descriptions

Field	Description
15-0 APIR[15:0]	**Autonomous Periodical Interrupt Rate Bits** — These bits define the time-out period of the API. See Table 8-22 for details of the effect of the autonomous periodical interrupt rate bits.

The period can be calculated as follows depending on logical value of the APICLK bit:

APICLK=0: Period = 2*(APIR[15:0] + 1) * (ACLK Clock Period * 2)
APICLK=1: Period = 2*(APIR[15:0] + 1) * Bus Clock Period

NOTE

For APICLK bit clear the first time-out period of the API will show a latency time between two to three f_{ACLK} cycles due to synchronous clock gate release when the API feature gets enabled (APIFE bit set).

Table 8-22. Selectable Autonomous Periodical Interrupt Periods

APICLK	APIR[15:0]	Selected Period
0	0000	0.2 ms[1]
0	0001	0.4 ms[1]
0	0002	0.6 ms[1]
0	0003	0.8 ms[1]
0	0004	1.0 ms[1]
0	0005	1.2 ms[1]
0	.....	.....
0	FFFD	13106.8 ms[1]
0	FFFE	13107.0 ms[1]
0	FFFF	13107.2 ms[1]
1	0000	2 * Bus Clock period
1	0001	4 * Bus Clock period
1	0002	6 * Bus Clock period
1	0003	8 * Bus Clock period
1	0004	10 * Bus Clock period
1	0005	12 * Bus Clock period
1	.....	.....
1	FFFD	131068 * Bus Clock period
1	FFFE	131070 * Bus Clock period
1	FFFF	131072 * Bus Clock period

[1] When f_{ACLK} is trimmed to 20KHz.

8.3.2.19 Reserved Register CPMUTEST3

NOTE

> This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in Special Mode can alter the S12CPMU_UHV_V7’s functionality.

Module Base + 0x0016

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-25. Reserved Register (CPMUTEST3)

Read: Anytime

Write: Only in Special Mode

8.3.2.20 High Temperature Trimming Register (CPMUHTTR)

The CPMUHTTR register configures the trimming of the S12CPMU_UHV_V7 temperature sense.

Module Base + 0x0017

	7	6	5	4	3	2	1	0
R	HTOE	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
W		▨	▨	▨				
Reset	0	0	0	0	F	F	F	F

After de-assert of System Reset a trim value is automatically loaded from the Flash memory. See Device specification for details.

▨ = Unimplemented or Reserved

Figure 8-26. High Temperature Trimming Register (CPMUHTTR)

Read: Anytime

Write: Anytime

Table 8-24. CPMUHTTR Field Descriptions

Field	Description
7 HTOE	**High Temperature Offset Enable Bit** — If set the temperature sense offset is enabled. 0 The temperature sense offset is disabled. HTTR[3:0] bits don't care. 1 The temperature sense offset is enabled. HTTR[3:0] select the temperature offset.
3–0 HTTR[3:0]	**High Temperature Trimming Bits** — See Table 8-25 for trimming effects.

Table 8-25. Trimming Effect of HTTR

Bit	Trimming Effect
HTTR[3]	Increases V_{HT} twice of HTTR[2]
HTTR[2]	Increases V_{HT} twice of HTTR[1]
HTTR[1]	Increases V_{HT} twice of HTTR[0]
HTTR[0]	Increases V_{HT} (to compensate Temperature Offset)

8.3.2.21 S12CPMU_UHV_V7 IRC1M Trim Registers (CPMUIRCTRIMH / CPMUIRCTRIML)

Module Base + 0x0018

	15	14	13	12	11	10	9	8
R	TCTRIM[4:0]					0	IRCTRIM[9:8]	
W								
Reset	F	F	F	F	F	0	F	F

After de-assert of System Reset a factory programmed trim value is automatically loaded from the Flash memory to provide trimmed Internal Reference Frequency f_{IRC1M_TRIM}.

Figure 8-27. S12CPMU_UHV_V7 IRC1M Trim High Register (CPMUIRCTRIMH)

Module Base + 0x0019

	7	6	5	4	3	2	1	0
R	IRCTRIM[7:0]							
W								
Reset	F	F	F	F	F	F	F	F

After de-assert of System Reset a factory programmed trim value is automatically loaded from the Flash memory to provide trimmed Internal Reference Frequency f_{IRC1M_TRIM}.

Figure 8-28. S12CPMU_UHV_V7 IRC1M Trim Low Register (CPMUIRCTRIML)

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register). Else write has no effect

NOTE

Writes to these registers while PLLSEL=1 clears the LOCK and UPOSC status bits.

Table 8-26. CPMUIRCTRIMH/L Field Descriptions

Field	Description
15-11 TCTRIM[4:0]	**IRC1M temperature coefficient Trim Bits** Trim bits for the Temperature Coefficient (TC) of the IRC1M frequency. Table 8-27 shows the influence of the bits TCTRIM[4:0] on the relationship between frequency and temperature. Figure 8-30 shows an approximate TC variation, relative to the nominal TC of the IRC1M (i.e. for TCTRIM[4:0]=0x00000 or 0x10000).
9-0 IRCTRIM[9:0]	**IRC1M Frequency Trim Bits** — Trim bits for Internal Reference Clock After System Reset the factory programmed trim value is automatically loaded into these registers, resulting in a Internal Reference Frequency f_{IRC1M_TRIM}.See device electrical characteristics for value of f_{IRC1M_TRIM}. The frequency trimming consists of two different trimming methods: A rough trimming controlled by bits IRCTRIM[9:6] can be done with frequency leaps of about 6% in average. A fine trimming controlled by bits IRCTRIM[5:0] can be done with frequency leaps of about 0.3% (this trimming determines the precision of the frequency setting of 0.15%, i.e. 0.3% is the distance between two trimming values). Figure 8-29 shows the relationship between the trim bits and the resulting IRC1M frequency.

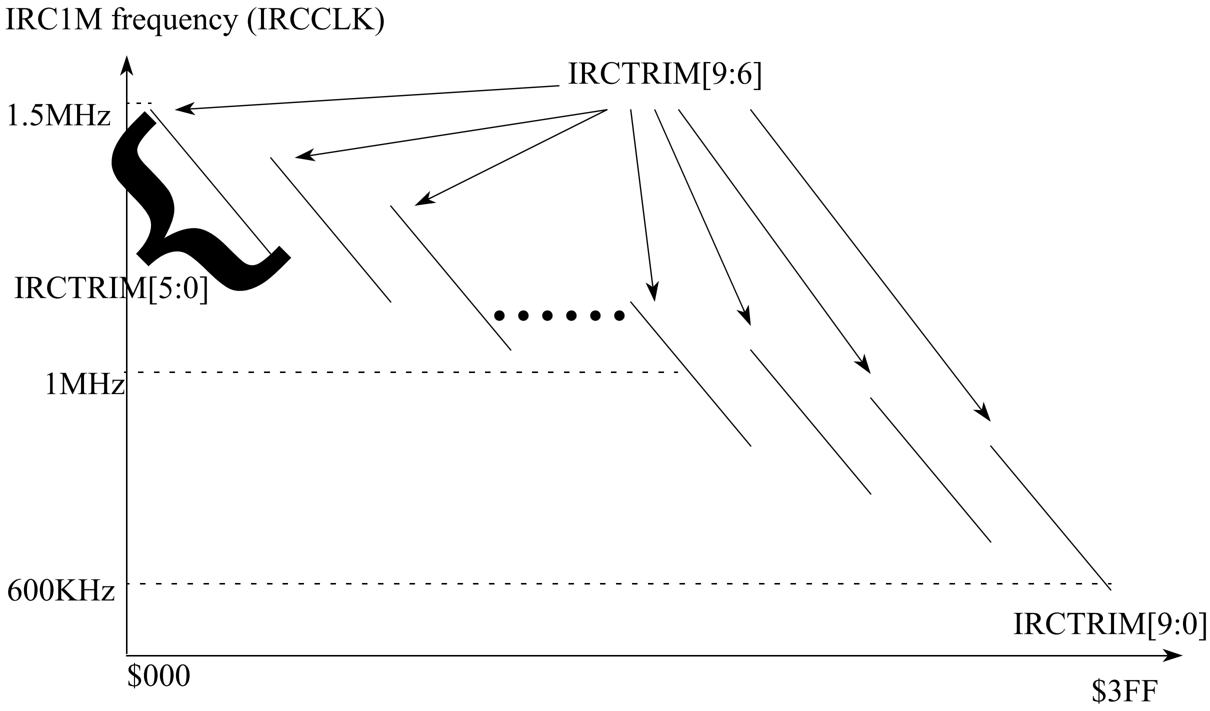


Figure 8-29. IRC1M Frequency Trimming Diagram

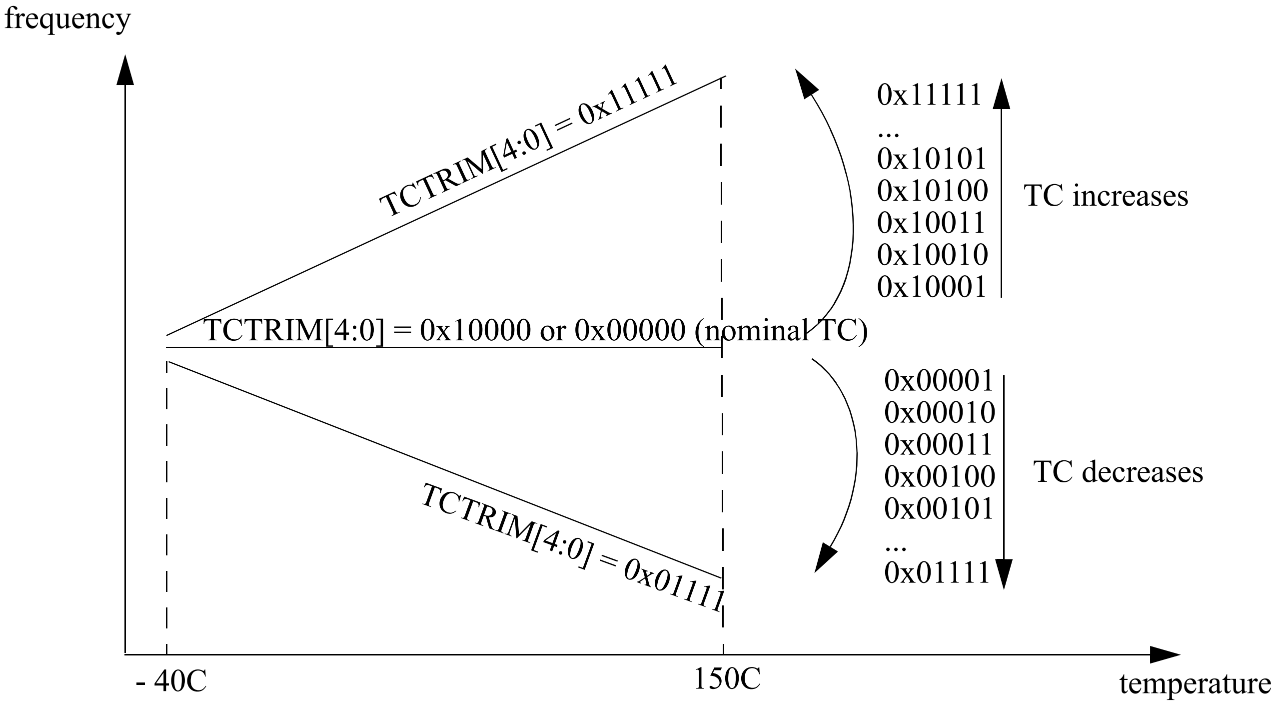


Figure 8-30. Influence of TCTRIM[4:0] on the Temperature Coefficient

NOTE

The frequency is not necessarily linear with the temperature (in most cases it will not be). The above diagram is meant only to give the direction (positive or negative) of the variation of the TC, relative to the nominal TC.

Setting TCTRIM[4:0] at 0x00000 or 0x10000 does not mean that the temperature coefficient will be zero. These two combinations basically switch off the TC compensation module, which results in the nominal TC of the IRC1M.

Table 8-27. TC trimming of the frequency of the IRC1M at ambient temperature

TCTRIM[4:0]	IRC1M Indicative relative TC variation	IRC1M indicative frequency drift for relative TC variation
00000	0 (nominal TC of the IRC)	0%
00001	-0.27%	-0.5%
00010	-0.54%	-0.9%
00011	-0.81%	-1.3%
00100	-1.08%	-1.7%
00101	-1.35%	-2.0%
00110	-1.63%	-2.2%
00111	-1.9%	-2.5%
01000	-2.20%	-3.0%
01001	-2.47%	-3.4%
01010	-2.77%	-3.9%
01011	-3.04	-4.3%
01100	-3.33%	-4.7%
01101	-3.6%	-5.1%
01110	-3.91%	-5.6%
01111	-4.18%	-5.9%
10000	0 (nominal TC of the IRC)	0%
10001	+0.27%	+0.5%
10010	+0.54%	+0.9%
10011	+0.81%	+1.3%
10100	+1.07%	+1.7%
10101	+1.34%	+2.0%
10110	+1.59%	+2.2%
10111	+1.86%	+2.5%
11000	+2.11%	+3.0%
11001	+2.38%	+3.4%
11010	+2.62%	+3.9%
11011	+2.89%	+4.3%
11100	+3.12%	+4.7%
11101	+3.39%	+5.1%
11110	+3.62%	+5.6%
11111	+3.89%	+5.9%

NOTE

Since the IRC1M frequency is not a linear function of the temperature, but more like a parabola, the above relative variation is only an indication and should be considered with care.

Be aware that the output frequency varies with the TC trimming. A frequency trimming correction is therefore necessary. The values provided in Table 8-27 are typical values at ambient temperature which can vary from device to device.

8.3.2.22 S12CPMU_UHV_V7 Oscillator Register (CPMUOSC)

This registers configures the external oscillator (XOSCLCP).

Module Base + 0x001A

	7	6	5	4	3	2	1	0
R	OSCE	0	Reserved	0	0	0	0	0
W		▨		▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-31. S12CPMU_UHV_V7 Oscillator Register (CPMUOSC)

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register). Else write has no effect.

NOTE.

Write to this register clears the LOCK and UPOSC status bits.

Table 8-28. CPMUOSC Field Descriptions

Field	Description
7 OSCE	**Oscillator Enable Bit** — This bit enables the external oscillator (XOSCLCP). The UPOSC status bit in the CPMIUFLG register indicates when the oscillation is stable and when OSCCLK can be selected as source of the Bus Clock or source of the COP or RTI.**If the oscillator clock monitor reset is enabled (OMRE = 1 in CPMUOSC2 register), then a loss of oscillation will lead to an oscillator clock monitor reset.** 0 External oscillator is disabled. REFCLK for PLL is IRCCLK. 1 External oscillator is enabled. Oscillator clock monitor is enabled. External oscillator is qualified by PLLCLK. REFCLK for PLL is the external oscillator clock divided by REFDIV. **If OSCE bit has been set (write “1”) the EXTAL and XTAL pins are exclusively reserved for the oscillator and they can not be used anymore as general purpose I/O until the next system reset.** **Note:** When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator t_{UPOSC} before entering Pseudo Stop Mode.
5 Reserved	Do not alter this bit from its reset value. It is for Manufacturer use only and can change the Oscillator behavior.

8.3.2.23 S12CPMU_UHV_V7 Protection Register (CPMUPROT)

This register protects the clock configuration registers from accidental overwrite:

CPMUSYNR, CPMUREFDIV, CPMUCLKS, CPMUPLL, CPMUIRCTRIMH/L, CPMUOSC and CPMUOSC2

Module Base + 0x001B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PROT
W								
Reset	0	0	0	0	0	0	0	0

Figure 8-32. S12CPMU_UHV_V7 Protection Register (CPMUPROT)

Read: Anytime

Write: Anytime

Field	Description
PROT	**Clock Configuration Registers Protection Bit** — This bit protects the clock configuration registers from accidental overwrite (see list of protected registers above): Writing 0x26 to the CPMUPROT register clears the PROT bit, other write accesses set the PROT bit. 0 Protection of clock configuration registers is disabled. 1 Protection of clock configuration registers is enabled. (see list of protected registers above).

8.3.2.24 Reserved Register CPMUTEST2

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in Special Mode can alter the S12CPMU_UHV_V7's functionality.

Module Base + 0x001C

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W			▨	▨	▨	▨	▨	
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 8-33. Reserved Register CPMUTEST2

Read: Anytime

Write: Only in Special Mode

8.3.2.25 Voltage Regulator Control Register (CPMUVREGCTL)

The CPMUVREGCTL allows to enable or disable certain parts of the voltage regulator.This register must be configured after system startup.

Module Base + 0x001D

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	EXTCON	EXTXON	INTXON
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	1	1	1

▨ = Unimplemented or Reserved

Figure 8-34. Voltage Regulator Control Register (CPMUVREGCTL)

Read: Anytime

Write: Once in normal modes, anytime in special modes

Table 8-29. Effects of writing the EXTXON and INTXON bits

value of EXTXON to be written	value of INTXON to be written	Write Access
0	0	blocked, no effect
0	1	legal access
1	0	legal access
1	1	blocked, no effect

Table 8-30. CPMUVREGCTL Field Descriptions

Field	Description
2 EXTCON	**External voltage regulator Enable Bit for VDDC domain** — Should be disabled after system startup if VDDC domain is not used. 0 VDDC domain disabled 1 VDDC domain enabled
1 EXTXON	**External voltage regulator Enable Bit for VDDX domain** — Should be set to 1 if external BJT is present on the PCB, cleared otherwise. 0 VDDX control loop does not use external BJT 1 VDDX control loop uses external BJT
0 INTXON	**Internal voltage regulator Enable Bit for VDDX domain**— Should be set to 1 if no external BJT is present on the PCB, cleared otherwise. 0 VDDX control loop does not use internal power transistor 1 VDDX control loop uses internal power transistor

8.3.2.26 S12CPMU_UHV_V7 Oscillator Register 2 (CPMUOSC2)

This registers configures the external oscillator (XOSCLCP).

Module Base + 0x001E

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	OMRE	OSCMOD
W								
Reset	0	0	0	0	0	0	0	0

Figure 8-35. S12CPMU_UHV_V7 Oscillator Register 2 (CPMUOSC2)

Read: Anytime

Write: Anytime if PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register). Else write has no effect.

Table 8-31. CPMUOSC2 Field Descriptions

Field	Description
0 OSCMOD	This bit selects the mode of the external oscillator (XOSCLCP) If OSCE bit in CPMUOSC register is 1, then the OSCMOD bit can not be changed (writes will have no effect). 0 External oscillator configured for loop controlled mode (reduced amplitude on EXTAL and XTAL)) 1 External oscillator configured for full swing mode (full swing amplitude on EXTAL and XTAL)
1 OMRE	This bit enables the oscillator clock monitor reset. If OSCE bit in CPMUOSC register is 1, then the OMRE bit can not be changed (writes will have no effect). 0 Oscillator clock monitor reset is disabled 1 Oscillator clock monitor reset is enabled

8.4 Functional Description

8.4.1 Phase Locked Loop with Internal Filter (PLL)

The PLL is used to generate a high speed PLLCLK based on a low frequency REFCLK.

The REFCLK is by default the IRCCLK which is trimmed to f_{IRC1M_TRIM}=1MHz.

If using the oscillator (OSCE=1) REFCLK will be based on OSCCLK. For increased flexibility, OSCCLK can be divided in a range of 1 to 16 to generate the reference frequency REFCLK using the REFDIV[3:0] bits. Based on the SYNDIV[5:0] bits the PLL generates the VCOCLK by multiplying the reference clock by a 2, 4, 6,... 126, 128. Based on the POSTDIV[4:0] bits the VCOCLK can be divided in a range of 1,2, 3, 4, 5, 6,... to 32 to generate the PLLCLK.

If oscillator is enabled (OSCE=1) $f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$

If oscillator is disabled (OSCE=0) $f_{REF} = f_{IRC1M}$

$$f_{VCO} = 2 \times f_{REF} \times (SYNDIV + 1)$$

If PLL is locked (LOCK=1) $f_{PLL} = \frac{f_{VCO}}{(POSTDIV + 1)}$

If PLL is not locked (LOCK=0) $f_{PLL} = \frac{f_{VCO}}{4}$

If PLL is selected (PLLSEL=1) $f_{bus} = \frac{f_{PLL}}{2}$

NOTE

Although it is possible to set the dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU.

Several examples of PLL divider settings are shown in Table 8-32. The following rules help to achieve optimum stability and shortest lock time:

- Use lowest possible f_{VCO} / f_{REF} ratio (SYNDIV value).
- Use highest possible REFCLK frequency f_{REF}.

Table 8-32. Examples of PLL Divider Settings

f_{osc}	REFDIV[3:0]	f_{REF}	REFFRQ[1:0]	SYNDIV[5:0]	f_{VCO}	VCOFRQ[1:0]	POSTDIV[4:0]	f_{PLL}	f_{bus}
off	\$00	1MHz	00	\$18	50MHz	01	\$03	12.5MHz	6.25MHz
off	\$00	1MHz	00	\$18	50MHz	01	\$00	50MHz	25MHz
4MHz	\$00	4MHz	01	\$05	48MHz	00	\$00	48MHz	24MHz

The phase detector inside the PLL compares the feedback clock (FBCLK = VCOCLK/(SYNDIV+1)) with the reference clock (REFCLK = (IRC1M or OSCCLK)/(REFDIV+1)). Correction pulses are generated based on the phase difference between the two signals. The loop filter alters the DC voltage on the internal filter capacitor, based on the width and direction of the correction pulse which leads to a higher or lower VCO frequency.

The user must select the range of the REFCLK frequency (REFFRQ[1:0] bits) and the range of the VCOCLK frequency (VCOFRQ[1:0] bits) to ensure that the correct PLL loop bandwidth is set.

The lock detector compares the frequencies of the FBCLK and the REFCLK. Therefore the speed of the lock detector is directly proportional to the reference clock frequency. The circuit determines the lock condition based on this comparison. So e.g. a failure in the reference clock will cause the PLL not to lock.

If PLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and for instance check the LOCK bit. If interrupt requests are disabled, software can poll the LOCK bit continuously (during PLL start-up) or at periodic intervals. In either case, only when the LOCK bit is set, the VCOCLK will have stabilized to the programmed frequency.

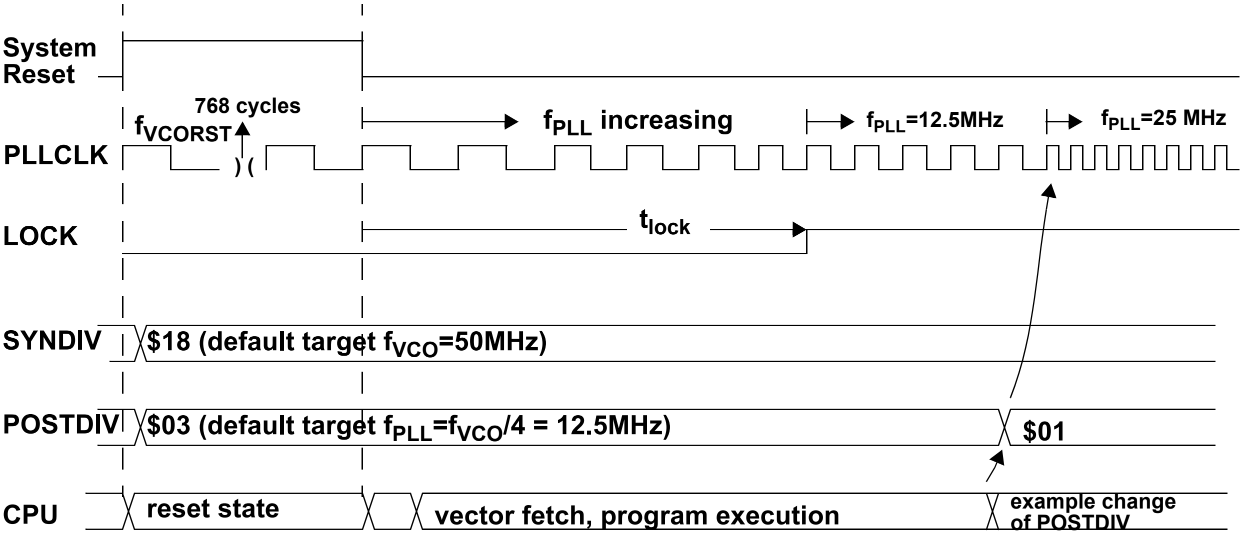
- The LOCK bit is a read-only indicator of the locked state of the PLL.
- The LOCK bit is set when the VCO frequency is within the tolerance, Δ_{Lock}, and is cleared when the VCO frequency is out of the tolerance, Δ_{unl}.
- Interrupt requests can occur if enabled (LOCKIE = 1) when the lock condition changes, toggling the LOCK bit.

In case of loss of reference clock (e.g. IRCCLK) the PLL will not lock or if already locked, then it will unlock. The frequency of the VCOCLK will be very low and will depend on the value of the VCOFRQ[1:0] bits.

8.4.2 Startup from Reset

An example for startup of the clock system from Reset is given in Figure 8-36.

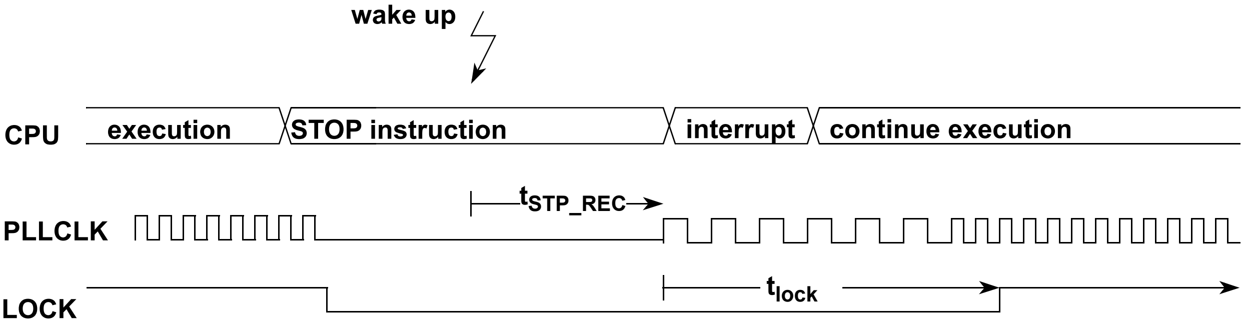
Figure 8-36. Startup of clock system after Reset



8.4.3 Stop Mode using PLLCLK as source of the Bus Clock

An example of what happens going into Stop Mode and exiting Stop Mode after an interrupt is shown in Figure 8-37. Disable PLL Lock interrupt (LOCKIE=0) before going into Stop Mode.

Figure 8-37. Stop Mode using PLLCLK as source of the Bus Clock



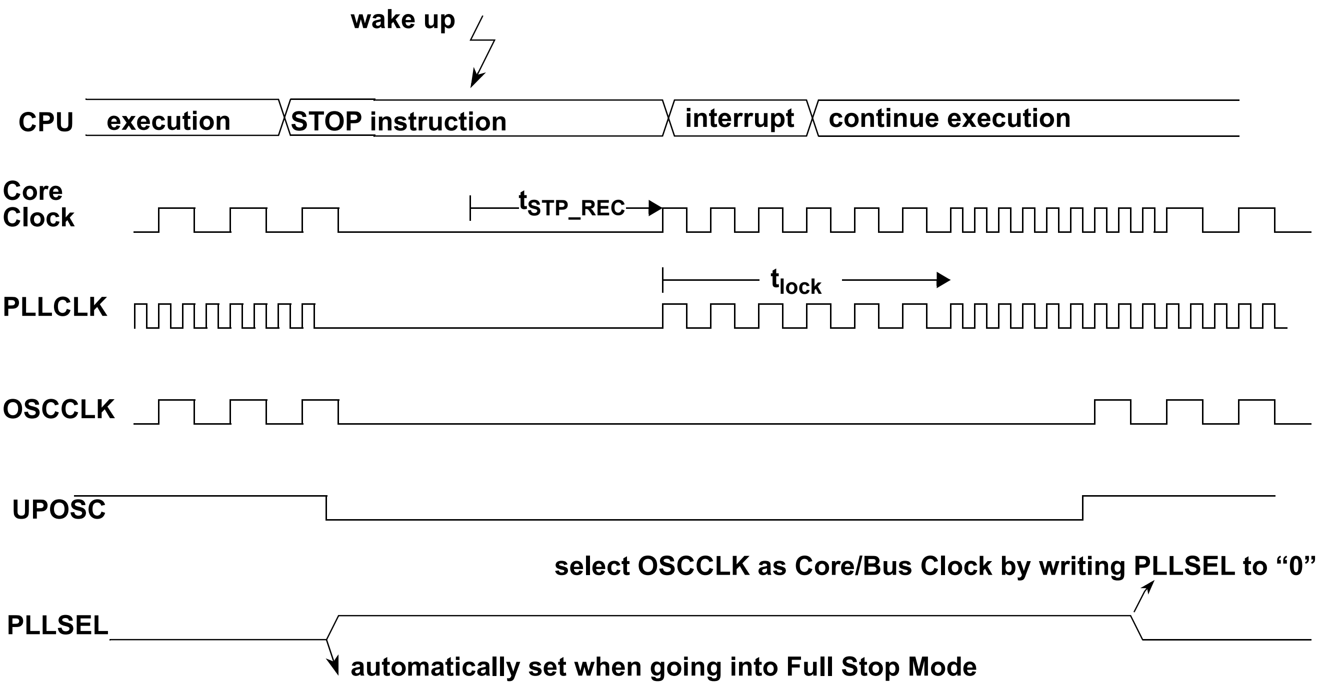
Depending on the COP configuration there might be an additional significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time of 2 ACLK cycles occurs if COP clock source is ACLK and the CSAD bit is set and must be added to the device Stop Mode recovery time t_{STP_REC}. After exit from Stop Mode (Pseudo, Full) for this latency time of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.

8.4.4 Full Stop Mode using Oscillator Clock as source of the Bus Clock

An example of what happens going into Full Stop Mode and exiting Full Stop Mode after an interrupt is shown in Figure 8-38.

Disable PLL Lock interrupt (LOCKIE=0) and oscillator status change interrupt (OSCIE=0) before going into Full Stop Mode.

Figure 8-38. Full Stop Mode using Oscillator Clock as source of the Bus Clock



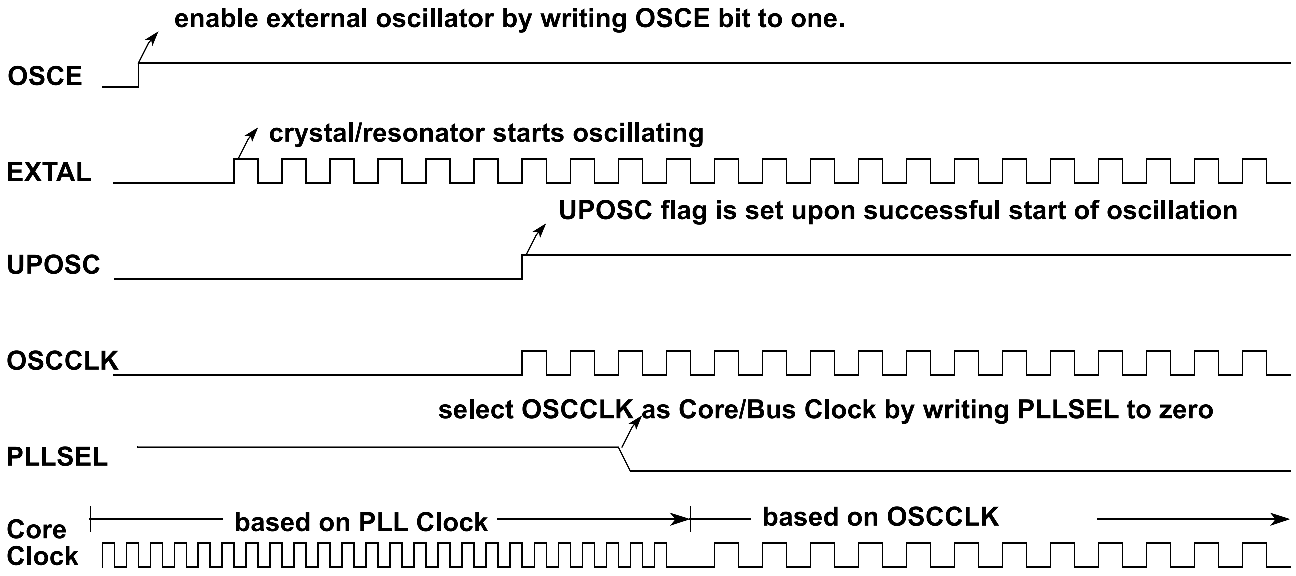
Depending on the COP configuration there might be a significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time of 2 ACLK cycles occurs if COP clock source is ACLK and the CSAD bit is set and must be added to the device Stop Mode recovery time t_{STP_REC}. After exit from Stop Mode (Pseudo, Full) for this latency time of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.

8.4.5 External Oscillator

8.4.5.1 Enabling the External Oscillator

An example of how to use the oscillator as source of the Bus Clock is shown in Figure 8-39.

Figure 8-39. Enabling the external oscillator



8.4.6 System Clock Configurations

8.4.6.1 PLL Engaged Internal Mode (PEI)

This mode is the default mode after System Reset or Power-On Reset.

The Bus Clock is based on the PLLCLK, the reference clock for the PLL is internally generated (IRC1M). The PLL is configured to 50 MHz VCOCLK with POSTDIV set to 0x03. If locked (LOCK=1) this results in a PLLCLK of 12.5 MHz and a Bus Clock of 6.25 MHz. The PLL can be re-configured to other bus frequencies.

The clock sources for COP and RTI can be based on the internal reference clock generator (IRC1M) or the RC-Oscillator (ACLK).

8.4.6.2 PLL Engaged External Mode (PEE)

In this mode, the Bus Clock is based on the PLLCLK as well (like PEI). The reference clock for the PLL is based on the external oscillator.

The clock sources for COP and RTI can be based on the internal reference clock generator or on the external oscillator clock or the RC-Oscillator (ACLK).

This mode can be entered from default mode PEI by performing the following steps:

1. Configure the PLL for desired bus frequency.
2. Enable the external Oscillator (OSCE bit).
3. Wait for oscillator to start-up and the PLL being locked (LOCK = 1) and (UPOSC =1).
4. Clear all flags in the CPMUIFLG register to be able to detect any future status bit change.
5. Optionally status interrupts can be enabled (CPMUINT register).

Loosing PLL lock status (LOCK=0) means loosing the oscillator status information as well (UPOSC=0).

The impact of loosing the oscillator status (UPOSC=0) in PEE mode is as follows:

- The PLLCLK is derived from the VCO clock (with its actual frequency) divided by four until the PLL locks again.

Application software needs to be prepared to deal with the impact of loosing the oscillator status at any time.

8.4.6.3 PLL Bypassed External Mode (PBE)

In this mode, the Bus Clock is based on the external oscillator clock. The reference clock for the PLL is based on the external oscillator.

The clock sources for COP and RTI can be based on the internal reference clock generator or on the external oscillator clock or the RC-Oscillator (ACLK).

This mode can be entered from default mode PEI by performing the following steps:

1. Make sure the PLL configuration is valid.
2. Enable the external Oscillator (OSCE bit)
3. Wait for the oscillator to start-up and the PLL being locked (LOCK = 1) and (UPOSC =1)
4. Clear all flags in the CPMUIFLG register to be able to detect any status bit change.
5. Optionally status interrupts can be enabled (CPMUINT register).
6. Select the Oscillator clock as source of the Bus clock (PLLSEL=0)

Loosing PLL lock status (LOCK=0) means loosing the oscillator status information as well (UPOSC=0).

The impact of loosing the oscillator status (UPOSC=0) in PBE mode is as follows:

- PLLSEL is set automatically and the source of the Bus clock is switched back to the PLL clock.
- The PLLCLK is derived from the VCO clock (with its actual frequency) divided by four until the PLL locks again.

Application software needs to be prepared to deal with the impact of loosing the oscillator status at any time.

8.5 Resets

8.5.1 General

All reset sources are listed in Table 8-33. There is only one reset vector for all these reset sources. Refer to MCU specification for reset vector address.

Table 8-33. Reset Summary

Reset Source	Local Enable
Power-On Reset (POR)	None
Low Voltage Reset (LVR)	None
External pin $\overline{\text{RESET}}$	None
PLL Clock Monitor Reset	None
Oscillator Clock Monitor Reset	OSCE Bit in CPMUOSC register and OMRE Bit in CPMUOSC2 register
COP Reset	CR[2:0] in CPMUCOP register

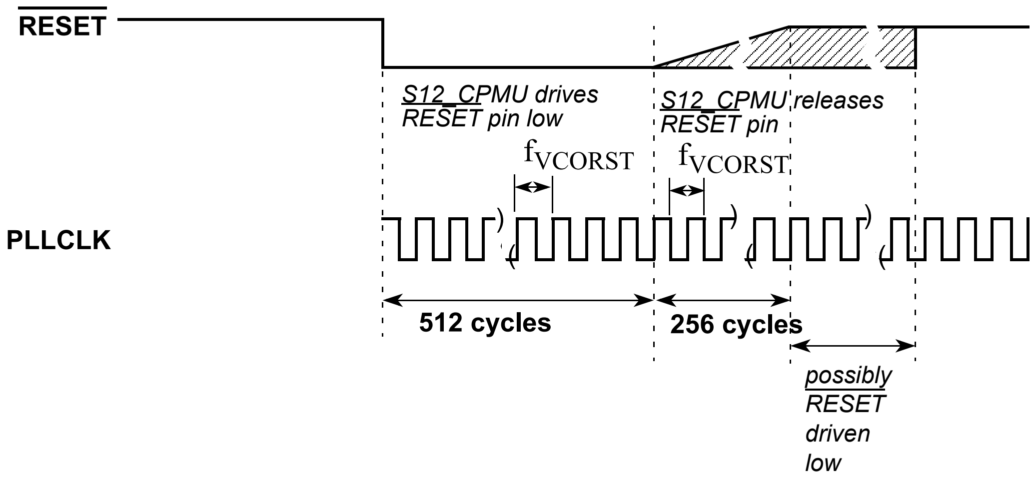
8.5.2 Description of Reset Operation

Upon detection of any reset of Table 8-33, an internal circuit drives the $\overline{\text{RESET}}$ pin low for 512 PLLCLK cycles. After 512 PLLCLK cycles the $\overline{\text{RESET}}$ pin is released. The internal reset of the MCU remains asserted while the reset generator completes the 768 PLLCLK cycles long reset sequence.In case the $\overline{\text{RESET}}$ pin is externally driven low for more than these 768 PLLCLK cycles (External Reset), the internal reset remains asserted longer.

NOTE

While System Reset is asserted the PLLCLK runs with the frequency f_{VCORST}.

Figure 8-40. RESET Timing



8.5.3 Oscillator Clock Monitor Reset

If the external oscillator is enabled (OSCE=1)and the oscillator clock monitor reset is enabled (OMRE=1), then in case of loss of oscillation or the oscillator frequency drops below the failure assert frequency f_{CMFA} (see device electrical characteristics for values), the S12CPMU_UHV_V7 generates an Oscillator Clock Monitor Reset. In Full Stop Mode the external oscillator and the oscillator clock monitor are disabled.

8.5.4 PLL Clock Monitor Reset

In case of loss of PLL clock oscillation or the PLL clock frequency is below the failure assert frequency f_{PMFA} (see device electrical characteristics for values), the S12CPMU_UHV_V7 generates a PLL Clock Monitor Reset[. In Full Stop Mode the PLL and the PLL clock monitor are disabled.

8.5.5 Computer Operating Properly Watchdog (COP) Reset

The COP (free running watchdog timer) enables the user to check that a program is running and sequencing properly. When the COP is being used, software is responsible for keeping the COP from timing out. If the COP times out it is an indication that the software is no longer being executed in the intended sequence; thus COP reset is generated.

The clock source for the COP is either ACLK, IRCCLK or OSCCLK depending on the setting of the COPOSCSEL0 and COPOSCSEL1 bit.

Due to clock domain crossing synchronization there is a latency time to enter and exit Stop Mode if the COP clock source is ACLK and this clock is stopped in Stop Mode. This maximum total latency time is 4 ACLK cycles (2 ACLK cycles for Stop Mode entry and exit each) which must be added to the Stop Mode recovery time t_{STP_REC} from exit of current Stop Mode to entry of next Stop Mode. This latency time occurs no matter which Stop Mode (Full, Pseudo) is currently exited or entered next.

After exit from Stop Mode (Pseudo, Full) for this latency time of 2 ACLK cycles no Stop Mode request (STOP instruction) should be generated to make sure the COP counter can increment at each Stop Mode exit.

Table 8-34 gives an overview of the COP condition (run, static) in Stop Mode depending on legal configuration and status bit settings:

Table 8-34. COP condition (run, static) in Stop Mode

COPOSCSEL1	CSAD	PSTP	PCE	COPOSCSEL0	OSCE	UPOSC	COP counter behavior in Stop Mode (clock source)
1	0	x	x	x	x	x	Run (ACLK)
1	1	x	x	x	x	x	Static (ACLK)
0	x	1	1	1	1	1	Run (OSCCLK)
0	x	1	1	0	0	x	Static (IRCCLK)
0	x	1	1	0	1	x	Static (IRCCLK)
0	x	1	0	0	x	x	Static (IRCCLK)
0	x	1	0	1	1	1	Static (OSCCLK)
0	x	0	1	1	1	1	Static (OSCCLK)
0	x	0	1	0	1	x	Static (IRCCLK)
0	x	0	1	0	0	0	Static (IRCCLK)
0	x	0	0	1	1	1	Satic (OSCCLK)
0	x	0	0	0	1	1	Static (IRCCLK)
0	x	0	0	0	1	0	Static (IRCCLK)
0	x	0	0	0	0	0	Static (IRCCLK)

Three control bits in the CPMUCOP register allow selection of seven COP time-out periods.

When COP is enabled, the program must write $55 and $AA (in this order) to the CPMUARMCOP register during the selected time-out period. Once this is done, the COP time-out period is restarted. If the program fails to do this and the COP times out, a COP reset is generated. Also, if any value other than $55 or $AA is written, a COP reset is generated.

Windowed COP operation is enabled by setting WCOP in the CPMUCOP register. In this mode, writes to the CPMUARMCOP register to clear the COP timer must occur in the last 25% of the selected time-out period. A premature write will immediately reset the part.

In MCU Normal Mode the COP time-out period (CR[2:0]) and COP window (WCOP) setting can be automatically pre-loaded at reset release from NVM memory (if values are defined in the NVM by the application). By default the COP is off and no window COP feature is enabled after reset release via NVM memory. The COP control register CPMUCOP can be written once in an application in MCU Normal Mode to update the COP time-out period (CR[2:0]) and COP window (WCOP) setting loaded from NVM memory at reset release. Any value for the new COP time-out period and COP window setting is allowed except COP off value if the COP was enabled during pre-load via NVM memory.
The COP clock source select bits can not be pre-loaded via NVM memory at reset release. The IRC clock is the default COP clock source out of reset.
The COP clock source select bits (COPOSCSEL0/1) and ACLK clock control bit in Stop Mode (CSAD) can be modified until the CPMUCOP register write once has taken place. Therefore these control bits should be modified before the final COP time-out period and window COP setting is written.
The CPMUCOP register access to modify the COP time-out period and window COP setting in MCU Normal Mode after reset release must be done with the WRTMASK bit cleared otherwise the update is ignored and this access does not count as the write once.

8.5.6 Power-On Reset (POR)

The on-chip POR circuitry detects when the internal supply VDD drops below an appropriate voltage level. The POR is deasserted, if the internal supply VDD exceeds an appropriate voltage level (voltage levels not specified, because the internal supply can not be monitored externally). The POR circuitry is always active. It acts as LVR in Stop Mode.

8.5.7 Low-Voltage Reset (LVR)

The on-chip LVR circuitry detects when one of the supply voltages VDD, VDDX and VDDF drops below an appropriate voltage level. If LVR is deasserted the MCU is fully operational at the specified maximum speed. The LVR assert and deassert levels for the supply voltage VDDX are V_{LVRXA} and V_{LVRXD} and are specified in the device Reference Manual.The LVR circuitry is active in Run- and Wait Mode.

8.6 Interrupts

The interrupt vectors requested by the S12CPMU_UHV_V7 are listed in Table 8-35. Refer to MCU specification for related vector addresses and priorities.

Table 8-35. S12CPMU_UHV_V7 Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
RTI time-out interrupt	I bit	CPMUINT (RTIE)
PLL lock interrupt	I bit	CPMUINT (LOCKIE)
Oscillator status interrupt	I bit	CPMUINT (OSCIE)
Low voltage interrupt	I bit	CPMULVCTL (LVIE)
High temperature interrupt	I bit	CPMUHTCTL (HTIE)
Autonomous Periodical Interrupt	I bit	CPMUAPICTL (APIE)

8.6.1 Description of Interrupt Operation

8.6.1.1 Real Time Interrupt (RTI)

The clock source for the RTI is either IRCCLK or OSCCLK depending on the setting of the RTIOSCSEL bit. In Stop Mode with PSTP=1 (Pseudo Stop Mode), RTIOSCSEL=1 and PRE=1 the RTI continues to run, else the RTI counter halts in Stop Mode.

The RTI can be used to generate hardware interrupts at a fixed periodic rate. If enabled (by setting RTIE=1), this interrupt will occur at the rate selected by the CPMURTI register. At the end of the RTI time-out period the RTIF flag is set to one and a new RTI time-out period starts immediately.

A write to the CPMURTI register restarts the RTI time-out period.

8.6.1.2 PLL Lock Interrupt

The S12CPMU_UHV_V7 generates a PLL Lock interrupt when the lock condition (LOCK status bit) of the PLL changes, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to zero. The PLL Lock interrupt flag (LOCKIF) is set to1 when the lock condition has changed, and is cleared to 0 by writing a 1 to the LOCKIF bit.

8.6.1.3 Oscillator Status Interrupt

When the OSCE bit is 0, then UPOSC stays 0. When OSCE=1 the UPOSC bit is set after the LOCK bit is set.

Upon detection of a status change (UPOSC) the OSCIF flag is set. Going into Full Stop Mode or disabling the oscillator can also cause a status change of UPOSC.

Any change in PLL configuration or any other event which causes the PLL lock status to be cleared leads to a loss of the oscillator status information as well (UPOSC=0).

Oscillator status change interrupts are locally enabled with the OSCIE bit.

NOTE

> Loosing the oscillator status (UPOSC=0) affects the clock configuration of the system[1]. This needs to be dealt with in application software.

8.6.1.4 Low-Voltage Interrupt (LVI)

In FPM the input voltage VDDA is monitored. Whenever VDDA drops below level $V_{LVIA,}$ the status bit LVDS is set to 1. When VDDA rises above level V_{LVID} the status bit LVDS is cleared to 0. An interrupt, indicated by flag LVIF = 1, is triggered by any change of the status bit LVDS if interrupt enable bit LVIE = 1.

8.6.1.5 HTI - High Temperature Interrupt

In FPM the junction temperature T_J is monitored. Whenever T_J exceeds level T_{HTIA} the status bit HTDS is set to 1. Vice versa, HTDS is reset to 0 when T_J get below level T_{HTID}. An interrupt, indicated by flag HTIF = 1, is triggered by any change of the status bit HTDS, if interrupt enable bit HTIE = 1.

8.6.1.6 Autonomous Periodical Interrupt (API)

The API sub-block can generate periodical interrupts independent of the clock source of the MCU. To enable the timer, the bit APIFE needs to be set.

The API timer is either clocked by the Autonomous Clock (ACLK - trimmable internal RC oscillator) or the Bus Clock. Timer operation will freeze when MCU clock source is selected and Bus Clock is turned off. The clock source can be selected with bit APICLK. APICLK can only be written when APIFE is not set.

The APIR[15:0] bits determine the interrupt period. APIR[15:0] can only be written when APIFE is cleared. As soon as APIFE is set, the timer starts running for the period selected by APIR[15:0] bits. When the configured time has elapsed, the flag APIF is set. An interrupt, indicated by flag APIF = 1, is triggered if interrupt enable bit APIE = 1. The timer is re-started automatically again after it has set APIF.

The procedure to change APICLK or APIR[15:0] is first to clear APIFE, then write to APICLK or APIR[15:0], and afterwards set APIFE.

The API Trimming bits ACLKTR[5:0] must be set so the minimum period equals 0.2 ms if stable frequency is desired.

See Table 8-20 for the trimming effect of ACLKTR[5:0].

---

[1] For details please refer to “8.4.6 System Clock Configurations”

NOTE

> The first period after enabling the counter by APIFE might be reduced by API start up delay t_{sdel}.

It is possible to generate with the API a waveform at the external pin API_EXTCLK by setting APIFE and enabling the external access with setting APIEA.

8.7 Initialization/Application Information

8.7.1 General Initialization Information

Usually applications run in MCU Normal Mode.

It is recommended to write the CPMUCOP register in any case from the application program initialization routine after reset no matter if the COP is used in the application or not, even if a configuration is loaded via the flash memory after reset. By doing a "controlled" write access in MCU Normal Mode (with the right value for the application) the write once for the COP configuration bits (WCOP,CR[2:0]) takes place which protects these bits from further accidental change. In case of a program sequencing issue (code runaway) the COP configuration can not be accidentally modified anymore.

8.7.2 Application information for COP and API usage

In many applications the COP is used to check that the program is running and sequencing properly. Often the COP is kept running during Stop Mode and periodic wake-up events are needed to service the COP on time and maybe to check the system status.

For such an application it is recommended to use the ACLK as clock source for both COP and API. This guarantees lowest possible IDD current during Stop Mode. Additionally it eases software implementation using the same clock source for both, COP and API.

The Interrupt Service Routine (ISR) of the Autonomous Periodic Interrupt API should contain the write instruction to the CPMUARMCOP register. The value (byte) written is derived from the "main routine" (alternating sequence of $55 and $AA) of the application software.

Using this method, then in the case of a runtime or program sequencing issue the application "main routine" is not executed properly anymore and the alternating values are not provided properly. Hence the COP is written at the correct time (due to independent API interrupt request) but the wrong value is written (alternating sequence of $55 and $AA is no longer maintained) which causes a COP reset.

If the COP is stopped during any Stop Mode it is recommended to service the COP shortly before Stop Mode is entered.

8.7.3 Application Information for PLL and Oscillator Startup

The following C-code example shows a recommended way of setting up the system clock system using the PLL and Oscillator:

```
/* Procedure proposed by to setup PLL and Oscillator */
/* example for OSC = 4 MHz and Bus Clock = 25MHz, That is VCOCLK = 50MHz */

/* Initialize */
/* PLL Clock = 50 MHz, divide by one */
CPMUPOSTDIV = 0x00;

/* Generally: Whenever changing PLL reference clock (REFCLK) frequency to a higher value */
/* it is recommended to write CPMUSYNR = 0x00 in order to stay within specified */
/* maximum frequency of the MCU */
CPMUSYNR = 0x00;

/* configure PLL reference clock (REFCLK) for usage with Oscillator */
/* OSC=4MHz divide by 4 (3+1) = 1MHz, REFCLK range 1MHz to 2 MHz (REFFRQ[1:0] = 00) */
CPMUREFDV = 0x03;

/* enable external Oscillator, switch PLL reference clock (REFCLK) to OSC */
CPMUOSC = 0x80;

/* multiply REFCLK = 1MHz by 2*(24+1)*1MHz = 50MHz */
/* VCO range 48 to 80 MHz (VCOFRQ[1:0] = 01) */
CPMUSYNR = 0x58;

/* clear all flags, especially LOCKIF and OSCIF */
CPMUIFLG = 0xFF;

/* put your code to loop and wait for the LOCKIF and OSCIF or */
/* poll CPMUIFLG register until both UPOSC and LOCK status are “1” */
/* that is CPMIFLG == 0x1B */

/*..............continue to your main code execution here..............*/

/* in case later in your code you want to disable the Oscillator and use the */
/* 1MHz IRCCLK as PLL reference clock */

/* Generally: Whenever changing PLL reference clock (REFCLK) frequency to a higher value */
/* it is recommended to write CPMUSYNR = 0x00 in order to stay within specified */
/* maximum frequency of the MCU */
CPMUSYNR = 0x00;

/* disable OSC and switch PLL reference clock to IRC */
CPMUOSC = 0x00;

/* multiply REFCLK = 1MHz by 2*(24+1)*1MHz = 50MHz */
/* VCO range 48 to 80 MHz (VCOFRQ[1:0] = 01) */
CPMUSYNR = 0x58;

/* clear all flags, especially LOCKIF and OSCIF */
CPMUIFLG = 0xFF;

/* put your code to loop and wait for the LOCKIF or */
/* poll CPMUIFLG register until both LOCK status is “1” */
/* that is CPMIFLG == 0x18 */

/*..............continue to your main code execution here..............*/
```

Chapter 9
Analog-to-Digital Converter (ADC12B_LBA_V1)

Table 9-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V1.40	02. Oct 2013	entire document	Updated formatting and wording correction for entire document (for technical publications).

9.1 Introduction

The ADC12B_LBA is an n-channel multiplexed input successive approximation analog-to-digital converter. Refer to device electrical specifications for ADC parameters and accuracy.

The List Based Architecture (LBA) provides flexible conversion sequence definition as well as flexible oversampling. The order of channels to be converted can be freely defined. Also, multiple instantiations of the module can be triggered simultaneously (matching sampling point across multiple module instantiations).

There are four register bits which control the conversion flow (please refer to the description of register ADCFLWCTL).
The four conversion flow control bits of register ADCFLWCTL can be modified in two different ways:

- Via data bus accesses
- Via internal interface Signals (Trigger, Restart, LoadOK, and Seq_Abort; see also Figure 9-2). Each Interface Signal is associated with one conversion flow control bit.

For information regarding internal interface connectivity related to the conversion flow control please refer to the device overview of the reference manual.

The ADCFLWCTL register can be controlled via internal interface only or via data bus only or by both depending on the register access configuration bits ACC_CFG[1:0].

The four bits of register ADCFLWCTL reflect the captured request and status of the four internal interface Signals (LoadOK, Trigger, Restart, and Seq_abort; see also Figure 9-2) if access configuration is set accordingly and indicate event progress (when an event is processed and when it is finished).

Conversion flow error situations are captured by corresponding interrupt flags in the ADCEIF register.

There are two conversion flow control modes (Restart Mode, Trigger Mode). Each mode causes a certain behavior of the conversion flow control bits which can be selected according to the application needs.

Please refer to Section 9.4.2.1, “ADC Control Register 0 (ADCCTL_0) and Section 9.5.3.2.4, “The two conversion flow control Mode Configurations for more information regarding conversion flow control.

Because internal components of the ADC are turned on/off with bit ADC_EN, the ADC requires a recovery time period (t_{REC}) after ADC is enabled until the first conversion can be launched via a trigger.

When bit ADC_EN gets cleared (transition from 1’b1 to 1’b0) any ongoing conversion sequence will be aborted and pending results, or the result of current conversion, gets discarded (not stored). The ADC cannot be re-enabled before any pending action or action in process is finished respectively aborted, which could take up to a maximum latency time of $t_{DISABLE}$ (see device level specification for more details).

9.2 Key Features

- Programmer’s Model with List Based Architecture for conversion command and result value organization
- Selectable resolution of 8-bit, 10-bit, or 12-bit
- Channel select control for n external analog input channels
- Provides up to eight device internal channels (please see the device reference manual for connectivity information and Figure 9-2)
- Programmable sample time
- A sample buffer amplifier for channel sampling (improved performance in view to influence of channel input path resistance versus conversion accuracy)
- Left/right justified result data
- Individual selectable VRH_0/1 and VRL_0/1 inputs on a conversion command basis (please see Figure 9-2)
- Special conversions for selected VRH_0/1, VRL_0/1, (VRL_0/1 + VRH_0/1) / 2
- 15 conversion interrupts with flexible interrupt organization per conversion result
- One dedicated interrupt for “End Of List” type commands
- Command Sequence List (CSL) with a maximum number of 64 command entries
- Provides conversion sequence abort
- Restart from top of active Command Sequence List (CSL)
- The Command Sequence List and Result Value List are implemented in double buffered manner (two lists in parallel for each function)
- Conversion Command (CSL) loading possible from System RAM or NVM
- Single conversion flow control register with software selectable access path
- Two conversion flow control modes optimized to different application use cases

9.2.1 Modes of Operation

9.2.1.1 Conversion Modes

This architecture provides **single**, **multiple,** or **continuous conversion** on a **single channel** or on **multiple channels based on the Command Sequence List**.

9.2.1.2 MCU Operating Modes

- **MCU Stop Mode**
  Before issuing an MCU Stop Mode request the ADC should be idle (no conversion or conversion sequence or Command Sequence List ongoing).
  If a conversion, conversion sequence, or CSL is in progress when an MCU Stop Mode request is issued, a Sequence Abort Event occurs automatically and any ongoing conversion finish. After the Sequence Abort Event finishes, if the STR_SEQA bit is set (STR_SEQA=1), then the conversion result is stored and the corresponding flags are set. If the STR_SEQA bit is cleared (STR_SEQA=0), then the conversion result is not stored and the corresponding flags are not set. The microcontroller then enters MCU Stop Mode without SEQAD_IF being set.
  Alternatively, the Sequence Abort Event can be issued by software before an MCU Stop Mode request. As soon as flag SEQAD_IF is set the MCU Stop Mode request can be is issued.
  With the occurrence of the MCU Stop Mode Request until exit from Stop Mode all flow control signals (RSTA, SEQA, LDOK, TRIG) are cleared.

  After exiting MCU Stop Mode, the following happens in the order given with expected event(s) depending on the conversion flow control mode:
  - In ADC conversion flow control mode “Trigger Mode” a Restart Event is expected to simultaneously set bits TRIG and RSTA, causing the ADC to execute the Restart Event (CMD_IDX and RVL_IDX cleared) followed by the Trigger Event. The Restart Event can be generated automatically after exit from MCU Stop Mode if bit AUT_RSTA is set.
  - In ADC conversion flow control mode “Restart Mode”, a Restart Event is expected to set bit RSTA only (ADC already aborted at MCU Stop Mode entry hence bit SEQA must not be set simultaneously) causing the ADC to execute the Restart Event (CDM_IDX and RVL_IDX cleared). The Restart Event can be generated automatically after exit from MCU Stop Mode if bit AUT_RSTA is set.
  - The RVL buffer select (RVL_SEL) is not changed if a CSL is in process at MCU Stop Mode request. Hence the same buffer will be used after exit from Stop Mode that was used when the Stop Mode request occurred.

- **MCU Wait Mode**
  Depending on the ADC Wait Mode configuration bit SWAI, the ADC either continues conversion in MCU Wait Mode or freezes conversion at the next conversion boundary before MCU Wait Mode is entered.

  ADC behavior for configuration SWAI =1'b0:
  The ADC continues conversion during Wait Mode according to the conversion flow control sequence. It is assumed that the conversion flow control sequence is continued (conversion flow control bits TRIG, RSTA, SEQA, and LDOK are serviced accordingly).

  ADC behavior for configuration SWAI = 1'b1:
  At MCU Wait Mode request the ADC should be idle (no conversion or conversion sequence or Command Sequence List ongoing).
  If a conversion, conversion sequence, or CSL is in progress when an MCU Wait Mode request is issued, a Sequence Abort Event occurs automatically and any ongoing conversion finish. After the Sequence Abort Event finishes, if the STR_SEQA bit is set (STR_SEQA=1), then the conversion result is stored and the corresponding flags are set. If the STR_SEQA bit is cleared (STR_SEQA=0), then the conversion result is not stored and the corresponding flags are not set.
  Alternatively the Sequence Abort Event can be issued by software before MCU Wait Mode request. As soon as flag SEQAD_IF is set, the MCU Wait Mode request can be issued.
  With the occurrence of the MCU Wait Mode request until exit from Wait Mode all flow control signals (RSTA, SEQA, LDOK, TRIG) are cleared.
  After exiting MCU Wait Mode, the following happens in the order given with expected event(s) depending on the conversion flow control mode:
  - In ADC conversion flow control mode “Trigger Mode”, a Restart Event is expected to occur. This simultaneously sets bit TRIG and RSTA causing the ADC to execute the Restart Event (CMD_IDX and RVL_IDX cleared) followed by the Trigger Event. The Restart Event can be generated automatically after exit from MCU Wait Mode if bit AUT_RSTA is set.
  - In ADC conversion flow control mode “Restart Mode”, a Restart Event is expected to set bit RSTA only (ADC already aborted at MCU Wait Mode entry hence bit SEQA must not be set simultaneously) causing the ADC to execute the Restart Event (CDM_IDX and RVL_IDX cleared). The Restart Event can be generated automatically after exit from MCU Wait Mode if bit AUT_RSTA is set.
  - The RVL buffer select (RVL_SEL) is not changed if a CSL is in process at MCU Wait Mode request. Hence the same RVL buffer will be used after exit from Wait Mode that was used when Wait Mode request occurred.

NOTE

In principle, the MCU could stay in Wait Mode for a shorter period of time than the ADC needs to abort an ongoing conversion (range of µµµµµs). Therefore in case a Sequence Abort Event is issued automatically due to MCU Wait Mode request a following Restart Event after exit from MCU Wait Mode can not be executed before ADC has finished this Sequence Abort Event. The Restart Event is detected but it is pending.
This applies in case MCU Wait Mode is exited before ADC has finished the Sequence Abort Event and a Restart Event is issued immediately after exit from MCU Wait Mode. Bit READY can be used by software to detect when the Restart Event can be issued without latency time in processing the event (see also Figure 9-1).

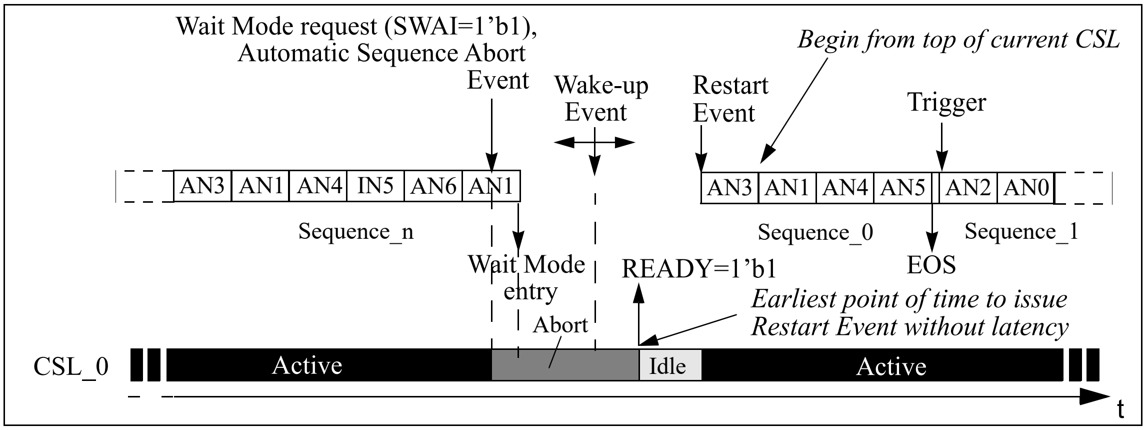


Figure 9-1. Conversion Flow Control Diagram - Wait Mode (SWAI=1'b1, AUT_RSTA=1'b0)

- **MCU Freeze Mode**
  Depending on the ADC Freeze Mode configuration bit FRZ_MOD, the ADC either continues conversion in Freeze Mode or freezes conversion at next conversion boundary before the MCU Freeze Mode is entered. After exit from MCU Freeze Mode with previously frozen conversion sequence the ADC continues the conversion with the next conversion command and all ADC interrupt flags are unchanged during MCU Freeze Mode.

9.2.2 Block Diagram

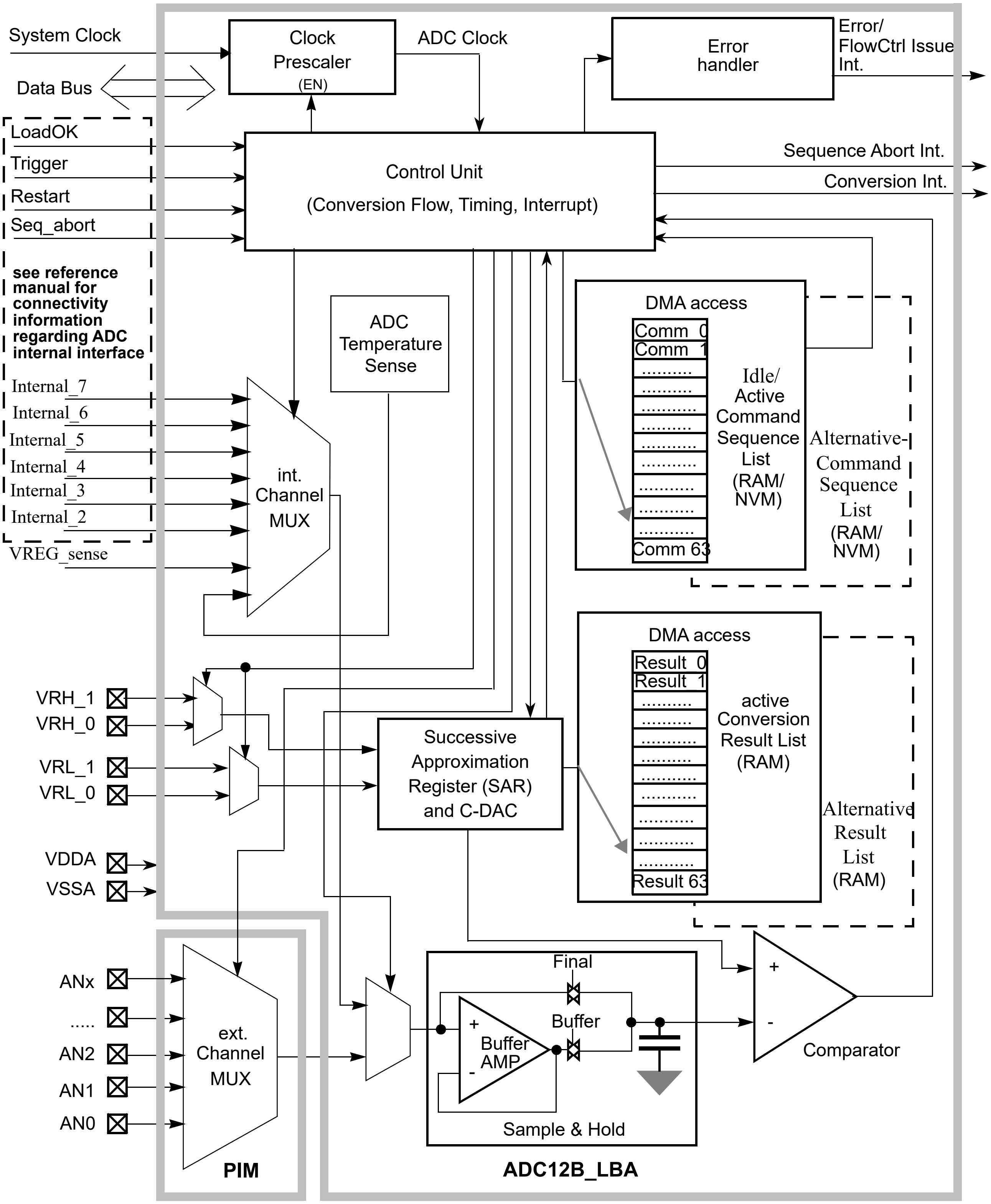


Figure 9-2. ADC12B_LBA Block Diagram

9.3 Signal Description

This section lists all inputs to the ADC12B_LBA block.

9.3.1 Detailed Signal Descriptions

9.3.1.1 AN*x* (*x* = *n*,..., 2, 1, 0)

This pin serves as the analog input Channel *x*. The maximum input channel number is *n*. Please refer to the device reference manual for the maximum number of input channels.

9.3.1.2 VRH_0, VRH_1, VRL_0, VRL_1

VRH_0/1 are the high reference voltages, VRL0/1 are the low reference voltages for a ADC conversion selectable on a conversion command basis. Please refer to the device reference manual for availability and connectivity of these pins.

9.3.1.3 VDDA, VSSA

These pins are the power supplies for the analog circuitry of the ADC12B_LBA block.

9.4 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the ADC12B_LBA.

9.4.1 Module Memory Map

Figure 9-3 gives an overview of all ADC12B_LBA registers.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	ADCCTL_0	R	ADC_EN	ADC_SR	FRZ_MOD	SWAI	ACC_CFG[1:0]		STR_SEQA	MOD_CFG
		W								
0x0001	ADCCTL_1	R	CSL_BMOD	RVL_BMOD	SMOD_ACC	AUT_RSTA	0	0	0	0
		W					▨	▨	▨	▨
0x0002	ADCSTS	R	CSL_SEL	RVL_SEL	DBECC_ERR	Reserved	READY	0	0	0
		W			▨	▨	▨	▨	▨	▨
0x0003	ADCTIM	R	0	PRS[6:0]						
		W	▨							
0x0004	ADCFMT	R	DJM	0	0	0	0	SRES[2:0]		
		W		▨	▨	▨	▨			
0x0005	ADCFLWCTL	R	SEQA	TRIG	RSTA	LDOK	0	0	0	0
		W					▨	▨	▨	▨
0x0006	ADCEIE	R	IA_EIE	CMD_EIE	EOL_EIE	Reserved	TRIG_EIE	RSTAR_EIE	LDOK_EIE	0
		W								▨
0x0007	ADCIE	R	SEQAD_IE	CONIF_OIE	Reserved	0	0	0	0	0
		W				▨	▨	▨	▨	▨
0x0008	ADCEiF	R	IA_EIF	CMD_EIF	EOL_EIF	Reserved	TRIG_EIF	RSTAR_EIF	LDOK_EIF	0
		W								▨
0x0009	ADCIF	R	SEQAD_IF	CONIF_OIF	Reserved	0	0	0	0	0
		W				▨	▨	▨	▨	▨
0x000A	ADCCONIE_0	R	CON_IE[15:8]							
		W								
0x000B	ADCCONIE_1	R	CON_IE[7:1]							EOL_IE
		W								
0x000C	ADCCONIF_0	R	CON_IF[15:8]							
		W								
0x000D	ADCCONIF_1	R	CON_IF[7:1]							EOL_IF
		W								
0x000E	ADCIMDRI_0	R	CSL_IMD	RVL_IMD	0	0	0	0	0	0
			▨	▨	▨	▨	▨	▨	▨	▨
0x000F	ADCIMDRI_1	R	0	0	RIDX_IMD[5:0]					
		W	▨	▨	▨					

▨ = Unimplemented or Reserved

Figure 9-3. ADC12B_LBA Register Summary (Sheet 1 of 3)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0010	ADCEOLRI	R	CSL_EOL	RVL_EOL	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0011	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0012	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0013	Reserved	R	Reserved	Reserved					0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0014	ADCCMD_0	R	CMD_SEL		0	0	INTFLG_SEL[3:0]			
		W			▨	▨				
0x0015	ADCCMD_1	R	VRH_SEL	VRL_SEL	CH_SEL[5:0]					
		W								
0x0016	ADCCMD_2	R	SMP[4:0]					0	0	Reserved
		W						▨	▨	
0x0017	ADCCMD_3	R	Reserved	Reserved	Reserved					
		W								
0x0018	Reserved	R	Reserved							
		W								
0x0019	Reserved	R	Reserved							
		W								
0x001A	Reserved	R	Reserved							
		W								
0x001B	Reserved	R	Reserved							
		W								
0x001C	ADCCIDX	R	0	0	CMD_IDX[5:0]					
		W	▨	▨	▨					
0x001D	ADCCBP_0	R	CMD_PTR[23:16]							
		W								
0x001E	ADCCBP_1	R	CMD_PTR[15:8]							
		W								
0x001F	ADCCBP_2	R	CMD_PTR[7:2]						0	0
		W							▨	▨
0x0020	ADCRIDX	R	0	0	RES_IDX[5:0]					
		W	▨	▨	▨					
0x0021	ADCRBP_0	R	0	0	0	0	RES_PTR[19:16]			
		W	▨	▨	▨	▨				
0x0022	ADCRBP_1	R	RES_PTR[15:8]							
		W								
0x0023	ADCRBP_2	R	RES_PTR[7:2]						0	0
		W							▨	▨
0x0024	ADCCROFF0	R	0	CMDRES_OFF0[6:0]						
		W	▨	▨						
0x0025	ADCCROFF1	R	0	CMDRES_OFF1[6:0]						
		W	▨							
0x0026	Reserved	R	0	0	0	0	Reserved			
		W	▨	▨	▨	▨				

▨ = Unimplemented or Reserved

Figure 9-3. ADC12B_LBA Register Summary (Sheet 2 of 3)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0027	Reserved	R	Reserved							
		W								
0x0028	Reserved	R	Reserved						0	0
		W							▨	▨
0x0029	Reserved	R	Reserved	0	Reserved					
		W	▨	▨	▨					
0x002A- 0x003F	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 9-3. ADC12B_LBA Register Summary (Sheet 3 of 3)

9.4.2 Register Descriptions

This section describes in address order all the ADC12B_LBA registers and their individual bits.

9.4.2.1 ADC Control Register 0 (ADCCTL_0)

Module Base + 0x0000

	15	14	13	12	11	10	9	8
R W	ADC_EN	ADC_SR	FRZ_MOD	SWAI	ACC_CFG[1:0]		STR_SEQA	MOD_CFG
Reset	0	0	0	0	0	0	0	0

░ = Unimplemented or Reserved

Figure 9-4. ADC Control Register 0 (ADCCTL_0)

Read: Anytime

Write:

- Bits ADC_EN, ADC_SR, FRZ_MOD and SWAI writable anytime
- Bits MOD_CFG, STR_SEQA and ACC_CFG[1:0] writable if bit ADC_EN clear or bit SMOD_ACC set

Table 9-2. ADCCTL_0 Field Descriptions

Field	Description
15 ADC_EN	**ADC Enable Bit** — This bit enables the ADC (e.g. sample buffer amplifier etc.) and controls accessibility of ADC register bits. When this bit gets cleared any ongoing conversion sequence will be aborted and pending results or the result of current conversion gets discarded (not stored). The ADC cannot be re-enabled before any pending action or action in process is finished or aborted, which could take up to a maximum latency time of $t_{DISABLE}$ (see device reference manual for more details). Because internal components of the ADC are turned on/off with this bit, the ADC requires a recovery time period (t_{REC}) after ADC is enabled until the first conversion can be launched via a trigger. 0 ADC disabled. 1 ADC enabled.
14 ADC_SR	**ADC Soft-Reset** — This bit causes an ADC Soft-Reset if set after a severe error occurred (see list of severe errors in Section 9.4.2.9, “ADC Error Interrupt Flag Register (ADCEIF) that causes the ADC to cease operation). It clears all overrun flags and error flags and forces the ADC state machine to its idle state. It also clears the Command Index Register, the Result Index Register, and the CSL_SEL and RVL_SEL bits (to be ready for a new control sequence to load new command and start execution again from top of selected CSL). A severe error occurs if an error flag is set which cause the ADC to cease operation. In order to make the ADC operational again an ADC Soft-Reset must be issued. Once this bit is set it can not be cleared by writing any value. It is cleared only by ADC hardware after the Soft-Reset has been executed. 0 No ADC Soft-Reset issued. 1 Issue ADC Soft-Reset.
13 FRZ_MOD	**Freeze Mode Configuration** — This bit influences conversion flow during Freeze Mode. 0 ADC continues conversion in Freeze Mode. 1 ADC freezes the conversion at next conversion boundary at Freeze Mode entry.
12 SWAI	**Wait Mode Configuration** — This bit influences conversion flow during Wait Mode. 0 ADC continues conversion in Wait Mode. 1 ADC halts the conversion at next conversion boundary at Wait Mode entry.

Table 9-2. ADCCTL_0 Field Descriptions (continued)

Field	Description
11-10 ACC_CFG[1:0]	**ADCFLWCTL Register Access Configuration** — These bits define if the register ADCFLWCTL is controlled via internal interface only or data bus only or both. See Table 9-3. for more details.
9 STR_SEQA	**Control Of Conversion Result Storage and RSTAR_EIF flag setting at Sequence Abort or Restart Event** — This bit controls conversion result storage and RSTAR_EIF flag setting when a Sequence Abort Event or Restart Event occurs as follows: *If STR_SEQA = 1'b0 and if a:* • Sequence Abort Event or Restart Event is issued during a conversion the data of this conversion is not stored and the respective conversion complete flag is not set • Restart Event only is issued before the last conversion of a CSL is finished and no Sequence Abort Event is in process (SEQA clear) causes the RSTA_EIF error flag to be asserted and bit SEQA gets set by hardware *If STR_SEQA = 1'b1 and if a:* • Sequence Abort Event or Restart Event is issued during a conversion the data of this conversion is stored and the respective conversion complete flag is set and Intermediate Result Information Register is updated. • Restart Event only occurs during the last conversion of a CSL and no Sequence Abort Event is in process (SEQA clear) does not set the RSTA_EIF error flag • Restart Event only is issued before the CSL is finished and no Sequence Abort Event is in process (SEQA clear) causes the RSTA_EIF error flag to be asserted and bit SEQA gets set by hardware
8 MOD_CFG	**(Conversion Flow Control) Mode Configuration** — This bit defines the conversion flow control after a Restart Event and after execution of the "End Of List" command type: - Restart Mode - Trigger Mode (For more details please see also section Section 9.5.3.2, "Introduction of the Programmer's Model and following.) 0 "Restart Mode" selected. 1 "Trigger Mode" selected.

Table 9-3. ADCFLWCTL Register Access Configurations

ACC_CFG[1]	ACC_CFG[0]	ADCFLWCTL Access Mode
0	0	None of the access paths is enabled (default / reset configuration)
0	1	Single Access Mode - Internal Interface (ADCFLWCTL access via internal interface only)
1	0	Single Access Mode - Data Bus (ADCFLWCTL access via data bus only)
1	1	Dual Access Mode (ADCFLWCTL register access via internal interface and data bus)

NOTE

Each conversion flow control bit (SEQA, RSTA, TRIG, LDOK) must be controlled by software or internal interface according to the requirements described in Section 9.5.3.2.4, “The two conversion flow control Mode Configurations and overview summary in Table 9-10.

9.4.2.2 ADC Control Register 1 (ADCCTL_1)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	CSL_BMOD	RVL_BMOD	SMOD_ACC	AUT_RSTA	0	0	0	0
W					▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-5. ADC Control Register 1 (ADCCTL_1)

Read: Anytime

Write:

- Bit CSL_BMOD and RVL_BMOD writable if bit ADC_EN clear or bit SMOD_ACC set
- Bit SMOD_ACC only writable in MCU Special Mode
- Bit AUT_RSTA writable anytime

Table 9-4. ADCCTL_1 Field Descriptions

Field	Description
7 CSL_BMOD	**CSL Buffer Mode Select Bit** — This bit defines the CSL buffer mode. This bit is only writable if ADC_EN is clear. 0 CSL single buffer mode. 1 CSL double buffer mode.
6 RVL_BMOD	**RVL Buffer Mode Select Bit** — This bit defines the RVL buffer mode. 0 RVL single buffer mode 1 RVL double buffer mode
5 SMOD_ACC	**Special Mode Access Control Bit** — This bit controls register access rights in MCU Special Mode. This bit is automatically cleared when leaving MCU Special Mode. Note: When this bit is set also the ADCCMD register is writeable via the data bus to allow modification of the current command for debugging purpose. But this is only possible if the current command is not already processed (conversion not started). Please see access details given for each register. Care must be taken when modifying ADC registers while bit SMOD_ACC is set to not corrupt a possible ongoing conversion. 0 Normal user access - Register write restrictions exist as specified for each bit. 1 Special access - Register write restrictions are lifted.
4 AUT_RSTA	**Automatic Restart Event after exit from MCU Stop and Wait Mode (SWAI set)** — This bit controls if a Restart Event is automatically generated after exit from MCU Stop Mode or Wait Mode with bit SWAI set. It can be configured for ADC conversion flow control mode “Trigger Mode” and “Restart Mode” (anytime during application runtime). 0 No automatic Restart Event after exit from MCU Stop Mode. 1 Automatic Restart Event occurs after exit from MCU Stop Mode.

9.4.2.3 ADC Status Register (ADCSTS)

It is important to note that if flag DBECC_ERR is set the ADC ceases operation. In order to make the ADC operational again an ADC Soft-Reset must be issued. An ADC Soft-Reset clears bits CSL_SEL and RVL_SEL.

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	CSL_SEL	RVL_SEL	DBECC_ERR	Reserved	READY	0	0	0
W			▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	1	0	0	0

▨ = Unimplemented or Reserved

Figure 9-6. ADC Status Register (ADCSTS)

Read: Anytime

Write:

- Bits CSL_SEL and RVL_SEL anytime if bit ADC_EN is clear or bit SMOD_ACC is set
- Bits DBECC_ERR and READY not writable

Table 9-5. ADCSTS Field Descriptions

Field	Description
7 CSL_SEL	**Command Sequence List Select bit** — This bit controls and indicates which ADC Command List is active. This bit can only be written if ADC_EN bit is clear. This bit toggles in CSL double buffer mode when no conversion or conversion sequence is ongoing and bit LDOK is set and bit RSTA is set. In CSL single buffer mode this bit is forced to 1’b0 by bit CSL_BMOD. 0 ADC Command List 0 is active. 1 ADC Command List 1 is active.
6 RVL_SEL	**Result Value List Select Bit** — This bit controls and indicates which ADC Result List is active. This bit can only be written if bit ADC_EN is clear. After storage of the initial Result Value List this bit toggles in RVL double buffer mode whenever the conversion result of the first conversion of the current CSL is stored or a CSL got aborted. In RVL single buffer mode this bit is forced to 1’b0 by bit RVL_BMOD. Please see also Section 9.2.1.2, “MCU Operating Modes for information regarding Result List usage in case of Stop or Wait Mode. 0 ADC Result List 0 is active. 1 ADC Result List 1 is active.
5 DBECC_ERR	**Double Bit ECC Error Flag** — This flag indicates that a double bit ECC error occurred during conversion command load or result storage and ADC ceases operation. In order to make the ADC operational again an ADC Soft-Reset must be issued. This bit is cleared if bit ADC_EN is clear. 0 No double bit ECC error occurred. 1 A double bit ECC error occurred.
3 READY	**Ready For Restart Event Flag** — This flag indicates that ADC is in its idle state and ready for a Restart Event. It can be used to verify after exit from Wait Mode if a Restart Event can be issued and processed immediately without any latency time due to an ongoing Sequence Abort Event after exit from MCU Wait Mode (see also the Note in Section 9.2.1.2, “MCU Operating Modes). 0 ADC not in idle state. 1 ADC is in idle state.

9.4.2.4 ADC Timing Register (ADCTIM)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	PRS[6:0]						
W	▨							
Reset	0	0	0	0	0	1	0	1

▨ = Unimplemented or Reserved

Figure 9-7. ADC Timing Register (ADCTIM))

Read: Anytime

Write: These bits are writable if bit ADC_EN is clear or bit SMOD_ACC is set

Table 9-6. ADCTIM Field Descriptions

Field	Description
6-0 PRS[6:0]	**ADC Clock Prescaler** — These 7bits are the binary prescaler value PRS. The ADC conversion clock frequency is calculated as follows: $f_{ATDCLK} = \frac{f_{BUS}}{2x(PRS+1)}$ Refer to Device Specification for allowed frequency range of f_{ATDCLK}.

9.4.2.5 ADC Format Register (ADCFMT)

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	DJM	0	0	0	0	SRES[2:0]		
W		▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-8. ADC Format Register (ADCFMT)

Read: Anytime

Write: Bits DJM and SRES[2:0] are writable if bit ADC_EN clear or bit SMOD_ACC set

Table 9-7. ADCFMT Field Descriptions

Field	Description
7 DJM	**Result Register Data Justification** — Conversion result data format is always unsigned. This bit controls justification of conversion result data in the conversion result list. 0 Left justified data in the conversion result list. 1 Right justified data in the conversion result list.
2-0 SRES[2:0]	**ADC Resolution Select** — These bits select the resolution of conversion results. See Table 9-8 for coding.

Table 9-8. Selectable Conversion Resolution

SRES[2]	SRES[1]	SRES[0]	ADC Resolution
0	0	0	8-bit data
0	0	1	Reserved[1]
0	1	0	10-bit data
0	1	1	Reserved[1]
1	0	0	12-bit data
1	x	x	Reserved[1]

[1] Reserved settings cause a severe error at ADC conversion start whereby the CMD_EIF flag is set and ADC ceases operation

9.4.2.6 ADC Conversion Flow Control Register (ADCFLWCTL)

Bit set and bit clear instructions should not be used to access this register.

When the ADC is enabled the bits of ADCFLWCTL register can be modified after a latency time of three Bus Clock cycles.
All bits are cleared if bit ADC_EN is clear or via ADC soft-reset.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	SEQA	TRIG	RSTA	LDOK	0	0	0	0
W					▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-9. ADC Conversion Flow Control Register (ADCFLWCTL)

Read: Anytime

Write:

- Bits SEQA, TRIG, RSTA, LDOK can only be set if bit ADC_EN is set.
- Writing 1’b0 to any of these bits does not have an effect

Timing considerations (Trigger Event - channel sample start) depending on ADC mode configuration:

- Restart Mode
  When the Restart Event has been processed (initial command of current CSL is loaded) it takes two Bus Clock cycles plus two ADC conversion clock cycles (pump phase) from the Trigger Event (bit TRIG set) until the select channel starts to sample.
  During a conversion sequence (back to back conversions) it takes five Bus Clock cycles plus two ADC conversion clock cycles (pump phase) from current conversion period end until the newly selected channel is sampled in the following conversion period.
- Trigger Mode
  When a Restart Event occurs a Trigger Event is issued simultaneously. The time required to process the Restart Event is mainly defined by the internal read data bus availability and therefore can vary. In this mode the Trigger Event is processed immediately after the Restart Event is finished and both conversion flow control bits are cleared simultaneously. From de-assert of bit TRIG until sampling begins five Bus Clock cycles are required. Hence from occurrence of a Restart Event until channel sampling it takes five Bus Clock cycles plus an uncertainty of a few Bus Clock cycles.

For more details regarding the sample phase please refer to Section 9.5.2.2, “Sample and Hold Machine with Sample Buffer Amplifier.

Table 9-9. ADCFLWCTL Field Descriptions

Field	Description
7 SEQA	**Conversion Sequence Abort Event** — This bit indicates that a conversion sequence abort event is in progress. When this bit is set the ongoing conversion sequence and current CSL will be aborted at the next conversion boundary. This bit gets cleared when the ongoing conversion sequence is aborted and ADC is idle. This bit can only be set if bit ADC_EN is set. This bit is cleared if bit ADC_EN is clear. *Data Bus Control:* This bit can be controlled via the data bus if access control is configured accordingly via ACC_CFG[1:0]. Writing a value of 1’b0 does not clear the flag. Writing a one to this bit does not clear it but causes an overrun if the bit has already been set. See Section 9.5.3.2.6, “Conversion flow control in case of conversion sequence control bit overrun scenarios for more details. *Internal Interface Control:* This bit can be controlled via the internal interface Signal “Seq_Abort” if access control is configured accordingly via ACC_CFG[1:0]. After being set an additional request via the internal interface Signal “Seq_Abort” causes an overrun. See also conversion flow control in case of overrun situations. *General:* In both conversion flow control modes (Restart Mode and Trigger Mode) when bit RSTA gets set automatically bit SEQA gets set when the ADC has not reached one of the following scenarios: - A Sequence Abort request is about to be executed or has been executed. - “End Of List” command type has been executed or is about to be executed In case bit SEQA is set automatically the Restart error flag RSTA_EIF is set to indicate an unexpected Restart Request. 0 No conversion sequence abort request. 1 Conversion sequence abort request.
6 TRIG	**Conversion Sequence Trigger Bit** — This bit starts a conversion sequence if set and no conversion or conversion sequence is ongoing. This bit is cleared when the first conversion of a sequence starts to sample. This bit can only be set if bit ADC_EN is set. This bit is cleared if bit ADC_EN is clear. *Data Bus Control:* This bit can be controlled via the data bus if access control is configured accordingly via ACC_CFG[1:0]. Writing a value of 1’b0 does not clear the flag. After being set this bit can not be cleared by writing a value of 1’b1 instead the error flag TRIG_EIF is set. See also Section 9.5.3.2.6, “Conversion flow control in case of conversion sequence control bit overrun scenarios for more details. *Internal Interface Control:* This bit can be controlled via the internal interface Signal “Trigger” if access control is configured accordingly via ACC_CFG[1:0]. After being set an additional request via internal interface Signal “Trigger“ causes the flag TRIG_EIF to be set. 0 No conversion sequence trigger. 1 Trigger to start conversion sequence.

Table 9-9. ADCFLWCTL Field Descriptions (continued)

Field	Description
5 RSTA	**Restart Event (Restart from Top of Command Sequence List)** — This bit indicates that a Restart Event is executed. The ADC loads the conversion command from top of the active Sequence Command List when no conversion or conversion sequence is ongoing. This bit is cleared when the first conversion command of the sequence from top of active Sequence Command List has been loaded into the ADCCMD register. This bit can only be set if bit ADC_EN is set. This bit is cleared if bit ADC_EN is clear. *Data Bus Control:* This bit can be controlled via the data bus if access control is configured accordingly via ACC_CFG[1:0]. Writing a value of 1'b0 does not clear the flag. Writing a one to this bit does not clear it but causes an overrun if the bit has already been set. See also Section 9.5.3.2.6, "Conversion flow control in case of conversion sequence control bit overrun scenarios for more details. *Internal Interface Control:* This bit can be controlled via the internal interface Signal "Restart" if access control is configured accordingly via ACC_CFG[1:0]. After being set an additional request via internal interface Signal "Restart" causes an overrun. See conversion flow control in case of overrun situations for more details. General: In conversion flow control mode "Trigger Mode" when bit RSTA gets set bit TRIG is set simultaneously if one of the following has been executed: - "End Of List" command type has been executed or is about to be executed - Sequence Abort Event 0 Continue with commands from active Sequence Command List. 1 Restart from top of active Sequence Command List.
4 LDOK	**Load OK for alternative Command Sequence List** — This bit indicates if the preparation of the alternative Sequence Command List is done and Command Sequence List must be swapped with the Restart Event. This bit is cleared when bit RSTA is set (Restart Event executed) and the Command Sequence List got swapped. This bit can only be set if bit ADC_EN is set. This bit is cleared if bit ADC_EN is clear. This bit is forced to zero if bit CSL_BMOD is clear. *Data Bus Control:* This bit can be controlled via the data bus if access control is configured accordingly via ACC_CFG[1:0]. Writing a value of 1'b0 does not clear the flag. To set bit LDOK the bits LDOK and RSTA must be written simultaneously. After being set this bit can not be cleared by writing a value of 1'b1. See also Section 9.5.3.2.6, "Conversion flow control in case of conversion sequence control bit overrun scenarios for more details. *Internal Interface Control:* This bit can be controlled via the internal interface Signal "LoadOK" and "Restart" if access control is configured accordingly via ACC_CFG[1:0]. With the assertion of Interface Signal "Restart" the interface Signal "LoadOK" is evaluated and bit LDOK set accordingly (bit LDOK set if Interface Signal "LoadOK" asserted when Interface Signal "Restart" asserts). General: Only in "Restart Mode" if a Restart Event occurs without bit LDOK being set the error flag LDOK_EIF is set except when the respective Restart Request occurred after or simultaneously with a Sequence Abort Request. The LDOK_EIF error flag is also not set in "Restart Mode" if the first Restart Event occurs after: - ADC got enabled - Exit from Stop Mode - ADC Soft-Reset 0 Load of alternative list done. 1 Load alternative list.

Table 9-10. Summary of Conversion Flow Control Bit Scenarios

RSTA	TRIG	SEQA	LDOK	Conversion Flow Control Mode	Conversion Flow Control Scenario
0	0	0	0	Both Modes	Valid
0	0	0	1	Both Modes	Can Not Occur
0	0	1	0	Both Modes	Valid[5]
0	0	1	1	Both Modes	Can Not Occur
0	1	0	0	Both Modes	Valid[2]
0	1	0	1	Both Modes	Can Not Occur
0	1	1	0	Both Modes	Can Not Occur
0	1	1	1	Both Modes	Can Not Occur
1	0	0	0	Both Modes	Valid[4]
1	0	0	1	Both Modes	Valid[1,4]
1	0	1	0	Both Modes	Valid[3,4,5]
1	0	1	1	Both Modes	Valid[1,3,4,5]
1	1	0	0	“Restart Mode”	Error flag TRIG_EIF set
				“Trigger Mode”	Valid[2,4,6]
1	1	0	1	“Restart Mode”	Error flag TRIG_EIF set
				“Trigger Mode”	Valid[1,2,4,6]
1	1	1	0	“Restart Mode”	Error flag TRIG_EIF set
				“Trigger Mode”	Valid[2,3,4,5,6]
1	1	1	1	“Restart Mode”	Error flag TRIG_EIF set
				“Trigger Mode”	Valid[1,2,3,4,5,6]

[1] Swap CSL buffer

[2] Start conversion sequence

[3] Prevent RSTA_EIF and LDOK_EIF

[4] Load conversion command from top of CSL

[5] Abort any ongoing conversion, conversion sequence and CSL

[6] Bit TRIG set automatically in Trigger Mode

For a detailed description of all conversion flow control bit scenarios please see also Section 9.5.3.2.4, “The two conversion flow control Mode Configurations, Section 9.5.3.2.5, “The four ADC conversion flow control bits and Section 9.5.3.2.6, “Conversion flow control in case of conversion sequence control bit overrun scenarios

9.4.2.7 ADC Error Interrupt Enable Register (ADCEIE)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	IA_EIE	CMD_EIE	EOL_EIE	Reserved	TRIG_EIE	RSTAR_EIE	LDOK_EIE	0
W								▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-10. ADC Error Interrupt Enable Register (ADCEIE)

Read: Anytime

Write: Anytime

Table 9-11. ADCEIE Field Descriptions

Field	Description
7 IA_EIE	**Illegal Access Error Interrupt Enable Bit** — This bit enables the illegal access error interrupt. 0 Illegal access error interrupt disabled. 1 Illegal access error interrupt enabled.
6 CMD_EIE	**Command Value Error Interrupt Enable Bit** — This bit enables the command value error interrupt. 0 Command value interrupt disabled. 1 Command value interrupt enabled.
5 EOL_EIE	**"End Of List" Error Interrupt Enable Bit** — This bit enables the "End Of List" error interrupt. 0 "End Of List" error interrupt disabled. 1 "End Of List" error interrupt enabled.
3 TRIG_EIE	**Conversion Sequence Trigger Error Interrupt Enable Bit** — This bit enables the conversion sequence trigger error interrupt. 0 Conversion sequence trigger error interrupt disabled. 1 Conversion sequence trigger error interrupt enabled.
2 RSTAR_EIE	**Restart Request Error Interrupt Enable Bit**— This bit enables the restart request error interrupt. 0 Restart Request error interrupt disabled. 1 Restart Request error interrupt enabled.
1 LDOK_EIE	**Load OK Error Interrupt Enable Bit** — This bit enables the Load OK error interrupt. 0 Load OK error interrupt disabled. 1 Load OK error interrupt enabled.

9.4.2.8 ADC Interrupt Enable Register (ADCIE)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	SEQAD_IE	CONIF_OIE	Reserved	0	0	0	0	0
W				▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-11. ADC Interrupt Enable Register (ADCIE)

Read: Anytime

Write: Anytime

Table 9-12. ADCIE Field Descriptions

Field	Description
7 SEQAD_IE	**Conversion Sequence Abort Done Interrupt Enable Bit** — This bit enables the conversion sequence abort event done interrupt. 0 Conversion sequence abort event done interrupt disabled. 1 Conversion sequence abort event done interrupt enabled.
6 CONIF_OIE	**ADCCONIF Register Flags Overrun Interrupt Enable** — This bit enables the flag which indicates if an overrun situation occurred for one of the CON_IF[15:1] flags or for the EOL_IF flag. 0 No ADCCONIF Register Flag overrun occurred. 1 ADCCONIF Register Flag overrun occurred.

9.4.2.9 ADC Error Interrupt Flag Register (ADCEIF)

If one of the following error flags is set the ADC ceases operation:

- IA_EIF
- CMD_EIF
- EOL_EIF
- TRIG_EIF

In order to make the ADC operational again an ADC Soft-Reset must be issued which clears above listed error interrupt flags.

The error interrupt flags RSTAR_EIF and LDOK_EIF do not cause the ADC to cease operation. If set the ADC continues operation. Each of the two bits can be cleared by writing a value of 1’b1. Both bits are also cleared if an ADC Soft-Reset is issued.

All bits are cleared if bit ADC_EN is clear. Writing any flag with value 1’b0 does not clear a flag. Writing any flag with value 1’b1 does not set the flag.

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	IA_EIF	CMD_EIF	EOL_EIF	Reserved	TRIG_EIF	RSTAR_EIF	LDOK_EIF	0
W								▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-12. ADC Error Interrupt Flag Register (ADCEIF)

Read: Anytime

Write:

- Bits RSTAR_EIF and LDOK_EIF are writable anytime
- Bits IA_EIF, CMD_EIF, EOL_EIF and TRIG_EIF are not writable

Table 9-13. ADCEIF Field Descriptions

Field	Description
7 IA_EIF	**Illegal Access Error Interrupt Flag** — This flag indicates that storing the conversion result caused an illegal access error or conversion command loading from outside system RAM or NVM area occurred. The ADC ceases operation if this error flag is set (issue of type severe). 0 No illegal access error occurred. 1 An illegal access error occurred.
6 CMD_EIF	**Command Value Error Interrupt Flag** — This flag indicates that an invalid command is loaded (Any command that contains reserved bit settings) or illegal format setting selected (reserved SRES[2:0] bit settings). The ADC ceases operation if this error flag is set (issue of type severe). 0 Valid conversion command loaded. 1 Invalid conversion command loaded.
5 EOL_EIF	**“End Of List” Error Interrupt Flag** — This flag indicates a missing “End Of List” command type in current executed CSL. The ADC ceases operation if this error flag is set (issue of type severe). 0 No “End Of List” error. 1 “End Of List” command type missing in current executed CSL.

Table 9-13. ADCEIF Field Descriptions (continued)

Field	Description
3 TRIG_EIF	**Trigger Error Interrupt Flag** — This flag indicates that a trigger error occurred. This flag is set in “Restart” Mode when a conversion sequence got aborted and no Restart Event occurred before the Trigger Event or if the Trigger Event occurred before the Restart Event was finished (conversion command has been loaded). This flag is set in “Trigger” Mode when a Trigger Event occurs before the Restart Event is issued to start conversion of the initial Command Sequence List. In “Trigger” Mode only a Restart Event is required to start conversion of the initial Command Sequence List. This flag is set when a Trigger Event occurs before a conversion sequence got finished. This flag is also set if a Trigger occurs while a Trigger Event is just processed - first conversion command of a sequence is beginning to sample (see also Section 9.5.3.2.6, “Conversion flow control in case of conversion sequence control bit overrun scenarios). This flag is also set if the Trigger Event occurs automatically generated by hardware in “Trigger Mode” due to a Restart Event and simultaneously a Trigger Event is generated via data bus or internal interface. The ADC ceases operation if this error flag is set (issue of type severe). 0 No trigger error occurred. 1 A trigger error occurred.
2 RSTAR_EIF	**Restart Request Error Interrupt Flag** — This flag indicates a flow control issue. It is set when a Restart Request occurs after a Trigger Event and before one of the following conditions was reached: - The “End Of List” command type has been executed - Depending on bit STR_SEQA if the “End Of List” command type is about to be executed - The current CSL has been aborted or is about to be aborted due to a Sequence Abort Request. The ADC continues operation if this error flag is set. This flag is not set for Restart Request overrun scenarios (see also Section 9.5.3.2.6, “Conversion flow control in case of conversion sequence control bit overrun scenarios). 0 No Restart request error situation occurred. 1 Restart request error situation occurred.
1 LDOK_EIF	**Load OK Error Interrupt Flag** — This flag can only be set in “Restart Mode”. It indicates that a Restart Request occurred without LDOK. This flag is not set if a Sequence Abort Event is already in process (bit SEQA set) when the Restart Request occurs or a Sequence Abort Request occurs simultaneously with the Restart Request. The LDOK_EIF error flag is also not set in “Restart Mode” if the first Restart Event occurs after: - ADC got enabled - Exit from Stop Mode - ADC Soft-Reset - ADC used in CSL single buffer mode The ADC continues operation if this error flag is set. 0 No Load OK error situation occurred. 1 Load OK error situation occurred.

9.4.2.10 ADC Interrupt Flag Register (ADCIF)

After being set any of these bits can be cleared by writing a value of 1’b1 or via ADC soft-reset (bit ADC_SR). All bits are cleared if bit ADC_EN is clear. Writing any flag with value 1’b0 does not clear the flag. Writing any flag with value 1’b1 does not set the flag.

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	SEQAD_IF	CONIF_OIF	Reserved	0	0	0	0	0
W				▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-13. ADC Interrupt Flag Register (ADCIF)

Read: Anytime

Write: Anytime

Table 9-14. ADCIF Field Descriptions

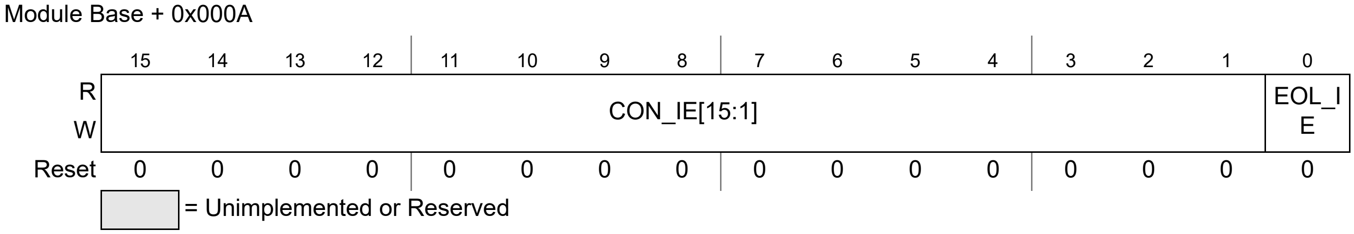
Field	Description
7 SEQAD_IF	**Conversion Sequence Abort Done Interrupt Flag** — This flag is set when the Sequence Abort Event has been executed except the Sequence Abort Event occurred by hardware in order to be able to enter MCU Stop Mode or Wait Mode with bit SWAI set.This flag is also not set if the Sequence Abort request occurs during execution of the last conversion command of a CSL and bit STR_SEQA being set. 0 No conversion sequence abort request occurred. 1 A conversion sequence abort request occurred.
6 CONIF_OIF	**ADCCONIF Register Flags Overrun Interrupt Flag** — This flag indicates if an overrun situation occurred for one of the CON_IF[15:1] flags or for the EOL_IF flag. In RVL single buffer mode (RVL_BMOD clear) an overrun of the EOL_IF flag is not indicated (For more information please see Note below). 0 No ADCCONIF Register Flag overrun occurred. 1 ADCCONIF Register Flag overrun occurred.

NOTE

In RVL double buffer mode a conversion interrupt flag (CON_IF[15:1]) or End Of List interrupt flag (EOL_IF) overrun is detected if one of these bits is set when it should be set again due to conversion command execution.

In RVL single buffer mode a conversion interrupt flag (CON_IF[15:1]) overrun is detected only. The overrun is detected if any of the conversion interrupt flags (CON_IF[15:1]) is set while the first conversion result of a CSL is stored (result of first conversion from top of CSL is stored).

9.4.2.11 ADC Conversion Interrupt Enable Register (ADCCONIE)



Module Base + 0x000A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CON_IE[15:1]															EOL_IE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-14. ADC Conversion Interrupt Enable Register (ADCCONIE)

Read: Anytime

Write: Anytime

Table 9-15. ADCCONIE Field Descriptions

Field	Description
15-1 CON_IE[15:1]	**Conversion Interrupt Enable Bits** — These bits enable the individual interrupts which can be triggered via interrupt flags CON_IF[15:1]. 0 ADC conversion interrupt disabled. 1 ADC conversion interrupt enabled.
0 EOL_IE	**End Of List Interrupt Enable Bit** — This bit enables the end of conversion sequence list interrupt. 0 End of list interrupt disabled. 1 End of list interrupt enabled.

9.4.2.12 ADC Conversion Interrupt Flag Register (ADCCONIF)

After being set any of these bits can be cleared by writing a value of 1’b1. All bits are cleared if bit ADC_EN is clear or via ADC soft-reset (bit ADC_SR set). Writing any flag with value 1’b0 does not clear the flag. Writing any flag with value 1’b1 does not set the flag.

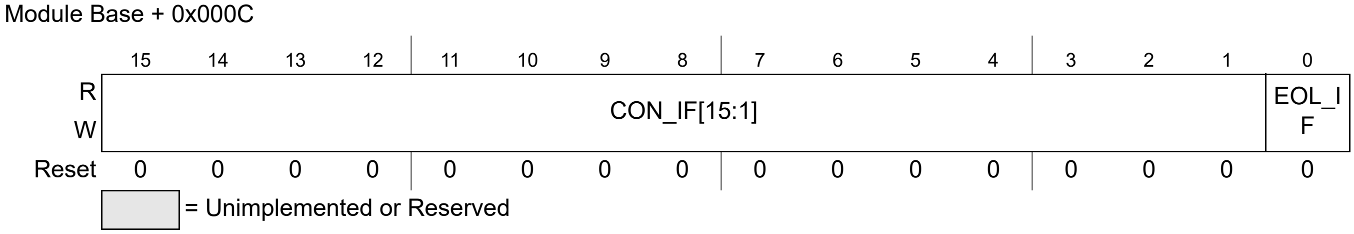


Figure 9-15. ADC Conversion Interrupt Flag Register (ADCCONIF)

Read: Anytime

Write: Anytime

Table 9-16. ADCCONIF Field Descriptions

Field	Description
15-1 CON_IF[15:1]	**Conversion Interrupt Flags** — These bits could be set by the binary coded interrupt select bits INTFLG_SEL[3:0] when the corresponding conversion command has been processed and related data has been stored to RAM. See also notes below.
0 EOL_IF	**End Of List Interrupt Flag** — This bit is set by the binary coded conversion command type select bits CMD_SEL[1:0] for “end of list” type of commands and after such a command has been processed and the related data has been stored RAM. See also second note below

NOTE

These bits can be used to indicate if a certain packet of conversion results is available. Clearing a flag indicates that conversion results have been retrieved by software and the flag can be used again (see also Section 9.8.6, “RVL swapping in RVL double buffer mode and related registers ADCIMDRI and ADCEOLRI.

NOTE

Overrun situation of a flag CON_IF[15:1] and EOL_IF are indicated by flag CONIF_OIF.

9.4.2.13 ADC Intermediate Result Information Register (ADCIMDRI)

This register is cleared when bit ADC_SR is set or bit ADC_EN is clear.

Module Base + 0x000E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSL_IMD	RVL_IMD	0	0	0	0	0	0	0	0	RIDX_IMD[5:0]					
W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-16. ADC Intermediate Result Information Register (ADCIMDRI)

Read: Anytime

Write: Never

Table 9-17. ADCIMDRI Field Descriptions

Field	Description
15 CSL_IMD	**Active CSL At Intermediate Event** — This bit indicates the active (used) CSL at the occurrence of a conversion interrupt flag (CON_IF[15:1]) (occurrence of an intermediate result buffer fill event) or when a Sequence Abort Event gets executed. 0 CSL_0 active (used) when a conversion interrupt flag (CON_IF[15:1]) got set. 1 CSL_1 active (used) when a conversion interrupt flag (CON_IF[15:1]) got set.
14 RVL_IMD	**Active RVL At Intermediate Event** — This bit indicates the active (used) RVL buffer at the occurrence of a conversion interrupt flag (CON_IF[15:1]) (occurrence of an intermediate result buffer fill event) or when a Sequence Abort Event gets executed. 0 RVL_0 active (used) when a conversion interrupt flag (CON_IF[15:1]) got set. 1 RVL_1 active (used) when a conversion interrupt flag (CON_IF[15:1]) got set.
5-0 RIDX_IMD[5:0]	**RES_IDX Value At Intermediate Event** — These bits indicate the result index (RES_IDX) value at the occurrence of a conversion interrupt flag (CON_IF[15:1]) (occurrence of an intermediate result buffer fill event) or occurrence of EOL_IF flag or when a Sequence Abort Event gets executed to abort an ongoing conversion (the result index RES_IDX is captured at the occurrence of a result data store). When a Sequence Abort Event has been processed flag SEQAD_IF is set and the RES_IDX value of the last stored result is provided. Hence in case an ongoing conversion is aborted the RES_IDX value captured in RIDX_IMD bits depends on bit STORE_SEQA: - STORE_SEQA =1: The result index of the aborted conversion is provided - STORE_SEQA =0: The result index of the last stored result at abort execution time is provided In case a CSL is aborted while no conversion is ongoing (ADC waiting for a Trigger Event) the last captured result index is provided. In case a Sequence Abort Event was initiated by hardware due to MCU entering Stop Mode or Wait Mode with bit SWAI set, the result index of the last stored result is captured by bits RIDX_IMD but flag SEQAD_IF is not set.

NOTE

The register ADCIMDRI is updated and simultaneously a conversion interrupt flag CON_IF[15:1] occurs when the corresponding conversion command (conversion command with INTFLG_SEL[3:0] set) has been processed and related data has been stored to RAM.

9.4.2.14 ADC End Of List Result Information Register (ADCEOLRI)

This register is cleared when bit ADC_SR is set or bit ADC_EN is clear.

Module Base + 0x0010

	7	6	5	4	3	2	1	0
R	CSL_EOL	RVL_EOL	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-17. ADC End Of List Result Information Register (ADCEOLRI)

Read: Anytime

Write: Never

Table 9-18. ADCEOLRI Field Descriptions

Field	Description
7 CSL_EOL	**Active CSL When "End Of List" Command Type Executed** — This bit indicates the active (used) CSL when a "End Of List" command type has been executed and related data has been stored to RAM. 0 CSL_0 active when "End Of List" command type executed. 1 CSL_1 active when "End Of List" command type executed.
6 RVL_EOL	**Active RVL When "End Of List" Command Type Executed** — This bit indicates the active (used) RVL when a "End Of List" command type has been executed and related data has been stored to RAM. 0 RVL_0 active when "End Of List" command type executed. 1 RVL_1 active when "End Of List" command type executed.

NOTE

The conversion interrupt EOL_IF occurs and simultaneously the register ADCEOLRI is updated when the "End Of List" conversion command type has been processed and related data has been stored to RAM.

9.4.2.15 ADC Command Register 0 (ADCCMD_0)

Module Base + 0x0014

	31	30	29	28	27	26	25	24
R	CMD_SEL		0	0	INTFLG_SEL[3:0]			
W			▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-18. ADC Command Register 0 (ADCCMD_0)

Read: Anytime

Write: Only writable if bit SMOD_ACC is set
(see also Section 9.4.2.2, “ADC Control Register 1 (ADCCTL_1) bit SMOD_ACC description for more details)

Table 9-19. ADCCMD_0 Field Descriptions

Field	Description
31-30 CMD_SEL[1:0]	**Conversion Command Select Bits** — These bits define the type of current conversion described in Table 9-20.
27-24 INTFLG_SEL[3:0]	**Conversion Interrupt Flag Select Bits** — These bits define which interrupt flag is set in the ADCIFH/L register at the end of current conversion.The interrupt flags ADCIF[15:1] are selected via binary coded bits INTFLG_SEL[3:0]. See also Table 9-21

NOTE

If bit SMOD_ACC is set modifying this register must be done carefully - only when no conversion and conversion sequence is ongoing.

Table 9-20. Conversion Command Type Select

CMD_SEL[1]	CMD_SEL[0]	Conversion Command Type Description
0	0	Normal Conversion
0	1	End Of Sequence (Wait for Trigger to execute next sequence or for a Restart)
1	0	End Of List (Automatic wrap to top of CSL and Continue Conversion)
1	1	End Of List (Wrap to top of CSL and: - In “Restart Mode” wait for Restart Event followed by a Trigger - In “Trigger Mode” wait for Trigger or Restart Event)

Table 9-21. Conversion Interrupt Flag Select

CON_IF[15:1]	INTFLG_SEL[3]	INTFLG_SEL[2]	INTFLG_SEL[1]	INTFLG_SEL[0]	Comment
0x0000	0	0	0	0	No flag set
0x0001	0	0	0	1	Only one flag can be set (one hot coding)
0x0002	0	0	1	0	
0x0004	0	0	1	1	
0x0008	0	1	0	0	
0x0010	0	1	0	1	
....	...	...	...	...	
0x0800	1	1	0	0	
0x1000	1	1	0	1	
0x2000	1	1	1	0	
0x4000	1	1	1	1	

9.4.2.16 ADC Command Register 1 (ADCCMD_1)

A command which contains reserved bit settings causes the error flag CMD_EIF being set and ADC cease operation.

Module Base + 0x0015

	23	22	21	20	19	18	17	16
R W	VRH_SEL	VRL_SEL	CH_SEL[5:0]					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-19. ADC Command Register 1 (ADCCMD_1)

Read: Anytime

Write: Only writable if bit SMOD_ACC is set
(see also Section 9.4.2.2, “ADC Control Register 1 (ADCCTL_1) bit SMOD_ACC description for more details)

Table 9-22. ADCCMD_1 Field Descriptions

Field	Description
23 VRH_SEL	**Reference High Voltage Select Bit** — This bit selects the high voltage reference for current conversion. 0 VRH_0 input selected as high voltage reference. 1 VRH_1 input selected as high voltage reference.
22 VRL_SEL	**Reference Low Voltage Select Bit** — This bit selects the voltage reference for current conversion. 0 VRL_0 input selected as low voltage reference. 1 VRL_1 input selected as low voltage reference.
21-16 CH_SEL[5:0]	**ADC Input Channel Select Bits** — These bits select the input channel for the current conversion. See Table 9-23 for channel coding information.

NOTE

If bit SMOD_ACC is set modifying this register must be done carefully - only when no conversion and conversion sequence is ongoing.

Table 9-23. Analog Input Channel Select

CH_SEL[5]	CH_SEL[4]	CH_SEL[3]	CH_SEL[2]	CH_SEL[1]	CH_SEL[0]	Analog Input Channel
0	0	0	0	0	0	VRL_0/1
0	0	0	0	0	1	VRH_0/1
0	0	0	0	1	0	(VRH_0/1 + VRL_0/1) / 2
0	0	0	0	1	1	Reserved
0	0	0	1	0	0	Reserved
0	0	0	1	0	1	Reserved
0	0	0	1	1	0	Reserved

Table 9-23. Analog Input Channel Select

CH_SEL[5]	CH_SEL[4]	CH_SEL[3]	CH_SEL[2]	CH_SEL[1]	CH_SEL[0]	Analog Input Channel
0	0	0	1	1	1	Reserved
0	0	1	0	0	0	Reserved
0	0	1	0	0	1	Internal_1 (Vreg_3v3 sense)
0	0	1	0	1	0	Internal_2
0	0	1	0	1	1	Internal_3
0	0	1	1	0	0	Internal_4
0	0	1	1	0	1	Internal_5
0	0	1	1	1	0	Internal_6
0	0	1	1	1	1	Internal_7
0	1	0	0	0	0	AN0
0	1	0	0	0	1	AN1
0	1	0	0	1	0	AN2
0	1	0	0	1	1	AN3
0	1	0	1	0	0	AN4
0	1	x	x	x	x	ANx
1	x	x	x	x	x	Reserved

NOTE

ANx in Table 9-23 is the maximum number of implemented analog input channels on the device. Please refer to the device overview of the reference manual for details regarding number of analog input channels.

9.4.2.17 ADC Command Register 2 (ADCCMD_2)

A command which contains reserved bit settings causes the error flag CMD_EIF being set and ADC cease operation.

Module Base + 0x0016

	15	14	13	12	11	10	9	8
R	SMP[4:0]					0	0	Reserved
W						▨	▨	
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-20. ADC Command Register 2 (ADCCMD_2)

Read: Anytime

Write: Only writable if bit SMOD_ACC is set
(see also Section 9.4.2.2, "ADC Control Register 1 (ADCCTL_1) bit SMOD_ACC description for more details)

Table 9-24. ADCCMD_2 Field Descriptions

Field	Description
15-11 SMP[4:0]	**Sample Time Select Bits** — These four bits select the length of the sample time in units of ADC conversion clock cycles. Note that the ADC conversion clock period is itself a function of the prescaler value (bits PRS[6:0]). Table 9-25 lists the available sample time lengths.

NOTE

If bit SMOD_ACC is set modifying this register must be done carefully - only when no conversion and conversion sequence is ongoing.

Table 9-25. Sample Time Select

SMP[4]	SMP[3]	SMP[2]	SMP[1]	SMP[0]	Sample Time in Number of ADC Clock Cycles
0	0	0	0	0	4
0	0	0	0	1	5
0	0	0	1	0	6
0	0	0	1	1	7
0	0	1	0	0	8
0	0	1	0	1	9
0	0	1	1	0	10
0	0	1	1	1	11
0	1	0	0	0	12
0	1	0	0	1	13

Table 9-25. Sample Time Select

SMP[4]	SMP[3]	SMP[2]	SMP[1]	SMP[0]	Sample Time in Number of ADC Clock Cycles
0	1	0	1	0	14
0	1	0	1	1	15
0	1	1	0	0	16
0	1	1	0	1	17
0	1	1	1	0	18
0	1	1	1	1	19
1	0	0	0	0	20
1	0	0	0	1	21
1	0	0	1	0	22
1	0	0	1	1	23
1	0	1	0	0	24
1	0	1	0	1	Reserved
1	0	1	1	0	Reserved
1	0	1	1	1	Reserved
1	1	x	x	x	Reserved

9.4.2.18 ADC Command Register 3 (ADCCMD_3)

Module Base + 0x0017

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-21. ADC Command Register 3 (ADCCMD_3)

9.4.2.19 ADC Command Index Register (ADCCIDX)

It is important to note that these bits do not represent absolute addresses instead it is a sample index (object size 32bit).

Module Base + 0x001C

	7	6	5	4	3	2	1	0
R	0	0	CMD_IDX[5:0]					
W	▨	▨	▨					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-22. ADC Command Index Register (ADCCIDX)

Read: Anytime

Write: NA

Table 9-26. ADCCIDX Field Descriptions

Field	Description
5-0 CMD_IDX [5:0]	**ADC Command Index Bits** — These bits represent the command index value for the conversion commands relative to the two CSL start addresses in the memory map. These bits do not represent absolute addresses instead it is a sample index (object size 32bit). See also Section 9.5.3.2.2, "Introduction of the two Command Sequence Lists (CSLs) for more details.

9.4.2.20 ADC Command Base Pointer Register (ADCCBP)

Module Base + 0x001D

	23	22	21	20	19	18	17	16
R W	CMD_PTR[23:16]							
Reset	0	0	0	0	0	0	0	0

Module Base + 0x001E

	15	14	13	12	11	10	9	8
R W	CMD_PTR[15:8]							
Reset	0	0	0	0	0	0	0	0

Module Base + 0x001F

	7	6	5	4	3	2	1	0
R	CMD_PTR[7:2]						0	0
W							▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-23. ADC Command Base Pointer Registers (ADCCBP_0, ADCCBP_1, ADCCBP_2))

Read: Anytime

Write: Bits CMD_PTR[23:2] writable if bit ADC_EN clear or bit SMOD_ACC set

Table 9-27. ADCCBP Field Descriptions

Field	Description
23-2 CMD_PTR [23:2]	**ADC Command Base Pointer Address** — These bits define the base address of the two CSL areas inside the system RAM or NVM of the memory map. They are used to calculate the final address from which the conversion commands will be loaded depending on which list is active. For more details see Section 9.5.3.2.2, “Introduction of the two Command Sequence Lists (CSLs).

9.4.2.21 ADC Result Index Register (ADCRIDX)

It is important to note that these bits do not represent absolute addresses instead it is a sample index (object size 16bit).

Module Base + 0x0020

	7	6	5	4	3	2	1	0
R	0	0	RES_IDX[5:0]					
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-24. ADC Result Index Register (ADCRIDX)

Read: Anytime

Write: NA

Table 9-28. ADCRIDX Field Descriptions

Field	Description
5-0 RES_IDX[5:0]	**ADC Result Index Bits** — These read only bits represent the index value for the conversion results relative to the two RVL start addresses in the memory map. These bits do not represent absolute addresses instead it is a sample index (object size 16bit). See also Section 9.5.3.2.3, “Introduction of the two Result Value Lists (RVLs) for more details.

9.4.2.22 ADC Result Base Pointer Register (ADCRBP)

Module Base + 0x0021

	23	22	21	20	19	18	17	16
R	0	0	0	0	RES_PTR[19:16]			
W	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

Module Base + 0x0022

	15	14	13	12	11	10	9	8
R	RES_PTR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x0023

	7	6	5	4	3	2	1	0
R	RES_PTR[7:2]						0	0
W							▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-25. ADC Result Base Pointer Registers (ADCRBP_0, ADCRBP_1, ADCRBP_2))

Read: Anytime

Write: Bits RES_PTR[19:2] writeable if bit ADC_EN clear or bit SMOD_ACC set

Table 9-29. ADCRBP Field Descriptions

Field	Description
19-2 RES_PTR[19:2]	**ADC Result Base Pointer Address** — These bits define the base address of the list areas inside the system RAM of the memory map to which conversion results will be stored to at the end of a conversion. These bits can only be written if bit ADC_EN is clear. See also Section 9.5.3.2.3, “Introduction of the two Result Value Lists (RVLs).

9.4.2.23 ADC Command and Result Offset Register 0 (ADCCROFF0)

Module Base + 0x0024

	7	6	5	4	3	2	1	0
R	0	CMDRES_OFF0[6:0]						
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-26. ADC Command and Result Offset Register 0 (ADCCROFF0)

Read: Anytime

Write: NA

Table 9-30. ADCCROFF0 Field Descriptions

Field	Description
6-0 CMDRES_OFF0 [6:0]	**ADC Command and Result Offset Value** — These read only bits represent the conversion command and result offset value relative to the conversion command base pointer address and result base pointer address in the memory map to refer to CSL_0 and RVL_0. It is used to calculate the address inside the system RAM to which the result at the end of the current conversion is stored to and the area (RAM or NVM) from which the conversion commands are loaded from. This is a zero offset (null offset) which can not be modified. These bits do not represent absolute addresses instead it is a sample offset (object size 16bit for RVL, object size 32bit for CSL). See also Section 9.5.3.2.2, "Introduction of the two Command Sequence Lists (CSLs) and Section 9.5.3.2.3, "Introduction of the two Result Value Lists (RVLs) for more details.

9.4.2.24 ADC Command and Result Offset Register 1 (ADCCROFF1)

It is important to note that these bits do not represent absolute addresses instead it is an sample offset (object size 16bit for RVL, object size 32bit for CSL).

Module Base + 0x0025

	7	6	5	4	3	2	1	0
R	0	CMDRES_OFF1[6:0]						
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-27. ADC Command and Result Offset Register 1 (ADCCROFF1)

Read: Anytime

Write: These bits are writable if bit ADC_EN clear or bit SMOD_ACC set

Table 9-31. ADCCROFF1 Field Descriptions

Field	Description
6-0 CMDRES_OFF1 [6:0]	**ADC Result Address Offset Value** — These bits represent the conversion command and result offset value relative to the conversion command base pointer address and result base pointer address in the memory map to refer to CSL_1 and RVL_1. It is used to calculate the address inside the system RAM to which the result at the end of the current conversion is stored to and the area (RAM or NVM) from which the conversion commands are loaded from. These bits do not represent absolute addresses instead it is an sample offset (object size 16bit for RVL, object size 32bit for CSL).,These bits can only be modified if bit ADC_EN is clear. See also Section 9.5.3.2.2, "Introduction of the two Command Sequence Lists (CSLs) and Section 9.5.3.2.3, "Introduction of the two Result Value Lists (RVLs) for more details.

9.5 Functional Description

9.5.1 Overview

The ADC12B_LBA consists of an analog sub-block and a digital sub-block. It is a successive approximation analog-to-digital converter including a sample-and-hold mechanism and an internal charge scaled C-DAC (switched capacitor scaled digital-to-analog converter) with a comparator to realize the successive approximation algorithm.

9.5.2 Analog Sub-Block

The analog sub-block contains all analog circuits (sample and hold, C-DAC, analog Comparator, and so on) required to perform a single conversion. Separate power supplies VDDA and VSSA allow noise from the MCU circuitry to be isolated from the analog sub-block for improved accuracy.

9.5.2.1 Analog Input Multiplexer

The analog input multiplexers connect one of the external or internal analog input channels to the sample and hold storage node.

9.5.2.2 Sample and Hold Machine with Sample Buffer Amplifier

The Sample and Hold Machine controls the storage and charge of the storage node (sample capacitor) to the voltage level of the analog signal at the selected ADC input channel. This architecture employs the advantage of reduced crosstalk between channels.

The sample buffer amplifier is used to raise the effective input impedance of the A/D machine, so that external components (higher bandwidth or higher impedance connected as specified) are less significant to accuracy degradation.

During the sample phase, the analog input connects first via a sample buffer amplifier with the storage node always for two ADC clock cycles (“Buffer” sample time). For the remaining sample time (“Final” sample time) the storage node is directly connected to the analog input source. Please see also Figure 9-28 for illustration and the Appendix of the device reference manual for more details.
The input analog signals are unipolar and must be within the potential range of VSSA to VDDA.
During the hold process, the analog input is disconnected from the storage node.

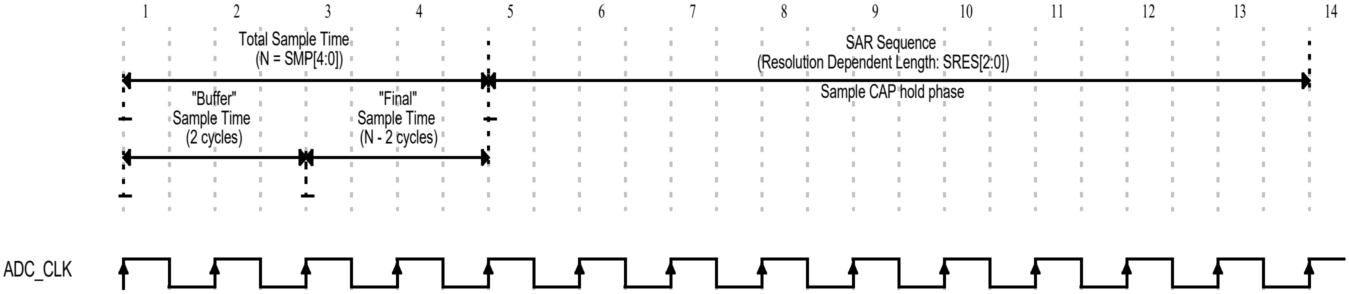


Figure 9-28. Sampling and Conversion Timing Example (8-bit Resolution, 4 Cycle Sampling)

Please note that there is always a pump phase of two ADC_CLK cycles before the sample phase begins, hence glitches during the pump phase could impact the conversion accuracy for short sample times.

9.5.3 Digital Sub-Block

The digital sub-block contains a list-based programmer's model and the control logic for the analog sub-block circuits.

9.5.3.1 Analog-to-Digital (A/D) Machine

The A/D machine performs the analog-to-digital conversion. The resolution is program selectable to be either 8- or 10- or 12 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the sampled and stored analog voltage with a series of binary coded discrete voltages.

By following a binary search algorithm, the A/D machine identifies the discrete voltage that is nearest to the sampled and stored voltage.

Only analog input signals within the potential range of VRL_0/1 to VRH_0/1 (A/D reference potentials) will result in a non-railed digital output code.

9.5.3.2 Introduction of the Programmer's Model

The ADC_LBA provides a programmer's model that uses a system memory list-based architecture for definition of the conversion command sequence and conversion result handling.
The Command Sequence List (CSL) and Result Value List (RVL) are implemented in double buffered manner and the buffer mode is user selectable for each list (bits CSL_BMOD, RVL_BMOD). The 32-bit wide conversion command is double buffered and the currently active command is visible in the ADC register map at ADCCMD register space.

9.5.3.2.1 Introduction of The Command Sequence List (CSL) Format

A Command Sequence List (CSL) contains up to 64 conversion commands. A user selectable number of successive conversion commands in the CSL can be grouped as a command sequence. This sequence of conversion commands is successively executed by the ADC at the occurrence of a Trigger Event. The commands of a sequence are successively executed until an “End Of Sequence” or “End Of List” command type identifier in a command is detected (command type is coded via bits CMD_SEL[1:0]). The number of successive conversion commands that belong to a command sequence and the number of command sequences inside the CSL can be freely defined by the user and is limited by the 64 conversion commands a CSL can contain. A CSL must contain at least one conversion command and one “end of list” command type identifier. The minimum number of command sequences inside a CSL is zero and the maximum number of command sequences is 63. A command sequence is defined with bits CMD_SEL[1:0] in the register ADCCMD_M by defining the end of a conversion sequence. The Figure 9-29 and Figure 9-30 provides examples of a CSL.

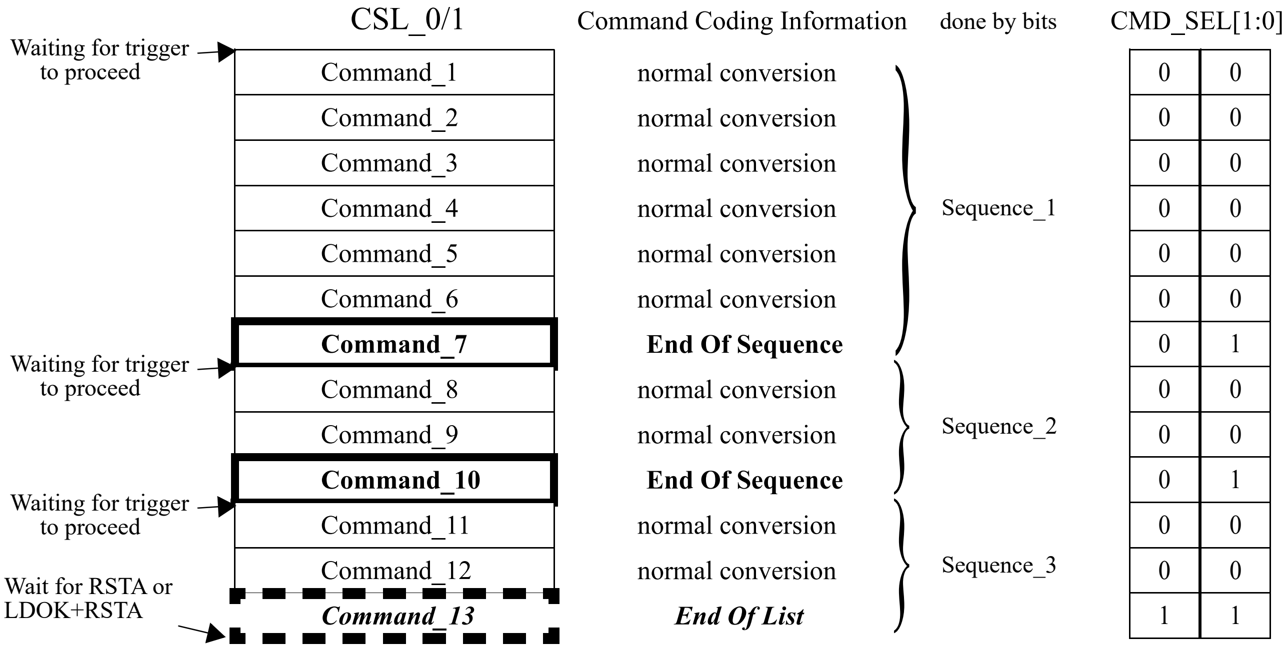


Figure 9-29. Example CSL with sequences and an “End Of List” command type identifier

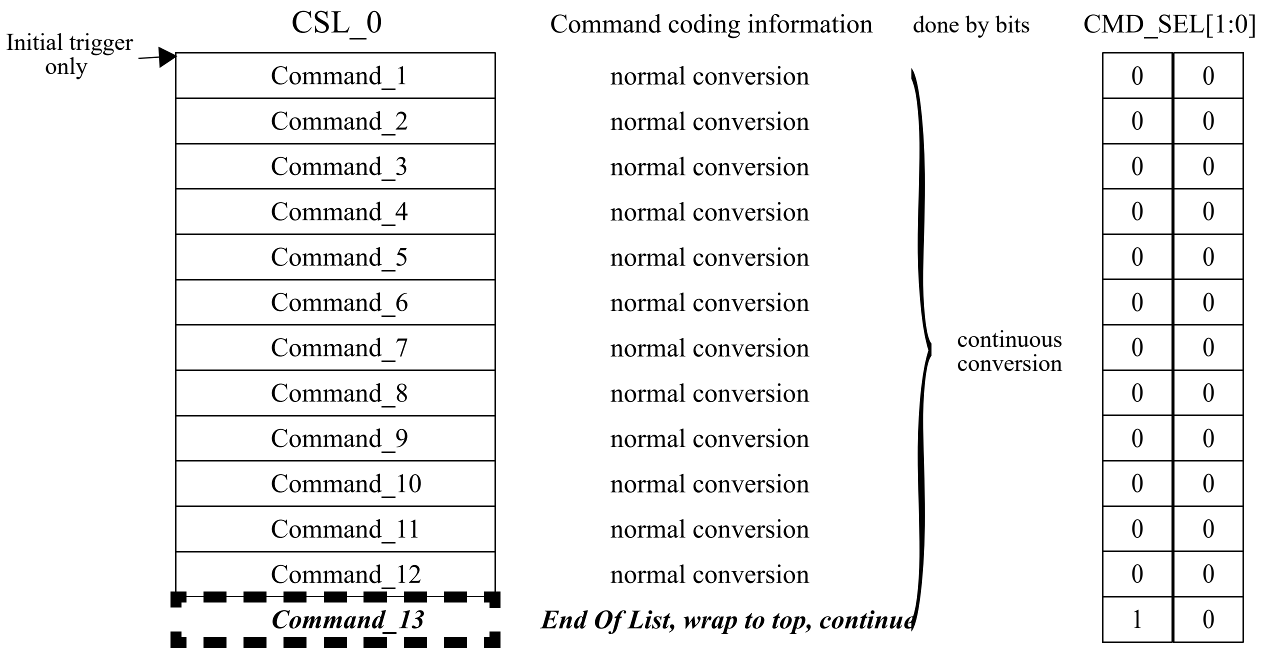


Figure 9-30. Example CSL for continues conversion

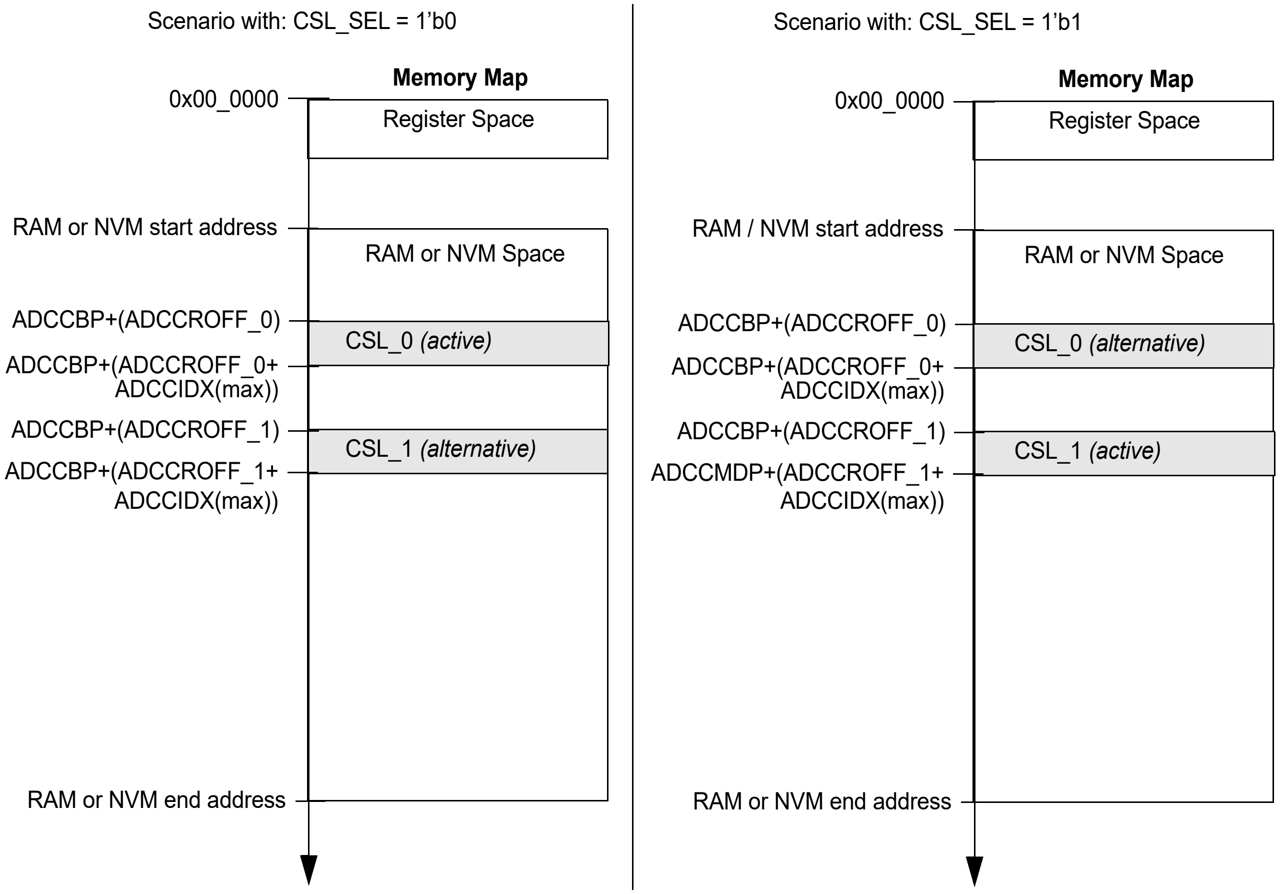
9.5.3.2.2 Introduction of the two Command Sequence Lists (CSLs)

The two Command Sequence Lists (CSLs) can be referred to via the Command Base Pointer Register plus the Command and Result Offset Registers plus the Command Index Register (ADCCBP, ADCCROFF_0/1, ADCCIDX).

The final address for conversion command loading is calculated by the sum of these registers (e.g.: ADCCBP+ADCCROFF_0+ADCCIDX or ADCCBP+ADCCROFF_1+ADCCIDX).

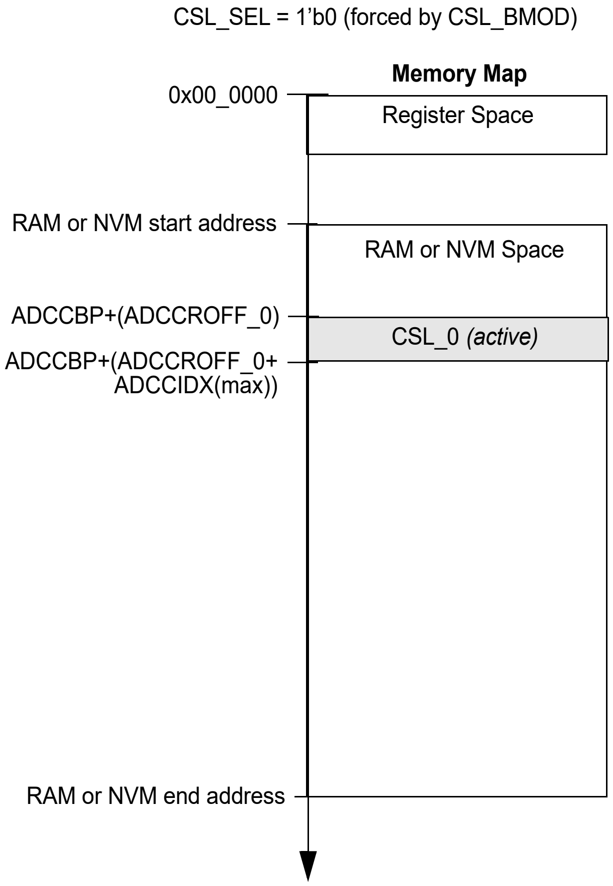
Bit CSL_BMOD selects if the CSL is used in double buffer or single buffer mode. In double buffer mode, the CSL can be swapped by flow control bits LDOK and RSTA. For detailed information about when and how the CSL is swapped, please refer to Section 9.5.3.2.5, “The four ADC conversion flow control bits - description of Restart Event + CSL Swap, Section 9.8.7.1, “Initial Start of a Command Sequence List and Section 9.8.7.3, “Restart CSL execution with new/other CSL (alternative CSL becomes active CSL) — CSL swapping

Which list is actively used for ADC command loading is indicated by bit CSL_SEL. The register to define the CSL start addresses (ADCCBP) can be set to any even location of the system RAM or NVM area. It is the user’s responsibility to make sure that the different ADC lists do not overlap or exceed the system RAM or the NVM area, respectively. The error flag IA_EIF will be set for accesses to ranges outside system RAM area and cause an error interrupt if enabled.



Note: Address register names in () are not absolute addresses instead they are a sample offset or sample index

Figure 9-31. Command Sequence List Schema in Double Buffer Mode



Note: Address register names in () are not absolute addresses instead they are a sample offset or sample index

Figure 9-32. Command Sequence List Schema in Single Buffer Mode

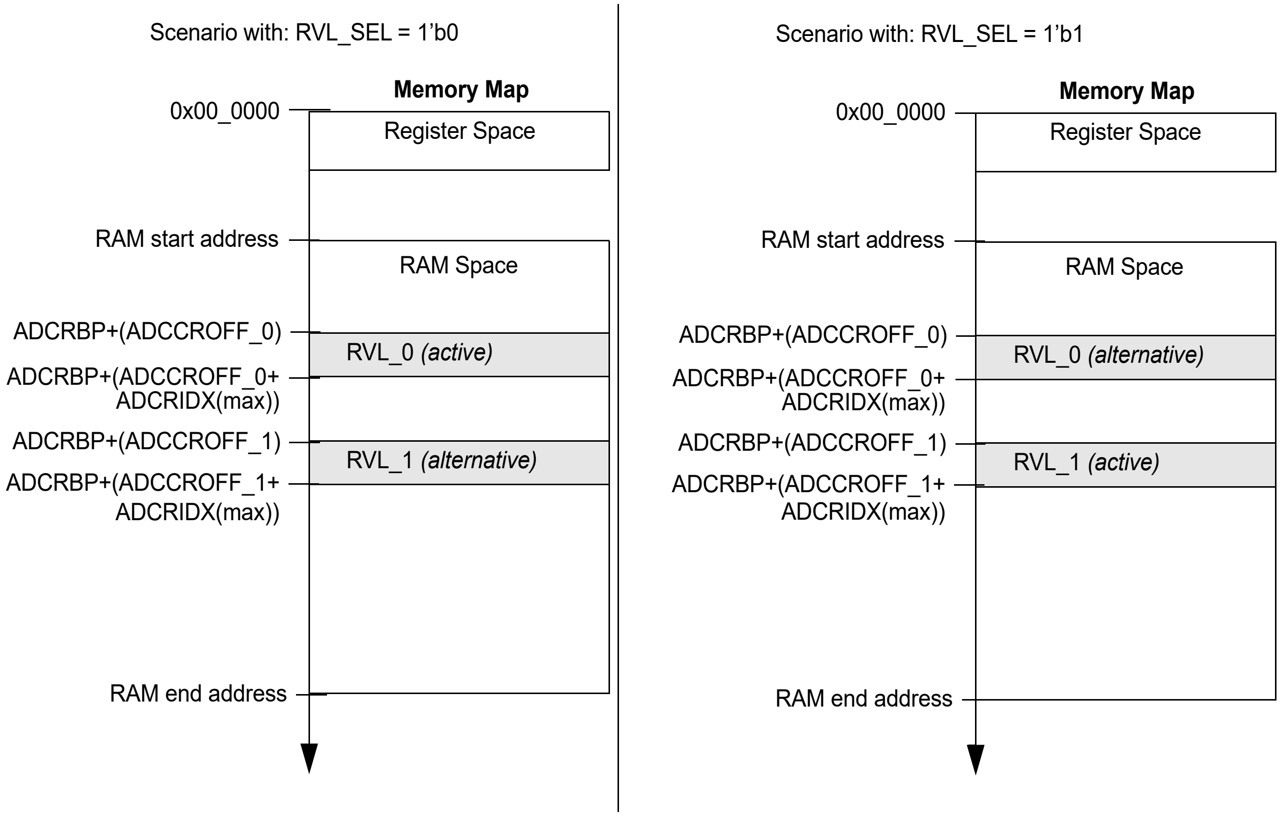
While the ADC is enabled, one CSL is active (indicated by bit CSL_SEL) and the corresponding list should not be modified anymore. At the same time the alternative CSL can be modified to prepare the ADC for new conversion sequences in CSL double buffered mode. When the ADC is enabled, the command address registers (ADCCBP, ADCCROFF_0/2, ADCCIDX) are read only and register ADCCIDX is under control of the ADC.

9.5.3.2.3 Introduction of the two Result Value Lists (RVLs)

The same list-based architecture as described above for the CSL has been implemented for the Result Value List (RVL) with corresponding address registers (ADCRBP, ADCCROFF_0/1, ADCRIDX).
The final address for conversion result storage is calculated by the sum of these registers (e.g.: ADCRBP+ADCCROFF_0+ADCRIDX or ADCRBP+ADCCROFF_1+ADCRIDX).
The RVL_BMOD bit selects if the RVL is used in double buffer or single buffer mode. In double buffer mode the RVL is swapped:

- Each time an “End Of List” command type got executed followed by the first conversion from top of the next CSL and related (first) result is about to be stored
- A CSL got aborted (bit SEQA=1’b1) and ADC enters idle state (becomes ready for new flow control events)

Using the RVL in double buffer mode the RVL is not swapped after exit from Stop Mode or Wait Mode with bit SWAI set. Hence the RVL used before entry of Stop or Wait Mode with bit SWAI set is overwritten after exit from the MCU Operating Mode (see also Section 9.2.1.2, “MCU Operating Modes).
Which list is actively used for the ADC conversion result storage is indicated by bit RVL_SEL. The register to define the RVL start addresses (ADCRBP) can be set to any even location of the system RAM area. It is the user’s responsibility to make sure that the different ADC lists do not overlap or exceed the system RAM area. The error flag IA_EIF will be set for accesses to ranges outside system RAM area and cause an error interrupt if enabled.



Note: Address register names in () are not absolute addresses instead they are a sample offset or sample index

Figure 9-33. Result Value List Schema in Double Buffer Mode

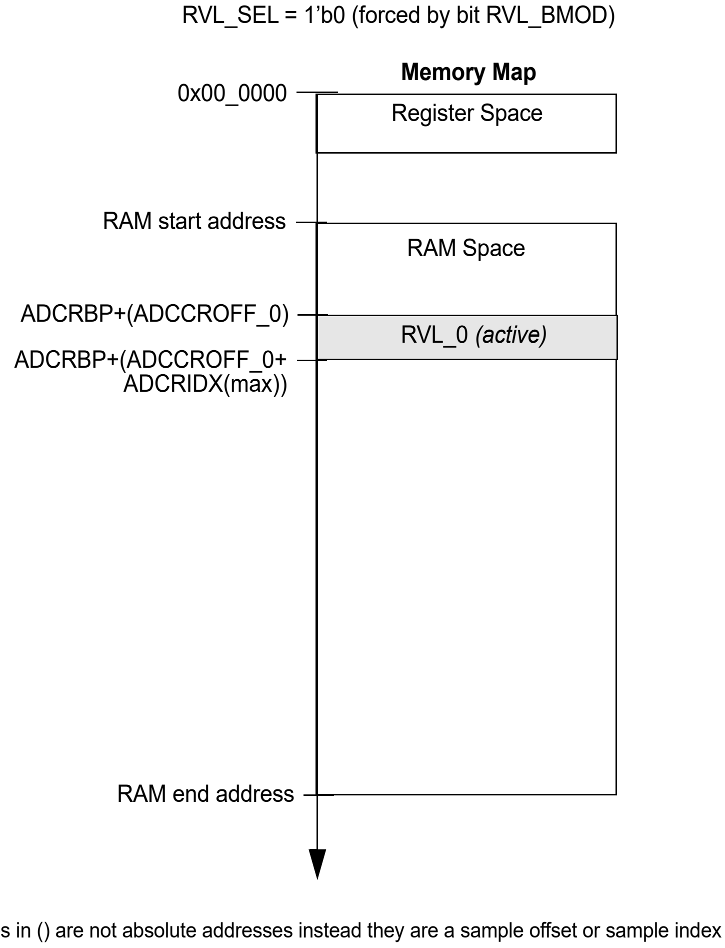


Figure 9-34. Result Value List Schema in Single Buffer Mode

While ADC is enabled, one Result Value List is active (indicated by bit RVL_SEL). The conversion Result Value List can be read anytime. When the ADC is enabled the conversion result address registers (ADCRBP, ADCCROFF_0/1, ADCRIDX) are read only and register ADCRIDX is under control of the ADC.

A conversion result is always stored as 16bit entity in unsigned data representation. Left and right justification inside the entity is selected via the DJM control bit. Unused bits inside an entity are stored zero.

Table 9-32. Conversion Result Justification Overview

Conversion Resolution (SRES[1:0])	Left Justified Result (DJM = 1’b0)	Right Justified Result (DJM = 1’b1)
8 bit	{Result[7:0],8’b00000000}	{8’b00000000,Result[7:0]}
10 bit	{Result[9:0],6’b000000}	{6’b000000,Result[9:0]}
12 bit	{Result[11:0],4’b0000}	{4’b0000,Result[11:0]}

9.5.3.2.4 The two conversion flow control Mode Configurations

The ADC provides two modes (“Trigger Mode” and “Restart Mode”) which are different in the conversion control flow. The “Restart Mode” provides precise timing control about the sample start point but is more complex from the flow control perspective, while the “Trigger Mode” is more simple from flow control point of view but is less controllable regarding conversion sample start.

Following are the key differences:

In “Trigger Mode” configuration, when conversion flow control bit RSTA gets set the bit TRIG gets set automatically. Hence in “Trigger Mode” the applications should not set the bit TRIG and bit RSTA simultaneously (via data bus or internal interface), because it is a flow control failure and the ADC will cease operation.

In “Trigger Mode” configuration, after the execution of the initial Restart Event the current CSL can be executed and controlled via Trigger Events only. Hence, if the “End Of List” command is reached a restart of conversion flow from top of current CSL does not require to set bit RSTA because returning to the top of current CSL is done automatically. Therefore the current CSL can be executed again after the “End Of List” command type is executed by a Trigger Event only.

In “Restart Mode” configuration, the execution of a CSL is controlled via Trigger Events and Restart Events. After execution of the “End Of List” command the conversion flow must be continued by a Restart Event followed by a Trigger Event and the Trigger Event must not occur before the Restart Event has finished.

For more details and examples regarding flow control and application use cases please see following section and Section 9.8.7, “Conversion flow control application information.

9.5.3.2.5 The four ADC conversion flow control bits

There are four bits to control conversion flow (execution of a CSL and CSL exchange in double buffer mode). Each bit is controllable via the data bus and internal interface depending on the setting of ACC_CFG[1:0] bits (see also Figure 9-2). In the following the conversion control event to control the conversion flow is given with the related internal interface signal and corresponding register bit name together with information regarding:

- Function of the conversion control event
- How to request the event
- When is the event finished
- Mandatory requirements to executed the event

A summary of all event combinations is provided by Table 9-10.

- **Trigger Event**
  Internal Interface Signal: Trigger
  Corresponding Bit Name: TRIG

- *Function:*
  Start the first conversion of a conversion sequence which is defined in the active Command Sequence List
- *Requested by:*
  - Positive edge of internal interface signal Trigger
  - Write Access via data bus to set control bit TRIG
- *When finished:*
  This bit is cleared by the ADC when the first conversion of the sequence is beginning to sample
- *Mandatory Requirements:*
  - In all ADC conversion flow control modes bit TRIG is only set (Trigger Event executed) if the Trigger Event occurs while no conversion or conversion sequence is ongoing (ADC idle)
  - In ADC conversion flow control mode “Restart Mode” with a Restart Event in progress it is not allowed that a Trigger Event occurs before the background command load phase has finished (Restart Event has been executed) else the error flag TRIG_EIF is set
  - In ADC conversion flow control mode “Trigger Mode” a Restart Event causes bit TRIG being set automatically. Bit TRIG is set when no conversion or conversion sequence is ongoing (ADC idle) and the RVL done condition is reached by one of the following:
  * A “End Of List” command type has been executed
  * A Sequence Abort Event is in progress or has been executed
  The ADC executes the Restart Event followed by the Trigger Event.
  - In ADC conversion flow control mode “Trigger Mode” a Restart Event and a simultaneous Trigger Event via internal interface or data bus causes the TRIG_EIF bit being set and ADC cease operation.

- **Restart Event** (with current active CSL)
  Internal Interface Signal: Restart
  Corresponding Bit Name: RSTA
  - *Function:*
    - Go to top of active CSL (clear index register for CSL)
    - Load one background command register and wait for Trigger (CSL offset register is not switched independent of bit CSL_BMOD)
    - Set error flag RSTA_EIF when a Restart Request occurs before one of the following conditions was reached:
    * The "End Of List" command type has been executed
    * Depending on bit STR_SEQA if the "End Of List" command type is about to be executed
    * The current CSL has been aborted or is about to be aborted due to a Sequence Abort Request.
  - *Requested by:*
    - Positive edge of internal interface signal Restart
    - Write Access via data bus to set control bit RSTA

- *When finished:*
  This bit is cleared when the first conversion command of the sequence from top of active Sequence Command List is loaded
- *Mandatory Requirement:*
  - In all ADC conversion flow control modes a Restart Event causes bit RSTA to be set. Bit SEQA is set simultaneously by ADC hardware if:
  * ADC not idle (a conversion or conversion sequence is ongoing and current CSL not finished) and no Sequence Abort Event in progress (bit SEQA not already set or set simultaneously via internal interface or data bus)
  * ADC idle but RVL done condition not reached
  The RVL done condition is reached by one of the following:
  * A “End Of List” command type has been executed
  * A Sequence Abort Event is in progress or has been executed (bit SEQA already set or set simultaneously via internal interface or data bus)
  The ADC executes the Sequence Abort Event followed by the Restart Event for the conditions described before or only a Restart Event.
  - In ADC conversion flow control mode “Trigger Mode” a Restart Event causes bit TRIG being set automatically. Bit TRIG is set when no conversion or conversion sequence is ongoing (ADC idle) and the RVL done condition is reached by one of the following:
  * A “End Of List” command type has been executed
  * A Sequence Abort Event is in progress or has been executed
  The ADC executes the Restart Event followed by the Trigger Event.
  - In ADC conversion flow control mode “Trigger Mode” a Restart Event and a simultaneous Trigger Event via internal interface or data bus causes the TRIG_EIF bit being set and ADC cease operation.

- **Restart Event + CSL Exchange (Swap)**
  Internal Interface Signals: Restart + LoadOK
  Corresponding Bit Names: RSTA + LDOK
  - *Function:*
    Go to top of active CSL (clear index register for CSL) and switch to other offset register for address calculation if configured for double buffer mode (exchange the CSL list)
    Requested by:
    - Internal interface with the assertion of Interface Signal Restart the interface Signal LoadOK is evaluated and bit LDOK is set accordingly (bit LDOK set if Interface Signal LoadOK asserted when Interface Signal Restart asserts).
    - Write Access via data bus to set control bit RSTA simultaneously with bit LDOK.
  - *When finished:*
    Bit LDOK can only be cleared if it was set as described before and both bits (LDOK, RSTA) are cleared when the first conversion command from top of active Sequence Command List is loaded
  - *Mandatory Requirement:*
    No ongoing conversion or conversion sequence
    Details if using the internal interface:

If signal Restart is asserted before signal LoadOK is set the conversion starts from top of currently active CSL at the next Trigger Event (no exchange of CSL list).
If signal Restart is asserted after or simultaneously with signal LoadOK the conversion starts from top of the other CSL at the next Trigger Event (CSL is switched) if CSL is configured for double buffer mode.

- **Sequence Abort Event**
  Internal Interface Signal: Seq_Abort
  Corresponding Bit Name: SEQA
  - *Function:*
    Abort any possible ongoing conversion at next conversion boundary and abort current conversion sequence and active CSL
  - *Requested by:*
    - Positive edge of internal interface signal Seq_Abort
    - Write Access via data bus to set control bit SEQA
  - *When finished:*
    This bit gets cleared when an ongoing conversion is finished and the result is stored and/or an ongoing conversion sequence is aborted and current active CSL is aborted (ADC idle, RVL done)
  - *Mandatory Requirement:*
    - In all ADC conversion flow control modes bit SEQA can only be set if:
    * ADC not idle (a conversion or conversion sequence is ongoing)
    * ADC idle but RVL done condition not reached
    The RVL done condition is not reached if:
    * An “End Of List” command type has not been executed
    * A Sequence Abort Event has not been executed (bit SEQA not already set)
    - In all ADC conversion flow control modes a Sequence Abort Event can be issued at any time
    - In ADC conversion flow control mode “Restart Mode” after a conversion sequence abort request has been executed it is mandatory to set bit RSTA. If a Trigger Event occurs before a Restart Event is executed (bit RSTA set and cleared by hardware), bit TRIG is set, error flag TRIG_EIF is set, and the ADC can only be continued by a Soft-Reset. After the Restart Event the ADC accepts new Trigger Events (bit TRIG set) and begins conversion from top of the currently active CSL.
    - In ADC conversion flow control mode “Restart Mode” after a Sequence Abort Event has been executed, a Restart Event causes only the RSTA bit being set. The ADC executes a Restart Event only.
  - In both conversion flow control modes (“Restart Mode” and “Trigger Mode”) when conversion flow control bit RSTA gets set automatically bit SEQA gets set when the ADC has not reached one of the following scenarios:
    * An “End Of List” command type has been executed or is about to be executed
    * A Sequence Abort request is about to be executed or has been executed.
    In case bit SEQA is set automatically the Restart error flag RSTA_EIF is set to indicate an unexpected Restart Request.

9.5.3.2.6 Conversion flow control in case of conversion sequence control bit overrun scenarios

Restart Request Overrun:
If a legal Restart Request is detected and no Restart Event is in progress, the RSTA bit is set due to the request. The set RSTA bit indicates that a Restart Request was detected and the Restart Event is in process. In case further Restart Requests occur while the RSTA bit is set, this is defined a overrun situation. This scenario is likely to occur when bit STR_SEQA is set or when a Restart Event causes a Sequence Abort Event. The request overrun is captured in a background register that always stores the last detected overrun request. Hence if the overrun situation occurs more than once while a Restart Event is in progress, only the latest overrun request is pending. When the RSTA bit is cleared, the latest overrun request is processed and RSTA is set again one cycle later.

LoadOK Overrun:
Simultaneously at any Restart Request overrun situation the LoadOK input is evaluated and the status is captured in a background register which is alternated anytime a Restart Request Overrun occurs while Load OK Request is asserted. The Load OK background register is cleared as soon as the pending Restart Request gets processed.

Trigger Overrun:
If a Trigger occurs whilst bit TRIG is already set, this is defined as a Trigger overrun situation and causes the ADC to cease conversion at the next conversion boundary and to set bit TRIG_EIF. A overrun is also detected if the Trigger Event occurs automatically generated by hardware in “Trigger Mode” due to a Restart Event and simultaneously a Trigger Event is generated via data bus or internal interface. In this case the ADC ceases operation before conversion begins to sample. In “Trigger Mode” a Restart Request Overrun does not cause a Trigger Overrun (bit TRIG_EIF not set).

Sequence Abort Request Overrun:
If a Sequence Abort Request occurs whilst bit SEQA is already set, this is defined as a Sequence Abort Request Overrun situation and the overrun request is ignored.

9.5.3.3 ADC List Usage and Conversion/Conversion Sequence Flow Description

It is the user's responsibility to make sure that the different lists do not overlap or exceed the system RAM area respectively the CSL does not exceed the NVM area if located in the NVM. The error flag IA_EIF will be set for accesses done outside the system RAM area and will cause an error interrupt if enabled for lists that are located in the system RAM.

Generic flow for ADC register load at conversion sequence start/restart:

- It is mandatory that the ADC is idle (no ongoing conversion or conversion sequence).
- It is mandatory to have at least one CSL with valid entries. See also Section 9.8.7.2, "Restart CSL execution with currently active CSL or Section 9.8.7.3, "Restart CSL execution with new/other CSL (alternative CSL becomes active CSL) — CSL swapping for more details on possible scenarios.
- A Restart Event occurs, which causes the index registers to be cleared (register ADCCIDX and ADCRIDX are cleared) and to point to the top of the corresponding lists (top of active RVL and CSL).
- Load conversion command to background conversion command register 1.
- The control bit(s) RSTA (and LDOK if set) are cleared.
- Wait for Trigger Event to start conversion.

Generic flow for ADC register load during conversion:

- The index registers ADCCIDX is incremented.
- The inactive background command register is loaded with a new conversion command.

Generic flow for ADC result storage at end of conversion:

- Index register ADCRIDX is incremented and the conversion result is stored in system RAM. As soon as the result is successfully stored, any conversion interrupt flags are set accordingly.
- At the conversion boundary the other background command register becomes active and visible in the ADC register map.
- If the last executed conversion command was of type "End Of Sequence", the ADC waits for the Trigger Event.
- If the last executed conversion command was of type "End Of List" and the ADC is configured in "Restart Mode", the ADC sets all related flags and stays idle awaiting a Restart Event to continue.
- If the last executed conversion command was of type "End Of List" and the ADC is configured in "Trigger Mode", the ADC sets all related flags and automatically returns to top of current CSL and is awaiting a Trigger Event to continue.
- If the last executed conversion command was of type "Normal Conversion" the ADC continues command execution in the order of the current CSL (continues conversion).

9.6 Resets

At reset the ADC12B_LBA is disabled and in a power down state. The reset state of each individual bit is listed within Section 9.4.2, “Register Descriptions” which details the registers and their bit-fields.

9.7 Interrupts

The ADC supports three types of interrupts:

- Conversion Interrupt
- Sequence Abort Interrupt
- Error and Conversion Flow Control Issue Interrupt

Each of the interrupt types is associated with individual interrupt enable bits and interrupt flags.

9.7.1 ADC Conversion Interrupt

The ADC provides one conversion interrupt associated to 16 interrupt enable bits with dedicated interrupt flags. The 16 interrupt flags consist of:

- 15 conversion interrupt flags which can be associated to any conversion completion.
- One additional interrupt flag which is fixed to the “End Of List” conversion command type within the active CSL.

The association of the conversion number with the interrupt flag number is done in the conversion command.

9.7.2 ADC Sequence Abort Done Interrupt

The ADC provides one sequence abort done interrupt associated with the sequence abort request for conversion flow control. Hence, there is only one dedicated interrupt flag and interrupt enable bit for conversion sequence abort and it occurs when the sequence abort is done.

9.7.3 ADC Error and Conversion Flow Control Issue Interrupt

The ADC provides one error interrupt for four error classes related to conversion interrupt overflow, command validness, DMA access status and Conversion Flow Control issues, and CSL failure. The following error interrupt flags belong to the group of severe issues which cause an error interrupt if enabled and cease ADC operation:

- IA_EIF
- CMD_EIF
- EOL_EIF
- TRIG_EIF

In order to make the ADC operational again, an ADC Soft-Reset must be issued which clears the above listed error interrupt flags.

NOTE

It is important to note that if flag DBECC_ERR is set, the ADC ceases operation as well, but does not cause an ADC error interrupt. Instead, a machine exception is issued. In order to make the ADC operational again an ADC Soft-Reset must be issued.

Remaining error interrupt flags cause an error interrupt if enabled, but ADC continues operation. The related interrupt flags are:

- RSTAR_EIF
- LDOK_EIF
- CONIF_OIF

9.8 Use Cases and Application Information

9.8.1 List Usage — CSL single buffer mode and RVL single buffer mode

In this use case both list types are configured for single buffer mode (CSL_BMOD=1'b0 and RVL_BMOD=1'b0, CSL_SEL and RVL_SEL are forced to 1'b0). The index register for the CSL and RVL are cleared to start from the top of the list with next conversion command and result storage in the following cases:

- The conversion flow reaches the command containing the "End-of-List" command type identifier
- A Restart Request occurs at a sequence boundary
- After an aborted conversion or conversion sequence

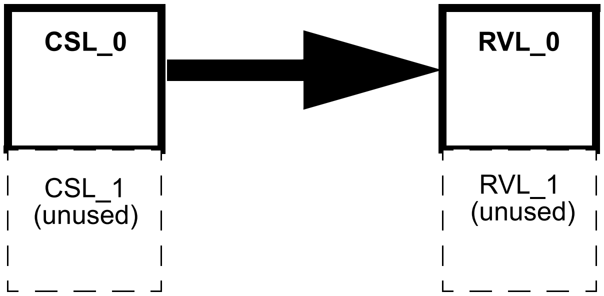


Figure 9-35. CSL Single Buffer Mode — RVL Single Buffer Mode Diagram

9.8.2 List Usage — CSL single buffer mode and RVL double buffer mode

In this use case the CSL is configured for single buffer mode (CSL_BMOD=1'b0) and the RVL is configured for double buffer mode (RVL_BMOD=1'b1). In this buffer configuration only the result list RVL is switched when the first conversion result of a CSL is stored after a CSL was successfully finished or a CSL got aborted.

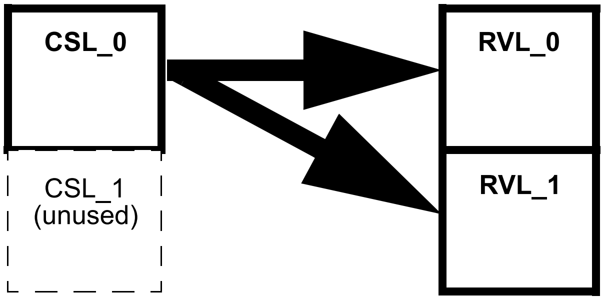


Figure 9-36. CSL Single Buffer Mode — RVL Single Buffer Mode Diagram

The last entirely filled RVL (an RVL where the corresponding CSL has been executed including the "End Of List " command type) is shown by register ADCEOLRI.

The CSL is used in single buffer mode and bit CSL_SEL is forced to 1'b0.

9.8.3 List Usage — CSL double buffer mode and RVL double buffer mode

In this use case both list types are configured for double buffer mode (CSL_BMOD=1'b1 and RVL_BMOD=1'b1) and whenever a Command Sequence List (CSL) is finished or aborted the command Sequence List is swapped by the simultaneous assertion of bits LDOK and RSTA.

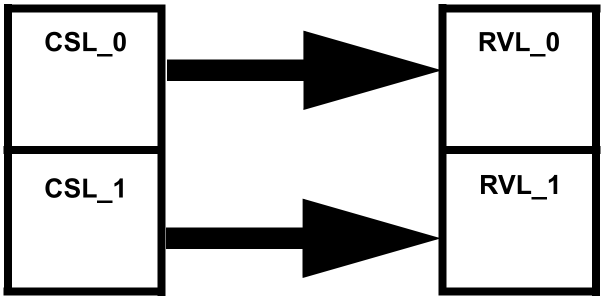


Figure 9-37. CSL Double Buffer Mode — RVL Double Buffer Mode Diagram

This use case can be used if the channel order or CSL length varies very frequently in an application.

9.8.4 List Usage — CSL double buffer mode and RVL single buffer mode

In this use case the CSL is configured for double buffer mode (CSL_BMOD=1'b1) and the RVL is configured for single buffer mode (RVL_BMOD=1'b0).

The two command lists can be different sizes and the allocated result list memory area in the RAM must be able to hold as many entries as the larger of the two command lists. Each time when the end of a Command Sequence List is reached, if bits LDOK and RSTA are set, the commands list is swapped.

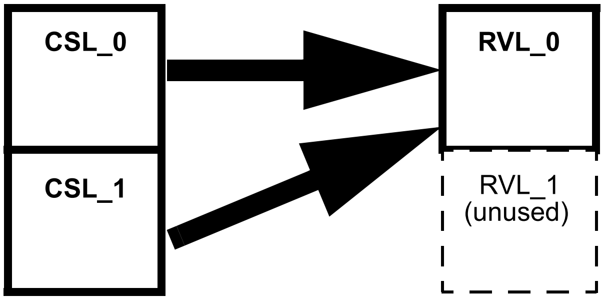


Figure 9-38. CSL Double Buffer Mode — RVL Single Buffer Mode Diagram

9.8.5 List Usage — CSL double buffer mode and RVL double buffer mode

In this use case both list types are configured for double buffer mode (CSL_BMOD=1’b1) and RVL_BMOD=1’b1).

This setup is the same as Section 9.8.3, “List Usage — CSL double buffer mode and RVL double buffer mode but at the end of a CSL the CSL is not always swapped (bit LDOK not always set with bit RSTA). The Result Value List is swapped whenever a CSL is finished or a CSL got aborted.

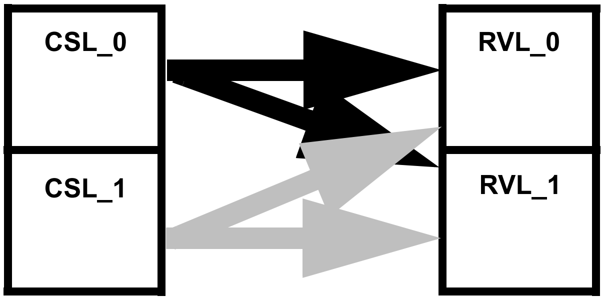
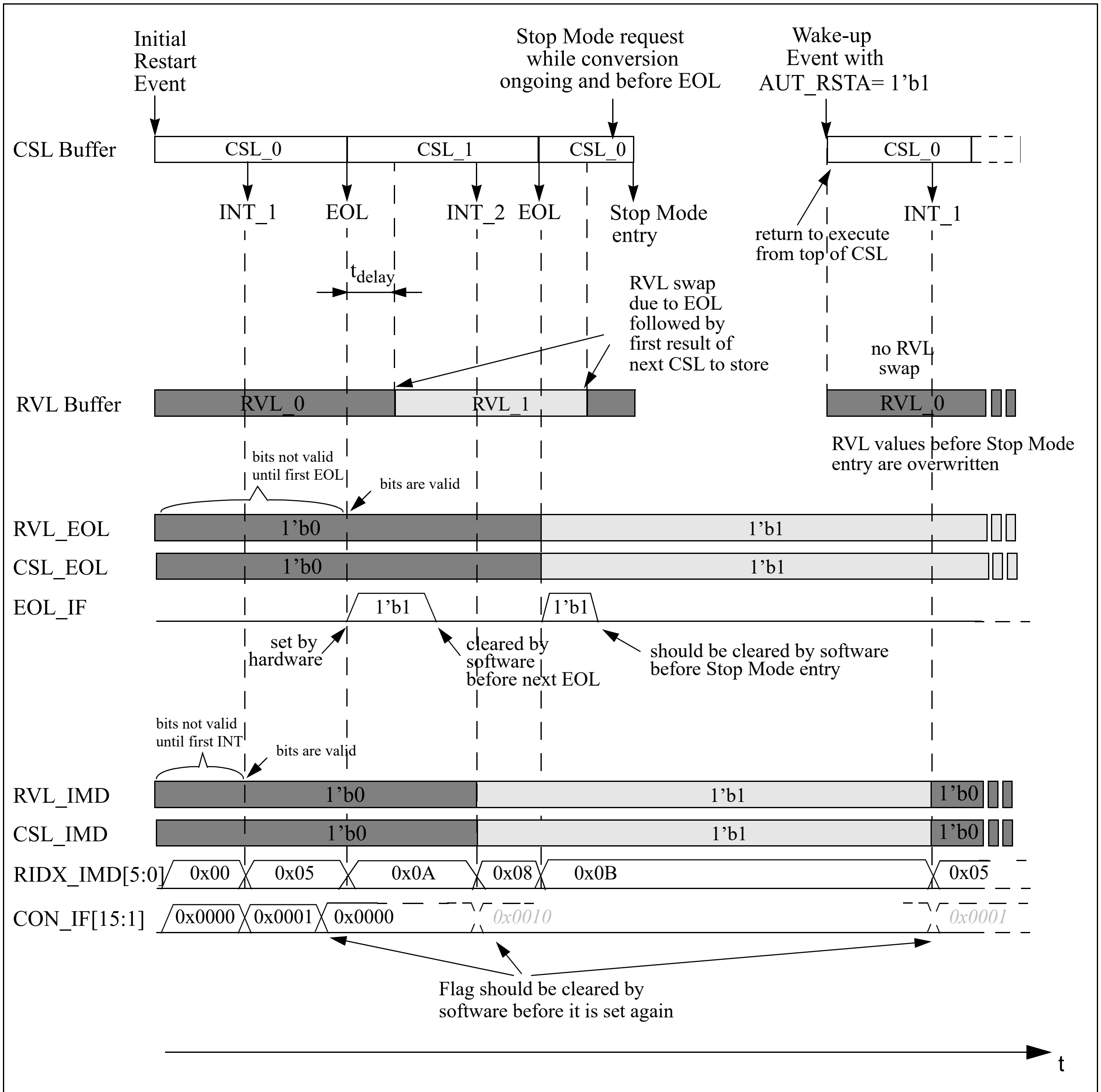


Figure 9-39. CSL Double Buffer Mode — RVL Double Buffer Mode Diagram

9.8.6 RVL swapping in RVL double buffer mode and related registers ADCIMDRI and ADCEOLRI

When using the RVL in double buffer mode, the registers ADCIMDRI and ADCEOLRI can be used by the application software to identify which RVL holds relevant and latest data and which CSL is related to this data. These registers are updated at the setting of one of the CON_IF[15:1] or the EOL_IF interrupt flags. As described in the register description Section 9.4.2.13, “ADC Intermediate Result Information Register (ADCIMDRI) and Section 9.4.2.14, “ADC End Of List Result Information Register (ADCEOLRI), the register ADCIMDRI, for instance, is always updated at the occurrence of a CON_IF[15:1] interrupt flag amongst other cases. Also each time the last conversion command of a CSL is finished and the corresponding result is stored, the related EOL_IF flag is set and register ADCEOLRI is updated. Hence application software can pick up conversion results, or groups of results, or an entire result list driven fully by interrupts. A use case example diagram is shown in Figure 9-40.



Comments:

EOL: ”End Of List” command type processed

INT_x: One of the CON_IF interrupt flags occurs

t_{delay}: Delay can vary depending on the DMA performance, and ADC configuration (conversion flow using the Trigger to proceed through the CSL)

Figure 9-40. RVL Swapping — Use Case Diagram

9.8.7 Conversion flow control application information

The ADC12B_LBA provides various conversion control scenarios to the user accomplished by the following features.

The ADC conversion flow control can be realized via the data bus only, the internal interface only, or by both access methods. The method used is software configurable via bits ACC_CFG[1:0].

The conversion flow is controlled via the four conversion flow control bits: SEQA, TRIG, RSTA, and LDOK.

Two different conversion flow control modes can be configured: Trigger Mode or Restart Mode

Single or double buffer configuration of CSL and RVL.

9.8.7.1 Initial Start of a Command Sequence List

At the initial start of a Command Sequence List after device reset all entries for at least one of the two CSL must have been completed and data must be valid. Depending on if the CSL_0 or the CSL_1 should be executed at the initial start of a Command Sequence List the following conversion control sequence must be applied:

If CSL_0 should be executed at the initial conversion start after device reset:
A Restart Event and a Trigger Event must occur (depending to the selected conversion flow control mode the events must occur one after the other or simultaneously) which causes the ADC to start conversion with commands loaded from CSL_0.

If CSL_1 should be executed at the initial conversion start after device reset:
Bit LDOK must be set simultaneously with the Restart Event followed by a Trigger Event (depending on the selected conversion flow control mode the Trigger events must occur simultaneously or after the Restart Event is finished). As soon as the Trigger Event gets executed the ADC starts conversion with commands loaded from CSL_1.

As soon as a new valid Restart Event occurs the flow for ADC register load at conversion sequence start as described in Section 9.5.3.3, “ADC List Usage and Conversion/Conversion Sequence Flow Description applies.

9.8.7.2 Restart CSL execution with currently active CSL

To restart a Command Sequence List execution it is mandatory that the ADC is idle (no conversion or conversion sequence is ongoing).

If necessary, a possible ongoing conversion sequence can be aborted by the Sequence Abort Event (setting bit SEQA). As soon as bit SEQA is cleared by the ADC, the current conversion sequence has been aborted and the ADC is idle (no conversion sequence or conversion ongoing).

After a conversion sequence abort is executed it is mandatory to request a Restart Event (bit RSTA set). After the Restart Event is finished (bit RSTA is cleared), the ADC accepts a new Trigger Event (bit TRIG can be set) and begins conversion from the top of the currently active CSL. In conversion flow control mode “Trigger Mode” only a Restart Event is necessary if ADC is idle to restart Conversion Sequence List execution (the Trigger Event occurs automatically).

It is possible to set bit RSTA and SEQA simultaneously, causing a Sequence Abort Event followed by a Restart Event. In this case the error flags behave differently depending on the selected conversion flow control mode:

- Setting both flow control bits simultaneously in conversion flow control mode “Restart Mode” prevents the error flags RSTA_EIF and LDOK_EIF from occurring.
- Setting both flow control bits simultaneously in conversion flow control mode “Trigger Mode” prevents the error flag RSTA_EIF from occurring.

If only a Restart Event occurs while ADC is not idle and bit SEQA is not set already (Sequence Abort Event in progress) a Sequence Abort Event is issued automatically and bit RSTAR_EIF is set.

Please see also the detailed conversion flow control bit mandatory requirements and execution information for bit RSTA and SEQA described in Section 9.5.3.2.5, “The four ADC conversion flow control bits.

9.8.7.3 Restart CSL execution with new/other CSL (alternative CSL becomes active CSL) — CSL swapping

After all alternative conversion command list entries are finished the bit LDOK can be set simultaneously with the next Restart Event to swap command buffers.

To start conversion command list execution it is mandatory that the ADC is idle (no conversion or conversion sequence is ongoing).

If necessary, a possible ongoing conversion sequence can be aborted by the Sequence Abort Event (setting bit SEQA). As soon as bit SEQA is cleared by the ADC, the current conversion sequence has been aborted and the ADC is idle (no conversion sequence or conversion ongoing).

After a conversion sequence abort is executed it is mandatory to request a Restart Event (bit RSTA set) and simultaneously set bit LDOK to swap the CSL buffer. After the Restart Event is finished (bit RSTA and LDOK are cleared), the ADC accepts a new Trigger Event (bit TRIG can be set) and begins conversion from the top of the newly selected CSL buffer. In conversion flow control mode “Trigger Mode” only a Restart Event (simultaneously with bit LDOK being set) is necessary to restart conversion command list execution with the newly selected CSL buffer (the Trigger Event occurs automatically).

It is possible to set bits RSTA, LDOK and SEQA simultaneously, causing a Sequence Abort Event followed by a Restart Event. In this case the error flags behave differently depending on the selected conversion flow control mode:

- Setting these three flow control bits simultaneously in “Restart Mode” prevents the error flags RSTA_EIF and LDOK_EIF from occurring.
- Setting these three flow control bits simultaneously in “Trigger Mode” prevents the error flag RSTA_EIF from occurring.

If only a Restart Event occurs while ADC is not idle and bit SEQA is not set already (Sequence Abort Event in progress) a Sequence Abort Event is issued automatically and bit RSTAR_EIF is set.

Please see also the detailed conversion flow control bit mandatory requirements and execution information for bit RSTA and SEQA described in Section 9.5.3.2.5, "The four ADC conversion flow control bits.

9.8.8 Continuous Conversion

Applications that only need to continuously convert a list of channels, without the need for timing control or the ability to perform different sequences of conversions (grouped number of different channels to convert) can make use of the following simple setup:

- "Trigger Mode" configuration
- Single buffer CSL
- Depending on data transfer rate either use single or double buffer RVL configuration
- Define a list of conversion commands which only contains the "End Of List" command with automatic wrap to top of CSL

After finishing the configuration and enabling the ADC an initial Restart Event is sufficient to launch the continuous conversion until next device reset or low power mode.

In case a Low Power Mode is used:
If bit AUT_RSTA is set before Low Power Mode is entered the conversion continues automatically as soon as a low power mode (Stop Mode or Wait Mode with bit SWAI set) is exited.

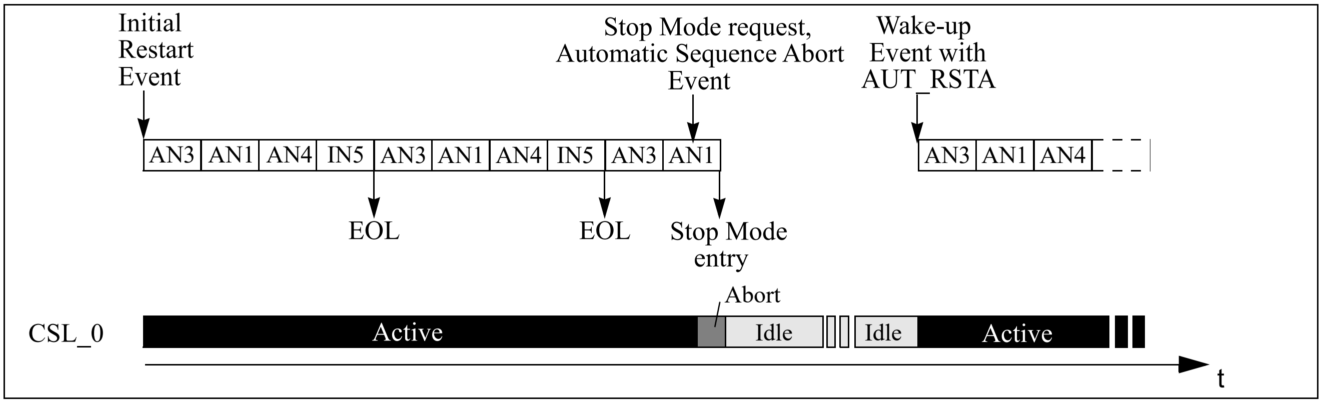


Figure 9-41. Conversion Flow Control Diagram — Continuous Conversion (with Stop Mode)

9.8.9 Triggered Conversion — Single CSL

Applications that require the conversion of one or more groups of different channels in a periodic and timed manner can make use of a configuration in “Trigger Mode” with a single CSL containing a list of sequences. This means the CSL consists of several sequences each separated by an “End of Sequence” command. The last command of the CSL uses the “End Of List” command with wrap to top of CSL and waiting for a Trigger (CMD_SEL[1:0] =2’b11). Hence after the initial Restart Event each sequence can be launched via a Trigger Event and repetition of the CSL can be launched via a Trigger after execution of the “End Of List” command.

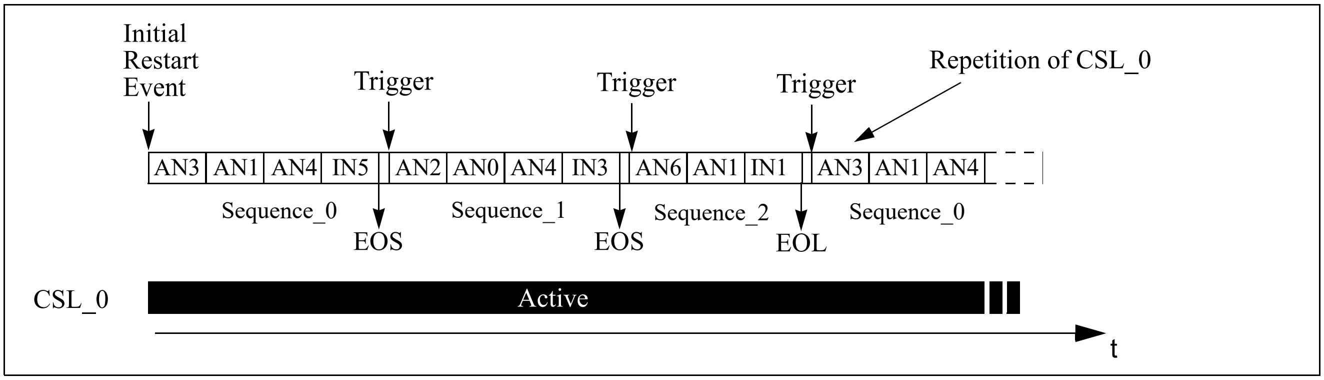


Figure 9-42. Conversion Flow Control Diagram — Triggered Conversion (CSL Repetition)

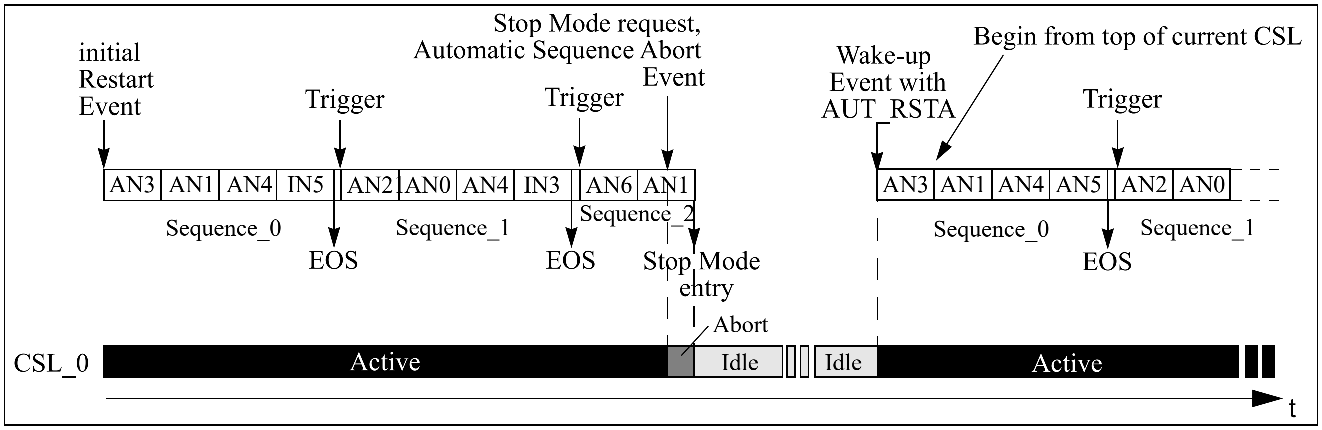


Figure 9-43. Conversion Flow Control Diagram — Triggered Conversion (with Stop Mode)

In case a Low Power Mode is used:
If bit AUT_RSTA is set before Low Power Mode is entered, the conversion continues automatically as soon as a low power mode (Stop Mode or Wait Mode with bit SWAI set) is exited.

9.8.10 Fully Timing Controlled Conversion

As described previously, in “Trigger Mode” a Restart Event automatically causes a trigger. To have full and precise timing control of the beginning of any conversion/sequence the “Restart Mode” is available. In “Restart Mode” a Restart Event does not cause a Trigger automatically; instead, the Trigger must be issued separately and with correct timing, which means the Trigger is not allowed before the Restart Event (conversion command loading) is finished (bit RSTA=1’b0 again). The time required from Trigger until sampling phase starts is given (refer to Section 9.4.2.6, “ADC Conversion Flow Control Register (ADCFLWCTL), Timing considerations) and hence timing is fully controllable by the application. Additionally, if a Trigger occurs before a Restart Event is finished, this causes the TRIG_EIF flag being set. This allows detection of false flow control sequences.

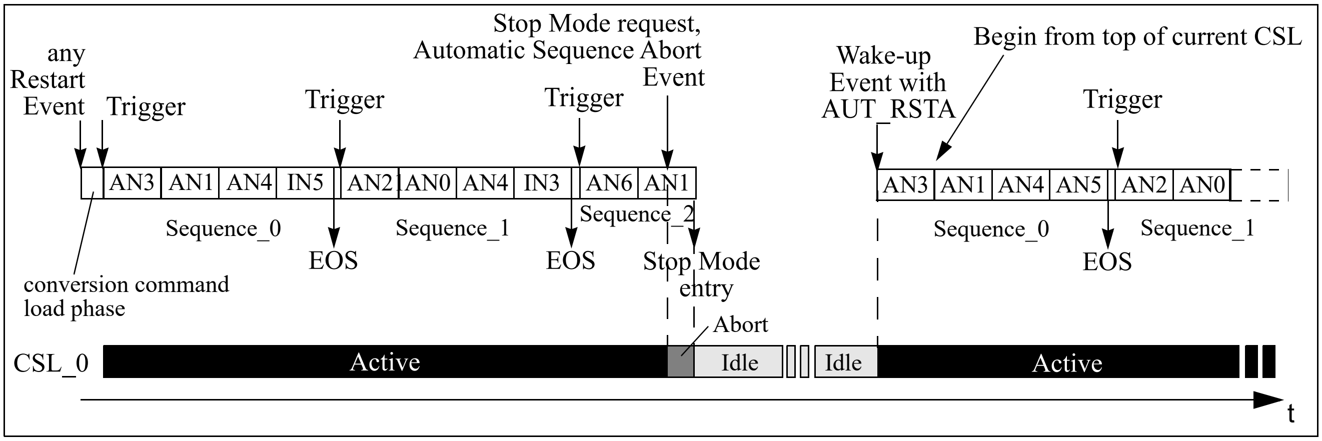


Figure 9-44. Conversion Flow Control Diagram — Fully Timing Controlled Conversion (with Stop Mode)

Unlike the Stop Mode entry shown in Figure 9-43 and Figure 9-44 it is recommended to issue the Stop Mode at sequence boundaries (when ADC is idle and no conversion/conversion sequence is ongoing).

Any of the Conversion flow control application use cases described above (Continuous, Triggered, or Fully Timing Controlled Conversion) can be used with CSL single buffer mode or with CSL double buffer mode. If using CSL double buffer mode, CSL swapping is performed by issuing a Restart Event with bit LDOK set.

Chapter 10
Supply Voltage Sensor - (BATSV3)

Table 10-1. Revision History Table

Rev. No. (Item No.)	Data	Sections Affected	Substantial Change(s)
V01.00	15 Dec 2010	all	Initial Version
V02.00	16 Mar 2011	10.3.2.1 10.4.2.1	- added BVLS[1] to support four voltage level - moved BVHS to register bit 6
V03.00	26 Apr 2011	all	- removed Vsense
V03.10	04 Oct 2011	10.4.2.1 and 10.4.2.2	- removed BSESE

10.1 Introduction

The BATS module provides the functionality to measure the voltage of the chip supply pin VSUP.

10.1.1 Features

The VSUP pin can be routed via an internal divider to the internal Analog to Digital Converter. Independent of the routing to the Analog to Digital Converter, it is possible to route this voltage to a comparator to generate a low or a high voltage interrupt to alert the MCU.

10.1.2 Modes of Operation

The BATS module behaves as follows in the system power modes:

1. Run mode

   The activation of the VSUP Level Sense Enable (BSUSE=1) or ADC connection Enable (BSUAE=1) closes the path from VSUP pin through the resistor chain to ground and enables the associated features if selected.

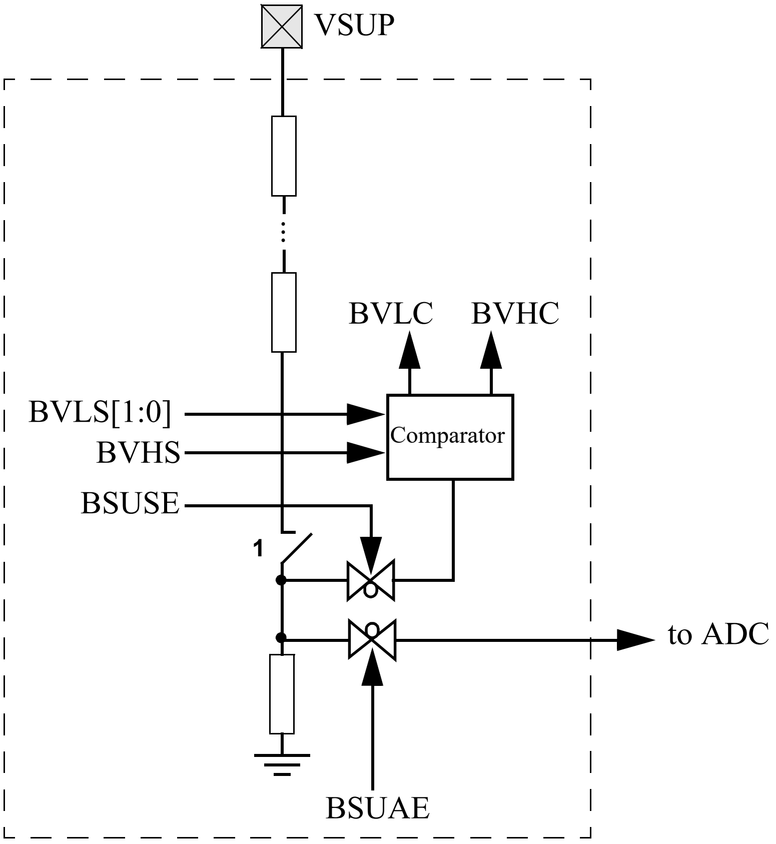
2. Stop mode

   During stop mode operation the path from the VSUP pin through the resistor chain to ground is opened and the low and high voltage sense features are disabled.
   The content of the configuration register is unchanged.

10.1.3 Block Diagram

Figure 10-1 shows a block diagram of the BATS module. See device guide for connectivity to ADC channel.

Figure 10-1. BATS Block Diagram



1automatically closed if BSUSE and/or BSUAE
is active, open during Stop mode

10.2 External Signal Description

This section lists the name and description of all external ports.

10.2.1 VSUP — Voltage Supply Pin

This pin is the chip supply. It can be internally connected for voltage measurement. The voltage present at this input is scaled down by an internal voltage divider, and can be routed to the internal ADC or to a comparator.

10.3 Memory Map and Register Definition

This section provides the detailed information of all registers for the BATS module.

10.3.1 Register Summary

Figure 10-2 shows the summary of all implemented registers inside the BATS module.

NOTE

Register Address = Module Base Address + Address Offset, where the Module Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 BATE	R	0	BVHS	BVLS[1:0]		BSUAE	BSUSE	0	0
	W	▨						▨	▨
0x0001 BATSR	R	0	0	0	0	0	0	BVHC	BVLC
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0002 BATIE	R	0	0	0	0	0	0	BVHIE	BVLIE
	W	▨	▨	▨	▨	▨	▨		
0x0003 BATIF	R	0	0	0	0	0	0	BVHIF	BVLIF
	W	▨	▨	▨	▨	▨	▨		
0x0004 - 0x0005 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0006 - 0x0007 Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
	W								

▨ = Unimplemented

Figure 10-2. BATS Register Summary

10.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order. Unused bits read back zero.

10.3.2.1 BATS Module Enable Register (BATE)

Module Base + 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	BVHS	BVLS[1:0]		BSUAE	BSUSE	0	0
W	▨						▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 10-3. BATS Module Enable Register (BATE)

[1] Read: Anytime
Write: Anytime

Table 10-2. BATE Field Description

Field	Description
6 BVHS	**BATS Voltage High Select** — This bit selects the trigger level for the Voltage Level High Condition (BVHC). 0 Voltage level V_{HBI1} is selected 1 Voltage level V_{HBI2} is selected
5:4 BVLS[1:0]	**BATS Voltage Low Select** — This bit selects the trigger level for the Voltage Level Low Condition (BVLC). 00 Voltage level V_{LBI1} is selected 01 Voltage level V_{LBI2} is selected 10 Voltage level V_{LBI3} is selected 11 Voltage level V_{LBI4} is selected
3 BSUAE	**BATS VSUP ADC Connection Enable** — This bit connects the VSUP pin through the resistor chain to ground and connects the ADC channel to the divided down voltage. 0 ADC Channel is disconnected 1 ADC Channel is connected
2 BSUSE	**BATS VSUP Level Sense Enable** — This bit connects the VSUP pin through the resistor chain to ground and enables the Voltage Level Sense features measuring BVLC and BVHC. 0 Level Sense features disabled 1 Level Sense features enabled

NOTE

When opening the resistors path to ground by changing BSUSE or BSUAE then for a time T_{EN_UNC} + two bus cycles the measured value is invalid. This is to let internal nodes be charged to correct value. BVHIE, BVLIE might be cleared for this time period to avoid false interrupts.

10.3.2.2 BATS Module Status Register (BATSR)

Module Base + 0x0001 Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	BVHC	BVLC
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

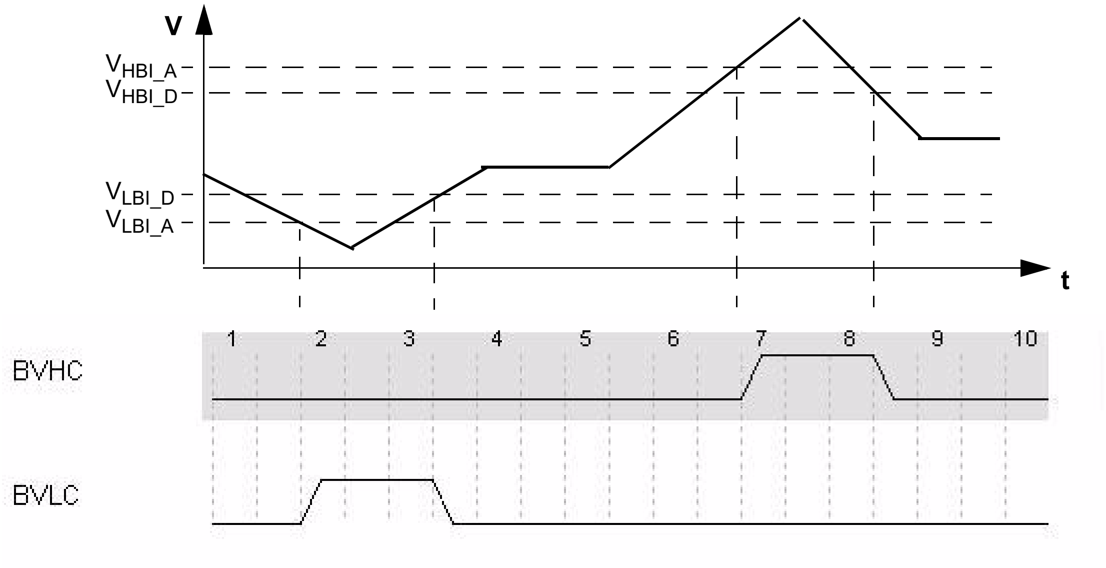
Figure 10-4. BATS Module Status Register (BATSR)

[1] Read: Anytime
Write: Never

Table 10-3. BATSR - Register Field Descriptions

Field	Description
1 BVHC	**BATS Voltage Sense High Condition Bit** — This status bit indicates that a high voltage at VSUP, depending on selection, is present. 0 $V_{measured} < V_{HBI\_A}$ (rising edge) or $V_{measured} < V_{HBI\_D}$ (falling edge) 1 $V_{measured} \geq V_{HBI\_A}$ (rising edge) or $V_{measured} \geq V_{HBI\_D}$ (falling edge)
0 BVLC	**BATS Voltage Sense Low Condition Bit** — This status bit indicates that a low voltage at VSUP, depending on selection, is present. 0 $V_{measured} \geq V_{LBI\_A}$ (falling edge) or $V_{measured} \geq V_{LBI\_D}$ (rising edge) 1 $V_{measured} < V_{LBI\_A}$ (falling edge) or $V_{measured} < V_{LBI\_D}$ (rising edge)

Figure 10-5. BATS Voltage Sensing



10.3.2.3 BATS Interrupt Enable Register (BATIE)

Module Base + 0x0002 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	BVHIE	BVLIE
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 10-6. BATS Interrupt Enable Register (BATIE)

[1] Read: Anytime
Write: Anytime

Table 10-4. BATIE Register Field Descriptions

Field	Description
1 BVHIE	**BATS Interrupt Enable High** — Enables High Voltage Interrupt **.** 0 No interrupt will be requested whenever BVHIF flag is set . 1 Interrupt will be requested whenever BVHIF flag is set
0 BVLIE	**BATS Interrupt Enable Low** — Enables Low Voltage Interrupt . 0 No interrupt will be requested whenever BVLIF flag is set . 1 Interrupt will be requested whenever BVLIF flag is set .

10.3.2.4 BATS Interrupt Flag Register (BATIF)

Module Base + 0x0003 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	BVHIF	BVLIF
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 10-7. BATS Interrupt Flag Register (BATIF)

[1] Read: Anytime
Write: Anytime, write 1 to clear

Table 10-5. BATIF Register Field Descriptions

Field	Description
1 BVHIF	**BATS Interrupt Flag High Detect** — The flag is set to 1 when BVHC status bit changes. 0 No change of the BVHC status bit since the last clearing of the flag. 1 BVHC status bit has changed since the last clearing of the flag.
0 BVLIF	**BATS Interrupt Flag Low Detect** — The flag is set to 1 when BVLC status bit changes. 0 No change of the BVLC status bit since the last clearing of the flag. 1 BVLC status bit has changed since the last clearing of the flag.

10.3.2.5 Reserved Register

Module Base + 0x0006
Module Base + 0x0007

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 10-8. Reserved Register

[1] Read: Anytime
Write: Only in special mode

NOTE

> These reserved registers are designed for factory test purposes only and are not intended for general user access. Writing to these registers when in special mode can alter the module's functionality.

10.4 Functional Description

10.4.1 General

The BATS module allows measuring the voltage on the VSUP pin. The voltage at the VSUP pin can be routed via an internal voltage divider to an internal Analog to Digital Converter Channel. Also the BATS module can be configured to generate a low and high voltage interrupt based on VSUP. The trigger level of the high and low interrupt are selectable.

10.4.2 Interrupts

This section describes the interrupt generated by the BATS module. The interrupt is only available in CPU run mode. Entering and exiting CPU stop mode has no effect on the interrupt flags.

To make sure the interrupt generation works properly the bus clock frequency must be higher than the Voltage Warning Low Pass Filter frequency (f_{VWLP_filter}).

The comparator outputs BVLC and BVHC are forced to zero if the comparator is disabled (configuration bit BSUSE is cleared). If the software disables the comparator during a high or low Voltage condition (BVHC or BVLC active), then an additional interrupt is generated. To avoid this behavior the software must disable the interrupt generation before disabling the comparator.

The BATS interrupt vector is named in Table 10-6. Vector addresses and interrupt priorities are defined at MCU level.

The module internal interrupt sources are combined into one module interrupt signal.

Table 10-6. BATS Interrupt Sources

Module Interrupt Source	Module Internal Interrupt Source	Local Enable
BATS Interrupt (BATI)	BATS Voltage Low Condition Interrupt (BVLI)	BVLIE = 1
	BATS Voltage High Condition Interrupt (BVHI)	BVHIE = 1

10.4.2.1 BATS Voltage Low Condition Interrupt (BVLI)

To use the Voltage Low Interrupt the Level Sensing must be enabled (BSUSE =1).

If measured when

a) V_{LBI1} selected with BVLS[1:0] = 0x0
$V_{measure} < V_{LBI1\_A}$ (falling edge) or $V_{measure} < V_{LBI1\_D}$ (rising edge)

or when

b) V_{LBI2} selected with BVLS[1:0] = 0x1 at pin VSUP
$V_{measure} < V_{LBI2\_A}$ (falling edge) or $V_{measure} < V_{LBI2\_D}$ (rising edge)

or when

c) V_{LBI3} selected with BVLS[1:0] = 0x2
$V_{measure} < V_{LBI3\_A}$ (falling edge) or $V_{measure} < V_{LBI3\_D}$ (rising edge)

or when

d) V_{LBI4} selected with BVLS[1:0] = 0x3
$V_{measure} < V_{LBI4\_A}$ (falling edge) or $V_{measure} < V_{LBI4\_D}$ (rising edge)

then BVLC is set. BVLC status bit indicates that a low voltage at pin VSUP is present. The Low Voltage Interrupt flag (BVLIF) is set to 1 when the Voltage Low Condition (BVLC) changes state . The Interrupt flag BVLIF can only be cleared by writing a 1. If the interrupt is enabled by bit BVLIE the module requests an interrupt to MCU (BATI).

10.4.2.2 BATS Voltage High Condition Interrupt (BVHI)

To use the Voltage High Interrupt the Level Sensing must be enabled (BSUSE=1).

If measured when

a) V_{HBI1} selected with BVHS = 0
$V_{measure} \geq V_{HBI1\_A}$ (rising edge) or $V_{measure} \geq V_{HBI1\_D}$ (falling edge)

or when

a) V_{HBI2} selected with BVHS = 1
$V_{measure} \geq V_{HBI2\_A}$ (rising edge) or $V_{measure} \geq V_{HBI2\_D}$ (falling edge)

then BVHC is set. BVHC status bit indicates that a high voltage at pin VSUP is present. The High Voltage Interrupt flag (BVHIF) is set to 1 when a Voltage High Condition (BVHC) changes state. The Interrupt flag BVHIF can only be cleared by writing a 1. If the interrupt is enabled by bit BVHIE the module requests an interrupt to MCU (BATI).

Chapter 11
Timer Module (TIM16B8CV3) Block Description

Table 11-1.

V03.00	Jan. 28, 2009		Initial version
V03.01	Aug. 26, 2009	11.1.2/11-360 Figure 1-4./1-8 11.3.2.15/11-376 11.3.2.2/11-366, 11.3.2.3/11-366, 11.3.2.4/11-367, 11.4.3/11-382	- Correct typo: TSCR ->TSCR1; - Correct typo: ECTxxx->TIMxxx - Correct reference: Figure 11-25 -> Figure 11-30 - Add description, “a counter overflow when TTOV[7] is set”, to be the condition of channel 7 override event. - Phrase the description of OC7M to make it more explicit
V03.02	Apri,12,2010	11.3.2.8/11-370 11.3.2.11/11-373 11.4.3/11-382	-Add Table 11-10 -update TCRE bit description -add Figure 11-31
V03.03	Jan,14,2013		-single source generate different channel guide

11.1 Introduction

The basic scalable timer consists of a 16-bit, software-programmable counter driven by a flexible programmable prescaler.

This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from microseconds to many seconds.

This timer could contain up to 8 input capture/output compare channels with one pulse accumulator available only on channel 7. The input capture function is used to detect a selected transition edge and record the time. The output compare function is used for generating output signals or for timer software delays. The 16-bit pulse accumulator is used to operate as a simple event counter or a gated time accumulator. The pulse accumulator shares timer channel 7 when the channel is available and when in event mode.

A full access for the counter registers or the input capture/output compare registers should take place in one clock cycle. Accessing high byte and low byte separately for all of these registers may not yield the same result as accessing them in one word.

11.1.1 Features

The TIM16B8CV3 includes these distinctive features:

- Up to 8 channels available. (refer to device specification for exact number)

- All channels have same input capture/output compare functionality.
- Clock prescaling.
- 16-bit counter.
- 16-bit pulse accumulator on channel 7 .

11.1.2 Modes of Operation

Stop: Timer is off because clocks are stopped.

Freeze: Timer counter keeps on running, unless TSFRZ in TSCR1 is set to 1.

Wait: Counters keeps on running, unless TSWAI in TSCR1 is set to 1.

Normal: Timer counter keep on running, unless TEN in TSCR1 is cleared to 0.

11.1.3 Block Diagrams

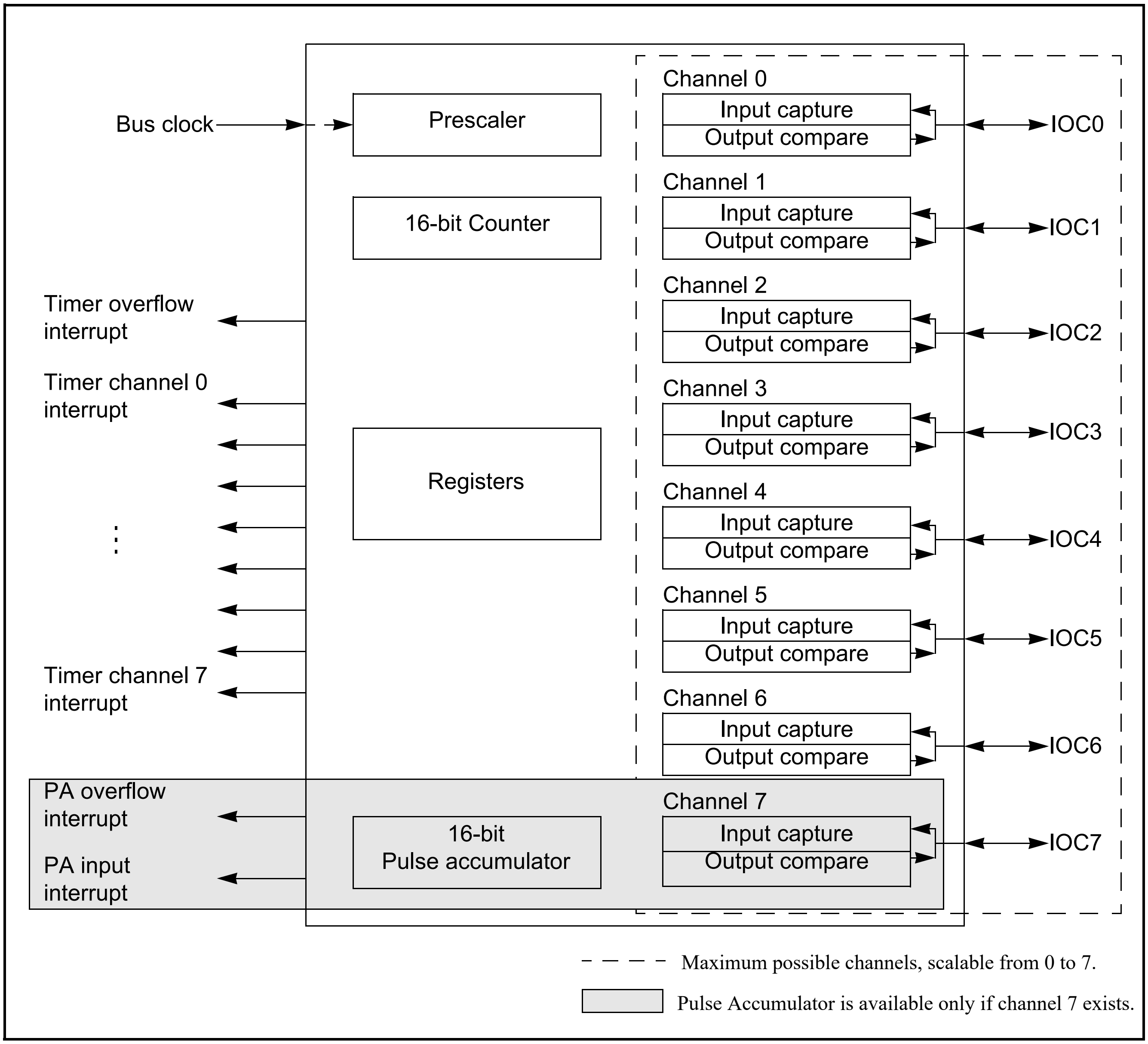


Figure 11-1. TIM16B8CV3 Block Diagram

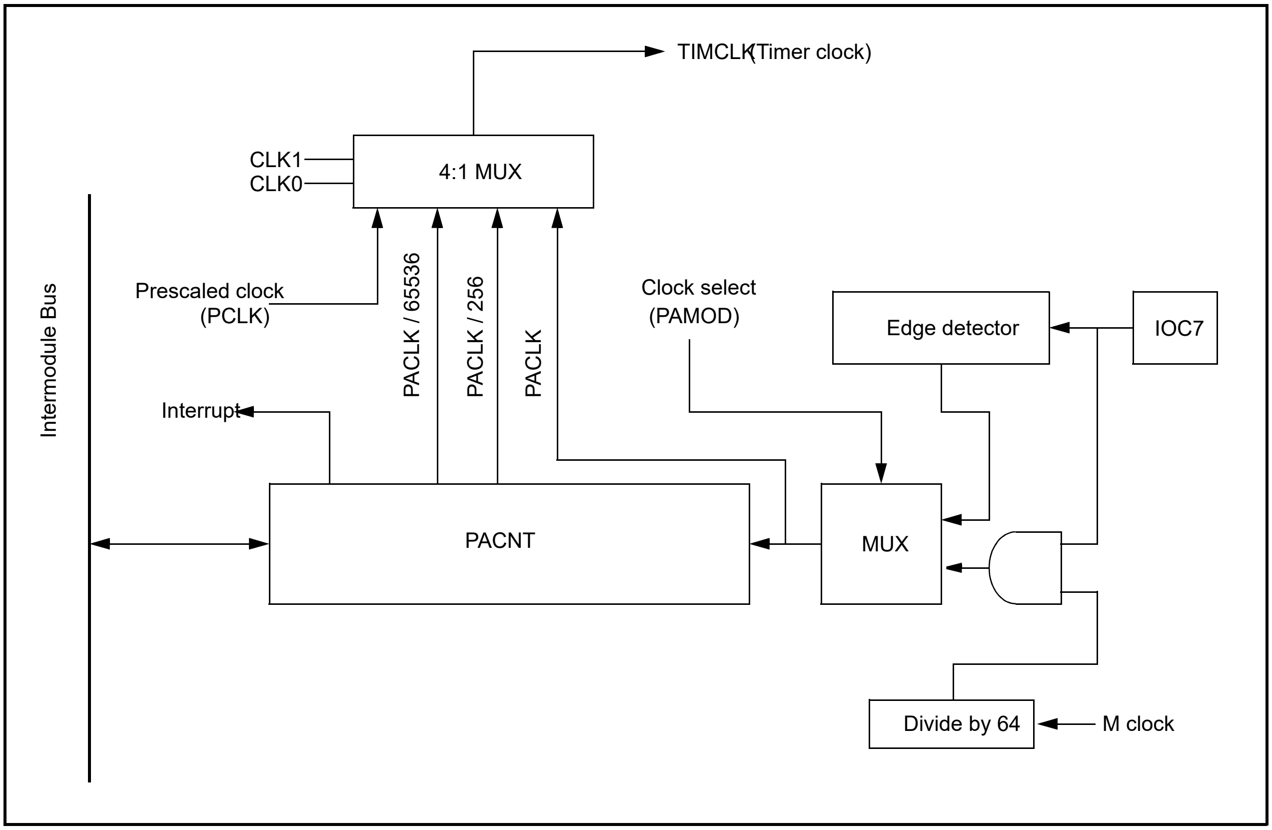


Figure 11-2. 16-Bit Pulse Accumulator Block Diagram

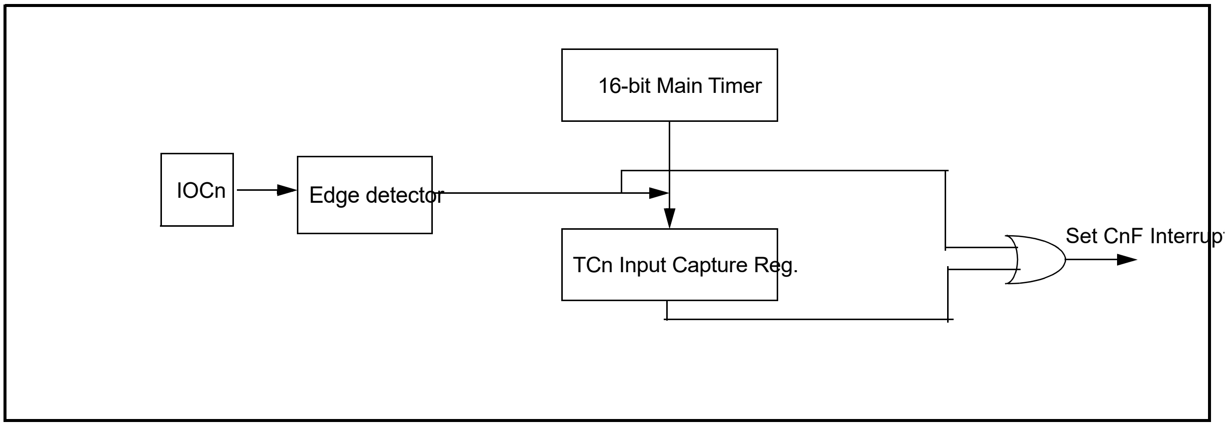


Figure 11-3. Interrupt Flag Setting

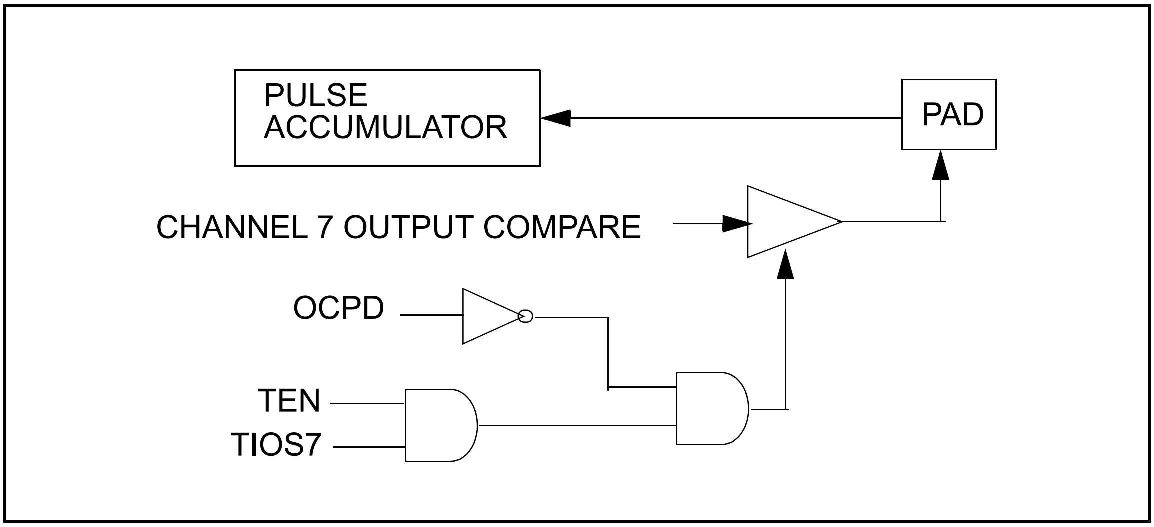


Figure 11-4. Channel 7 Output Compare/Pulse Accumulator Logic

11.2 External Signal Description

The TIM16B8CV3 module has a selected number of external pins. Refer to device specification for exact number.

11.2.1 IOC7 — Input Capture and Output Compare Channel 7

This pin serves as input capture or output compare for channel 7 . This can also be configured as pulse accumulator input.

11.2.2 IOC6 - IOC0 — Input Capture and Output Compare Channel 6-0

Those pins serve as input capture or output compare for TIM16B8CV3 channel .

NOTE

For the description of interrupts see Section 11.6, “Interrupts”.

11.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers.

11.3.1 Module Memory Map

The memory map for the TIM16B8CV3 module is given below in Figure 11-5. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the TIM16B8CV3 module and the address offset for each register.

11.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Only bits related to implemented channels are valid.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 TIOS	R	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
	W								
0x0001 CFORC	R	0	0	0	0	0	0	0	0
	W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
0x0002 OC7M	R	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
	W								
0x0003 OC7D	R	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
	W								
0x0004 TCNTH	R	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
	W								
0x0005 TCNTL	R	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
	W								
0x0006 TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
	W								
0x0007 TTOV	R	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
	W								
0x0008 TCTL1	R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
	W								
0x0009 TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
	W								
0x000A TCTL3	R	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
	W								
0x000B TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
	W								
0x000C TIE	R	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
	W								
0x000D TSCR2	R	TOI	0	0	0	TCRE	PR2	PR1	PR0
	W								
0x000E TFLG1	R	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
	W								
0x000F TFLG2	R	TOF	0	0	0	0	0	0	0
	W								
0x0010–0x001F TCxH–TCxL[1]	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0020 PACTL	R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
	W								

Figure 11-5. TIM16B8CV3 Register Summary (Sheet 1 of 2)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0021 PAFLG	R	0	0	0	0	0	0	PAOVF	PAIF
	W								
0x0022 PACNTH	R	PACNT15	PACNT14	PACNT13	PACNT12	PACNT11	PACNT10	PACNT9	PACNT8
	W								
0x0023 PACNTL	R	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
	W								
0x0024–0x002B Reserved	R								
	W								
0x002C OCPD	R	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0
	W								
0x002D Reserved	R								
0x002E PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
	W								
0x002F Reserved	R								
	W								

Figure 11-5. TIM16B8CV3 Register Summary (Sheet 2 of 2)

[1] The register is available only if corresponding channel exists.

11.3.2.1 Timer Input Capture/Output Compare Select (TIOS)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
Reset	0	0	0	0	0	0	0	0

Figure 11-6. Timer Input Capture/Output Compare Select (TIOS)

Read: Anytime

Write: Anytime

Table 11-2. TIOS Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 IOS[7:0]	**Input Capture or Output Compare Channel Configuration** 0 The corresponding implemented channel acts as an input capture. 1 The corresponding implemented channel acts as an output compare.

11.3.2.2 Timer Compare Force Register (CFORC)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
Reset	0	0	0	0	0	0	0	0

Figure 11-7. Timer Compare Force Register (CFORC)

Read: Anytime but will always return 0x0000 (1 state is transient)

Write: Anytime

Table 11-3. CFORC Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 FOC[7:0]	**Note: Force Output Compare Action for Channel** 7:**0** — A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set. A channel 7 event, which can be a counter overflow when TTOV[7] is set or a successful output compare on channel 7, overrides any channel 6:0 compares. If forced output compare on any channel occurs at the same time as the successful output compare then forced output compare action will take precedence and interrupt flag won’t get set.

11.3.2.3 Output Compare 7 Mask Register (OC7M)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
Reset	0	0	0	0	0	0	0	0

Figure 11-8. Output Compare 7 Mask Register (OC7M)

Read: Anytime

Write: Anytime

Table 11-4. OC7M Field Descriptions

Field	Description
7:0 OC7M[7:0]	**Output Compare 7 Mask** — A channel 7 event, which can be a counter overflow when TTOV[7] is set or a successful output compare on channel 7, overrides any channel 6:0 compares. For each OC7M bit that is set, the output compare action reflects the corresponding OC7D bit. 0 The corresponding OC7Dx bit in the output compare 7 data register will not be transferred to the timer port on a channel 7 event, even if the corresponding pin is setup for output compare. 1 The corresponding OC7Dx bit in the output compare 7 data register will be transferred to the timer port on a channel 7 event. **Note:** The corresponding channel must also be setup for output compare (IOSx = 1 and OCPDx = 0) for data to be transferred from the output compare 7 data register to the timer port.

11.3.2.4 Output Compare 7 Data Register (OC7D)

1 .

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
Reset	0	0	0	0	0	0	0	0

Figure 11-9. Output Compare 7 Data Register (OC7D)

Read: Anytime

Write: Anytime

Table 11-5. OC7D Field Descriptions

Field	Description
7:0 OC7D[7:0]	**Output Compare 7 Data** — A channel 7 event, which can be a counter overflow when TTOV[7] is set or a successful output compare on channel 7, can cause bits in the output compare 7 data register to transfer to the timer port data register depending on the output compare 7 mask register.

11.3.2.5 Timer Count Register (TCNT)

Module Base + 0x0004

	15	14	13	12	11	10	9	9
R W	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
Reset	0	0	0	0	0	0	0	0

Figure 11-10. Timer Count Register High (TCNTH)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
Reset	0	0	0	0	0	0	0	0

Figure 11-11. Timer Count Register Low (TCNTL)

The 16-bit main timer is an up counter.

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

Read: Anytime

Write: Has no meaning or effect in the normal mode; only writable in special modes .

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

11.3.2.6 Timer System Control Register 1 (TSCR1)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
W						▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 11-12. Timer System Control Register 1 (TSCR1)

Read: Anytime

Write: Anytime

Table 11-6. TSCR1 Field Descriptions

Field	Description
7 TEN	**Timer Enable** 0 Disables the main timer, including the counter. Can be used for reducing power consumption. 1 Allows the timer to function normally. If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator because the ÷64 is generated by the timer prescaler.
6 TSWAI	**Timer Module Stops While in Wait** 0 Allows the timer module to continue running during wait. 1 Disables the timer module when the MCU is in the wait mode. Timer interrupts cannot be used to get the MCU out of wait. TSWAI also affects pulse accumulator.

Table 11-6. TSCR1 Field Descriptions (continued)

Field	Description
5 TSFRZ	**Timer Stops While in Freeze Mode** 0 Allows the timer counter to continue running while in freeze mode. 1 Disables the timer counter whenever the MCU is in freeze mode. This is useful for emulation. TSFRZ does not stop the pulse accumulator.
4 TFFCA	**Timer Fast Flag Clear All** 0 Allows the timer flag clearing to function normally. 1 For TFLG1(0x000E), a read from an input capture or a write to the output compare channel (0x0010–0x001F) causes the corresponding channel flag, CnF, to be cleared. For TFLG2 (0x000F), any access to the TCNT register (0x0004, 0x0005) clears the TOF flag. Any access to the PACNT registers (0x0022, 0x0023) clears the PAOVF and PAIF flags in the PAFLG register (0x0021) if channel 7 exists. This has the advantage of eliminating software overhead in a separate clear sequence. Extra care is required to avoid accidental flag clearing due to unintended accesses.
3 PRNT	**Precision Timer** 0 Enables legacy timer. PR0, PR1, and PR2 bits of the TSCR2 register are used for timer counter prescaler selection. 1 Enables precision timer. All bits of the PTPSR register are used for Precision Timer Prescaler Selection, and all bits. This bit is writable only once out of reset.

11.3.2.7 Timer Toggle On Overflow Register 1 (TTOV)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
Reset	0	0	0	0	0	0	0	0

Figure 11-13. Timer Toggle On Overflow Register 1 (TTOV)

Read: Anytime

Write: Anytime

Table 11-7. TTOV Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 TOV[7:0]	**Toggle On Overflow Bits** — TOVx toggles output compare pin on overflow. This feature only takes effect when in output compare mode. When set, it takes precedence over forced output compare but not channel 7 override events. 0 Toggle output compare pin on overflow feature disabled. 1 Toggle output compare pin on overflow feature enabled.

11.3.2.8 Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R W	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
Reset	0	0	0	0	0	0	0	0

Figure 11-14. Timer Control Register 1 (TCTL1)

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R W	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
Reset	0	0	0	0	0	0	0	0

Figure 11-15. Timer Control Register 2 (TCTL2)

Read: Anytime

Write: Anytime

Table 11-8. TCTL1/TCTL2 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

Field	Description
7:0 OMx	**Output Mode** — These eight pairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** To enable output action by OMx bits on timer port, the corresponding bit in OC7M should be cleared. For an output line to be driven by an OCx the OCPDx must be cleared.
7:0 OLx	**Output Level** — These eightpairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** To enable output action by OLx bits on timer port, the corresponding bit in OC7M should be cleared. For an output line to be driven by an OCx the OCPDx must be cleared.

Table 11-9. Compare Result Output Action

OMx	OLx	Action
0	0	No output compare action on the timer output signal
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

Note: To enable output action using the OM7 and OL7 bits on the timer port,the corresponding bit OC7M7 in the OC7M register must also be cleared. The settings for these bits can be seen inTable 11-10.

Table 11-10. The OC7 and OCx event priority

OC7M7=0				OC7M7=1			
OC7Mx=1		OC7Mx=0		OC7Mx=1		OC7Mx=0	
TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx
IOCx=OC7Dx IOC7=OM7/OL7	IOCx=OC7Dx +OMx/OLx IOC7=OM7/OL7	IOCx=OMx/OLx IOC7=OM7/OL7		IOCx=OC7Dx IOC7=OC7D7	IOCx=OC7Dx +OMx/OLx IOC7=OC7D7	IOCx=OMx/OLx IOC7=OC7D7	

Note: in Table 11-10, the IOS7 and IOSx should be set to 1

IOSx is the register TIOS bit x,

OC7Mx is the register OC7M bit x,

TCx is timer Input Capture/Output Compare register,

IOCx is channel x,

OMx/OLx is the register TCTL1/TCTL2,

OC7Dx is the register OC7D bit x.

IOCx = OC7Dx+ OMx/OLx, means that both OC7 event and OCx event will change channel x value.

11.3.2.9 Timer Control Register 3/Timer Control Register 4 (TCTL3 and TCTL4)

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
Reset	0	0	0	0	0	0	0	0

Figure 11-16. Timer Control Register 3 (TCTL3)

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
Reset	0	0	0	0	0	0	0	0

Figure 11-17. Timer Control Register 4 (TCTL4)

Read: Anytime

Write: Anytime.

Table 11-11. TCTL3/TCTL4 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 EDGnB EDGnA	**Input Capture Edge Control** — These eight pairs of control bits configure the input capture edge detector circuits.

Table 11-12. Edge Detector Circuit Configuration

EDGnB	EDGnA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

11.3.2.10 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

	7	6	5	4	3	2	1	0
R W	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
Reset	0	0	0	0	0	0	0	0

Figure 11-18. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 11-13. TIE Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

Field	Description
7:0 C7I:C0I	**Input Capture/Output Compare “x” Interrupt Enable —** The bits in TIE correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause a interrupt.

11.3.2.11 Timer System Control Register 2 (TSCR2)

Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	TOI	0	0	0	TCRE	PR2	PR1	PR0
W		▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 11-19. Timer System Control Register 2 (TSCR2)

Read: Anytime

Write: Anytime.

Table 11-14. TSCR2 Field Descriptions

Field	Description
7 TOI	**Timer Overflow Interrupt Enable** 0 Interrupt inhibited. 1 Hardware interrupt requested when TOF flag set.
3 TCRE	**Timer Counter Reset Enable** — This bit allows the timer counter to be reset by a successful output compare 7 event. This mode of operation is similar to an up-counting modulus counter. 0 Counter reset inhibited and counter free runs. 1 Counter reset by a successful output compare 7. **Note:** If TC7 = 0x0000 and TCRE = 1, TCNT will stay at 0x0000 continuously. If TC7 = 0xFFFF and TCRE = 1, TOF will never be set when TCNT is reset from 0xFFFF to 0x0000. **Note:** TCRE=1 and TC7!=0, the TCNT cycle period will be TC7 x "prescaler counter width" + "1 Bus Clock", for a more detail explanation please refer to Section 11.4.3, “Output Compare **Note:** This bit and feature is available only when channel 7 exists. If channel 7 doesn’t exist, this bit is reserved. Writing to reserved bit has no effect. Read from reserved bit return a zero.
2:0 PR[2:0]	**Timer Prescaler Select** — These three bits select the frequency of the timer prescaler clock derived from the Bus Clock as shown in Table 11-15.

Table 11-15. Timer Clock Selection

PR2	PR1	PR0	Timer Clock
0	0	0	Bus Clock / 1
0	0	1	Bus Clock / 2
0	1	0	Bus Clock / 4
0	1	1	Bus Clock / 8
1	0	0	Bus Clock / 16
1	0	1	Bus Clock / 32
1	1	0	Bus Clock / 64
1	1	1	Bus Clock / 128

NOTE

The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

11.3.2.12 Main Timer Interrupt Flag 1 (TFLG1)

Module Base + 0x000E

	7	6	5	4	3	2	1	0
R W	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
Reset	0	0	0	0	0	0	0	0

Figure 11-20. Main Timer Interrupt Flag 1 (TFLG1)

Read: Anytime

Write: Used in the clearing mechanism (set bits cause corresponding bits to be cleared). Writing a zero will not affect current status of the bit.

Table 11-16. TRLG1 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 C[7:0]F	**Input Capture/Output Compare Channel "x" Flag** — These flags are set when an input capture or output compare event occurs. Clearing requires writing a one to the corresponding flag bit while TEN or PAEN is set to one. **Note:** When TFFCA bit in TSCR register is set, a read from an input capture or a write into an output compare channel (0x0010–0x001F) will cause the corresponding channel flag CxF to be cleared.

11.3.2.13 Main Timer Interrupt Flag 2 (TFLG2)

Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	TOF	0	0	0	0	0	0	0
W		▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ Unimplemented or Reserved

Figure 11-21. Main Timer Interrupt Flag 2 (TFLG2)

TFLG2 indicates when interrupt conditions have occurred. To clear a bit in the flag register, write the bit to one while TEN bit of TSCR1 or PAEN bit of PACTL is set to one.

Read: Anytime

Write: Used in clearing mechanism (set bits cause corresponding bits to be cleared).

Any access to TCNT will clear TFLG2 register if the TFFCA bit in TSCR register is set.

Table 11-17. TRLG2 Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — Set when 16-bit free-running timer overflows from 0xFFFF to 0x0000. Clearing this bit requires writing a one to bit 7 of TFLG2 register while the TEN bit of TSCR1 or PAEN bit of PACTL is set to one (See also TCRE control bit explanation) .

11.3.2.14 Timer Input Capture/Output Compare Registers High and Low 0–7(TCxH and TCxL)

Module Base +		
	0x0010 = TC0H	0x0018=TC4H
	0x0012 = TC1H	0x001A=TC5H
	0x0014=TC2H	0x001C=TC6H
	0x0016=TC3H	0x001E=TC7H

	15	14	13	12	11	10	9	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 11-22. Timer Input Capture/Output Compare Register x High (TCxH)

Module Base +		
	0x0011 = TC0L	0x0019 =TC4L
	0x0013 = TC1L	0x001B=TC5L
	0x0015 =TC2L	0x001D=TC6L
	0x0017=TC3L	0x001F=TC7L

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 11-23. Timer Input Capture/Output Compare Register x Low (TCxL)

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

Read: Anytime

Write: Anytime for output compare function.Writes to these registers have no meaning or effect during input capture. All timer input capture/output compare registers are reset to 0x0000.

NOTE

> Read/Write access in byte mode for high byte should take place before low byte otherwise it will give a different result.

11.3.2.15 16-Bit Pulse Accumulator Control Register (PACTL)

Module Base + 0x0020

	7	6	5	4	3	2	1	0
R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ Unimplemented or Reserved

Figure 11-24. 16-Bit Pulse Accumulator Control Register (PACTL)

Read: Any time

Write: Any time

When PAEN is set, the Pulse Accumulator counter is enabled. The Pulse Accumulator counter shares the input pin with IOC7.

Table 11-18. PACTL Field Descriptions

Field	Description
6 PAEN	**Pulse Accumulator System Enable** — PAEN is independent from TEN. With timer disabled, the pulse accumulator can function unless pulse accumulator is disabled. 0 16-Bit Pulse Accumulator system disabled. 1 Pulse Accumulator system enabled.
5 PAMOD	**Pulse Accumulator Mode** — This bit is active only when the Pulse Accumulator is enabled (PAEN = 1). See Table 11-19. 0 Event counter mode. 1 Gated time accumulation mode.
4 PEDGE	**Pulse Accumulator Edge Control** — This bit is active only when the Pulse Accumulator is enabled (PAEN = 1). For PAMOD bit = 0 (event counter mode). See Table 11-19. 0 Falling edges on IOC7 pin cause the count to be increased. 1 Rising edges on IOC7 pin cause the count to be increased. For PAMOD bit = 1 (gated time accumulation mode). 0 IOC7 input pin high enables M (Bus clock) divided by 64 clock to Pulse Accumulator and the trailing falling edge on IOC7 sets the PAIF flag. 1 IOC7 input pin low enables M (Bus clock) divided by 64 clock to Pulse Accumulator and the trailing rising edge on IOC7 sets the PAIF flag.
3:2 CLK[1:0]	**Clock Select Bits —** Refer to Table 11-20.
1 PAOVI	**Pulse Accumulator Overflow Interrupt Enable** 0 Interrupt inhibited. 1 Interrupt requested if PAOVF is set.
0 PAI	**Pulse Accumulator Input Interrupt Enable** 0 Interrupt inhibited. 1 Interrupt requested if PAIF is set.

Table 11-19. Pin Action

PAMOD	PEDGE	Pin Action
0	0	Falling edge
0	1	Rising edge
1	0	Div. by 64 clock enabled with pin high level
1	1	Div. by 64 clock enabled with pin low level

NOTE

If the timer is not active (TEN = 0 in TSCR), there is no divide-by-64 because the ÷64 clock is generated by the timer prescaler.

Table 11-20. Timer Clock Selection

CLK1	CLK0	Timer Clock
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

For the description of PACLK please refer Figure 11-30**.**

If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written.

11.3.2.16 Pulse Accumulator Flag Register (PAFLG)

1 .

Module Base + 0x0021

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PAOVF	PAIF
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ Unimplemented or Reserved

Figure 11-25. Pulse Accumulator Flag Register (PAFLG)

Read: Anytime

Write: Anytime

When the TFFCA bit in the TSCR register is set, any access to the PACNT register will clear all the flags in the PAFLG register. Timer module or Pulse Accumulator must stay enabled (TEN=1 or PAEN=1) while clearing these bits.

Table 11-21. PAFLG Field Descriptions

Field	Description
1 PAOVF	**Pulse Accumulator Overflow Flag** — Set when the 16-bit pulse accumulator overflows from 0xFFFF to 0x0000. Clearing this bit requires writing a one to this bit in the PAFLG register while TEN bit of TSCR1 or PAEN bit of PACTL register is set to one.
0 PAIF	**Pulse Accumulator Input edge Flag** — Set when the selected edge is detected at the IOC7 input pin.In event mode the event edge triggers PAIF and in gated time accumulation mode the trailing edge of the gate signal at the IOC7 input pin triggers PAIF. Clearing this bit requires writing a one to this bit in the PAFLG register while TEN bit of TSCR1 or PAEN bit of PACTL register is set to one. Any access to the PACNT register will clear all the flags in this register when TFFCA bit in register TSCR(0x0006) is set.

11.3.2.17 Pulse Accumulators Count Registers (PACNT)

Module Base + 0x0022

	15	14	13	12	11	10	9	0
R W	PACNT15	PACNT14	PACNT13	PACNT12	PACNT11	PACNT10	PACNT9	PACNT8
Reset	0	0	0	0	0	0	0	0

Figure 11-26. Pulse Accumulator Count Register High (PACNTH)

1 .

Module Base + 0x0023

	7	6	5	4	3	2	1	0
R W	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
Reset	0	0	0	0	0	0	0	0

Figure 11-27. Pulse Accumulator Count Register Low (PACNTL)

Read: Anytime

Write: Anytime

These registers contain the number of active input edges on its input pin since the last reset.

When PACNT overflows from 0xFFFF to 0x0000, the Interrupt flag PAOVF in PAFLG (0x0021) is set.

Full count register access should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

NOTE

> Reading the pulse accumulator counter registers immediately after an active edge on the pulse accumulator input pin may miss the last count because the input has to be synchronized with the Bus clock first.

11.3.2.18 Output Compare Pin Disconnect Register(OCPD)

Module Base + 0x002C

	7	6	5	4	3	2	1	0
R W	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0
Reset	0	0	0	0	0	0	0	0

Figure 11-28. Output Compare Pin Disconnect Register (OCPD)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 11-22. OCPD Field Description

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 OCPD[7:0]	**Output Compare Pin Disconnect Bits** 0 Enables the timer channel port. Output Compare action will occur on the channel pin. These bits do not affect the input capture or pulse accumulator functions. 1 Disables the timer channel port. Output Compare action will not occur on the channel pin, but the output compare flag still become set.

11.3.2.19 Precision Timer Prescaler Select Register (PTPSR)

Module Base + 0x002E

	7	6	5	4	3	2	1	0
R W	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
Reset	0	0	0	0	0	0	0	0

Figure 11-29. Precision Timer Prescaler Select Register (PTPSR)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 11-23. PTPSR Field Descriptions

Field	Description
7:0 PTPS[7:0]	**Precision Timer Prescaler Select Bits** — These eight bits specify the division rate of the main Timer prescaler. These are effective only when the PRNT bit of TSCR1 is set to 1. Table 11-24 shows some selection examples in this case. The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

The Prescaler can be calculated as follows depending on logical value of the PTPS[7:0] and PRNT bit:

PRNT = 1 : Prescaler = PTPS[7:0] + 1

Table 11-24. Precision Timer Prescaler Selection Examples when PRNT = 1

PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0	Prescale Factor
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
0	0	0	1	0	0	1	1	20
0	0	0	1	0	1	0	0	21
0	0	0	1	0	1	0	1	22
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	0	0	253
1	1	1	1	1	1	0	1	254
1	1	1	1	1	1	1	0	255
1	1	1	1	1	1	1	1	256

11.4 Functional Description

This section provides a complete functional description of the timer TIM16B8CV3 block. Please refer to the detailed timer block diagram in Figure 11-30 as necessary.

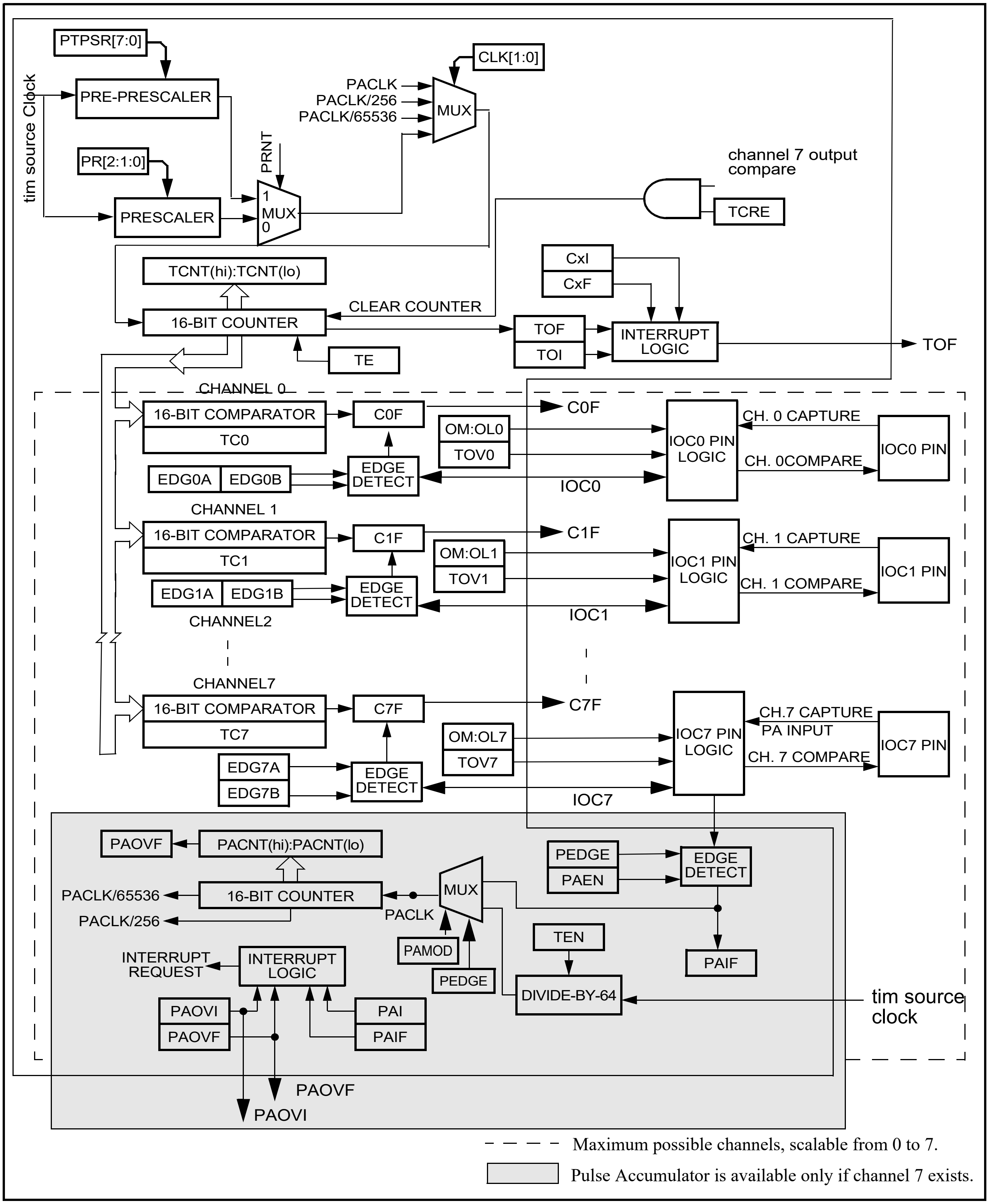


Figure 11-30. Detailed Timer Block Diagram

11.4.1 Prescaler

The prescaler divides the Bus clock by 1, 2, 4, 8, 16, 32, 64 or 128. The prescaler select bits, PR[2:0], select the prescaler divisor. PR[2:0] are in timer system control register 2 (TSCR2).

The prescaler divides the Bus clock by a prescalar value. Prescaler select bits PR[2:0] of in timer system control register 2 (TSCR2) are set to define a prescalar value that generates a divide by 1, 2, 4, 8, 16, 32, 64 and 128 when the PRNT bit in TSCR1 is disabled.

By enabling the PRNT bit of the TSCR1 register, the performance of the timer can be enhanced. In this case, it is possible to set additional prescaler settings for the main timer counter in the present timer by using PTPSR[7:0] bits of PTPSR register generating divide by 1, 2, 3, 4,....20, 21, 22, 23,......255, or 256.

11.4.2 Input Capture

Clearing the I/O (input/output) select bit, IOSx, configures channel x as an input capture channel. The input capture function captures the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the timer transfers the value in the timer counter into the timer channel registers, TCx.

The minimum pulse width for the input capture input is greater than two Bus clocks.

An input capture on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests. Timer module or Pulse Accumulator must stay enabled (TEN bit of TSCR1 or PAEN bit of PACTL register must be set to one) while clearing CxF (writing one to CxF).

11.4.3 Output Compare

Setting the I/O select bit, IOSx, configures channel x when available as an output compare channel. The output compare function can generate a periodic pulse with a programmable polarity, duration, and frequency. When the timer counter reaches the value in the channel registers of an output compare channel, the timer can set, clear, or toggle the channel pin if the corresponding OCPDx bit is set to zero. An output compare on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests. Timer module or Pulse Accumulator must stay enabled (TEN bit of TSCR1 or PAEN bit of PACTL register must be set to one) while clearing CxF (writing one to CxF).

The output mode and level bits, OMx and OLx, select set, clear, toggle on output compare. Clearing both OMx and OLx results in no output compare action on the output compare channel pin.

Setting a force output compare bit, FOCx, causes an output compare on channel x. A forced output compare does not set the channel flag.

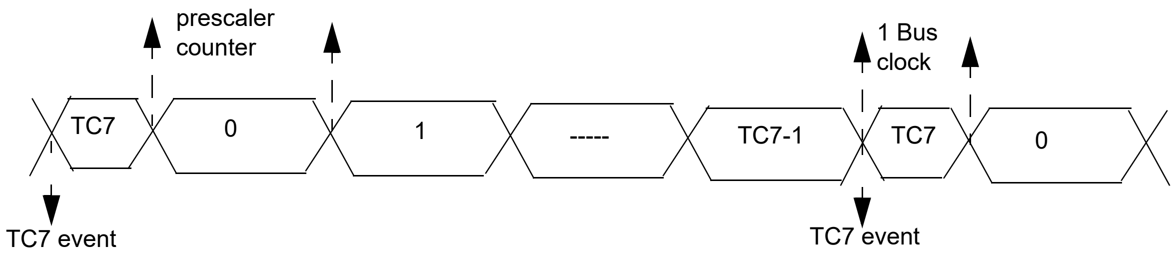
A channel 7 event, which can be a counter overflow when TTOV[7] is set or a successful output compare on channel 7, overrides output compares on all other output compare channels. The output compare 7 mask register masks the bits in the output compare 7 data register. The timer counter reset enable bit, TCRE, enables channel 7 output compares to reset the timer counter. A channel 7 output compare can reset the timer counter even if the IOC7 pin is being used as the pulse accumulator input.

Writing to the timer port bit of an output compare pin does not affect the pin state. The value written is stored in an internal latch. When the pin becomes available for general-purpose output, the last value written to the bit appears at the pin.

When TCRE is set and TC7 is not equal to 0, then TCNT will cycle from 0 to TC7. When TCNT reaches TC7 value, it will last only one Bus cycle then reset to 0.

Note: in Figure 11-31,if PR[2:0] is equal to 0, one prescaler counter equal to one Bus clock

Figure 11-31. The TCNT cycle diagram under TCRE=1 condition



11.4.3.1 OC Channel Initialization

The internal register whose output drives OCx can be programmed before the timer drives OCx. The desired state can be programmed to this internal register by writing a one to CFORCx bit with TIOSx, OCPDx and TEN bits set to one.

Set OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=1 and OCPDx=1

Clear OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=0 and OCPDx=1

Setting OCPDx to zero allows the internal register to drive the programmed state to OCx. This allows a glitch free switch over of port from general purpose I/O to timer output once the OCPDx bit is set to zero.

11.4.4 Pulse Accumulator

The pulse accumulator (PACNT) is a 16-bit counter that can operate in two modes:

Event counter mode — Counting edges of selected polarity on the pulse accumulator input pin, PAI.

Gated time accumulation mode — Counting pulses from a divide-by-64 clock. The PAMOD bit selects the mode of operation.

The minimum pulse width for the PAI input is greater than two Bus clocks.

11.4.5 Event Counter Mode

Clearing the PAMOD bit configures the PACNT for event counter operation. An active edge on the IOC7 pin increments the pulse accumulator counter. The PEDGE bit selects falling edges or rising edges to increment the count.

NOTE

The PACNT input and timer channel 7 use the same pin IOC7. To use the IOC7, disconnect it from the output logic by clearing the channel 7 output mode and output level bits, OM7 and OL7. Also clear the channel 7 output compare 7 mask bit, OC7M7.

The Pulse Accumulator counter register reflect the number of active input edges on the PACNT input pin since the last reset.

The PAOVF bit is set when the accumulator rolls over from 0xFFFF to 0x0000. The pulse accumulator overflow interrupt enable bit, PAOVI, enables the PAOVF flag to generate interrupt requests.

NOTE

The pulse accumulator counter can operate in event counter mode even when the timer enable bit, TEN, is clear.

11.4.6 Gated Time Accumulation Mode

Setting the PAMOD bit configures the pulse accumulator for gated time accumulation operation. An active level on the PACNT input pin enables a divided-by-64 clock to drive the pulse accumulator. The PEDGE bit selects low levels or high levels to enable the divided-by-64 clock.

The trailing edge of the active level at the IOC7 pin sets the PAIF. The PAI bit enables the PAIF flag to generate interrupt requests.

The pulse accumulator counter register reflect the number of pulses from the divided-by-64 clock since the last reset.

NOTE

The timer prescaler generates the divided-by-64 clock. If the timer is not active, there is no divided-by-64 clock.

11.5 Resets

The reset state of each individual bit is listed within Section 11.3, “Memory Map and Register Definition” which details the registers and their bit fields

11.6 Interrupts

This section describes interrupts originated by the TIM16B8CV3 block. Table 11-25 lists the interrupts generated by the TIM16B8CV3 to communicate with the MCU.

Table 11-25. TIM16B8CV3 **Interrupts**

Interrupt	Offset	Vector	Priority	Source	Description
C[7:0]F	—	—	—	Timer Channel 7–0	Active high timer channel interrupts 7–0
PAOVI	—	—	—	Pulse Accumulator Input	Active high pulse accumulator input interrupt
PAOVF	—	—	—	Pulse Accumulator Overflow	Pulse accumulator overflow interrupt
TOF	—	—	—	Timer Overflow	Timer Overflow interrupt

The TIM16B8CV3 could use up to 11 interrupt vectors. The interrupt vector offsets and interrupt numbers are chip dependent.

11.6.1 Channel [7:0] Interrupt (C[7:0]F)

This active high outputs will be asserted by the module to request a timer channel 7 – 0 interrupt. The TIM block only generates the interrupt and does not service it. Only bits related to implemented channels are valid.

11.6.2 Pulse Accumulator Input Interrupt (PAOVI)

This active high output will be asserted by the module to request a timer pulse accumulator input interrupt. The TIM block only generates the interrupt and does not service it.

11.6.3 Pulse Accumulator Overflow Interrupt (PAOVF)

This active high output will be asserted by the module to request a timer pulse accumulator overflow interrupt. The TIM block only generates the interrupt and does not service it.

11.6.4 Timer Overflow Interrupt (TOF)

This active high output will be asserted by the module to request a timer overflow interrupt. The TIM block only generates the interrupt and does not service it.

Chapter 12
Timer Module (TIM16B4CV3) Block Description

Table 12-1.

V03.00	Jan. 28, 2009		Initial version
V03.01	Aug. 26, 2009	12.1.2/12-388 Figure 1-4./1-8 1.3.2.15/1-18 12.3.2.2/12-391, 1.3.2.3/1-8, Chapter 12/12-387, 12.4.3/12-403	- Correct typo: TSCR ->TSCR1; - Correct typo: ECTxxx->TIMxxx - Correct reference: Figure 1-25 -> Figure 12-22 - Add description, “a counter overflow when TTOV[7] is set”, to be the condition of channel 7 override event. - Phrase the description of OC7M to make it more explicit
V03.02	Apri,12,2010	12.3.2.6/12-394 12.3.2.9/12-396 12.4.3/12-403	-Add Table 1-10 -update TCRE bit description -add Figure 1-31
V03.03	Jan,14,2013		-single source generate different channel guide

12.1 Introduction

The basic scalable timer consists of a 16-bit, software-programmable counter driven by a flexible programmable prescaler.

This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform.

This timer could contain up to 4 input capture/output compare channels . The input capture function is used to detect a selected transition edge and record the time. The output compare function is used for generating output signals or for timer software delays.

A full access for the counter registers or the input capture/output compare registers should take place in one clock cycle. Accessing high byte and low byte separately for all of these registers may not yield the same result as accessing them in one word.

12.1.1 Features

The TIM16B4CV3 includes these distinctive features:

- Up to 4 channels available. (refer to device specification for exact number)
- All channels have same input capture/output compare functionality.
- Clock prescaling.

- 16-bit counter.

12.1.2 Modes of Operation

Stop:	Timer is off because clocks are stopped.
Freeze:	Timer counter keeps on running, unless TSFRZ in TSCR1 is set to 1.
Wait:	Counters keeps on running, unless TSWAI in TSCR1 is set to 1.
Normal:	Timer counter keep on running, unless TEN in TSCR1 is cleared to 0.

12.1.3 Block Diagrams

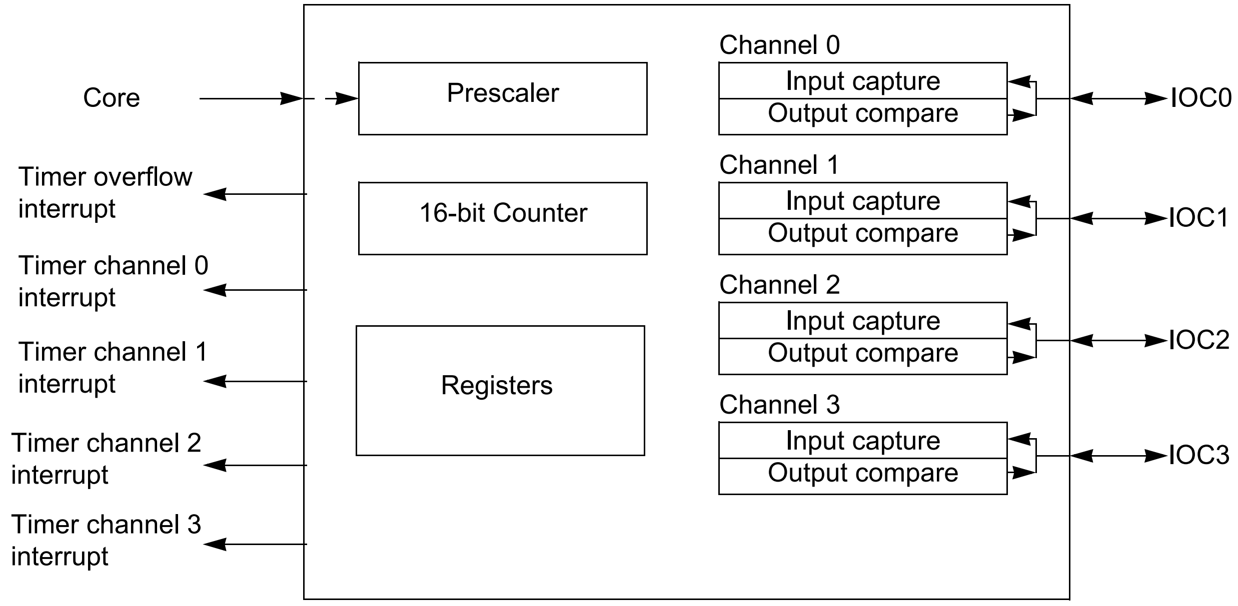


Figure 12-1. TIM16B4CV3 Block Diagram

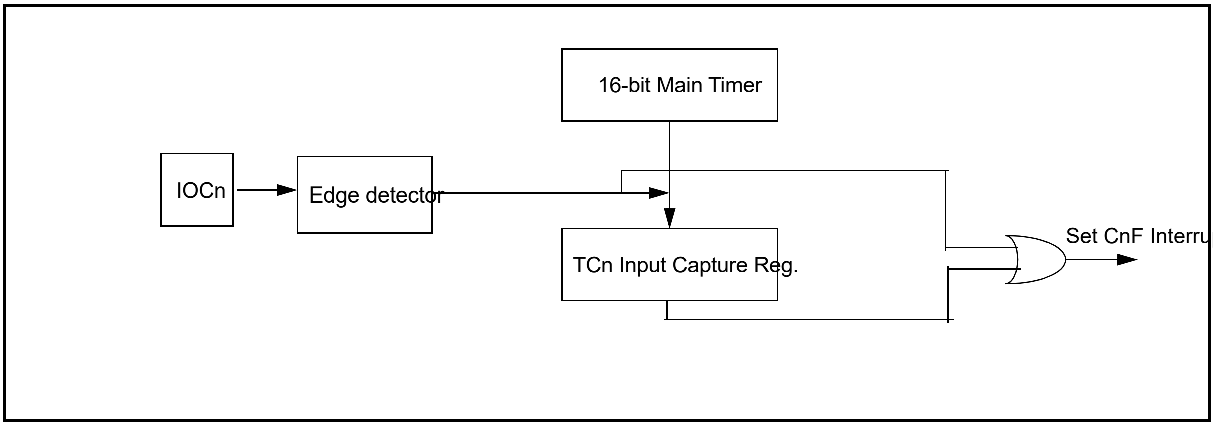


Figure 12-2. Interrupt Flag Setting

12.2 External Signal Description

The TIM16B4CV3 module has a selected number of external pins. Refer to device specification for exact number.

12.2.1 IOC3 - IOC0 — Input Capture and Output Compare Channel 3-0

Those pins serve as input capture or output compare for TIM16B4CV3 channel .

NOTE

For the description of interrupts see Section 12.6, “Interrupts”.

12.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers.

12.3.1 Module Memory Map

The memory map for the TIM16B4CV3 module is given below in Figure 12-3. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the TIM16B4CV3 module and the address offset for each register.

12.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Only bits related to implemented channels are valid.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 TIOS	R W	RESERVED	RESERVED	RESERVED	RESERVED	IOS3	IOS2	IOS1	IOS0
0x0001 CFORC	R	0	0	0	0	0	0	0	0
	W	RESERVED	RESERVED	RESERVED	RESERVED	FOC3	FOC2	FOC1	FOC0
0x0004 TCNTH	R W	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
0x0005 TCNTL	R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
0x0006 TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
	W								
0x0007 TTOV	R W	RESERVED	RESERVED	RESERVED	RESERVED	TOV3	TOV2	TOV1	TOV0
0x0008 TCTL1	R W	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
0x0009 TCTL2	R W	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
0x000A TCTL3	R W	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
0x000B TCTL4	R W	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
0x000C TIE	R W	RESERVED	RESERVED	RESERVED	RESERVED	C3I	C2I	C1I	C0I
0x000D TSCR2	R	TOI	0	0	0	RESERVED	PR2	PR1	PR0
	W								
0x000E TFLG1	R W	RESERVED	RESERVED	RESERVED	RESERVED	C3F	C2F	C1F	C0F
0x000F TFLG2	R	TOF	0	0	0	0	0	0	0
	W								
0x0010–0x001F TCxH–TCxL[1]	R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0024–0x002B Reserved	R W								
0x002C OCPD	R W	RESERVED	RESERVED	RESERVED	RESERVED	OCPD3	OCPD2	OCPD1	OCPD0
0x002D Reserved	R								
0x002E PTPSR	R W	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
0x002F Reserved	R W								

Figure 12-3. TIM16B4CV3 Register Summary

[1] The register is available only if corresponding channel exists.

12.3.2.1 Timer Input Capture/Output Compare Select (TIOS)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	IOS3	IOS2	IOS1	IOS0
Reset	0	0	0	0	0	0	0	0

Figure 12-4. Timer Input Capture/Output Compare Select (TIOS)

Read: Anytime

Write: Anytime

Table 12-2. TIOS Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 IOS[3:0]	**Input Capture or Output Compare Channel Configuration** 0 The corresponding implemented channel acts as an input capture. 1 The corresponding implemented channel acts as an output compare.

12.3.2.2 Timer Compare Force Register (CFORC)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	RESERVED	RESERVED	RESERVED	RESERVED	FOC3	FOC2	FOC1	FOC0
Reset	0	0	0	0	0	0	0	0

Figure 12-5. Timer Compare Force Register (CFORC)

Read: Anytime but will always return 0x0000 (1 state is transient)

Write: Anytime

Table 12-3. CFORC Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 FOC[3:0]	**Note: Force Output Compare Action for Channel** 3:**0** — A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set. If forced output compare on any channel occurs at the same time as the successful output compare then forced output compare action will take precedence and interrupt flag won’t get set.

12.3.2.3 Timer Count Register (TCNT)

Module Base + 0x0004

	15	14	13	12	11	10	9	9
R W	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
Reset	0	0	0	0	0	0	0	0

Figure 12-6. Timer Count Register High (TCNTH)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
Reset	0	0	0	0	0	0	0	0

Figure 12-7. Timer Count Register Low (TCNTL)

The 16-bit main timer is an up counter.

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

Read: Anytime

Write: Has no meaning or effect in the normal mode; only writable in special modes .

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

12.3.2.4 Timer System Control Register 1 (TSCR1)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
W						▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 12-8. Timer System Control Register 1 (TSCR1)

Read: Anytime

Write: Anytime

Table 12-4. TSCR1 Field Descriptions

Field	Description
7 TEN	**Timer Enable** 0 Disables the main timer, including the counter. Can be used for reducing power consumption. 1 Allows the timer to function normally. If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator because the ÷64 is generated by the timer prescaler.
6 TSWAI	**Timer Module Stops While in Wait** 0 Allows the timer module to continue running during wait. 1 Disables the timer module when the MCU is in the wait mode. Timer interrupts cannot be used to get the MCU out of wait. TSWAI also affects pulse accumulator.
5 TSFRZ	**Timer Stops While in Freeze Mode** 0 Allows the timer counter to continue running while in freeze mode. 1 Disables the timer counter whenever the MCU is in freeze mode. This is useful for emulation. TSFRZ does not stop the pulse accumulator.
4 TFFCA	**Timer Fast Flag Clear All** 0 Allows the timer flag clearing to function normally. 1 For TFLG1(0x000E), a read from an input capture or a write to the output compare channel (0x0010–0x001F) causes the corresponding channel flag, CnF, to be cleared. For TFLG2 (0x000F), any access to the TCNT register (0x0004, 0x0005) clears the TOF flag. This has the advantage of eliminating software overhead in a separate clear sequence. Extra care is required to avoid accidental flag clearing due to unintended accesses.
3 PRNT	**Precision Timer** 0 Enables legacy timer. PR0, PR1, and PR2 bits of the TSCR2 register are used for timer counter prescaler selection. 1 Enables precision timer. All bits of the PTPSR register are used for Precision Timer Prescaler Selection, and all bits. This bit is writable only once out of reset.

12.3.2.5 Timer Toggle On Overflow Register 1 (TTOV)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	TOV3	TOV2	TOV1	TOV0
Reset	0	0	0	0	0	0	0	0

Figure 12-9. Timer Toggle On Overflow Register 1 (TTOV)

Read: Anytime

Write: Anytime

Table 12-5. TTOV Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 TOV[3:0]	**Toggle On Overflow Bits** — TOVx toggles output compare pin on overflow. This feature only takes effect when in output compare mode. When set, it takes precedence over forced output compare 0 Toggle output compare pin on overflow feature disabled. 1 Toggle output compare pin on overflow feature enabled.

12.3.2.6 Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
Reset	0	0	0	0	0	0	0	0

Figure 12-10. Timer Control Register 1 (TCTL1)

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R W	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
Reset	0	0	0	0	0	0	0	0

Figure 12-11. Timer Control Register 2 (TCTL2)

Read: Anytime

Write: Anytime

Table 12-6. TCTL1/TCTL2 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

Field	Description
3:0 OMx	**Output Mode** — These four pairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** For an output line to be driven by an OCx the OCPDx must be cleared.
3:0 OLx	**Output Level** — These fourpairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is 1, the pin associated with OCx becomes an output tied to OCx. **Note:** For an output line to be driven by an OCx the OCPDx must be cleared.

Table 12-7. Compare Result Output Action

OMx	OLx	Action
0	0	No output compare action on the timer output signal
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

12.3.2.7 Timer Control Register 3/Timer Control Register 4 (TCTL3 and TCTL4)

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
Reset	0	0	0	0	0	0	0	0

Figure 12-12. Timer Control Register 3 (TCTL3)

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
Reset	0	0	0	0	0	0	0	0

Figure 12-13. Timer Control Register 4 (TCTL4)

Read: Anytime

Write: Anytime.

Table 12-8. TCTL3/TCTL4 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 EDGnB EDGnA	**Input Capture Edge Control** — These four pairs of control bits configure the input capture edge detector circuits.

Table 12-9. Edge Detector Circuit Configuration

EDGnB	EDGnA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

12.3.2.8 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	C3I	C2I	C1I	C0I
Reset	0	0	0	0	0	0	0	0

Figure 12-14. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 12-10. TIE Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero

Field	Description
3:0 C3I:C0I	**Input Capture/Output Compare "x" Interrupt Enable —** The bits in TIE correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause a interrupt.

12.3.2.9 Timer System Control Register 2 (TSCR2)

Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	TOI	0	0	0	RESERVED	PR2	PR1	PR0
W		▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 12-15. Timer System Control Register 2 (TSCR2)s12tim16b4c

Read: Anytime

Write: Anytime.

Table 12-11. TSCR2 Field Descriptions

Field	Description
7 TOI	**Timer Overflow Interrupt Enable** 0 Interrupt inhibited. 1 Hardware interrupt requested when TOF flag set.
2:0 PR[2:0]	**Timer Prescaler Select** — These three bits select the frequency of the timer prescaler clock derived from the Core Clock as shown in Table 12-12.

Table 12-12. Timer Clock Selection

PR2	PR1	PR0	Timer Clock
0	0	0	Core Clock / 1
0	0	1	Core Clock / 2
0	1	0	Core Clock / 4
0	1	1	Core Clock / 8
1	0	0	Core Clock / 16
1	0	1	Core Clock / 32
1	1	0	Core Clock / 64
1	1	1	Core Clock / 128

NOTE

The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

12.3.2.10 Main Timer Interrupt Flag 1 (TFLG1)

Module Base + 0x000E

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	C3F	C2F	C1F	C0F
Reset	0	0	0	0	0	0	0	0

Figure 12-16. Main Timer Interrupt Flag 1 (TFLG1)

Read: Anytime

Write: Used in the clearing mechanism (set bits cause corresponding bits to be cleared). Writing a zero will not affect current status of the bit.

Table 12-13. TRLG1 Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 C[3:0]F	**Input Capture/Output Compare Channel “x” Flag** — These flags are set when an input capture or output compare event occurs. Clearing requires writing a one to the corresponding flag bit while TEN is set to one. **Note:** When TFFCA bit in TSCR register is set, a read from an input capture or a write into an output compare channel (0x0010–0x001F) will cause the corresponding channel flag CxF to be cleared.

12.3.2.11 Main Timer Interrupt Flag 2 (TFLG2)

Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	TOF	0	0	0	0	0	0	0
W		▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ Unimplemented or Reserved

Figure 12-17. Main Timer Interrupt Flag 2 (TFLG2)

TFLG2 indicates when interrupt conditions have occurred. To clear a bit in the flag register, write the bit to one while TEN bit of TSCR1 .

Read: Anytime

Write: Used in clearing mechanism (set bits cause corresponding bits to be cleared).

Any access to TCNT will clear TFLG2 register if the TFFCA bit in TSCR register is set.

Table 12-14. TRLG2 Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — Set when 16-bit free-running timer overflows from 0xFFFF to 0x0000. Clearing this bit requires writing a one to bit 7 of TFLG2 register while the TEN bit of TSCR1 is set to one .

12.3.2.12 Timer Input Capture/Output Compare Registers High and Low 0–3(TCxH and TCxL)

Module Base + 0x0010 = TC0H 0x0018=RESERVD
0x0012 = TC1H 0x001A=RESERVD
0x0014=TC2H 0x001C=RESERVD
0x0016=TC3H 0x001E=RESERVD

	15	14	13	12	11	10	9	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 12-18. Timer Input Capture/Output Compare Register x High (TCxH)

Module Base + 0x0011 = TC0L 0x0019 =RESERVD
0x0013 = TC1L 0x001B=RESERVD
0x0015 =TC2L 0x001D=RESERVD
0x0017=TC3L 0x001F=RESERVD

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 12-19. Timer Input Capture/Output Compare Register x Low (TCxL)

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

Read: Anytime

Write: Anytime for output compare function.Writes to these registers have no meaning or effect during input capture. All timer input capture/output compare registers are reset to 0x0000.

NOTE

Read/Write access in byte mode for high byte should take place before low byte otherwise it will give a different result.

12.3.2.13 Output Compare Pin Disconnect Register(OCPD)

Module Base + 0x002C

	7	6	5	4	3	2	1	0
R W	RESERVED	RESERVED	RESERVED	RESERVED	OCPD3	OCPD2	OCPD1	OCPD0
Reset	0	0	0	0	0	0	0	0

Figure 12-20. Output Compare Pin Disconnect Register (OCPD)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 12-15. OCPD Field Description

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
3:0 OCPD[3:0]	**Output Compare Pin Disconnect Bits** 0 Enables the timer channel port. Output Compare action will occur on the channel pin. These bits do not affect the input capture . 1 Disables the timer channel port. Output Compare action will not occur on the channel pin, but the output compare flag still become set.

12.3.2.14 Precision Timer Prescaler Select Register (PTPSR)

Module Base + 0x002E

	7	6	5	4	3	2	1	0
R W	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
Reset	0	0	0	0	0	0	0	0

Figure 12-21. Precision Timer Prescaler Select Register (PTPSR)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 12-16. PTPSR Field Descriptions

Field	Description
7:0 PTPS[7:0]	**Precision Timer Prescaler Select Bits** — These eight bits specify the division rate of the main Timer prescaler. These are effective only when the PRNT bit of TSCR1 is set to 1. Table 12-17 shows some selection examples in this case. The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

The Prescaler can be calculated as follows depending on logical value of the PTPS[7:0] and PRNT bit:

PRNT = 1 : Prescaler = PTPS[7:0] + 1

Table 12-17. Precision Timer Prescaler Selection Examples when PRNT = 1

PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0	Prescale Factor
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
0	0	0	1	0	0	1	1	20
0	0	0	1	0	1	0	0	21
0	0	0	1	0	1	0	1	22
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	0	0	253
1	1	1	1	1	1	0	1	254
1	1	1	1	1	1	1	0	255
1	1	1	1	1	1	1	1	256

12.4 Functional Description

This section provides a complete functional description of the timer TIM16B4CV3 block. Please refer to the detailed timer block diagram in Figure 12-22 as necessary.

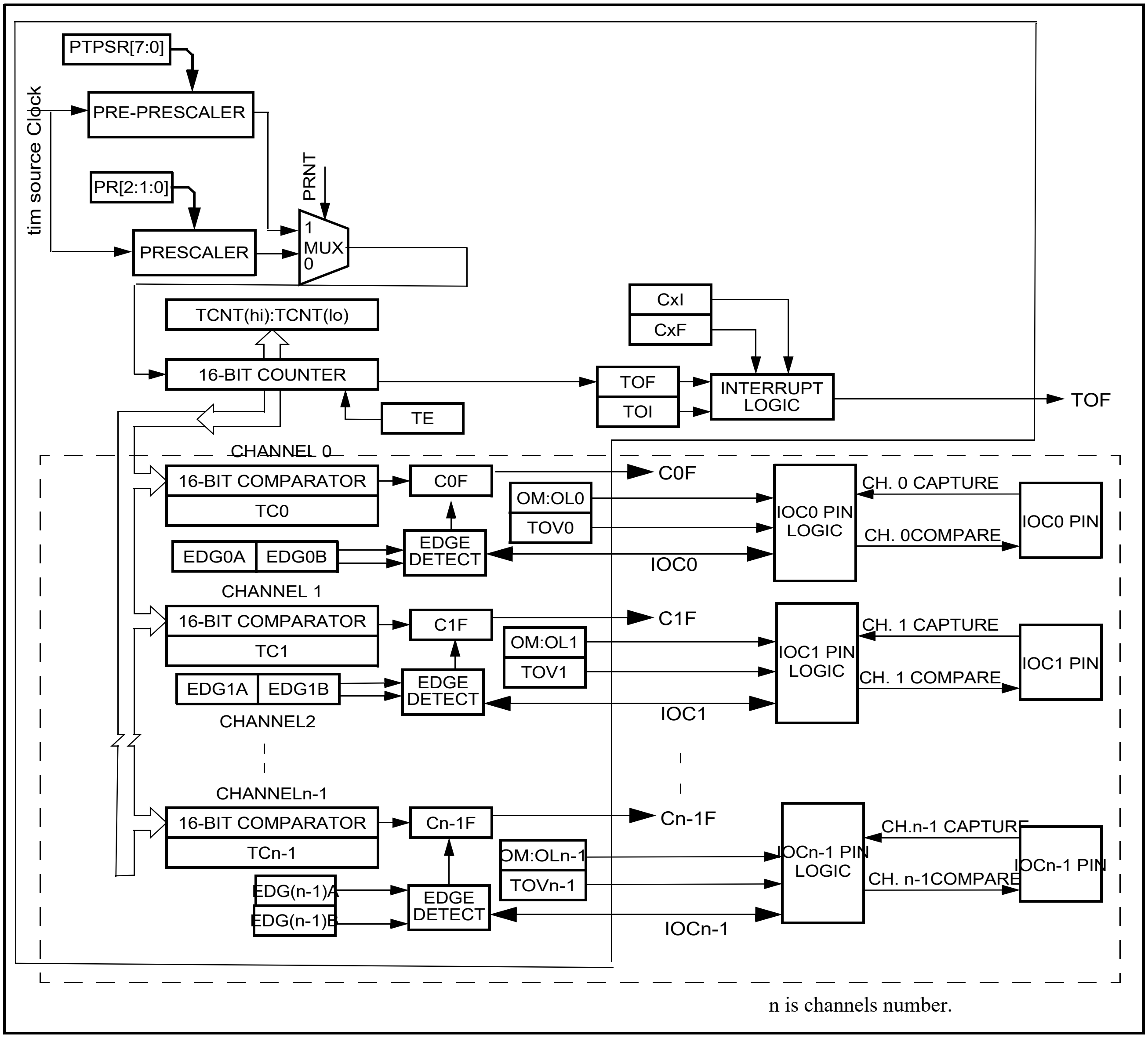


Figure 12-22. Detailed Timer Block Diagram

12.4.1 Prescaler

The prescaler divides the Core clock by 1, 2, 4, 8, 16, 32, 64 or 128. The prescaler select bits, PR[2:0], select the prescaler divisor. PR[2:0] are in timer system control register 2 (TSCR2).

The prescaler divides the Core clock by a prescalar value. Prescaler select bits PR[2:0] of in timer system control register 2 (TSCR2) are set to define a prescalar value that generates a divide by 1, 2, 4, 8, 16, 32, 64 and 128 when the PRNT bit in TSCR1 is disabled.

By enabling the PRNT bit of the TSCR1 register, the performance of the timer can be enhanced. In this case, it is possible to set additional prescaler settings for the main timer counter in the present timer by using PTPSR[7:0] bits of PTPSR register generating divide by 1, 2, 3, 4,....20, 21, 22, 23,......255, or 256.

12.4.2 Input Capture

Clearing the I/O (input/output) select bit, IOSx, configures channel x as an input capture channel. The input capture function captures the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the timer transfers the value in the timer counter into the timer channel registers, TCx.

The minimum pulse width for the input capture input is greater than two Core clocks.

An input capture on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests. Timer module must stay enabled (TEN bit of TSCR1 register must be set to one) while clearing CxF (writing one to CxF).

12.4.3 Output Compare

Setting the I/O select bit, IOSx, configures channel x when available as an output compare channel. The output compare function can generate a periodic pulse with a programmable polarity, duration, and frequency. When the timer counter reaches the value in the channel registers of an output compare channel, the timer can set, clear, or toggle the channel pin if the corresponding OCPDx bit is set to zero. An output compare on channel x sets the CxF flag. The CxI bit enables the CxF flag to generate interrupt requests. Timer module must stay enabled (TEN bit of TSCR1 register must be set to one) while clearing CxF (writing one to CxF).

The output mode and level bits, OMx and OLx, select set, clear, toggle on output compare. Clearing both OMx and OLx results in no output compare action on the output compare channel pin.

Setting a force output compare bit, FOCx, causes an output compare on channel x. A forced output compare does not set the channel flag.

Writing to the timer port bit of an output compare pin does not affect the pin state. The value written is stored in an internal latch. When the pin becomes available for general-purpose output, the last value written to the bit appears at the pin.

12.4.3.1 OC Channel Initialization

The internal register whose output drives OCx can be programmed before the timer drives OCx. The desired state can be programmed to this internal register by writing a one to CFORCx bit with TIOSx, OCPDx and TEN bits set to one.

Set OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=1 and OCPDx=1

Clear OCx: Write a 1 to FOCx while TEN=1, IOSx=1, OMx=1, OLx=0 and OCPDx=1

Setting OCPDx to zero allows the internal register to drive the programmed state to OCx. This allows a glitch free switch over of port from general purpose I/O to timer output once the OCPDx bit is set to zero.

12.5 Resets

The reset state of each individual bit is listed within Section 12.3, "Memory Map and Register Definition" which details the registers and their bit fields

12.6 Interrupts

This section describes interrupts originated by the TIM16B4CV3 block. Table 12-18 lists the interrupts generated by the TIM16B4CV3 to communicate with the MCU.

Table 12-18. TIM16B4CV3 **Interrupts**

Interrupt	Offset	Vector	Priority	Source	Description
C[3:0]F	—	—	—	Timer Channel 3–0	Active high timer channel interrupts 3–0
TOF	—	—	—	Timer Overflow	Timer Overflow interrupt

The TIM16B4CV3 could use up to 5 interrupt vectors. The interrupt vector offsets and interrupt numbers are chip dependent.

12.6.1 Channel [3:0] Interrupt (C[3:0]F)

This active high outputs will be asserted by the module to request a timer channel 7 – 0 interrupt. The TIM block only generates the interrupt and does not service it. Only bits related to implemented channels are valid.

12.6.2 Timer Overflow Interrupt (TOF)

This active high output will be asserted by the module to request a timer overflow interrupt. The TIM block only generates the interrupt and does not service it.

Chapter 13
Pulse-Width Modulator (S12PWM8B8CV2)

Table 13-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
v02.00	Feb. 20, 2009	All	Initial revision of scalable PWM. Started from pwm_8b8c (v01.08).

13.1 Introduction

The Version 2 of S12 PWM module is a channel scalable and optimized implementation of S12 PWM8B8C Version 1. The channel is scalable in pairs from PWM0 to PWM7 and the available channel number is 2, 4, 6 and 8. The shutdown feature has been removed and the flexibility to select one of four clock sources per channel has improved. If the corresponding channels exist and shutdown feature is not used, the Version 2 is fully software compatible to Version 1.

13.1.1 Features

The scalable PWM block includes these distinctive features:

- Up to eight independent PWM channels, scalable in pairs (PWM0 to PWM7)
- Available channel number could be 2, 4, 6, 8 (refer to device specification for exact number)
- Programmable period and duty cycle for each channel
- Dedicated counter for each PWM channel
- Programmable PWM enable/disable for each channel
- Software selection of PWM duty pulse polarity for each channel
- Period and duty cycle are double buffered. Change takes effect when the end of the effective period is reached (PWM counter reaches zero) or when the channel is disabled.
- Programmable center or left aligned outputs on individual channels
- Up to eight 8-bit channel or four 16-bit channel PWM resolution
- Four clock sources (A, B, SA, and SB) provide for a wide range of frequencies
- Programmable clock select logic

13.1.2 Modes of Operation

There is a software programmable option for low power consumption in wait mode that disables the input clock to the prescaler.

In freeze mode there is a software programmable option to disable the input clock to the prescaler. This is useful for emulation.

Wait:	The prescaler keeps on running, unless PSWAI in PWMCTL is set to 1.
Freeze:	The prescaler keeps on running, unless PFRZ in PWMCTL is set to 1.

13.1.3 Block Diagram

Figure 13-1 shows the block diagram for the 8-bit up to 8-channel scalable PWM block.

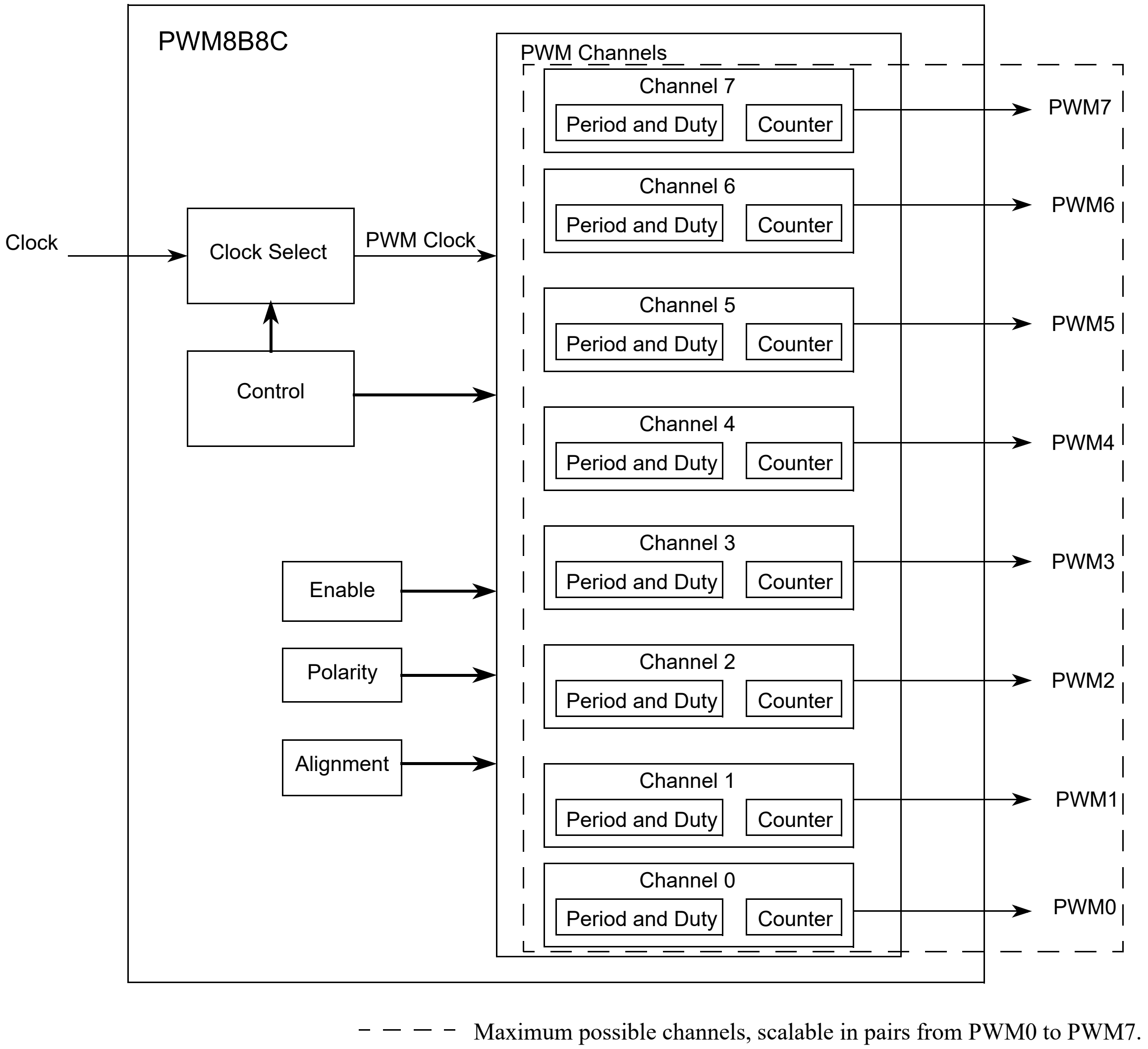


Figure 13-1. Scalable PWM Block Diagram

13.2 External Signal Description

The scalable PWM module has a selected number of external pins. Refer to device specification for exact number.

13.2.1 PWM7 - PWM0 — PWM Channel 7 - 0

Those pins serve as waveform output of PWM channel 7 - 0.

13.3 Memory Map and Register Definition

13.3.1 Module Memory Map

This section describes the content of the registers in the scalable PWM module. The base address of the scalable PWM module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset. The figure below shows the registers associated with the scalable PWM and their relative offset from the base address. The register detail description follows the order they appear in the register map.

Reserved bits within a register will always read as 0 and the write will be unimplemented. Unimplemented functions are indicated by shading the bit.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

13.3.2 Register Descriptions

This section describes in detail all the registers and register bits in the scalable PWM module.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 PWME[1]	R	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
	W								
0x0001 PWMPOL[1]	R	PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0
	W								
0x0002 PWMCLK[1]	R	PCLK7	PCLKL6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0
	W								
0x0003 PWMPRCLK	R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
	W	▨				▨			
0x0004 PWMCAE[1]	R	CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0
	W								
0x0005 PWMCTL[1]	R	CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0
	W							▨	▨

▨ = Unimplemented or Reserved

Figure 13-2. The scalable PWM Register Summary (Sheet 1 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0006 PWMCLKAB[1]	R	PCLKAB7	PCLKAB6	PCLKAB5	PCLKAB4	PCLKAB3	PCLKAB2	PCLKAB1	PCLKAB0
	W								
0x0007 RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0008 PWMSCLA	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0009 PWMSCLB	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x000A RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000B RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000C PWMCNT0[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x000D PWMCNT1[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x000E PWMCNT2[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x000F PWMCNT3[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x0010 PWMCNT4[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x0011 PWMCNT5[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x0012 PWMCNT6[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x0013 PWMCNT7[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W	0	0	0	0	0	0	0	0
0x0014 PWMPER0[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								

▨ = Unimplemented or Reserved

Figure 13-2. The scalable PWM Register Summary (Sheet 2 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0015 PWMPER1[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0016 PWMPER2[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0017 PWMPER3[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0018 PWMPER4[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0019 PWMPER5[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001A PWMPER6[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001B PWMPER7[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001C PWMDTY0[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001D PWMDTY1[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001E PWMDTY2[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001F PWMDTY3[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0010 PWMDTY4[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0021 PWMDTY5[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0022 PWMDTY6[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0023 PWMDTY7[2]	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								

▨ = Unimplemented or Reserved

Figure 13-2. The scalable PWM Register Summary (Sheet 3 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0024 RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0025 RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0026 RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0027 RESERVED	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 13-2. The scalable PWM Register Summary (Sheet 4 of 4)

[1] The related bit is available only if corresponding channel exists.

[2] The register is available only if corresponding channel exists.

13.3.2.1 PWM Enable Register (PWME)

Each PWM channel has an enable bit (PWMEx) to start its waveform output. When any of the PWMEx bits are set (PWMEx = 1), the associated PWM output is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWMEx and the clock source.

NOTE

The first PWM cycle after enabling the channel can be irregular.

An exception to this is when channels are concatenated. Once concatenated mode is enabled (CONxx bits set in PWMCTL register), enabling/disabling the corresponding 16-bit PWM channel is controlled by the low order PWMEx bit. In this case, the high order bytes PWMEx bits have no effect and their corresponding PWM output lines are disabled.

While in run mode, if all existing PWM channels are disabled (PWMEx–0 = 0), the prescaler counter shuts off for power savings.

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
Reset	0	0	0	0	0	0	0	0

Figure 13-3. PWM Enable Register (PWME)

Read: Anytime

Write: Anytime

Table 13-2. PWME Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7 PWME7	**Pulse Width Channel 7 Enable** 0 Pulse width channel 7 is disabled. 1 Pulse width channel 7 is enabled. The pulse modulated signal becomes available at PWM output bit 7 when its clock source begins its next cycle.
6 PWME6	**Pulse Width Channel 6 Enable** 0 Pulse width channel 6 is disabled. 1 Pulse width channel 6 is enabled. The pulse modulated signal becomes available at PWM output bit 6 when its clock source begins its next cycle. If CON67=1, then bit has no effect and PWM output line 6 is disabled.
5 PWME5	**Pulse Width Channel 5 Enable** 0 Pulse width channel 5 is disabled. 1 Pulse width channel 5 is enabled. The pulse modulated signal becomes available at PWM output bit 5 when its clock source begins its next cycle.
4 PWME4	**Pulse Width Channel 4 Enable** 0 Pulse width channel 4 is disabled. 1 Pulse width channel 4 is enabled. The pulse modulated signal becomes available at PWM, output bit 4 when its clock source begins its next cycle. If CON45 = 1, then bit has no effect and PWM output line 4 is disabled.
3 PWME3	**Pulse Width Channel 3 Enable** 0 Pulse width channel 3 is disabled. 1 Pulse width channel 3 is enabled. The pulse modulated signal becomes available at PWM, output bit 3 when its clock source begins its next cycle.
2 PWME2	**Pulse Width Channel 2 Enable** 0 Pulse width channel 2 is disabled. 1 Pulse width channel 2 is enabled. The pulse modulated signal becomes available at PWM, output bit 2 when its clock source begins its next cycle. If CON23 = 1, then bit has no effect and PWM output line 2 is disabled.
1 PWME1	**Pulse Width Channel 1 Enable** 0 Pulse width channel 1 is disabled. 1 Pulse width channel 1 is enabled. The pulse modulated signal becomes available at PWM, output bit 1 when its clock source begins its next cycle.
0 PWME0	**Pulse Width Channel 0 Enable** 0 Pulse width channel 0 is disabled. 1 Pulse width channel 0 is enabled. The pulse modulated signal becomes available at PWM, output bit 0 when its clock source begins its next cycle. If CON01 = 1, then bit has no effect and PWM output line 0 is disabled.

13.3.2.2 PWM Polarity Register (PWMPOL)

The starting polarity of each PWM channel waveform is determined by the associated PPOLx bit in the PWMPOL register. If the polarity bit is one, the PWM channel output is high at the beginning of the cycle and then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0
Reset	0	0	0	0	0	0	0	0

Figure 13-4. PWM Polarity Register (PWMPOL)

Read: Anytime

Write: Anytime

NOTE

PPOLx register bits can be written anytime. If the polarity is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition

Table 13-3. PWMPOL Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7–0 PPOL[7:0]	**Pulse Width Channel 7–0 Polarity Bits** 0 PWM channel 7–0 outputs are low at the beginning of the period, then go high when the duty count is reached. 1 PWM channel 7–0 outputs are high at the beginning of the period, then go low when the duty count is reached.

13.3.2.3 PWM Clock Select Register (PWMCLK)

Each PWM channel has a choice of four clocks to use as the clock source for that channel as described below.

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	PCLK7	PCLKL6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0
Reset	0	0	0	0	0	0	0	0

Figure 13-5. PWM Clock Select Register (PWMCLK)

Read: Anytime

Write: Anytime

NOTE

Register bits PCLK0 to PCLK7 can be written anytime. If a clock select is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

Table 13-4. PWMCLK Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7-0 PCLK[7:0]	**Pulse Width Channel 7-0 Clock Select** 0 Clock A or B is the clock source for PWM channel 7-0, as shown in Table 13-5 and Table 13-6. 1 Clock SA or SB is the clock source for PWM channel 7-0, as shown in Table 13-5 and Table 13-6.

The clock source of each PWM channel is determined by PCLKx bits in PWMCLK and PCLKABx bits in PWMCLKAB (see Section 13.3.2.7, “PWM Clock A/B Select Register (PWMCLKAB)). For Channel 0, 1, 4, 5, the selection is shown in Table 13-5; For Channel 2, 3, 6, 7, the selection is shown in Table 13-6.

Table 13-5. PWM Channel 0, 1, 4, 5 Clock Source Selection

PCLKAB[0,1,4,5]	PCLK[0,1,4,5]	Clock Source Selection
0	0	Clock A
0	1	Clock SA
1	0	Clock B
1	1	Clock SB

Table 13-6. PWM Channel 2, 3, 6, 7 Clock Source Selection

PCLKAB[2,3,6,7]	PCLK[2,3,6,7]	Clock Source Selection
0	0	Clock B
0	1	Clock SB
1	0	Clock A
1	1	Clock SA

13.3.2.4 PWM Prescale Clock Select Register (PWMPRCLK)

This register selects the prescale clock source for clocks A and B independently.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
W	▨				▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 13-6. PWM Prescale Clock Select Register (PWMPRCLK)

Read: Anytime

Write: Anytime

NOTE

PCKB2–0 and PCKA2–0 register bits can be written anytime. If the clock pre-scale is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

Table 13-7. PWMPRCLK Field Descriptions

Field	Description
6–4 PCKB[2:0]	**Prescaler Select for Clock B** — Clock B is one of two clock sources which can be used for all channels. These three bits determine the rate of clock B, as shown in Table 13-8.
2–0 PCKA[2:0]	**Prescaler Select for Clock A** — Clock A is one of two clock sources which can be used for all channels. These three bits determine the rate of clock A, as shown in Table 13-8.

Table 13-8. Clock A or Clock B Prescaler Selects

PCKA/B2	PCKA/B1	PCKA/B0	Value of Clock A/B
0	0	0	PWM clock
0	0	1	PWM clock / 2
0	1	0	PWM clock / 4
0	1	1	PWM clock / 8
1	0	0	PWM clock / 16
1	0	1	PWM clock / 32
1	1	0	PWM clock / 64
1	1	1	PWM clock / 128

13.3.2.5 PWM Center Align Enable Register (PWMCAE)

The PWMCAE register contains eight control bits for the selection of center aligned outputs or left aligned outputs for each PWM channel. If the CAEx bit is set to a one, the corresponding PWM output will be center aligned. If the CAEx bit is cleared, the corresponding PWM output will be left aligned. See Section 13.4.2.5, “Left Aligned Outputs” and Section 13.4.2.6, “Center Aligned Outputs” for a more detailed description of the PWM output modes.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0
Reset	0	0	0	0	0	0	0	0

Figure 13-7. PWM Center Align Enable Register (PWMCAE)

Read: Anytime

Write: Anytime

NOTE

Write these bits only when the corresponding channel is disabled.

Table 13-9. PWMCAE Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7–0 CAE[7:0]	**Center Aligned Output Modes on Channels 7–0** 0 Channels 7–0 operate in left aligned output mode. 1 Channels 7–0 operate in center aligned output mode.

13.3.2.6 PWM Control Register (PWMCTL)

The PWMCTL register provides for various control of the PWM module.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0
W							▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 13-8. PWM Control Register (PWMCTL)

Read: Anytime

Write: Anytime

There are up to four control bits for concatenation, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. If the corresponding channels do not exist on a particular derivative, then writes to these bits have no effect and reads will return zeroes. When channels 6 and 7are concatenated, channel 6 registers become the high order bytes of the double byte channel. When channels 4 and 5 are concatenated, channel 4 registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

See Section 13.4.2.7, “PWM 16-Bit Functions” for a more detailed description of the concatenation PWM Function.

NOTE

Change these bits only when both corresponding channels are disabled.

Table 13-10. PWMCTL Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7 CON67	**Concatenate Channels 6 and 7** 0 Channels 6 and 7 are separate 8-bit PWMs. 1 Channels 6 and 7 are concatenated to create one 16-bit PWM channel. Channel 6 becomes the high order byte and channel 7 becomes the low order byte. Channel 7 output pin is used as the output for this 16-bit PWM (bit 7 of port PWMP). Channel 7 clock select control-bit determines the clock source, channel 7 polarity bit determines the polarity, channel 7 enable bit enables the output and channel 7 center aligned enable bit determines the output mode.
6 CON45	**Concatenate Channels 4 and 5** 0 Channels 4 and 5 are separate 8-bit PWMs. 1 Channels 4 and 5 are concatenated to create one 16-bit PWM channel. Channel 4 becomes the high order byte and channel 5 becomes the low order byte. Channel 5 output pin is used as the output for this 16-bit PWM (bit 5 of port PWMP). Channel 5 clock select control-bit determines the clock source, channel 5 polarity bit determines the polarity, channel 5 enable bit enables the output and channel 5 center aligned enable bit determines the output mode.
5 CON23	**Concatenate Channels 2 and 3** 0 Channels 2 and 3 are separate 8-bit PWMs. 1 Channels 2 and 3 are concatenated to create one 16-bit PWM channel. Channel 2 becomes the high order byte and channel 3 becomes the low order byte. Channel 3 output pin is used as the output for this 16-bit PWM (bit 3 of port PWMP). Channel 3 clock select control-bit determines the clock source, channel 3 polarity bit determines the polarity, channel 3 enable bit enables the output and channel 3 center aligned enable bit determines the output mode.
4 CON01	**Concatenate Channels 0 and 1** 0 Channels 0 and 1 are separate 8-bit PWMs. 1 Channels 0 and 1 are concatenated to create one 16-bit PWM channel. Channel 0 becomes the high order byte and channel 1 becomes the low order byte. Channel 1 output pin is used as the output for this 16-bit PWM (bit 1 of port PWMP). Channel 1 clock select control-bit determines the clock source, channel 1 polarity bit determines the polarity, channel 1 enable bit enables the output and channel 1 center aligned enable bit determines the output mode.
3 PSWAI	**PWM Stops in Wait Mode —** Enabling this bit allows for lower power consumption in wait mode by disabling the input clock to the prescaler. 0 Allow the clock to the prescaler to continue while in wait mode. 1 Stop the input clock to the prescaler whenever the MCU is in wait mode.
2 PFRZ	**PWM Counters Stop in Freeze Mode** — In freeze mode, there is an option to disable the input clock to the prescaler by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode, the input clock to the prescaler is disabled. This feature is useful during emulation as it allows the PWM function to be suspended. In this way, the counters of the PWM can be stopped while in freeze mode so that once normal program flow is continued, the counters are re-enabled to simulate real-time operations. Since the registers can still be accessed in this mode, to re-enable the prescaler clock, either disable the PFRZ bit or exit freeze mode. 0 Allow PWM to continue while in freeze mode. 1 Disable PWM input clock to the prescaler whenever the part is in freeze mode. This is useful for emulation.

13.3.2.7 PWM Clock A/B Select Register (PWMCLKAB)

Each PWM channel has a choice of four clocks to use as the clock source for that channel as described below.

Module Base + 0x00006

	7	6	5	4	3	2	1	0
R W	PCLKAB7	PCLKAB6	PCLKAB5	PCLKAB4	PCLKAB3	PCLKAB2	PCLKAB1	PCLKAB0
Reset	0	0	0	0	0	0	0	0

Figure 13-9. PWM Clock Select Register (PWMCLK)

Read: Anytime

Write: Anytime

NOTE

Register bits PCLKAB0 to PCLKAB7 can be written anytime. If a clock select is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

Table 13-11. PWMCLK Field Descriptions

Note: Bits related to available channels have functional significance. Writing to unavailable bits has no effect. Read from unavailable bits return a zero

Field	Description
7 PCLKAB7	**Pulse Width Channel 7 Clock A/B Select** 0 Clock B or SB is the clock source for PWM channel 7, as shown in Table 13-6. 1 Clock A or SA is the clock source for PWM channel 7, as shown in Table 13-6.
6 PCLKAB6	**Pulse Width Channel 6 Clock A/B Select** 0 Clock B or SB is the clock source for PWM channel 6, as shown in Table 13-6. 1 Clock A or SA is the clock source for PWM channel 6, as shown in Table 13-6.
5 PCLKAB5	**Pulse Width Channel 5 Clock A/B Select** 0 Clock A or SA is the clock source for PWM channel 5, as shown in Table 13-5. 1 Clock B or SB is the clock source for PWM channel 5, as shown in Table 13-5.
4 PCLKAB4	**Pulse Width Channel 4 Clock A/B Select** 0 Clock A or SA is the clock source for PWM channel 4, as shown in Table 13-5. 1 Clock B or SB is the clock source for PWM channel 4, as shown in Table 13-5.
3 PCLKAB3	**Pulse Width Channel 3 Clock A/B Select** 0 Clock B or SB is the clock source for PWM channel 3, as shown in Table 13-6. 1 Clock A or SA is the clock source for PWM channel 3, as shown in Table 13-6.
2 PCLKAB2	**Pulse Width Channel 2 Clock A/B Select** 0 Clock B or SB is the clock source for PWM channel 2, as shown in Table 13-6. 1 Clock A or SA is the clock source for PWM channel 2, as shown in Table 13-6.
1 PCLKAB1	**Pulse Width Channel 1 Clock A/B Select** 0 Clock A or SA is the clock source for PWM channel 1, as shown in Table 13-5. 1 Clock B or SB is the clock source for PWM channel 1, as shown in Table 13-5.
0 PCLKAB0	**Pulse Width Channel 0 Clock A/B Select** 0 Clock A or SA is the clock source for PWM channel 0, as shown in Table 13-5. 1 Clock B or SB is the clock source for PWM channel 0, as shown in Table 13-5.

The clock source of each PWM channel is determined by PCLKx bits in PWMCLK (see Section 13.3.2.3, “PWM Clock Select Register (PWMCLK)) and PCLKABx bits in PWMCLKAB as shown in Table 13-5 and Table 13-6.

13.3.2.8 PWM Scale A Register (PWMSCLA)

PWMSCLA is the programmable scale value used in scaling clock A to generate clock SA. Clock SA is generated by taking clock A, dividing it by the value in the PWMSCLA register and dividing that by two.

Clock SA = Clock A / (2 * PWMSCLA)

NOTE

When PWMSCLA = $00, PWMSCLA value is considered a full scale value of 256. Clock A is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLA).

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 13-10. PWM Scale A Register (PWMSCLA)

Read: Anytime

Write: Anytime (causes the scale counter to load the PWMSCLA value)

13.3.2.9 PWM Scale B Register (PWMSCLB)

PWMSCLB is the programmable scale value used in scaling clock B to generate clock SB. Clock SB is generated by taking clock B, dividing it by the value in the PWMSCLB register and dividing that by two.

Clock SB = Clock B / (2 * PWMSCLB)

NOTE

When PWMSCLB = $00, PWMSCLB value is considered a full scale value of 256. Clock B is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLB).

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 13-11. PWM Scale B Register (PWMSCLB)

Read: Anytime

Write: Anytime (causes the scale counter to load the PWMSCLB value).

13.3.2.10 PWM Channel Counter Registers (PWMCNTx)

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source. The counter can be read at any time without affecting the count or the operation of the PWM channel. In left aligned output mode, the counter counts from 0 to the value in the period register - 1. In center aligned output mode, the counter counts from 0 up to the value in the period register and then back down to 0.

Any value written to the counter causes the counter to reset to $00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. The counter is also cleared at the end of the effective period (see Section 13.4.2.5, “Left Aligned Outputs” and Section 13.4.2.6, “Center Aligned Outputs” for more details). When the channel is disabled (PWMEx = 0), the PWMCNTx register does not count. When a channel becomes enabled (PWMEx = 1), the associated PWM counter starts at the count in the PWMCNTx register. For more detailed information on the operation of the counters, see Section 13.4.2.4, “PWM Timer Counters”.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

NOTE

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

Module Base + 0x000C = PWMCNT0, 0x000D = PWMCNT1, 0x000E = PWMCNT2, 0x000F = PWMCNT3
Module Base + 0x0010 = PWMCNT4, 0x0011 = PWMCNT5, 0x0012 = PWMCNT6, 0x0013 = PWMCNT7

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0

Figure 13-12. PWM Channel Counter Registers (PWMCNTx)

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime (any value written causes PWM counter to be reset to $00).

13.3.2.11 PWM Channel Period Registers (PWMPERx)

There is a dedicated period register for each channel. The value in this register determines the period of the associated PWM channel.

The period registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends

- The counter is written (counter resets to $00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period register will go directly to the latches as well as the buffer.

NOTE

> Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active period due to the double buffering scheme.

See Section 13.4.2.3, “PWM Period and Duty” for more information.

To calculate the output period, take the selected clock source period for the channel of interest (A, B, SA, or SB) and multiply it by the value in the period register for that channel:

- Left aligned output (CAEx = 0)
  PWMx Period = Channel Clock Period * PWMPERx
- Center Aligned Output (CAEx = 1)
  PWMx Period = Channel Clock Period * (2 * PWMPERx)

For boundary case programming values, please refer to Section 13.4.2.8, “PWM Boundary Cases”.

Module Base + 0x0014 = PWMPER0, 0x0015 = PWMPER1, 0x0016 = PWMPER2, 0x0017 = PWMPER3
Module Base + 0x0018 = PWMPER4, 0x0019 = PWMPER5, 0x001A = PWMPER6, 0x001B = PWMPER7

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	1	1	1	1	1	1	1	1

Figure 13-13. PWM Channel Period Registers (PWMPERx)

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime

13.3.2.12 PWM Channel Duty Registers (PWMDTYx)

There is a dedicated duty register for each channel. The value in this register determines the duty of the associated PWM channel. The duty value is compared to the counter and if it is equal to the counter value a match occurs and the output changes state.

The duty registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to $00)

- The channel is disabled

In this way, the output of the PWM will always be either the old duty waveform or the new duty waveform, not some variation in between. If the channel is not enabled, then writes to the duty register will go directly to the latches as well as the buffer.

NOTE

Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active duty due to the double buffering scheme.

See Section 13.4.2.3, “PWM Period and Duty” for more information.

NOTE

Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time. If the polarity bit is one, the output starts high and then goes low when the duty count is reached, so the duty registers contain a count of the high time. If the polarity bit is zero, the output starts low and then goes high when the duty count is reached, so the duty registers contain a count of the low time.

To calculate the output duty cycle (high time as a% of period) for a particular channel:

- Polarity = 0 (PPOL x =0)
  Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
- Polarity = 1 (PPOLx = 1)
  Duty Cycle = [PWMDTYx / PWMPERx] * 100%

For boundary case programming values, please refer to Section 13.4.2.8, “PWM Boundary Cases”.

Module Base + 0x001C = PWMDTY0, 0x001D = PWMDTY1, 0x001E = PWMDTY2, 0x001F = PWMDTY3
Module Base + 0x0020 = PWMDTY4, 0x0021 = PWMDTY5, 0x0022 = PWMDTY6, 0x0023 = PWMDTY7

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	1	1	1	1	1	1	1	1

Figure 13-14. PWM Channel Duty Registers (PWMDTYx)

[1] This register is available only when the corresponding channel exists and is reserved if that channel does not exist. Writes to a reserved register have no functional effect. Reads from a reserved register return zeroes.

Read: Anytime

Write: Anytime

13.4 Functional Description

13.4.1 PWM Clock Select

There are four available clocks: clock A, clock B, clock SA (scaled A), and clock SB (scaled B). These four clocks are based on the PWM clock.

Clock A and B can be software selected to be 1, 1/2, 1/4, 1/8,..., 1/64, 1/128 times the PWM clock. Clock SA uses clock A as an input and divides it further with a reloadable counter. Similarly, clock SB uses clock B as an input and divides it further with a reloadable counter. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB. Each PWM channel has the capability of selecting one of four clocks, clock A, Clock B, clock SA or clock SB.

The block diagram in Figure 13-15 shows the four different clocks and how the scaled clocks are created.

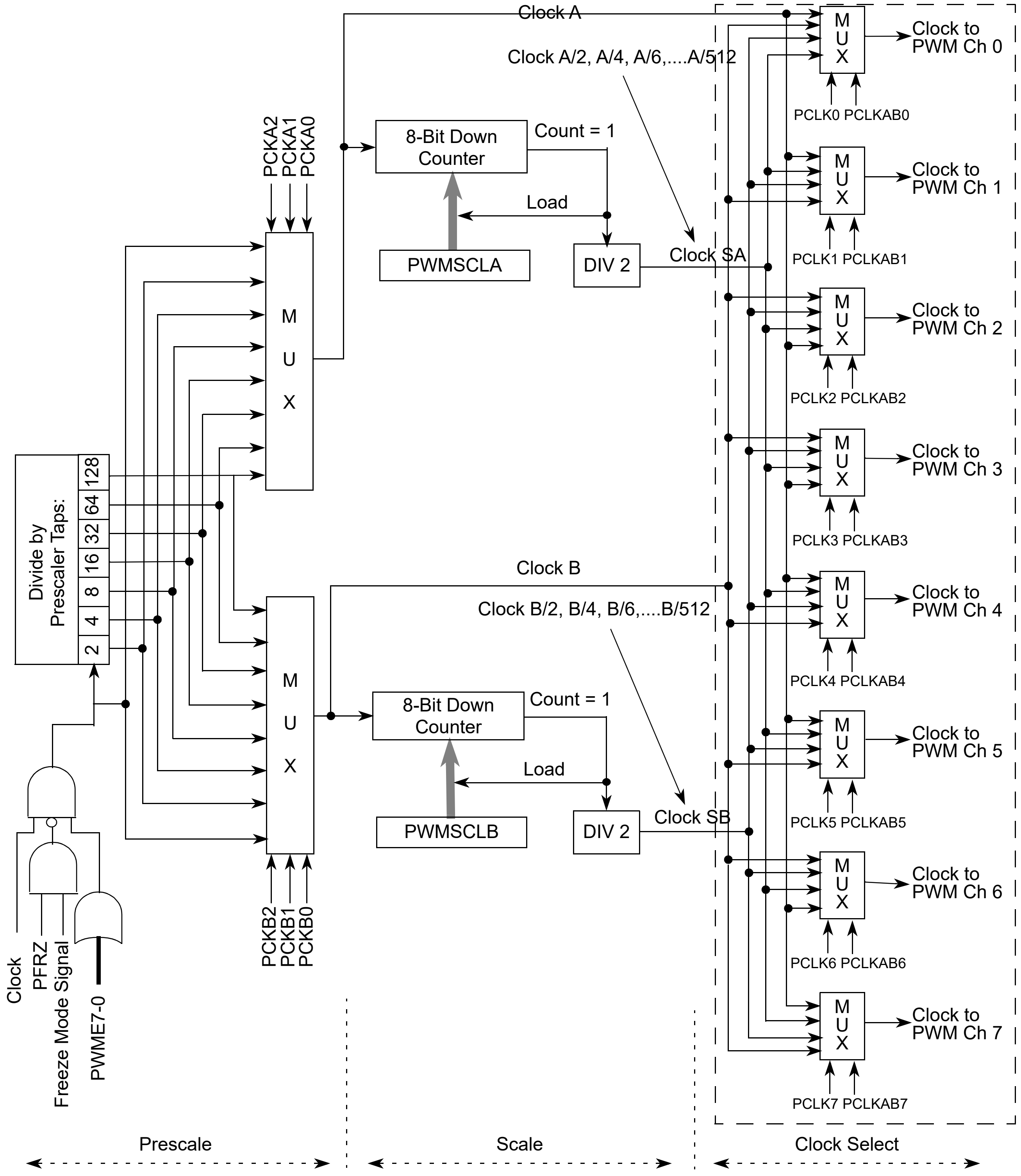
13.4.1.1 Prescale

The input clock to the PWM prescaler is the PWM clock. It can be disabled whenever the part is in freeze mode by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode (freeze mode signal active) the input clock to the prescaler is disabled. This is useful for emulation in order to freeze the PWM. The input clock can also be disabled when all available PWM channels are disabled (PWMEx-0 = 0). This is useful for reducing power by disabling the prescale counter.

Clock A and clock B are scaled values of the input clock. The value is software selectable for both clock A and clock B and has options of 1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, or 1/128 times the PWM clock. The value selected for clock A is determined by the PCKA2, PCKA1, PCKA0 bits in the PWMPRCLK register. The value selected for clock B is determined by the PCKB2, PCKB1, PCKB0 bits also in the PWMPRCLK register.

13.4.1.2 Clock Scale

The scaled A clock uses clock A as an input and divides it further with a user programmable value and then divides this by 2. The scaled B clock uses clock B as an input and divides it further with a user programmable value and then divides this by 2. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB.



– – – – Maximum possible channels, scalable in pairs from PWM0 to PWM7.

Figure 13-15. PWM Clock Select Block Diagram

Clock A is used as an input to an 8-bit down counter. This down counter loads a user programmable scale value from the scale register (PWMSCLA). When the down counter reaches one, a pulse is output and the 8-bit counter is re-loaded. The output signal from this circuit is further divided by two. This gives a greater range with only a slight reduction in granularity. Clock SA equals clock A divided by two times the value in the PWMSCLA register.

NOTE

Clock SA = Clock A / (2 * PWMSCLA)

When PWMSCLA = $00, PWMSCLA value is considered a full scale value of 256. Clock A is thus divided by 512.

Similarly, clock B is used as an input to an 8-bit down counter followed by a divide by two producing clock SB. Thus, clock SB equals clock B divided by two times the value in the PWMSCLB register.

NOTE

Clock SB = Clock B / (2 * PWMSCLB)

When PWMSCLB = $00, PWMSCLB value is considered a full scale value of 256. Clock B is thus divided by 512.

As an example, consider the case in which the user writes $FF into the PWMSCLA register. Clock A for this case will be PWM clock divided by 4. A pulse will occur at a rate of once every 255x4 PWM cycles. Passing this through the divide by two circuit produces a clock signal at an PWM clock divided by 2040 rate. Similarly, a value of $01 in the PWMSCLA register when clock A is PWM clock divided by 4 will produce a clock at an PWM clock divided by 8 rate.

Writing to PWMSCLA or PWMSCLB causes the associated 8-bit down counter to be re-loaded. Otherwise, when changing rates the counter would have to count down to $01 before counting at the proper rate. Forcing the associated counter to re-load the scale register value every time PWMSCLA or PWMSCLB is written prevents this.

NOTE

Writing to the scale registers while channels are operating can cause irregularities in the PWM outputs.

13.4.1.3 Clock Select

Each PWM channel has the capability of selecting one of four clocks, clock A, clock SA, clock B or clock SB. The clock selection is done with the PCLKx control bits in the PWMCLK register and PCLKABx control bits in PWMCLKAB register. For backward compatibility consideration, the reset value of PWMCLK and PWMCLKAB configures following default clock selection.

For channels 0, 1, 4, and 5 the clock choices are clock A.

For channels 2, 3, 6, and 7 the clock choices are clock B.

NOTE

Changing clock control bits while channels are operating can cause irregularities in the PWM outputs.

13.4.2 PWM Channel Timers

The main part of the PWM module are the actual timers. Each of the timer channels has a counter, a period register and a duty register (each are 8-bit). The waveform output period is controlled by a match between the period register and the value in the counter. The duty is controlled by a match between the duty register and the counter value and causes the state of the output to change during the period. The starting polarity of the output is also selectable on a per channel basis. Shown below in Figure 13-16 is the block diagram for the PWM timer.

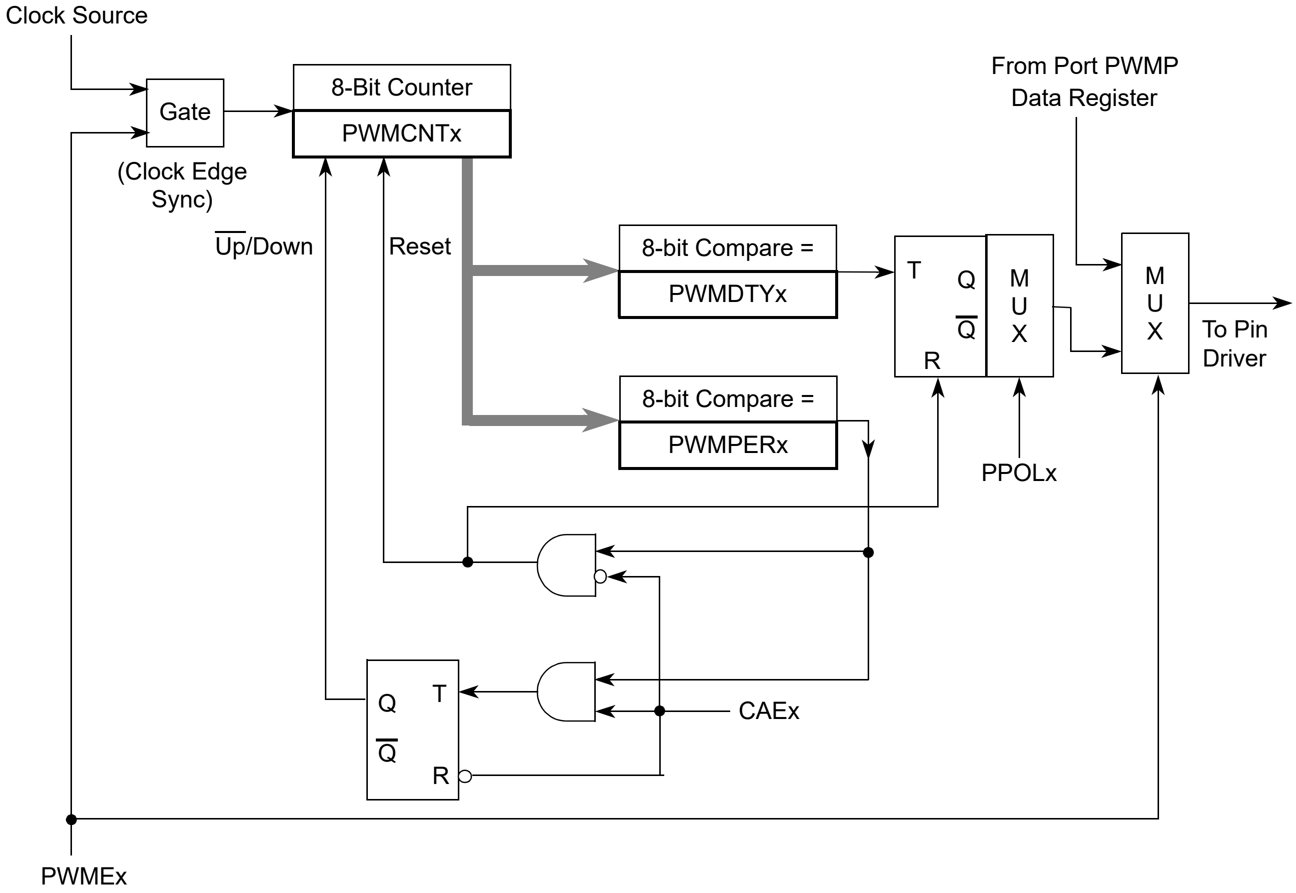


Figure 13-16. PWM Timer Channel Block Diagram

13.4.2.1 PWM Enable

Each PWM channel has an enable bit (PWMEx) to start its waveform output. When any of the PWMEx bits are set (PWMEx = 1), the associated PWM output signal is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWMEx and the clock source. An exception to this is when channels are concatenated. Refer to Section 13.4.2.7, “PWM 16-Bit Functions” for more detail.

NOTE

The first PWM cycle after enabling the channel can be irregular.

On the front end of the PWM timer, the clock is enabled to the PWM circuit by the PWMEx bit being high. There is an edge-synchronizing circuit to guarantee that the clock will only be enabled or disabled at an edge. When the channel is disabled (PWMEx = 0), the counter for the channel does not count.

13.4.2.2 PWM Polarity

Each channel has a polarity bit to allow starting a waveform cycle with a high or low signal. This is shown on the block diagram Figure 13-16 as a mux select of either the Q output or the $\overline{Q}$ output of the PWM output flip flop. When one of the bits in the PWMPOL register is set, the associated PWM channel output is high at the beginning of the waveform, then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

13.4.2.3 PWM Period and Duty

Dedicated period and duty registers exist for each channel and are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to $00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period and duty registers will go directly to the latches as well as the buffer.

A change in duty or period can be forced into effect “immediately” by writing the new value to the duty and/or period registers and then writing to the counter. This forces the counter to reset and the new duty and/or period values to be latched. In addition, since the counter is readable, it is possible to know where the count is with respect to the duty value and software can be used to make adjustments

NOTE

When forcing a new period or duty into effect immediately, an irregular PWM cycle can occur.

Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time.

13.4.2.4 PWM Timer Counters

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source (see Section 13.4.1, “PWM Clock Select” for the available clock sources and rates). The counter compares to two registers, a duty register and a period register as shown in Figure 13-16. When the PWM counter matches the duty register, the output flip-flop changes state, causing the PWM waveform to also change state. A match between the PWM counter and the period register behaves differently depending on what output mode is selected as shown in Figure 13-16 and described in Section 13.4.2.5, “Left Aligned Outputs” and Section 13.4.2.6, “Center Aligned Outputs”.

Each channel counter can be read at anytime without affecting the count or the operation of the PWM channel.

Any value written to the counter causes the counter to reset to $00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. When the channel is disabled (PWMEx = 0), the counter stops. When a channel becomes enabled (PWMEx = 1), the associated PWM counter continues from the count in the PWMCNTx register. This allows the waveform to continue where it left off when the channel is re-enabled. When the channel is disabled, writing "0" to the period register will cause the counter to reset on the next selected clock.

NOTE

If the user wants to start a new "clean" PWM waveform without any "history" from the old waveform, the user must write to channel counter (PWMCNTx) prior to enabling the PWM channel (PWMEx = 1).

Generally, writes to the counter are done prior to enabling a channel in order to start from a known state. However, writing a counter can also be done while the PWM channel is enabled (counting). The effect is similar to writing the counter when the channel is disabled, except that the new period is started immediately with the output set according to the polarity bit.

NOTE

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

The counter is cleared at the end of the effective period (see Section 13.4.2.5, "Left Aligned Outputs" and Section 13.4.2.6, "Center Aligned Outputs" for more details).

Table 13-12. PWM Timer Counter Conditions

Counter Clears ($00)	Counter Counts	Counter Stops
When PWMCNTx register written to any value	When PWM channel is enabled (PWMEx = 1). Counts from last value in PWMCNTx.	When PWM channel is disabled (PWMEx = 0)
Effective period ends		

13.4.2.5 Left Aligned Outputs

The PWM timer provides the choice of two types of outputs, left aligned or center aligned. They are selected with the CAEx bits in the PWMCAE register. If the CAEx bit is cleared (CAEx = 0), the corresponding PWM output will be left aligned.

In left aligned output mode, the 8-bit counter is configured as an up counter only. It compares to two registers, a duty register and a period register as shown in the block diagram in Figure 13-16. When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register resets the counter and the output flip-flop, as shown in Figure 13-16, as well as performing a load from the double buffer period and duty register to the associated registers, as described in Section 13.4.2.3, "PWM Period and Duty". The counter counts from 0 to the value in the period register – 1.

NOTE

Changing the PWM output mode from left aligned to center aligned output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.

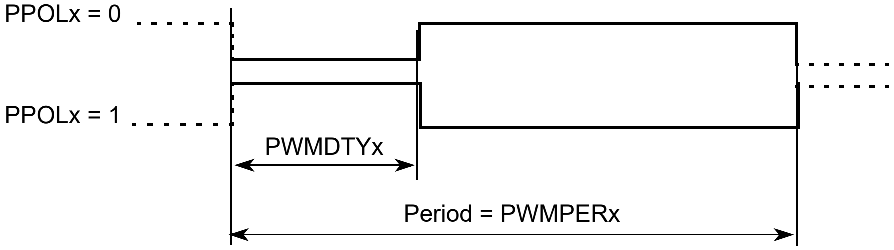


Figure 13-17. PWM Left Aligned Output Waveform

To calculate the output frequency in left aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by the value in the period register for that channel.

- PWMx Frequency = Clock (A, B, SA, or SB) / PWMPERx
- PWMx Duty Cycle (high time as a% of period):
  - Polarity = 0 (PPOLx = 0)
    Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
  - Polarity = 1 (PPOLx = 1)
    Duty Cycle = [PWMDTYx / PWMPERx] * 100%

As an example of a left aligned output, consider the following case:

Clock Source = PWM clock, where PWM clock = 10 MHz (100 ns period)
PPOLx = 0
PWMPERx = 4
PWMDTYx = 1
PWMx Frequency = 10 MHz/4 = 2.5 MHz
PWMx Period = 400 ns
PWMx Duty Cycle = 3/4 *100% = 75%

The output waveform generated is shown in Figure 13-18.

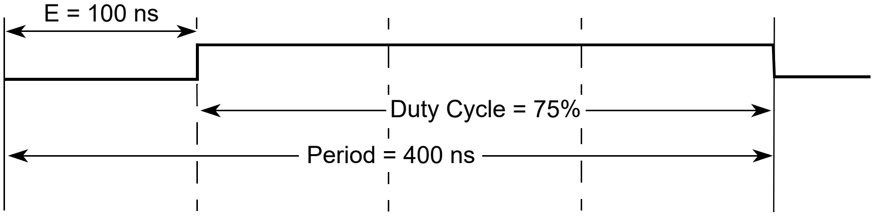


Figure 13-18. PWM Left Aligned Output Example Waveform

13.4.2.6 Center Aligned Outputs

For center aligned output mode selection, set the CAEx bit (CAEx = 1) in the PWMCAE register and the corresponding PWM output will be center aligned.

The 8-bit counter operates as an up/down counter in this mode and is set to up whenever the counter is equal to $00. The counter compares to two registers, a duty register and a period register as shown in the block diagram in Figure 13-16. When the PWM counter matches the duty register, the output flip-flop changes state, causing the PWM waveform to also change state. A match between the PWM counter and the period register changes the counter direction from an up-count to a down-count. When the PWM counter decrements and matches the duty register again, the output flip-flop changes state causing the PWM output to also change state. When the PWM counter decrements and reaches zero, the counter direction changes from a down-count back to an up-count and a load from the double buffer period and duty registers to the associated registers is performed, as described in Section 13.4.2.3, “PWM Period and Duty”. The counter counts from 0 up to the value in the period register and then back down to 0. Thus the effective period is PWMPERx*2.

NOTE

> Changing the PWM output mode from left aligned to center aligned output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.

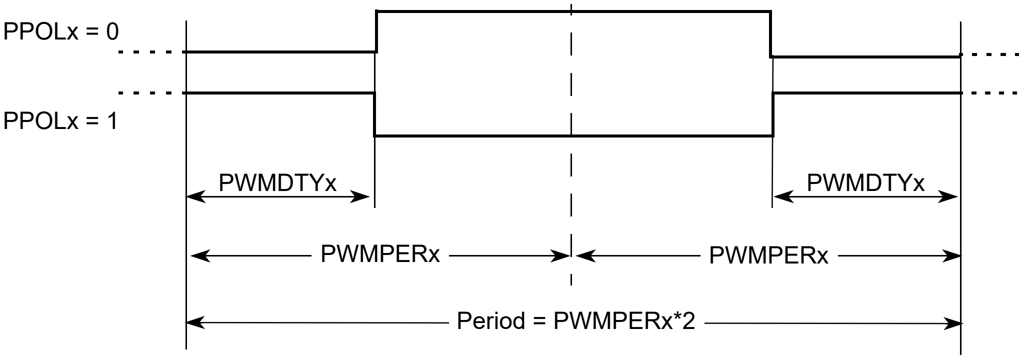


Figure 13-19. PWM Center Aligned Output Waveform

To calculate the output frequency in center aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by twice the value in the period register for that channel.

- PWMx Frequency = Clock (A, B, SA, or SB) / (2*PWMPERx)
- PWMx Duty Cycle (high time as a% of period):
  - Polarity = 0 (PPOLx = 0)

    Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 100%
  - Polarity = 1 (PPOLx = 1)

    Duty Cycle = [PWMDTYx / PWMPERx] * 100%

As an example of a center aligned output, consider the following case:

Clock Source = PWM clock, where PWM clock= 10 MHz (100 ns period)
PPOLx = 0
PWMPERx = 4
PWMDTYx = 1
PWMx Frequency = 10 MHz/8 = 1.25 MHz
PWMx Period = 800 ns
PWMx Duty Cycle = 3/4 *100% = 75%

Shown in Figure 13-20 is the output waveform generated.

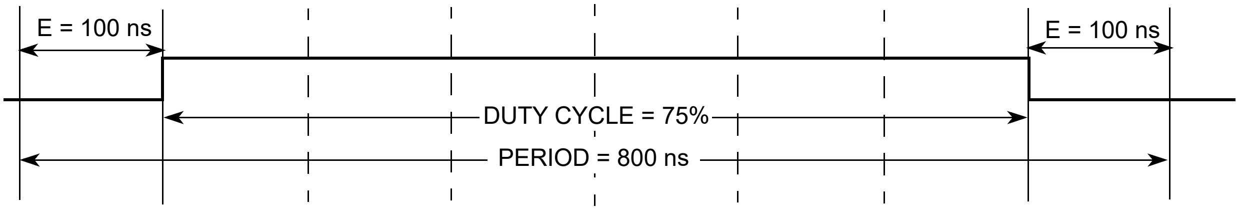


Figure 13-20. PWM Center Aligned Output Example Waveform

13.4.2.7 PWM 16-Bit Functions

The scalable PWM timer also has the option of generating up to 8-channels of 8-bits or 4-channels of 16-bits for greater PWM resolution. This 16-bit channel option is achieved through the concatenation of two 8-bit channels.

The PWMCTL register contains four control bits, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. Channels 6 and 7 are concatenated with the CON67 bit, channels 4 and 5 are concatenated with the CON45 bit, channels 2 and 3 are concatenated with the CON23 bit, and channels 0 and 1 are concatenated with the CON01 bit.

NOTE

Change these bits only when both corresponding channels are disabled.

When channels 6 and 7 are concatenated, channel 6 registers become the high order bytes of the double byte channel, as shown in Figure 13-21. Similarly, when channels 4 and 5 are concatenated, channel 4 registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

When using the 16-bit concatenated mode, the clock source is determined by the low order 8-bit channel clock select control bits. That is channel 7 when channels 6 and 7 are concatenated, channel 5 when channels 4 and 5 are concatenated, channel 3 when channels 2 and 3 are concatenated, and channel 1 when channels 0 and 1 are concatenated. The resulting PWM is output to the pins of the corresponding low order 8-bit channel as also shown in Figure 13-21. The polarity of the resulting PWM output is controlled by the PPOLx bit of the corresponding low order 8-bit channel as well.

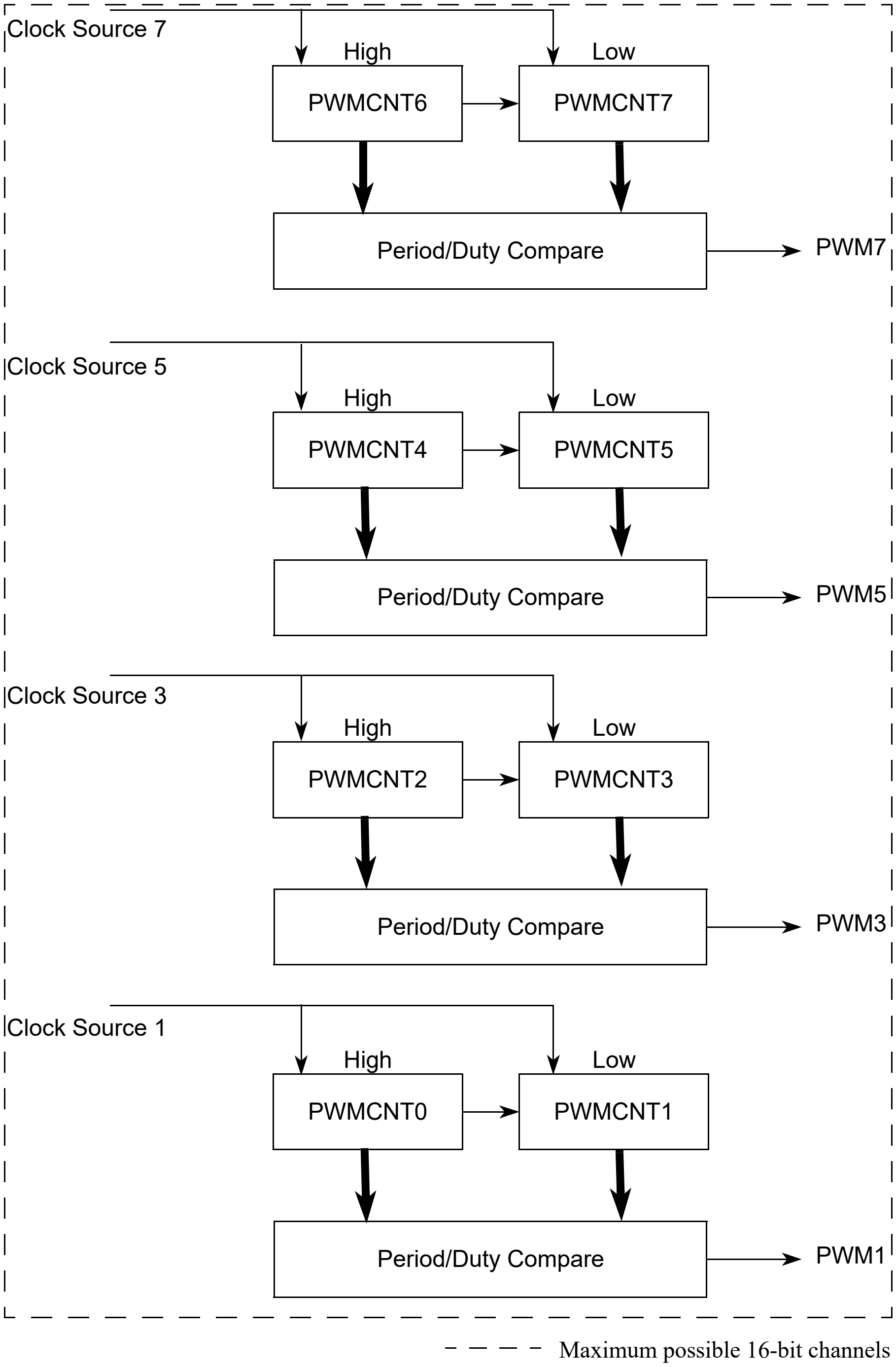


Figure 13-21. PWM 16-Bit Mode

Once concatenated mode is enabled (CONxx bits set in PWMCTL register), enabling/disabling the corresponding 16-bit PWM channel is controlled by the low order PWMEx bit. In this case, the high order bytes PWMEx bits have no effect and their corresponding PWM output is disabled.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

Either left aligned or center aligned output mode can be used in concatenated mode and is controlled by the low order CAEx bit. The high order CAEx bit has no effect.

Table 13-13 is used to summarize which channels are used to set the various control bits when in 16-bit mode.

Table 13-13. 16-bit Concatenation Mode Summary

Note: Bits related to available channels have functional significance.

CONxx	PWMEx	PPOLx	PCLKx	CAEx	PWMx Output
CON67	PWME7	PPOL7	PCLK7	CAE7	PWM7
CON45	PWME5	PPOL5	PCLK5	CAE5	PWM5
CON23	PWME3	PPOL3	PCLK3	CAE3	PWM3
CON01	PWME1	PPOL1	PCLK1	CAE1	PWM1

13.4.2.8 PWM Boundary Cases

Table 13-14 summarizes the boundary conditions for the PWM regardless of the output mode (left aligned or center aligned) and 8-bit (normal) or 16-bit (concatenation).

Table 13-14. PWM Boundary Cases

PWMDTYx	PWMPERx	PPOLx	PWMx Output
\$00 (indicates no duty)	>\$00	1	Always low
\$00 (indicates no duty)	>\$00	0	Always high
XX	\$00[1] (indicates no period)	1	Always high
XX	\$00[1] (indicates no period)	0	Always low
>= PWMPERx	XX	1	Always high
>= PWMPERx	XX	0	Always low

[1] Counter = \$00 and does not count.

13.5 Resets

The reset state of each individual bit is listed within the Section 13.3.2, “Register Descriptions” which details the registers and their bit-fields. All special functions or modes which are initialized during or just following reset are described within this section.

- The 8-bit up/down counter is configured as an up counter out of reset.
- All the channels are disabled and all the counters do not count.

- For channels 0, 1, 4, and 5 the clock choices are clock A.
- For channels 2, 3, 6, and 7 the clock choices are clock B.

13.6 Interrupts

The PWM module has no interrupt.

Chapter 14
Serial Communication Interface (S12SCIV6)

Table 14-1. Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
06.06	03/11/2013			fix typo of BDL reset value,Figure 14-4 fix typo of Table 14-2,Table 14-16,reword 14.4.4/14-454
06.07	09/03/2013			update Figure 14-14./14-451 Figure 14-16./14-455 Figure 14-20./14-460 update 14.4.4/14-454,more detail for two baud add note for Table 14-16./14-454 update Figure 14-2./14-439,Figure 14-12./14-450
06.08	10/14/2013			update Figure 14-4./14-440 14.3.2.9/14-450

14.1 Introduction

This block guide provides an overview of the serial communication interface (SCI) module.

The SCI allows asynchronous serial communications with peripheral devices and other CPUs.

14.1.1 Glossary

IR: InfraRed

IrDA: Infrared Design Associate

IRQ: Interrupt Request

LIN: Local Interconnect Network

LSB: Least Significant Bit

MSB: Most Significant Bit

NRZ: Non-Return-to-Zero

RZI: Return-to-Zero-Inverted

RXD: Receive Pin

SCI : Serial Communication Interface

TXD: Transmit Pin

14.1.2 Features

The SCI includes these distinctive features:

- Full-duplex or single-wire operation
- Standard mark/space non-return-to-zero (NRZ) format
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse widths
- 16-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable polarity for transmitter and receiver
- Programmable transmitter output parity
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Interrupt-driven operation with eight flags:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Idle receiver input
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
  - Receive wakeup on active edge
  - Transmit collision detect supporting LIN
  - Break Detect supporting LIN
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

14.1.3 Modes of Operation

The SCI functions the same in normal, special, and emulation modes. It has two low power modes, wait and stop modes.

- Run mode
- Wait mode
- Stop mode

14.1.4 Block Diagram

Figure 14-1 is a high level block diagram of the SCI module, showing the interaction of various function blocks.

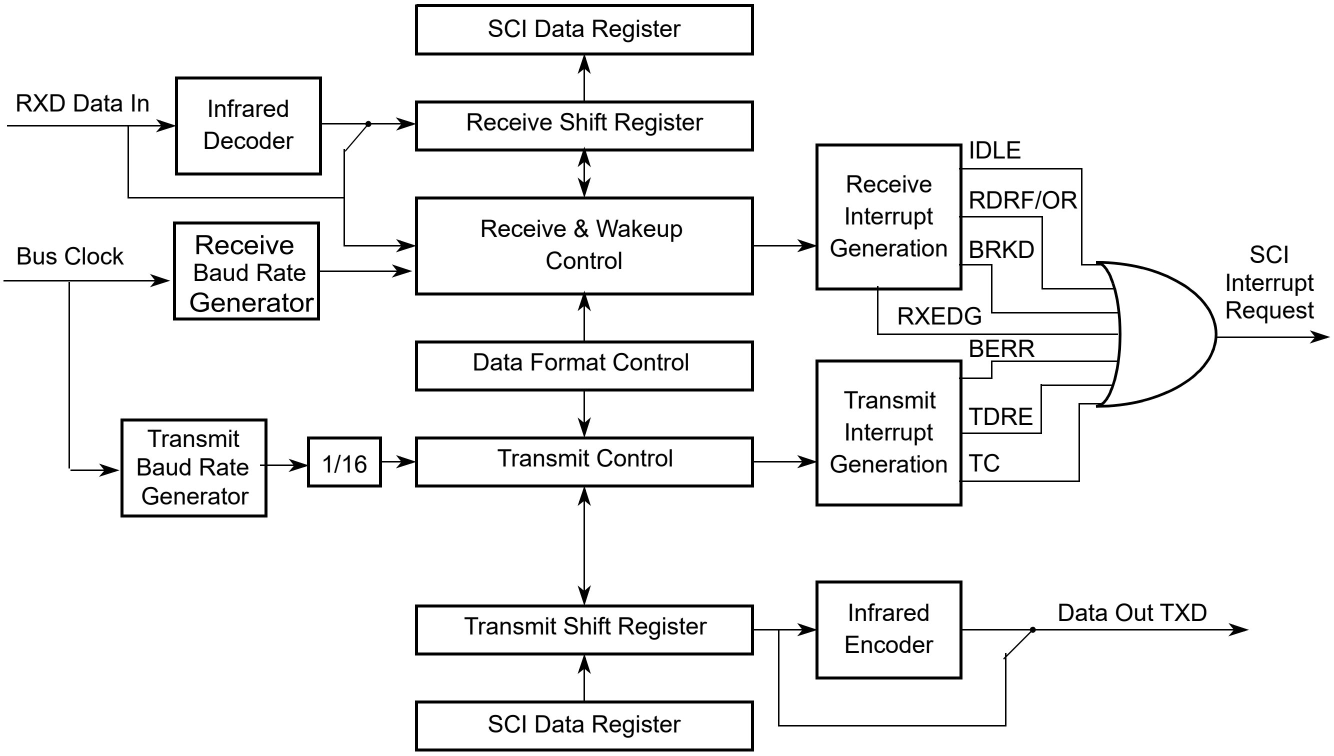


Figure 14-1. SCI Block Diagram

14.2 External Signal Description

The SCI module has a total of two external pins.

14.2.1 TXD — Transmit Pin

The TXD pin transmits SCI (standard or infrared) data. It will idle high in either mode and is high impedance anytime the transmitter is disabled.

14.2.2 RXD — Receive Pin

The RXD pin receives SCI (standard or infrared) data. An idle line is detected as a line high. This input is ignored when the receiver is disabled and should be terminated to a known voltage.

14.3 Memory Map and Register Definition

This section provides a detailed description of all the SCI registers.

14.3.1 Module Memory Map and Register Definition

The memory map for the SCI module is given below in Figure 14-2. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the SCI module and the address offset for each register.

14.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Writes to a reserved register locations do not have any effect and reads of these locations return a zero. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 SCIBDH[1]	R	SBR15	SBR14	SBR13	SBR12	SBR11	SBR10	SBR9	SBR8
	W								
0x0001 SCIBDL[1]	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
	W								
0x0002 SCICR1[1]	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
	W								
0x0000 SCIASR1[2]	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
	W		▨	▨	▨	▨			
0x0001 SCIACR1[2]	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
	W		▨	▨	▨	▨	▨		
0x0002 SCIACR2[2]	R	IREN	TNP1	TNP0	0	0	BERRM1	BERRM0	BKDFE
	W				▨	▨			
0x0003 SCICR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	W								
0x0004 SCISR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0005 SCISR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
	W		▨	▨					▨
0x0007 SCIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
	W	T7	T6	T5	T4	T3	T2	T1	T0

1.These registers are accessible if the AMAP bit in the SCISR2 register is set to zero.

2,These registers are accessible if the AMAP bit in the SCISR2 register is set to one.

▨ = Unimplemented or Reserved

Figure 14-2. SCI Register Summary

14.3.2.1 SCI Baud Rate Registers (SCIBDH, SCIBDL)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	SBR15	SBR14	SBR13	SBR12	SBR11	SBR10	SBR9	SBR8
Reset	0	0	0	0	0	0	0	0

Figure 14-3. SCI Baud Rate Register (SCIBDH)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
Reset	0	1	0	0	0	0	0	0

Figure 14-4. SCI Baud Rate Register (SCIBDL)

Read: Anytime, if AMAP = 0.

Write: Anytime, if AMAP = 0.

NOTE

Those two registers are only visible in the memory map if AMAP = 0 (reset condition).

The SCI baud rate register is used by to determine the baud rate of the SCI, and to control the infrared modulation/demodulation submodule.

Table 14-2. SCIBDH and SCIBDL Field Descriptions

Field	Description
SBR[15:0]	**SCI Baud Rate Bits** — The baud rate for the SCI is determined by the bits in this register. The baud rate is calculated two different ways depending on the state of the IREN bit. The formulas for calculating the baud rate are: When IREN = 0 then, SCI baud rate = SCI bus clock / (SBR[15:0]) When IREN = 1 then, SCI baud rate = SCI bus clock / (2 x SBR[15:1]) **Note:** The baud rate generator is disabled after reset and not started until the TE bit or the RE bit is set for the first time. The baud rate generator is disabled when (SBR[15:4] = 0 and IREN = 0) or (SBR[15:5] = 0 and IREN = 1). **Note:** . User should write SCIBD by word access. The updated SCIBD may take effect until next RT clock start, write SCIBDH or SCIBDL separately may cause baud generator load wrong data at that time,if second write later then RT clock.

14.3.2.2 SCI Control Register 1 (SCICR1)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

Figure 14-5. SCI Control Register 1 (SCICR1)

Read: Anytime, if AMAP = 0.

Write: Anytime, if AMAP = 0.

NOTE

This register is only visible in the memory map if AMAP = 0 (reset condition).

Table 14-3. SCICR1 Field Descriptions

Field	Description
7 LOOPS	**Loop Select Bit** — LOOPS enables loop operation. In loop operation, the RXD pin is disconnected from the SCI and the transmitter output is internally connected to the receiver input. Both the transmitter and the receiver must be enabled to use the loop function. 0 Normal operation enabled 1 Loop operation enabled The receiver input is determined by the RSRC bit.
6 SCISWAI	**SCI Stop in Wait Mode Bit** — SCISWAI disables the SCI in wait mode. 0 SCI enabled in wait mode 1 SCI disabled in wait mode
5 RSRC	**Receiver Source Bit** — When LOOPS = 1, the RSRC bit determines the source for the receiver shift register input. See Table 14-4. 0 Receiver input internally connected to transmitter output 1 Receiver input connected externally to transmitter
4 M	**Data Format Mode Bit** — MODE determines whether data characters are eight or nine bits long. 0 One start bit, eight data bits, one stop bit 1 One start bit, nine data bits, one stop bit
3 WAKE	**Wakeup Condition Bit** — WAKE determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received data character or an idle condition on the RXD pin. 0 Idle line wakeup 1 Address mark wakeup
2 ILT	**Idle Line Type Bit** — ILT determines when the receiver starts counting logic 1s as idle character bits. The counting begins either after the 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 may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. 0 Idle character bit count begins after start bit 1 Idle character bit count begins after stop bit

Table 14-3. SCICR1 Field Descriptions (continued)

Field	Description
1 PE	**Parity Enable Bit** — PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position. 0 Parity function disabled 1 Parity function enabled
0 PT	**Parity Type Bit** — PT determines whether the SCI generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity 1 Odd parity

Table 14-4. Loop Functions

LOOPS	RSRC	Function
0	x	Normal operation
1	0	Loop mode with transmitter output internally connected to receiver input
1	1	Single-wire mode with TXD pin connected to receiver input

14.3.2.3 SCI Alternative Status Register 1 (SCIASR1)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
W		▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-6. SCI Alternative Status Register 1 (SCIASR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 14-5. SCIASR1 Field Descriptions

Field	Description
7 RXEDGIF	**Receive Input Active Edge Interrupt Flag** — RXEDGIF is asserted, if an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD input occurs. RXEDGIF bit is cleared by writing a “1” to it. 0 No active receive on the receive input has occurred 1 An active edge on the receive input has occurred
2 BERRV	**Bit Error Value** — BERRV reflects the state of the RXD input when the bit error detect circuitry is enabled and a mismatch to the expected value happened. The value is only meaningful, if BERRIF = 1. 0 A low input was sampled, when a high was expected 1 A high input reassembled, when a low was expected
1 BERRIF	**Bit Error Interrupt Flag** — BERRIF is asserted, when the bit error detect circuitry is enabled and if the value sampled at the RXD input does not match the transmitted value. If the BERRIE interrupt enable bit is set an interrupt will be generated. The BERRIF bit is cleared by writing a “1” to it. 0 No mismatch detected 1 A mismatch has occurred
0 BKDIF	**Break Detect Interrupt Flag** — BKDIF is asserted, if the break detect circuitry is enabled and a break signal is received. If the BKDIE interrupt enable bit is set an interrupt will be generated. The BKDIF bit is cleared by writing a “1” to it. 0 No break signal was received 1 A break signal was received

14.3.2.4 SCI Alternative Control Register 1 (SCIACR1)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
W		▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-7. SCI Alternative Control Register 1 (SCIACR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 14-6. SCIACR1 Field Descriptions

Field	Description
7 RXEDGIE	**Receive Input Active Edge Interrupt Enable** — RXEDGIE enables the receive input active edge interrupt flag, RXEDGIF, to generate interrupt requests. 0 RXEDGIF interrupt requests disabled 1 RXEDGIF interrupt requests enabled
1 BERRIE	**Bit Error Interrupt Enable** — BERRIE enables the bit error interrupt flag, BERRIF, to generate interrupt requests. 0 BERRIF interrupt requests disabled 1 BERRIF interrupt requests enabled
0 BKDIE	**Break Detect Interrupt Enable** — BKDIE enables the break detect interrupt flag, BKDIF, to generate interrupt requests. 0 BKDIF interrupt requests disabled 1 BKDIF interrupt requests enabled

14.3.2.5 SCI Alternative Control Register 2 (SCIACR2)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	IREN	TNP1	TNP0	0	0	BERRM1	BERRM0	BKDFE
W				▨	▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-8. SCI Alternative Control Register 2 (SCIACR2)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 14-7. SCIACR2 Field Descriptions

Field	Description
7 IREN	**Infrared Enable Bit** — This bit enables/disables the infrared modulation/demodulation submodule. 0 IR disabled 1 IR enabled
6:5 TNP[1:0]	**Transmitter Narrow Pulse Bits** — These bits enable whether the SCI transmits a 1/16, 3/16, 1/32 or 1/4 narrow pulse. See Table 14-8.
2:1 BERRM[1:0]	**Bit Error Mode** — Those two bits determines the functionality of the bit error detect feature. See Table 14-9.
0 BKDFE	**Break Detect Feature Enable** — BKDFE enables the break detect circuitry. 0 Break detect circuit disabled 1 Break detect circuit enabled

Table 14-8. IRSCI Transmit Pulse Width

TNP[1:0]	Narrow Pulse Width
11	1/4
10	1/32
01	1/16
00	3/16

Table 14-9. Bit Error Mode Coding

BERRM1	BERRM0	Function
0	0	Bit error detect circuit is disabled
0	1	Receive input sampling occurs during the 9th time tick of a transmitted bit (refer to Figure 14-19)
1	0	Receive input sampling occurs during the 13th time tick of a transmitted bit (refer to Figure 14-19)
1	1	Reserved

14.3.2.6 SCI Control Register 2 (SCICR2)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

Figure 14-9. SCI Control Register 2 (SCICR2)

Read: Anytime

Write: Anytime

Table 14-10. SCICR2 Field Descriptions

Field	Description
7 TIE	**Transmitter Interrupt Enable Bit** — TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests. 0 TDRE interrupt requests disabled 1 TDRE interrupt requests enabled
6 TCIE	**Transmission Complete Interrupt Enable Bit** — TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 TC interrupt requests disabled 1 TC interrupt requests enabled
5 RIE	**Receiver Full Interrupt Enable Bit** — RIE enables the receive data register full flag, RDRF, or the overrun flag, OR, to generate interrupt requests. 0 RDRF and OR interrupt requests disabled 1 RDRF and OR interrupt requests enabled
4 ILIE	**Idle Line Interrupt Enable Bit** — ILIE enables the idle line flag, IDLE, to generate interrupt requests. 0 IDLE interrupt requests disabled 1 IDLE interrupt requests enabled
3 TE	**Transmitter Enable Bit** — TE enables the SCI transmitter and configures the TXD pin as being controlled by the SCI. The TE bit can be used to queue an idle preamble. 0 Transmitter disabled 1 Transmitter enabled
2 RE	**Receiver Enable Bit** — RE enables the SCI receiver. 0 Receiver disabled 1 Receiver enabled
1 RWU	**Receiver Wakeup Bit** — Standby state 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	**Send Break Bit** — Toggling SBK sends one break character (10 or 11 logic 0s, respectively 13 or 14 logics 0s if BRK13 is set). Toggling implies clearing the SBK bit before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10 or 11 bits, respectively 13 or 14 bits). 0 No break characters 1 Transmit break characters

14.3.2.7 SCI Status Register 1 (SCISR1)

The SCISR1 and SCISR2 registers provides inputs to the MCU for generation of SCI interrupts. Also, these registers can be polled by the MCU to check the status of these bits. The flag-clearing procedures require that the status register be read followed by a read or write to the SCI data register.It is permissible to execute other instructions between the two steps as long as it does not compromise the handling of I/O, but the order of operations is important for flag clearing.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	1	1	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-10. SCI Status Register 1 (SCISR1)

Read: Anytime

Write: Has no meaning or effect

Table 14-11. SCISR1 Field Descriptions

Field	Description
7 TDRE	**Transmit Data Register Empty Flag** — TDRE is set when the transmit shift register receives a byte from the SCI data register. When TDRE is 1, the transmit data register (SCIDRH/L) is empty and can receive a new value to transmit.Clear TDRE by reading SCI status register 1 (SCISR1), with TDRE set and then writing to SCI data register low (SCIDRL). 0 No byte transferred to transmit shift register 1 Byte transferred to transmit shift register; transmit data register empty
6 TC	**Transmit Complete Flag** — TC is set low when there is a transmission in progress or when a preamble or break character is loaded. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted.When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL). TC is cleared automatically when data, preamble, or break is queued and ready to be sent. TC is cleared in the event of a simultaneous set and clear of the TC flag (transmission not complete). 0 Transmission in progress 1 No transmission in progress
5 RDRF	**Receive Data Register Full Flag** — RDRF is set when the data in the receive shift register transfers to the SCI data register. Clear RDRF by reading SCI status register 1 (SCISR1) with RDRF set and then reading SCI data register low (SCIDRL). 0 Data not available in SCI data register 1 Received data available in SCI data register
4 IDLE	**Idle Line Flag** — IDLE is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M =1) appear on the receiver input. Once the IDLE flag is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag.Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL). 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared 1 Receiver input has become idle **Note:** When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag.

Table 14-11. SCISR1 Field Descriptions (continued)

Field	Description
3 OR	**Overrun Flag** — OR is set when software fails to read the SCI data register before the receive shift register receives the next frame. The OR bit is set immediately after the stop bit has been completely received for the second frame. The data in the shift register is lost, but the data already in the SCI data registers is not affected. Clear OR by reading SCI status register 1 (SCISR1) with OR set and then reading SCI data register low (SCIDRL). 0 No overrun 1 Overrun **Note:** OR flag may read back as set when RDRF flag is clear. This may happen if the following sequence of events occurs: 1. After the first frame is received, read status register SCISR1 (returns RDRF set and OR flag clear); 2. Receive second frame without reading the first frame in the data register (the second frame is not received and OR flag is set); 3. Read data register SCIDRL (returns first frame and clears RDRF flag in the status register); 4. Read status register SCISR1 (returns RDRF clear and OR set). Event 3 may be at exactly the same time as event 2 or any time after. When this happens, a dummy SCIDRL read following event 4 will be required to clear the OR flag if further frames are to be received.
2 NF	**Noise Flag** — NF is set when the SCI detects noise on the receiver input. NF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear NF by reading SCI status register 1(SCISR1), and then reading SCI data register low (SCIDRL). 0 No noise 1 Noise
1 FE	**Framing Error Flag** — FE is set when a logic 0 is accepted as the stop bit. FE bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading SCI status register 1 (SCISR1) with FE set and then reading the SCI data register low (SCIDRL). 0 No framing error 1 Framing error
0 PF	**Parity Error Flag** — PF is set when the parity enable bit (PE) is set and the parity of the received data does not match the parity type bit (PT). PF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear PF by reading SCI status register 1 (SCISR1), and then reading SCI data register low (SCIDRL). 0 No parity error 1 Parity error

14.3.2.8 SCI Status Register 2 (SCISR2)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
W		▨	▨					▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-11. SCI Status Register 2 (SCISR2)

Read: Anytime

Write: Anytime

Table 14-12. SCISR2 Field Descriptions

Field	Description
7 AMAP	**Alternative Map** — This bit controls which registers sharing the same address space are accessible. In the reset condition the SCI behaves as previous versions. Setting AMAP=1 allows the access to another set of control and status registers and hides the baud rate and SCI control Register 1. 0 The registers labelled SCIBDH (0x0000),SCIBDL (0x0001), SCICR1 (0x0002) are accessible 1 The registers labelled SCIASR1 (0x0000),SCIACR1 (0x0001), SCIACR2 (0x00002) are accessible
4 TXPOL	**Transmit Polarity** — This bit control the polarity of the transmitted data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity. 0 Normal polarity 1 Inverted polarity
3 RXPOL	**Receive Polarity** — This bit control the polarity of the received data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity. 0 Normal polarity 1 Inverted polarity
2 BRK13	**Break Transmit Character Length** — This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bits long. The detection of a framing error is not affected by this bit. 0 Break character is 10 or 11 bit long 1 Break character is 13 or 14 bit long
1 TXDIR	**Transmitter Pin Data Direction in Single-Wire Mode** — This bit determines whether the TXD pin is going to be used as an input or output, in the single-wire mode of operation. This bit is only relevant in the single-wire mode of operation. 0 TXD pin to be used as an input in single-wire mode 1 TXD pin to be used as an output in single-wire mode
0 RAF	**Receiver Active Flag** — RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 No reception in progress 1 Reception in progress

14.3.2.9 SCI Data Registers (SCIDRH, SCIDRL)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 14-12. SCI Data Registers (SCIDRH)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

Figure 14-13. SCI Data Registers (SCIDRL)

Read: Anytime; reading accesses SCI receive data register

Write: Anytime; writing accesses SCI transmit data register; writing to R8 has no effect

NOTE

The reserved bit SCIDRH[2:0] are designed for factory test purposes only, and are not intended for general user access. Writing to these bit is possible when in special mode and can alter the modules functionality.

Table 14-13. SCIDRH and SCIDRL Field Descriptions

Field	Description
SCIDRH 7 R8	**Received Bit 8** — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).
SCIDRH 6 T8	**Transmit Bit 8** — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).
SCIDRL 7:0 R[7:0] T[7:0]	**R7:R0** — Received bits seven through zero for 9-bit or 8-bit data formats **T7:T0** — Transmit bits seven through zero for 9-bit or 8-bit formats

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten.The same value is transmitted until T8 is rewritten

In 8-bit data format, only SCI data register low (SCIDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCIDRH), then SCIDRL.

14.4 Functional Description

This section provides a complete functional description of the SCI block, detailing the operation of the design from the end user perspective in a number of subsections.

Figure 14-14 shows the structure of the SCI module. The SCI allows full duplex, asynchronous, serial communication between the CPU and remote devices, including other CPUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

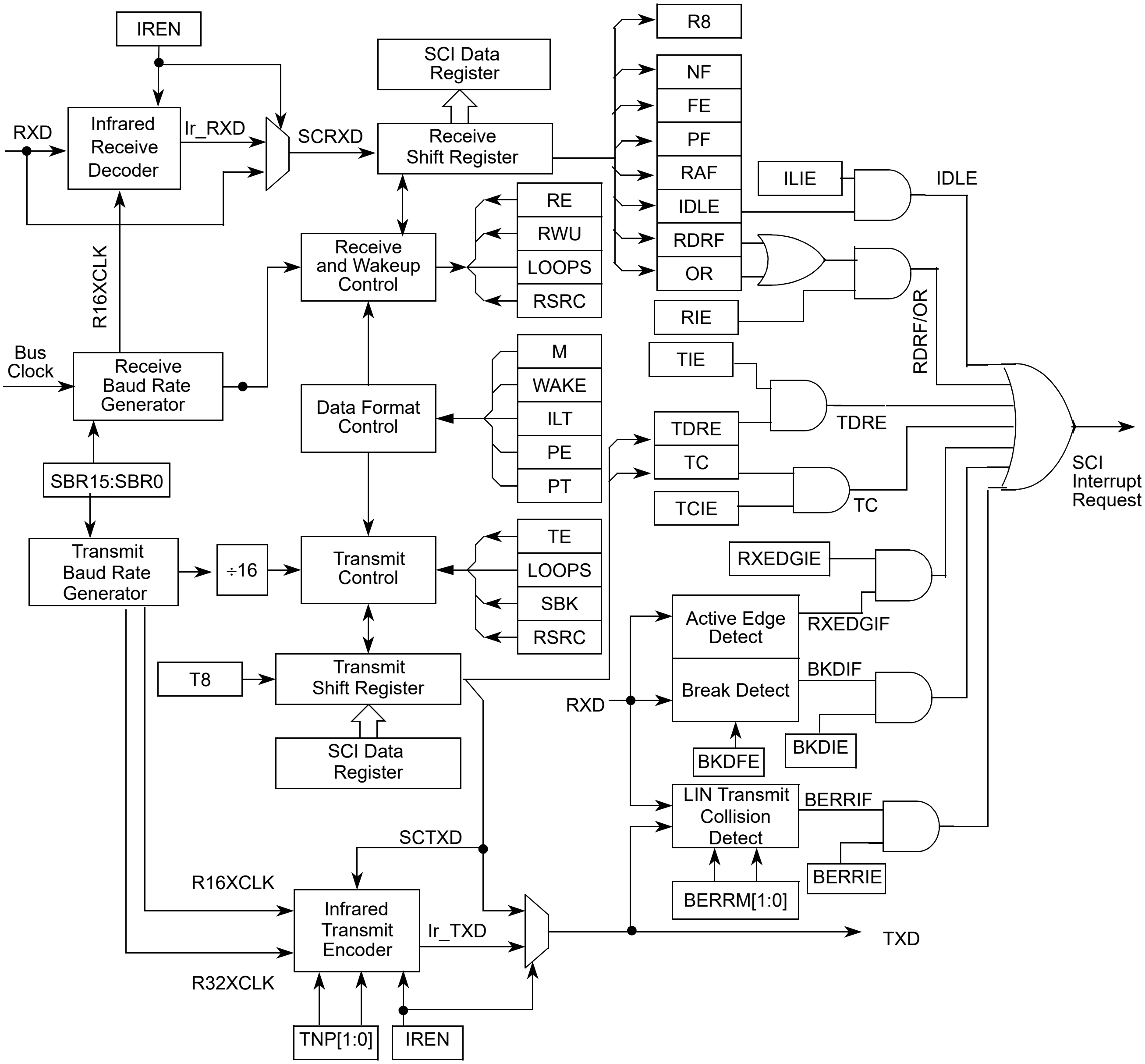


Figure 14-14. Detailed SCI Block Diagram

14.4.1 Infrared Interface Submodule

This module 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 SCI. The IrDA physical layer specification defines a half-duplex infrared communication link for exchange data. The full standard includes data rates up to 16 Mbits/s. This design covers only data rates between 2.4 Kbits/s and 115.2 Kbits/s.

The infrared submodule consists of two major blocks: the transmit encoder and the receive decoder. The SCI transmits serial bits of data which are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses should be detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder (external from the MCU). The narrow pulses are then stretched by the infrared submodule to get back to a serial bit stream to be received by the SCI.The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active low pulses.

The infrared submodule receives its clock sources from the SCI. One of these two clocks are selected in the infrared submodule in order to generate either 3/16, 1/16, 1/32 or 1/4 narrow pulses during transmission. The infrared block receives two clock sources from the SCI, R16XCLK and R32XCLK, which are configured to generate the narrow pulse width during transmission. The R16XCLK and R32XCLK are internal clocks with frequencies 16 and 32 times the baud rate respectively. Both R16XCLK and R32XCLK clocks are used for transmitting data. The receive decoder uses only the R16XCLK clock.

14.4.1.1 Infrared Transmit Encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD pin. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, 3/16 or 1/4 of a bit time. A narrow high pulse is transmitted for a zero bit when TXPOL is cleared, while a narrow low pulse is transmitted for a zero bit when TXPOL is set.

14.4.1.2 Infrared Receive Decoder

The infrared receive block converts data from the RXD pin to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow high pulse is expected for a zero bit when RXPOL is cleared, while a narrow low pulse is expected for a zero bit when RXPOL is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

14.4.2 LIN Support

This module provides some basic support for the LIN protocol. At first this is a break detect circuitry making it easier for the LIN software to distinguish a break character from an incoming data stream. As a further addition is supports a collision detection at the bit level as well as cancelling pending transmissions.

14.4.3 Data Format

The SCI uses the standard NRZ mark/space data format. When Infrared is enabled, the SCI uses RZI data format where zeroes are represented by light pulses and ones remain low. See Figure 14-15 below.

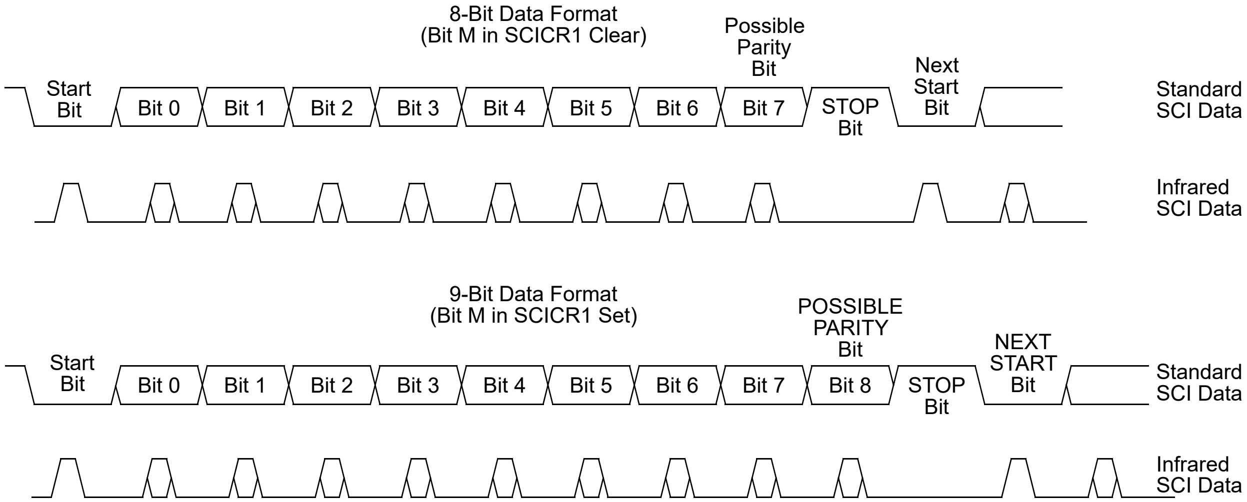


Figure 14-15. SCI Data Formats

Each data character is contained in a frame that includes a start bit, eight or nine data bits, and a stop bit. Clearing the M bit in SCI control register 1 configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Setting the M bit configures the SCI for nine-bit data characters. A frame with nine data bits has a total of 11 bits.

Table 14-14. Example of 8-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	8	0	0	1
1	7	0	1	1
1	7	1[1]	0	1

[1] The address bit identifies the frame as an address character. See Section 14.4.6.6, “Receiver Wakeup”.

When the SCI is configured for 9-bit data characters, the ninth data bit is the T8 bit in SCI data register high (SCIDRH). It remains unchanged after transmission and can be used repeatedly without rewriting it. A frame with nine data bits has a total of 11 bits.

Table 14-15. Example of 9-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	9	0	0	1
1	8	0	1	1
1	8	1[1]	0	1

[1] The address bit identifies the frame as an address character. See Section 14.4.6.6, “Receiver Wakeup”.

14.4.4 Baud Rate Generation

A 16-bit modulus counter in the two baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 65535 written to the SBR15:SBR0 bits determines the baud rate. The value from 0 to 4095 written to the SBR15:SBR4 bits determines the baud rate clock with SBR3:SBR0 for fine adjust. The SBR bits are in the SCI baud rate registers (SCIBDH and SCIBDL) for both transmit and receive baud generator. The baud rate clock is synchronized with the bus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to one source of error:

- Integer division of the bus clock may not give the exact target frequency.

Table 14-16 lists some examples of achieving target baud rates with a bus clock frequency of 25 MHz.

When IREN = 0 then,

SCI baud rate = SCI bus clock / (SCIBR[15:0])

Table 14-16. Baud Rates (Example: Bus Clock = 25 MHz)

Bits SBR[15:0]	Receiver[1] Clock (Hz)	Transmitter[2] Clock (Hz)	Target Baud Rate	Error (%)
109	3669724.8	229,357.8	230,400	.452
217	1843318.0	115,207.4	115,200	.006
651	614439.3	38,402.5	38,400	.006
1302	307219.7	19,201.2	19,200	.006
2604	153,609.8	9600.6	9,600	.006
5208	76,804.9	4800.3	4,800	.006
10417	38,398.8	2399.9	2,400	.003
20833	19,200.3	1200.02	1,200	.00
41667	9599.9	600.0	600	.00
65535	6103.6	381.5		

[1] 16x faster then baud rate

[2] divide 1/16 form transmit baud generator

14.4.5 Transmitter

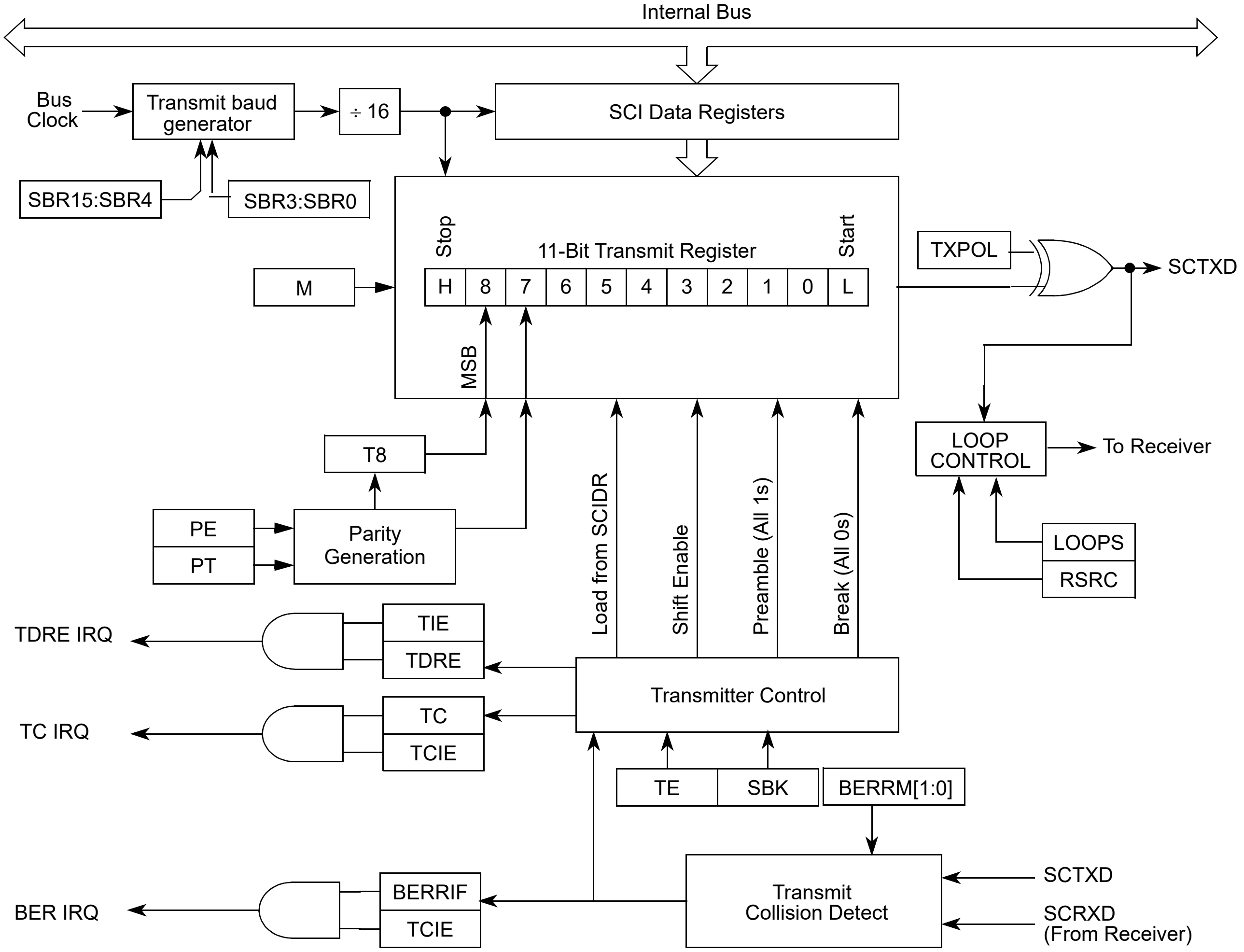


Figure 14-16. Transmitter Block Diagram

14.4.5.1 Transmitter Character Length

The SCI transmitter can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When transmitting 9-bit data, bit T8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

14.4.5.2 Character Transmission

To transmit data, the MCU writes the data bits to the SCI data registers (SCIDRH/SCIDRL), which in turn are transferred to the transmitter shift register. The transmit shift register then shifts a frame out through the TXD pin, after it has prefaced them with a start bit and appended them with a stop bit. The SCI data registers (SCIDRH and SCIDRL) are the write-only buffers between the internal data bus and the transmit shift register.

The SCI also sets a flag, the transmit data register empty flag (TDRE), every time it transfers data from the buffer (SCIDRH/L) to the transmitter shift register.The transmit driver routine may respond to this flag by writing another byte to the Transmitter buffer (SCIDRH/SCIDRL), while the shift register is still shifting out the first byte.

To initiate an SCI transmission:

1. Configure the SCI:
   a) Select a baud rate. Write this value to the SCI baud registers (SCIBDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the SCIBDH has no effect without also writing to SCIBDL.
   b) Write to SCICR1 to configure word length, parity, and other configuration bits (LOOPS,RSRC,M,WAKE,ILT,PE,PT).
   c) Enable the transmitter, interrupts, receive, and wake up as required, by writing to the SCICR2 register bits (TIE,TCIE,RIE,ILIE,TE,RE,RWU,SBK). A preamble or idle character will now be shifted out of the transmitter shift register.
2. Transmit Procedure for each byte:
   a) Poll the TDRE flag by reading the SCISR1 or responding to the TDRE interrupt. Keep in mind that the TDRE bit resets to one.
   b) If the TDRE flag is set, write the data to be transmitted to SCIDRH/L, where the ninth bit is written to the T8 bit in SCIDRH if the SCI is in 9-bit data format. A new transmission will not result until the TDRE flag has been cleared.
3. Repeat step 2 for each subsequent transmission.

NOTE

The TDRE flag is set when the shift register is loaded with the next data to be transmitted from SCIDRH/L, which happens, generally speaking, a little over half-way through the stop bit of the previous frame. Specifically, this transfer occurs 9/16ths of a bit time AFTER the start of the stop bit of the previous frame.

Writing the TE bit from 0 to a 1 automatically loads the transmit shift register with a preamble of 10 logic 1s (if M = 0) or 11 logic 1s (if M = 1). After the preamble shifts out, control logic transfers the data from the SCI data register into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit position of the transmit shift register. A logic 1 stop bit goes into the most significant bit position.

Hardware supports odd or even parity. When parity is enabled, the most significant bit (MSB) of the data character is the parity bit.

The transmit data register empty flag, TDRE, in SCI status register 1 (SCISR1) becomes set when the SCI data register transfers a byte to the transmit shift register. The TDRE flag indicates that the SCI data register can accept new data from the internal data bus. If the transmit interrupt enable bit, TIE, in SCI control register 2 (SCICR2) is also set, the TDRE flag generates a transmitter interrupt request.

When the transmit shift register is not transmitting a frame, the TXD pin goes to the idle condition, logic 1. If at any time software clears the TE bit in SCI control register 2 (SCICR2), the transmitter enable signal goes low and the transmit signal goes idle.

If software clears TE while a transmission is in progress (TC = 0), the frame in the transmit shift register continues to shift out. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last byte of the first message to SCIDRH/L.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the TE bit.
4. Write the first byte of the second message to SCIDRH/L.

14.4.5.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCI control register 2 (SCICR2) loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCI control register 1 (SCICR1). As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next frame.

The SCI recognizes a break character when there are 10 or 11(M = 0 or M = 1) consecutive zero received. Depending if the break detect feature is enabled or not receiving a break character has these effects on SCI registers.

If the break detect feature is disabled (BKDFE = 0):

- Sets the framing error flag, FE
- Sets the receive data register full flag, RDRF
- Clears the SCI data registers (SCIDRH/L)
- May set the overrun flag, OR, noise flag, NF, parity error flag, PE, or the receiver active flag, RAF (see 3.4.4 and 3.4.5 SCI Status Register 1 and 2)

If the break detect feature is enabled (BKDFE = 1) there are two scenarios[1]

The break is detected right from a start bit or is detected during a byte reception.

- Sets the break detect interrupt flag, BKDIF
- Does not change the data register full flag, RDRF or overrun flag OR
- Does not change the framing error flag FE, parity error flag PE.
- Does not clear the SCI data registers (SCIDRH/L)
- May set noise flag NF, or receiver active flag RAF.

---

[1] A Break character in this context are either 10 or 11 consecutive zero received bits

Figure 14-17 shows two cases of break detect. In trace RXD_1 the break symbol starts with the start bit, while in RXD_2 the break starts in the middle of a transmission. If BRKDFE = 1, in RXD_1 case there will be no byte transferred to the receive buffer and the RDRF flag will not be modified. Also no framing error or parity error will be flagged from this transfer. In RXD_2 case, however the break signal starts later during the transmission. At the expected stop bit position the byte received so far will be transferred to the receive buffer, the receive data register full flag will be set, a framing error and if enabled and appropriate a parity error will be set. Once the break is detected the BRKDIF flag will be set.

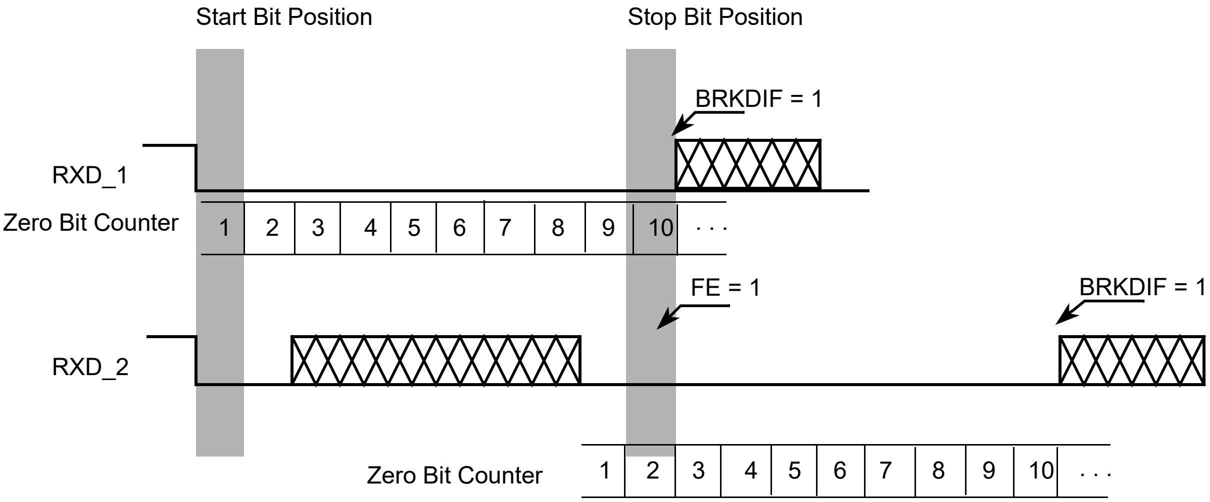


Figure 14-17. Break Detection if BRKDFE = 1 (M = 0)

14.4.5.4 Idle Characters

An idle character (or preamble) contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCI control register 1 (SCICR1). The preamble is a synchronizing idle character that begins the first transmission initiated after writing the TE bit from 0 to 1.

If the TE bit is cleared during a transmission, the TXD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the frame currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current frame shifts out through the TXD pin. Setting TE after the stop bit appears on TXD causes data previously written to the SCI data register to be lost. Toggle the TE bit for a queued idle character while the TDRE flag is set and immediately before writing the next byte to the SCI data register.

If the TE bit is clear and the transmission is complete, the SCI is not the master of the TXD pin

14.4.5.5 LIN Transmit Collision Detection

This module allows to check for collisions on the LIN bus.

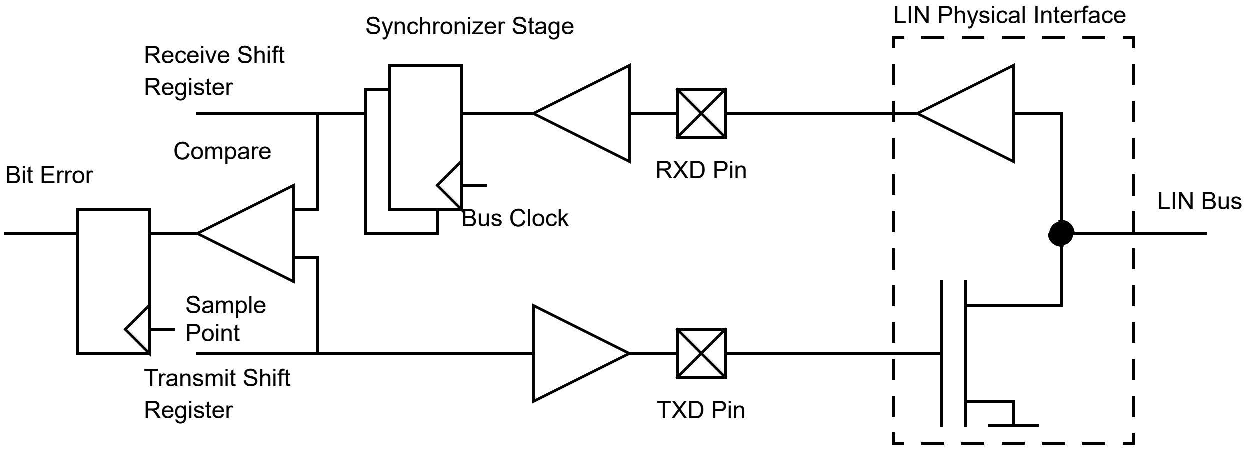


Figure 14-18. Collision Detect Principle

If the bit error circuit is enabled (BERRM[1:0] = 0:1 or = 1:0]), the error detect circuit will compare the transmitted and the received data stream at a point in time and flag any mismatch. The timing checks run when transmitter is active (not idle). As soon as a mismatch between the transmitted data and the received data is detected the following happens:

- The next bit transmitted will have a high level (TXPOL = 0) or low level (TXPOL = 1)
- The transmission is aborted and the byte in transmit buffer is discarded.
- the transmit data register empty and the transmission complete flag will be set
- The bit error interrupt flag, BERRIF, will be set.
- No further transmissions will take place until the BERRIF is cleared.

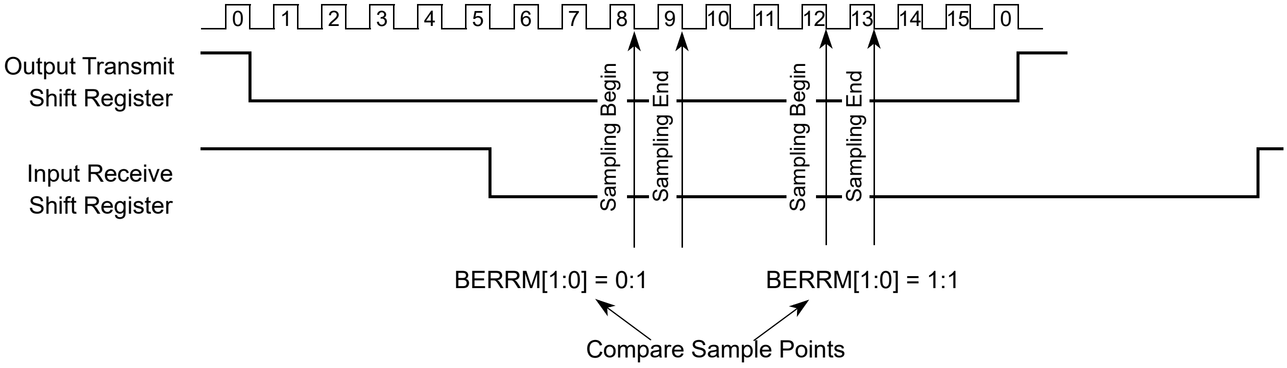


Figure 14-19. Timing Diagram Bit Error Detection

If the bit error detect feature is disabled, the bit error interrupt flag is cleared.

NOTE

The RXPOL and TXPOL bit should be set the same when transmission collision detect feature is enabled, otherwise the bit error interrupt flag may be set incorrectly.

14.4.6 Receiver

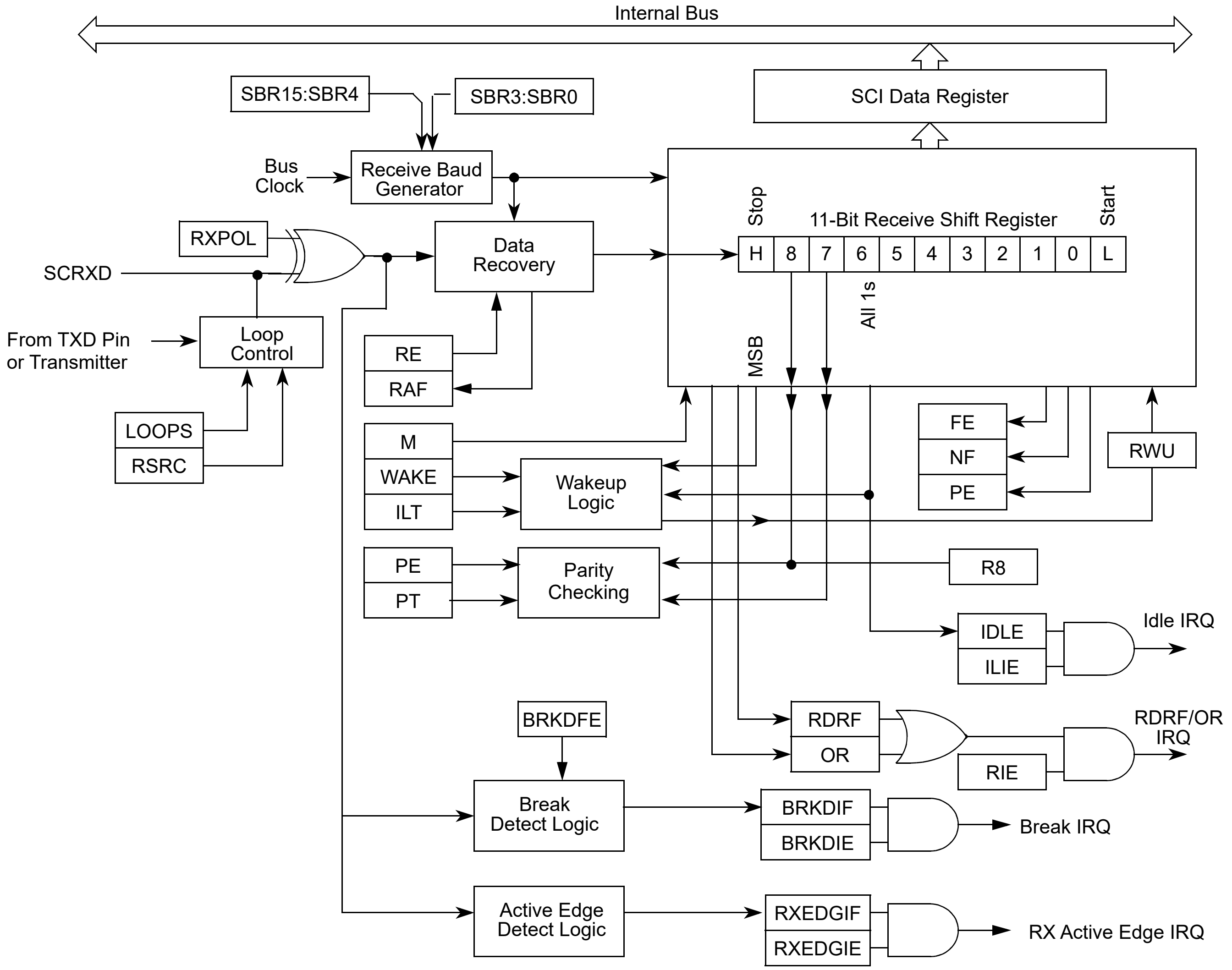


Figure 14-20. SCI Receiver Block Diagram

14.4.6.1 Receiver Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

14.4.6.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the RXD pin. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCISR1) becomes set, indicating that the received byte can be read. If the receive interrupt enable bit, RIE, in SCI control register 2 (SCICR2) is also set, the RDRF flag generates an RDRF interrupt request.

14.4.6.3 Data Sampling

The RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see Figure 14-21) is re-synchronized immediatelly at bus clock edge:

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s.When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

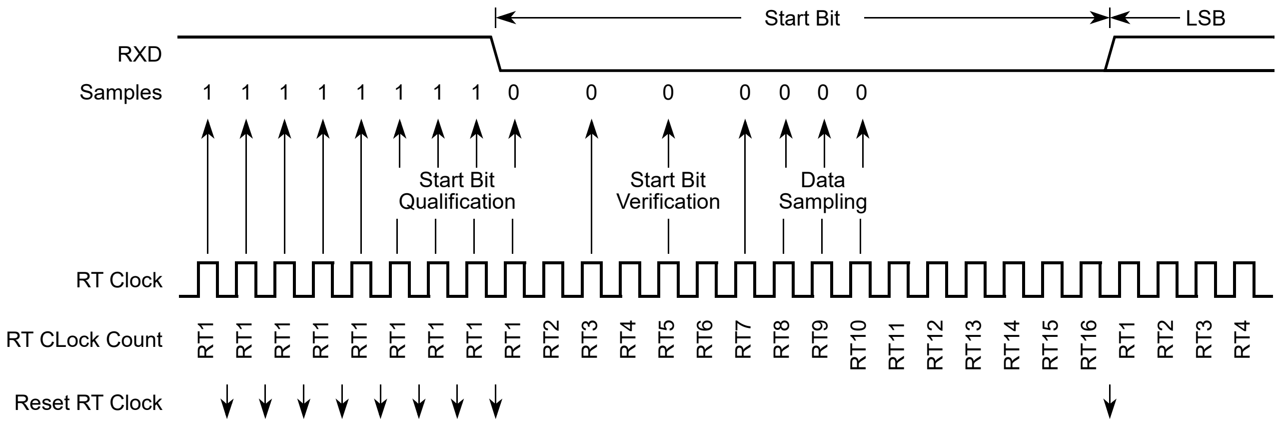


Figure 14-21. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Figure 14-17 summarizes the results of the start bit verification samples.

Table 14-17. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 14-18 summarizes the results of the data bit samples.

Table 14-18. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 14-19 summarizes the results of the stop bit samples.

Table 14-19. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In Figure 14-22 the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

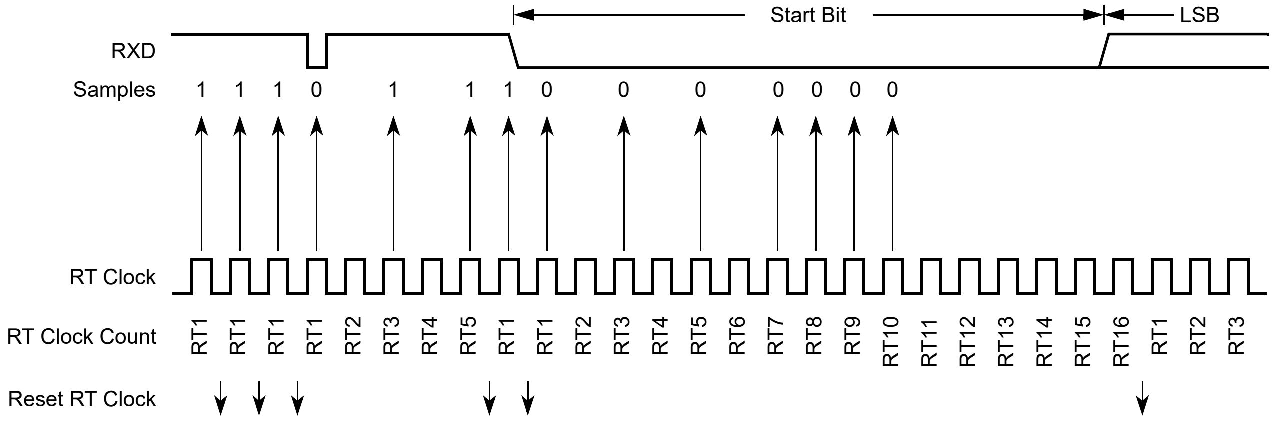


Figure 14-22. Start Bit Search Example 1

In Figure 14-23, verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

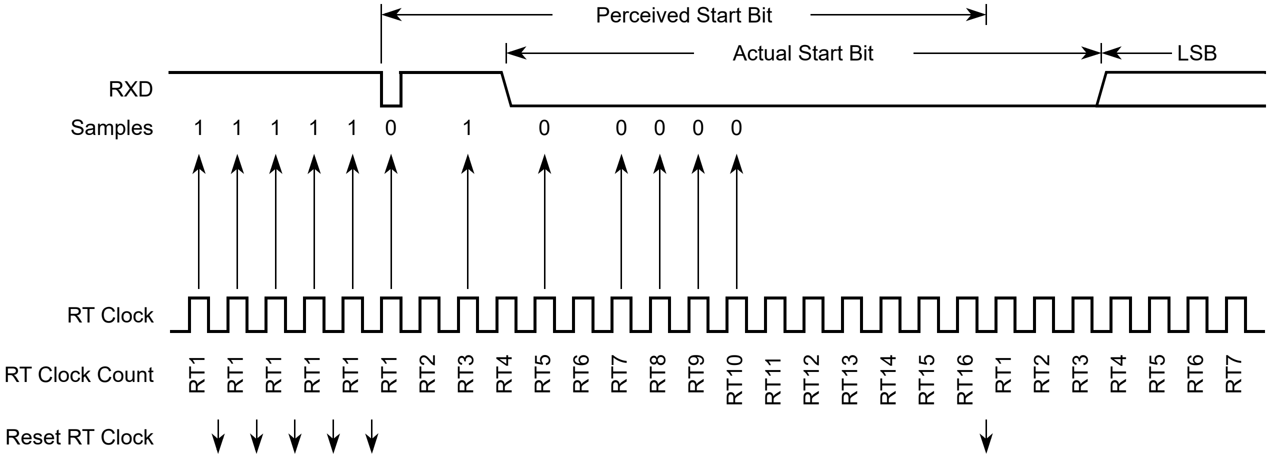


Figure 14-23. Start Bit Search Example 2

In Figure 14-24, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

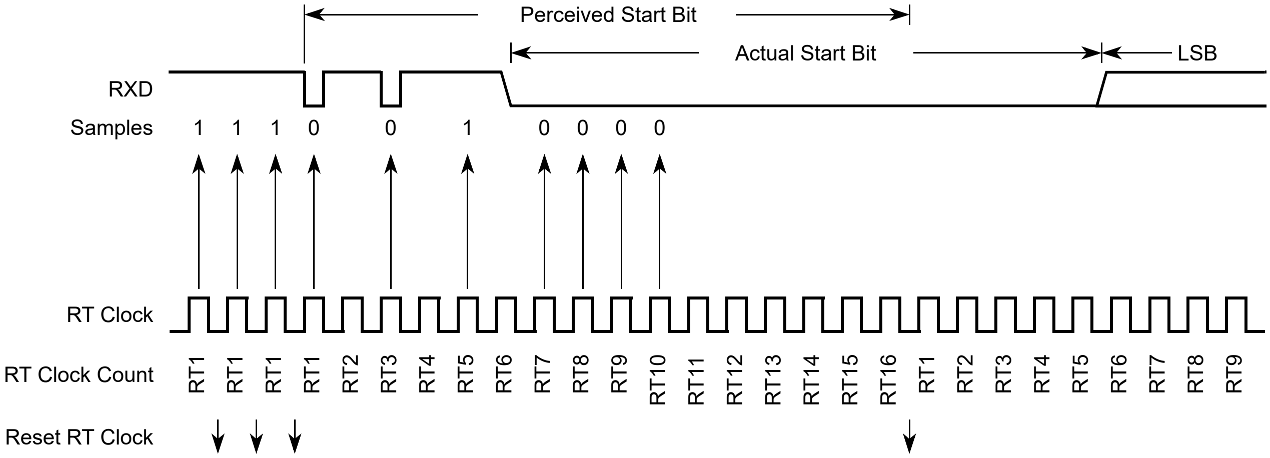


Figure 14-24. Start Bit Search Example 3

Figure 14-25 shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

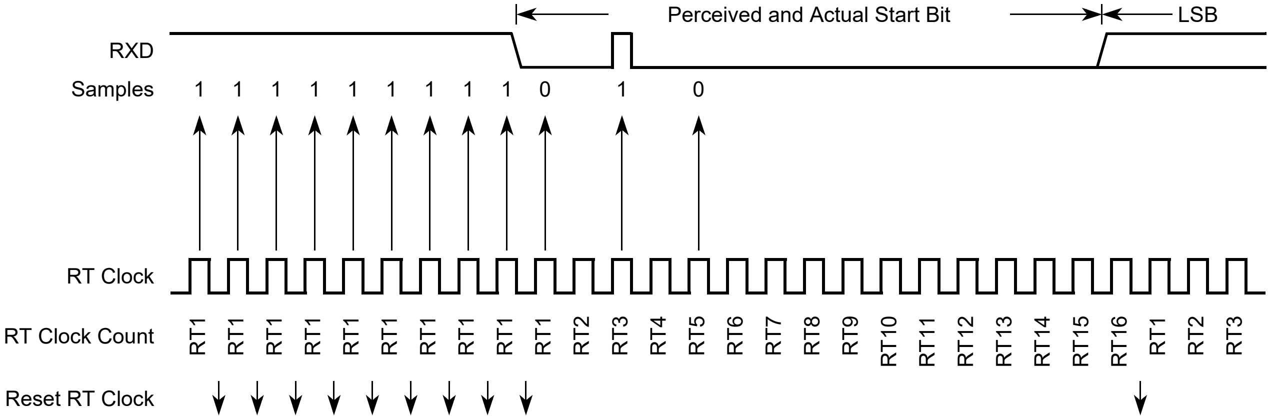


Figure 14-25. Start Bit Search Example 4

Figure 14-26 shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

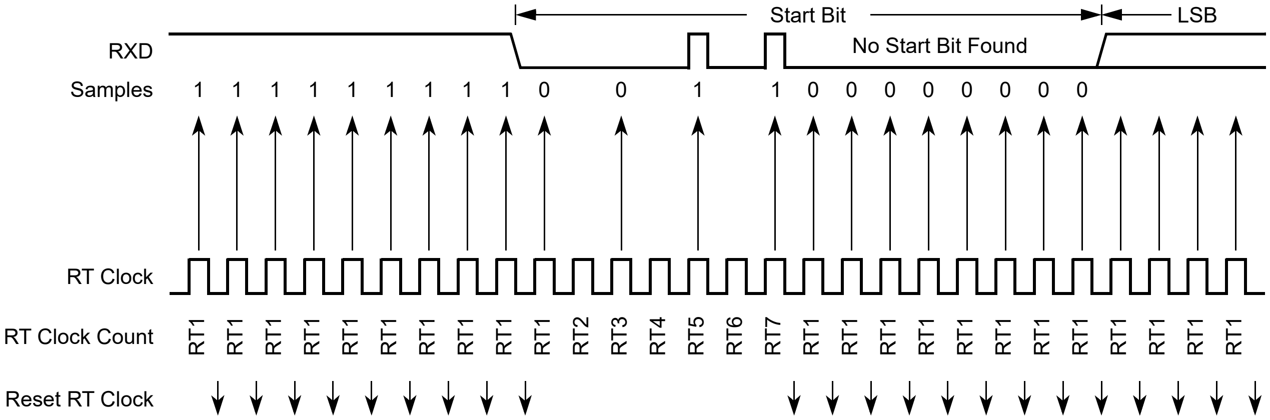


Figure 14-26. Start Bit Search Example 5

In Figure 14-27, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

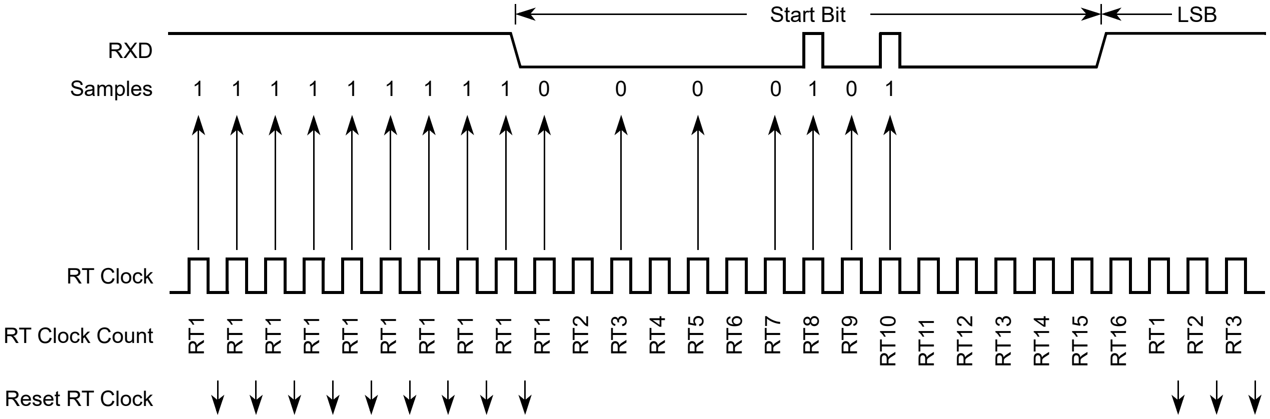


Figure 14-27. Start Bit Search Example 6

14.4.6.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, FE, in SCI status register 1 (SCISR1). A break character also sets the FE flag because a break character has no stop bit. The FE flag is set at the same time that the RDRF flag is set.

14.4.6.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic zero.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Re synchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

14.4.6.5.1 Slow Data Tolerance

Figure 14-28 shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

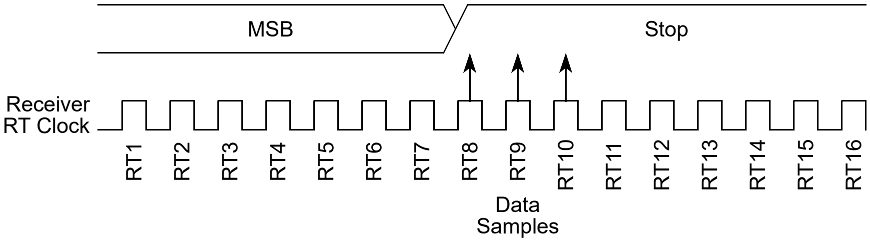


Figure 14-28. Slow Data

Let’s take RTr as receiver RT clock and RTt as transmitter RT clock.

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles +7 RTr cycles = 151 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 14-28, the receiver counts 151 RTr cycles at the point when the count of the transmitting device is 9 bit times x 16 RTt cycles = 144 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((151 - 144) / 151) \times 100 = 4.63\%$$

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 7 RTr cycles = 167 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 14-28, the receiver counts 167 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((167 - 160) / 167) \text{ X } 100 = 4.19\%$$

14.4.6.5.2 Fast Data Tolerance

Figure 14-29 shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

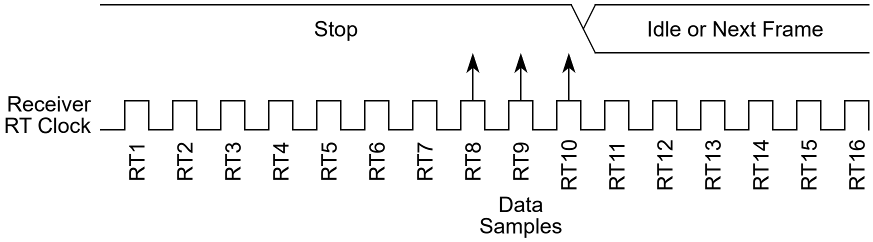


Figure 14-29. Fast Data

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 9 RTr cycles = 153 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 14-29, the receiver counts 153 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$((160 - 153) / 160) \times 100 = 4.375\%$$

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 9 RTr cycles = 169 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 14-29, the receiver counts 169 RTr cycles at the point when the count of the transmitting device is 11 bit times x 16 RTt cycles = 176 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$((176 - 169) /176) \times 100 = 3.98\%$$

NOTE

Due to asynchronous sample and internal logic, there is maximal 2 bus cycles between startbit edge and 1st RT clock, and cause to additional tolerance loss at worst case. The loss should be 2/SBR/10*100%, it is small.For example, for highspeed baud=230400 with 25MHz bus, SBR should be 109, and the tolerance loss is 2/109/10*100=0.18%, and fast data tolerance is 4.375%-0.18%=4.195%.

14.4.6.6 Receiver Wakeup

To enable the SCI to ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCI control register 2 (SCICR2) puts the receiver into standby state during which receiver interrupts are disabled.The SCI will still load the receive data into the SCIDRH/L registers, but it will not set the RDRF flag.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in SCI control register 1 (SCICR1) determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either idle line wakeup or address mark wakeup.

14.4.6.6.1 Idle Input line Wakeup (WAKE = 0)

In this wakeup method, an idle condition on the RXD pin clears the RWU bit and wakes up the SCI. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another idle character appears on the RXD pin.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

The idle character that wakes a receiver does not set the receiver idle bit, IDLE, or the receive data register full flag, RDRF.

The idle line type bit, ILT, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit. ILT is in SCI control register 1 (SCICR1).

14.4.6.6.2 Address Mark Wakeup (WAKE = 1)

In this wakeup method, a logic 1 in the most significant bit (MSB) position of a frame clears the RWU bit and wakes up the SCI. The logic 1 in the MSB position marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow.Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the RXD pin.

The logic 1 MSB of an address frame clears the receiver’s RWU bit before the stop bit is received and sets the RDRF flag.

Address mark wakeup allows messages to contain idle characters but requires that the MSB be reserved for use in address frames.

NOTE

With the WAKE bit clear, setting the RWU bit after the RXD pin has been idle can cause the receiver to wake up immediately.

14.4.7 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI. The SCI uses the TXD pin for both receiving and transmitting.

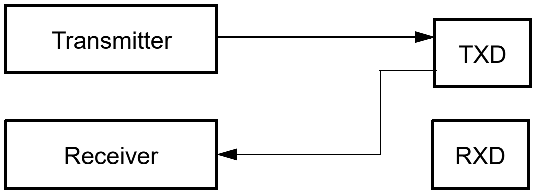


Figure 14-30. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Setting the RSRC bit connects the TXD pin to the receiver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).The TXDIR bit (SCISR2[1]) determines whether the TXD pin is going to be used as an input (TXDIR = 0) or an output (TXDIR = 1) in this mode of operation.

NOTE

In single-wire operation data from the TXD pin is inverted if RXPOL is set.

14.4.8 Loop Operation

In loop operation the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI.

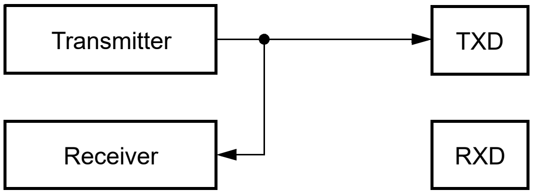


Figure 14-31. Loop Operation (LOOPS = 1, RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

NOTE

In loop operation data from the transmitter is not recognized by the receiver if RXPOL and TXPOL are not the same.

14.5 Initialization/Application Information

14.5.1 Reset Initialization

See Section 14.3.2, “Register Descriptions”.

14.5.2 Modes of Operation

14.5.2.1 Run Mode

Normal mode of operation.

To initialize a SCI transmission, see Section 14.4.5.2, “Character Transmission”.

14.5.2.2 Wait Mode

SCI operation in wait mode depends on the state of the SCISWAI bit in the SCI control register 1 (SCICR1).

- If SCISWAI is clear, the SCI operates normally when the CPU is in wait mode.
- If SCISWAI is set, SCI clock generation ceases and the SCI module enters a power-conservation state when the CPU is in wait mode. Setting SCISWAI does not affect the state of the receiver enable bit, RE, or the transmitter enable bit, TE.

  If SCISWAI is set, any transmission or reception in progress stops at wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of wait mode. Exiting wait mode by reset aborts any transmission or reception in progress and resets the SCI.

14.5.2.3 Stop Mode

The SCI is inactive during stop mode for reduced power consumption. The STOP instruction does not affect the SCI register states, but the SCI bus clock will be disabled. The SCI operation resumes from where it left off after an external interrupt brings the CPU out of stop mode. Exiting stop mode by reset aborts any transmission or reception in progress and resets the SCI.

The receive input active edge detect circuit is still active in stop mode. An active edge on the receive input can be used to bring the CPU out of stop mode.

14.5.3 Interrupt Operation

This section describes the interrupt originated by the SCI block.The MCU must service the interrupt requests. Table 14-20 lists the eight interrupt sources of the SCI.

Table 14-20. SCI Interrupt Sources

Interrupt	Source	Local Enable	Description
TDRE	SCISR1[7]	TIE	Active high level. Indicates that a byte was transferred from SCIDRH/L to the transmit shift register.
TC	SCISR1[6]	TCIE	Active high level. Indicates that a transmit is complete.
RDRF	SCISR1[5]	RIE	Active high level. The RDRF interrupt indicates that received data is available in the SCI data register.
OR	SCISR1[3]		Active high level. This interrupt indicates that an overrun condition has occurred.
IDLE	SCISR1[4]	ILIE	Active high level. Indicates that receiver input has become idle.

Table 14-20. SCI Interrupt Sources

RXEDGIF	SCIASR1[7]	RXEDGIE	Active high level. Indicates that an active edge (falling for RXPOL = 0, rising for RXPOL = 1) was detected.
BERRIF	SCIASR1[1]	BERRIE	Active high level. Indicates that a mismatch between transmitted and received data in a single wire application has happened.
BKDIF	SCIASR1[0]	BRKDIE	Active high level. Indicates that a break character has been received.

14.5.3.1 Description of Interrupt Operation

The SCI only originates interrupt requests. The following is a description of how the SCI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are chip dependent. The SCI only has a single interrupt line (SCI Interrupt Signal, active high operation) and all the following interrupts, when generated, are ORed together and issued through that port.

14.5.3.1.1 TDRE Description

The TDRE interrupt is set high by the SCI when the transmit shift register receives a byte from the SCI data register. A TDRE interrupt indicates that the transmit data register (SCIDRH/L) is empty and that a new byte can be written to the SCIDRH/L for transmission.Clear TDRE by reading SCI status register 1 with TDRE set and then writing to SCI data register low (SCIDRL).

14.5.3.1.2 TC Description

The TC interrupt is set by the SCI when a transmission has been completed. Transmission is completed when all bits including the stop bit (if transmitted) have been shifted out and no data is queued to be transmitted. No stop bit is transmitted when sending a break character and the TC flag is set (providing there is no more data queued for transmission) when the break character has been shifted out. A TC interrupt indicates that there is no transmission in progress. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL).TC is cleared automatically when data, preamble, or break is queued and ready to be sent.

14.5.3.1.3 RDRF Description

The RDRF interrupt is set when the data in the receive shift register transfers to the SCI data register. A RDRF interrupt indicates that the received data has been transferred to the SCI data register and that the byte can now be read by the MCU. The RDRF interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

14.5.3.1.4 OR Description

The OR interrupt is set when software fails to read the SCI data register before the receive shift register receives the next frame. The newly acquired data in the shift register will be lost in this case, but the data already in the SCI data registers is not affected. The OR interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

14.5.3.1.5 IDLE Description

The IDLE interrupt is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M = 1) appear on the receiver input. Once the IDLE is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag. Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL).

14.5.3.1.6 RXEDGIF Description

The RXEDGIF interrupt is set when an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD pin is detected. Clear RXEDGIF by writing a “1” to the SCIASR1 SCI alternative status register 1.

14.5.3.1.7 BERRIF Description

The BERRIF interrupt is set when a mismatch between the transmitted and the received data in a single wire application like LIN was detected. Clear BERRIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if the bit error detect feature is disabled.

14.5.3.1.8 BKDIF Description

The BKDIF interrupt is set when a break signal was received. Clear BKDIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if break detect feature is disabled.

14.5.4 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

14.5.5 Recovery from Stop Mode

An active edge on the receive input can be used to bring the CPU out of stop mode.

Chapter 15
Serial Peripheral Interface (S12SPIV5)

15.1 Introduction

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or the SPI operation can be interrupt driven.

15.1.1 Glossary of Terms

SPI	Serial Peripheral Interface
$\overline{\text{SS}}$	Slave Select
SCK	Serial Clock
MOSI	Master Output, Slave Input
MISO	Master Input, Slave Output
MOMI	Master Output, Master Input
SISO	Slave Input, Slave Output

15.1.2 Features

The SPI includes these distinctive features:

- Master mode and slave mode
- Selectable 8 or 16-bit transfer width
- Bidirectional mode
- Slave select output
- Mode fault error flag with CPU interrupt capability
- Double-buffered data register
- Serial clock with programmable polarity and phase
- Control of SPI operation during wait mode

15.1.3 Modes of Operation

The SPI functions in three modes: run, wait, and stop.

- Run mode

  This is the basic mode of operation.
- Wait mode

SPI operation in wait mode is a configurable low power mode, controlled by the SPISWAI bit located in the SPICR2 register. In wait mode, if the SPISWAI bit is clear, the SPI operates like in run mode. If the SPISWAI bit is set, the SPI goes into a power conservative state, with the SPI clock generation turned off. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into run mode. If the SPI is configured as a slave, reception and transmission of data continues, so that the slave stays synchronized to the master.

- Stop mode

  The SPI is inactive in stop mode for reduced power consumption. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into run mode. If the SPI is configured as a slave, reception and transmission of data continues, so that the slave stays synchronized to the master.

For a detailed description of operating modes, please refer to Section 15.4.7, “Low Power Mode Options”.

15.1.4 Block Diagram

Figure 15-1 gives an overview on the SPI architecture. The main parts of the SPI are status, control and data registers, shifter logic, baud rate generator, master/slave control logic, and port control logic.

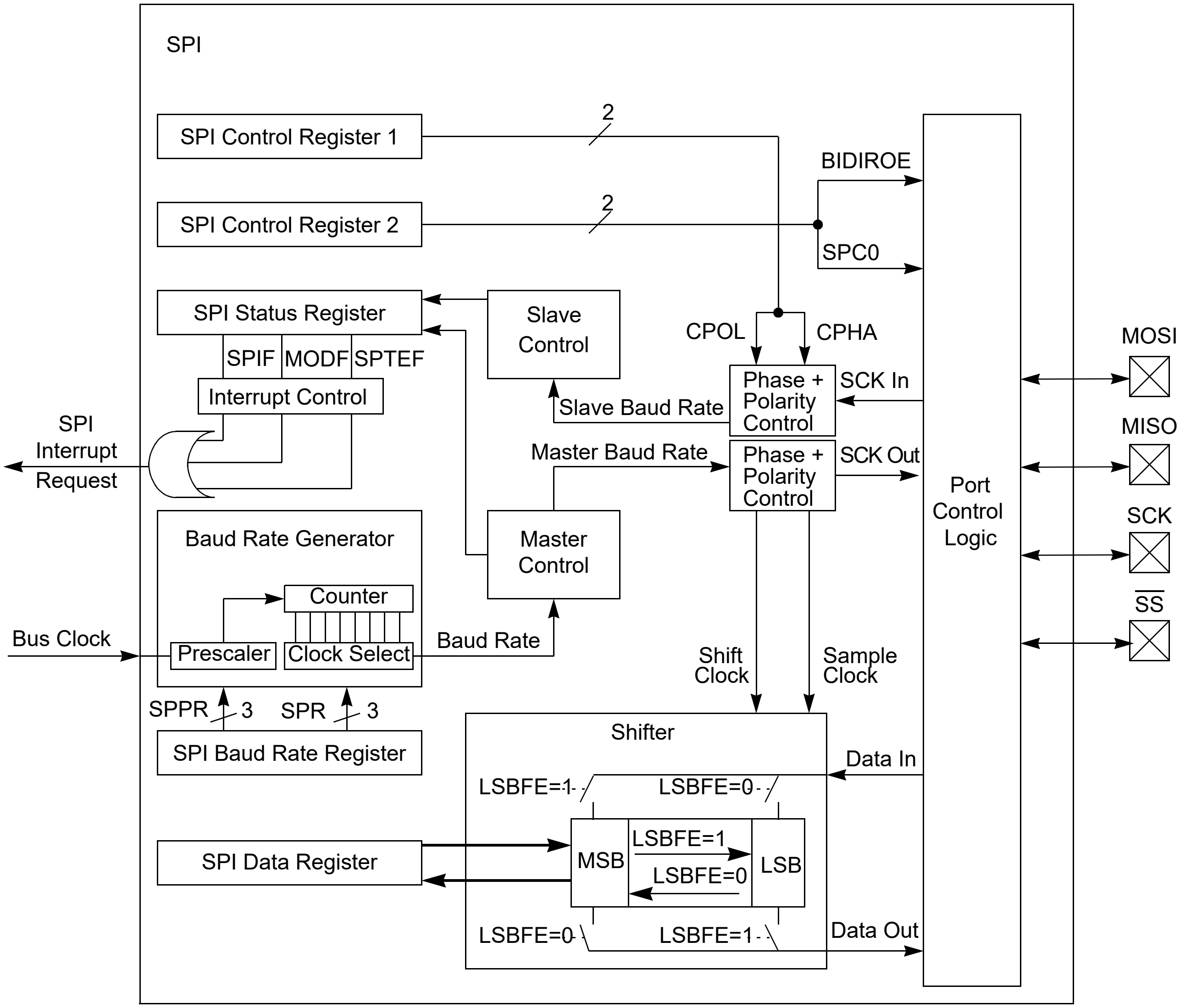


Figure 15-1. SPI Block Diagram

15.2 External Signal Description

This section lists the name and description of all ports including inputs and outputs that do, or may, connect off chip. The SPI module has a total of four external pins.

15.2.1 MOSI — Master Out/Slave In Pin

This pin is used to transmit data out of the SPI module when it is configured as a master and receive data when it is configured as slave.

15.2.2 MISO — Master In/Slave Out Pin

This pin is used to transmit data out of the SPI module when it is configured as a slave and receive data when it is configured as master.

15.2.3 $\overline{\text{SS}}$ — Slave Select Pin

This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place when it is configured as a master and it is used as an input to receive the slave select signal when the SPI is configured as slave.

15.2.4 SCK — Serial Clock Pin

In master mode, this is the synchronous output clock. In slave mode, this is the synchronous input clock.

15.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

15.3.1 Module Memory Map

The memory map for the SPI is given in Figure 15-2. The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reads from the reserved bits return zeros and writes to the reserved bits have no effect.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 SPICR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
	W								
0x0001 SPICR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
	W	▨		▨			▨		
0x0002 SPIBR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
	W	▨				▨			
0x0003 SPISR	R	SPIF	0	SPTEF	MODF	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0004 SPIDRH	R	R15	R14	R13	R12	R11	R10	R9	R8
	W	T15	T14	T13	T12	T11	T10	T9	T8
0x0005 SPIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
	W	T7	T6	T5	T4	T3	T2	T1	T0
0x0006 Reserved	R	▨	▨	▨	▨	▨	▨	▨	▨
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0007 Reserved	R	▨	▨	▨	▨	▨	▨	▨	▨
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 15-2. SPI Register Summary

15.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

15.3.2.1 SPI Control Register 1 (SPICR1)

Module Base +0x0000

	7	6	5	4	3	2	1	0
R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset	0	0	0	0	0	1	0	0

Figure 15-3. SPI Control Register 1 (SPICR1)

Read: Anytime

Write: Anytime

Table 15-1. SPICR1 Field Descriptions

Field	Description
7 SPIE	**SPI Interrupt Enable Bit** — This bit enables SPI interrupt requests, if SPIF or MODF status flag is set. 0 SPI interrupts disabled. 1 SPI interrupts enabled.
6 SPE	**SPI System Enable Bit** — This bit enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, SPI is disabled and forced into idle state, status bits in SPISR register are reset. 0 SPI disabled (lower power consumption). 1 SPI enabled, port pins are dedicated to SPI functions.
5 SPTIE	**SPI Transmit Interrupt Enable** — This bit enables SPI interrupt requests, if SPTEF flag is set. 0 SPTEF interrupt disabled. 1 SPTEF interrupt enabled.
4 MSTR	**SPI Master/Slave Mode Select Bit** — This bit selects whether the SPI operates in master or slave mode. Switching the SPI from master to slave or vice versa forces the SPI system into idle state. 0 SPI is in slave mode. 1 SPI is in master mode.
3 CPOL	**SPI Clock Polarity Bit** — This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Active-high clocks selected. In idle state SCK is low. 1 Active-low clocks selected. In idle state SCK is high.
2 CPHA	**SPI Clock Phase Bit** — This bit is used to select the SPI clock format. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Sampling of data occurs at odd edges (1,3,5,...) of the SCK clock. 1 Sampling of data occurs at even edges (2,4,6,...) of the SCK clock.

Table 15-1. SPICR1 Field Descriptions (continued)

Field	Description
1 SSOE	**Slave Select Output Enable** — The $\overline{\text{SS}}$ output feature is enabled only in master mode, if MODFEN is set, by asserting the SSOE as shown in Table 15-2. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.
0 LSBFE	**LSB-First Enable** — This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in the highest bit position. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Data is transferred most significant bit first. 1 Data is transferred least significant bit first.

Table 15-2. $\overline{\text{SS}}$ Input / Output Selection

MODFEN	SSOE	Master Mode	Slave Mode
0	0	$\overline{\text{SS}}$ not used by SPI	$\overline{\text{SS}}$ input
0	1	$\overline{\text{SS}}$ not used by SPI	$\overline{\text{SS}}$ input
1	0	$\overline{\text{SS}}$ input with MODF feature	$\overline{\text{SS}}$ input
1	1	$\overline{\text{SS}}$ is slave select output	$\overline{\text{SS}}$ input

15.3.2.2 SPI Control Register 2 (SPICR2)

Module Base +0x0001

	7	6	5	4	3	2	1	0
R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
W	▨		▨			▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 15-4. SPI Control Register 2 (SPICR2)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 15-3. SPICR2 Field Descriptions

Field	Description
6 XFRW	**Transfer Width —** This bit is used for selecting the data transfer width. If 8-bit transfer width is selected, SPIDRL becomes the dedicated data register and SPIDRH is unused. If 16-bit transfer width is selected, SPIDRH and SPIDRL form a 16-bit data register. Please refer to Section 15.3.2.4, "SPI Status Register (SPISR) for information about transmit/receive data handling and the interrupt flag clearing mechanism. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 8-bit Transfer Width (n = 8)[1] 1 16-bit Transfer Width (n = 16)[1]
4 MODFEN	**Mode Fault Enable Bit** — This bit allows the MODF failure to be detected. If the SPI is in master mode and MODFEN is cleared, then the $\overline{\text{SS}}$ port pin is not used by the SPI. In slave mode, the $\overline{\text{SS}}$ is available only as an input regardless of the value of MODFEN. For an overview on the impact of the MODFEN bit on the $\overline{\text{SS}}$ port pin configuration, refer to Table 15-2. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 $\overline{\text{SS}}$ port pin is not used by the SPI. 1 $\overline{\text{SS}}$ port pin with MODF feature.
3 BIDIROE	**Output Enable in the Bidirectional Mode of Operation** — This bit controls the MOSI and MISO output buffer of the SPI, when in bidirectional mode of operation (SPC0 is set). In master mode, this bit controls the output buffer of the MOSI port, in slave mode it controls the output buffer of the MISO port. In master mode, with SPC0 set, a change of this bit will abort a transmission in progress and force the SPI into idle state. 0 Output buffer disabled. 1 Output buffer enabled.
1 SPISWAI	**SPI Stop in Wait Mode Bit** — This bit is used for power conservation while in wait mode. 0 SPI clock operates normally in wait mode. 1 Stop SPI clock generation when in wait mode.
0 SPC0	**Serial Pin Control Bit 0** — This bit enables bidirectional pin configurations as shown in Table 15-4. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.

[1] n is used later in this document as a placeholder for the selected transfer width.

Table 15-4. Bidirectional Pin Configurations

Pin Mode	SPC0	BIDIROE	MISO	MOSI
Master Mode of Operation				
Normal	0	X	Master In	Master Out
Bidirectional	1	0	MISO not used by SPI	Master In
		1		Master I/O
Slave Mode of Operation				
Normal	0	X	Slave Out	Slave In
Bidirectional	1	0	Slave In	MOSI not used by SPI
		1	Slave I/O	

15.3.2.3 SPI Baud Rate Register (SPIBR)

Module Base +0x0002

	7	6	5	4	3	2	1	0
R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
W	▨				▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 15-5. SPI Baud Rate Register (SPIBR)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 15-5. SPIBR Field Descriptions

Field	Description
6–4 SPPR[2:0]	**SPI Baud Rate Preselection Bits** — These bits specify the SPI baud rates as shown in Table 15-6. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.
2–0 SPR[2:0]	**SPI Baud Rate Selection Bits** — These bits specify the SPI baud rates as shown in Table 15-6. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.

The baud rate divisor equation is as follows:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \text{Eqn. 15-1}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor} \qquad \text{Eqn. 15-2}$$

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

Table 15-6. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 1 of 3)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
0	0	0	0	0	0	2	12.5 Mbit/s
0	0	0	0	0	1	4	6.25 Mbit/s
0	0	0	0	1	0	8	3.125 Mbit/s
0	0	0	0	1	1	16	1.5625 Mbit/s
0	0	0	1	0	0	32	781.25 kbit/s
0	0	0	1	0	1	64	390.63 kbit/s
0	0	0	1	1	0	128	195.31 kbit/s
0	0	0	1	1	1	256	97.66 kbit/s
0	0	1	0	0	0	4	6.25 Mbit/s
0	0	1	0	0	1	8	3.125 Mbit/s

Table 15-6. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 2 of 3)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
0	0	1	0	1	0	16	1.5625 Mbit/s
0	0	1	0	1	1	32	781.25 kbit/s
0	0	1	1	0	0	64	390.63 kbit/s
0	0	1	1	0	1	128	195.31 kbit/s
0	0	1	1	1	0	256	97.66 kbit/s
0	0	1	1	1	1	512	48.83 kbit/s
0	1	0	0	0	0	6	4.16667 Mbit/s
0	1	0	0	0	1	12	2.08333 Mbit/s
0	1	0	0	1	0	24	1.04167 Mbit/s
0	1	0	0	1	1	48	520.83 kbit/s
0	1	0	1	0	0	96	260.42 kbit/s
0	1	0	1	0	1	192	130.21 kbit/s
0	1	0	1	1	0	384	65.10 kbit/s
0	1	0	1	1	1	768	32.55 kbit/s
0	1	1	0	0	0	8	3.125 Mbit/s
0	1	1	0	0	1	16	1.5625 Mbit/s
0	1	1	0	1	0	32	781.25 kbit/s
0	1	1	0	1	1	64	390.63 kbit/s
0	1	1	1	0	0	128	195.31 kbit/s
0	1	1	1	0	1	256	97.66 kbit/s
0	1	1	1	1	0	512	48.83 kbit/s
0	1	1	1	1	1	1024	24.41 kbit/s
1	0	0	0	0	0	10	2.5 Mbit/s
1	0	0	0	0	1	20	1.25 Mbit/s
1	0	0	0	1	0	40	625 kbit/s
1	0	0	0	1	1	80	312.5 kbit/s
1	0	0	1	0	0	160	156.25 kbit/s
1	0	0	1	0	1	320	78.13 kbit/s
1	0	0	1	1	0	640	39.06 kbit/s
1	0	0	1	1	1	1280	19.53 kbit/s
1	0	1	0	0	0	12	2.08333 Mbit/s
1	0	1	0	0	1	24	1.04167 Mbit/s
1	0	1	0	1	0	48	520.83 kbit/s
1	0	1	0	1	1	96	260.42 kbit/s
1	0	1	1	0	0	192	130.21 kbit/s
1	0	1	1	0	1	384	65.10 kbit/s
1	0	1	1	1	0	768	32.55 kbit/s
1	0	1	1	1	1	1536	16.28 kbit/s
1	1	0	0	0	0	14	1.78571 Mbit/s
1	1	0	0	0	1	28	892.86 kbit/s
1	1	0	0	1	0	56	446.43 kbit/s
1	1	0	0	1	1	112	223.21 kbit/s

Table 15-6. Example SPI Baud Rate Selection (25 MHz Bus Clock) (Sheet 3 of 3)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
1	1	0	1	0	0	224	111.61 kbit/s
1	1	0	1	0	1	448	55.80 kbit/s
1	1	0	1	1	0	896	27.90 kbit/s
1	1	0	1	1	1	1792	13.95 kbit/s
1	1	1	0	0	0	16	1.5625 Mbit/s
1	1	1	0	0	1	32	781.25 kbit/s
1	1	1	0	1	0	64	390.63 kbit/s
1	1	1	0	1	1	128	195.31 kbit/s
1	1	1	1	0	0	256	97.66 kbit/s
1	1	1	1	0	1	512	48.83 kbit/s
1	1	1	1	1	0	1024	24.41 kbit/s
1	1	1	1	1	1	2048	12.21 kbit/s

15.3.2.4 SPI Status Register (SPISR)

Module Base +0x0003

	7	6	5	4	3	2	1	0
R	SPIF	0	SPTEF	MODF	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	1	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 15-6. SPI Status Register (SPISR)

Read: Anytime

Write: Has no effect

Table 15-7. SPISR Field Descriptions

Field	Description
7 SPIF	**SPIF Interrupt Flag** — This bit is set after received data has been transferred into the SPI data register. For information about clearing SPIF Flag, please refer to Table 15-8. 0 Transfer not yet complete. 1 New data copied to SPIDR.

Table 15-7. SPISR Field Descriptions (continued)

Field	Description
5 SPTEF	**SPI Transmit Empty Interrupt Flag** — If set, this bit indicates that the transmit data register is empty. For information about clearing this bit and placing data into the transmit data register, please refer to Table 15-9. 0 SPI data register not empty. 1 SPI data register empty.
4 MODF	**Mode Fault Flag** — This bit is set if the $\overline{SS}$ input becomes low while the SPI is configured as a master and mode fault detection is enabled, MODFEN bit of SPICR2 register is set. Refer to MODFEN bit description in Section 15.3.2.2, "SPI Control Register 2 (SPICR2)". The flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to the SPI control register 1. 0 Mode fault has not occurred. 1 Mode fault has occurred.

Table 15-8. SPIF Interrupt Flag Clearing Sequence

<table>
<tr><th>XFRW Bit</th><th colspan="4">SPIF Interrupt Flag Clearing Sequence</th></tr>
<tr><td>0</td><td>Read SPISR with SPIF == 1</td><td>then</td><td colspan="2">Read SPIDRL</td></tr>
<tr><td rowspan="5">1</td><td rowspan="5">Read SPISR with SPIF == 1</td><td rowspan="5">then</td><td colspan="2">Byte Read SPIDRL [1]</td></tr>
<tr><td colspan="2">or</td></tr>
<tr><td>Byte Read SPIDRH [2]</td><td>Byte Read SPIDRL</td></tr>
<tr><td colspan="2">or</td></tr>
<tr><td colspan="2">Word Read (SPIDRH:SPIDRL)</td></tr>
</table>

[1] Data in SPIDRH is lost in this case.

[2] SPIDRH can be read repeatedly without any effect on SPIF. SPIF Flag is cleared only by the read of SPIDRL after reading SPISR with SPIF == 1.

Table 15-9. SPTEF Interrupt Flag Clearing Sequence

<table>
<tr><th>XFRW Bit</th><th colspan="4">SPTEF Interrupt Flag Clearing Sequence</th></tr>
<tr><td>0</td><td>Read SPISR with SPTEF == 1</td><td>then</td><td colspan="2">Write to SPIDRL [1]</td></tr>
<tr><td rowspan="5">1</td><td rowspan="5">Read SPISR with SPTEF == 1</td><td rowspan="5">then</td><td colspan="2">Byte Write to SPIDRL [1,2]</td></tr>
<tr><td colspan="2">or</td></tr>
<tr><td>Byte Write to SPIDRH [1,3]</td><td>Byte Write to SPIDRL [1]</td></tr>
<tr><td colspan="2">or</td></tr>
<tr><td colspan="2">Word Write to (SPIDRH:SPIDRL) [1]</td></tr>
</table>

[1] Any write to SPIDRH or SPIDRL with SPTEF == 0 is effectively ignored.

[2] Data in SPIDRH is undefined in this case.

[3] SPIDRH can be written repeatedly without any effect on SPTEF. SPTEF Flag is cleared only by writing to SPIDRL after reading SPISR with SPTEF == 1.

15.3.2.5 SPI Data Register (SPIDR = SPIDRH:SPIDRL)

Module Base +0x0004

	7	6	5	4	3	2	1	0
R	R15	R14	R13	R12	R11	R10	R9	R8
W	T15	T14	T13	T12	T11	T10	T9	T8
Reset	0	0	0	0	0	0	0	0

Figure 15-7. SPI Data Register High (SPIDRH)

Module Base +0x0005

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

Figure 15-8. SPI Data Register Low (SPIDRL)

Read: Anytime; read data only valid when SPIF is set

Write: Anytime

The SPI data register is both the input and output register for SPI data. A write to this register allows data to be queued and transmitted. For an SPI configured as a master, queued data is transmitted immediately after the previous transmission has completed. The SPI transmitter empty flag SPTEF in the SPISR register indicates when the SPI data register is ready to accept new data.

Received data in the SPIDR is valid when SPIF is set.

If SPIF is cleared and data has been received, the received data is transferred from the receive shift register to the SPIDR and SPIF is set.

If SPIF is set and not serviced, and a second data value has been received, the second received data is kept as valid data in the receive shift register until the start of another transmission. The data in the SPIDR does not change.

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced before the start of a third transmission, the data in the receive shift register is transferred into the SPIDR and SPIF remains set (see Figure 15-9).

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced after the start of a third transmission, the data in the receive shift register has become invalid and is not transferred into the SPIDR (see Figure 15-10).

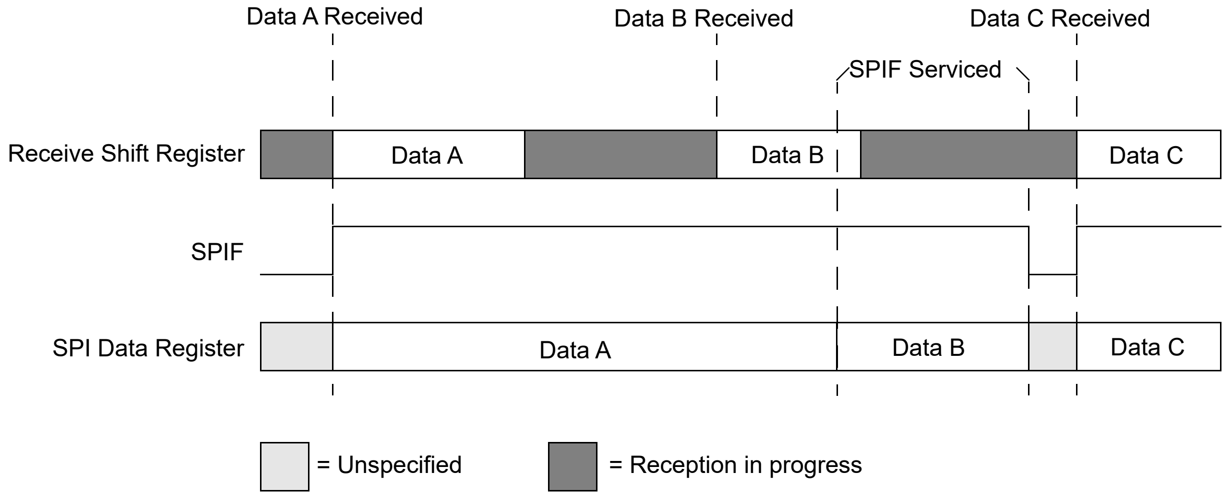


Figure 15-9. Reception with SPIF serviced in Time

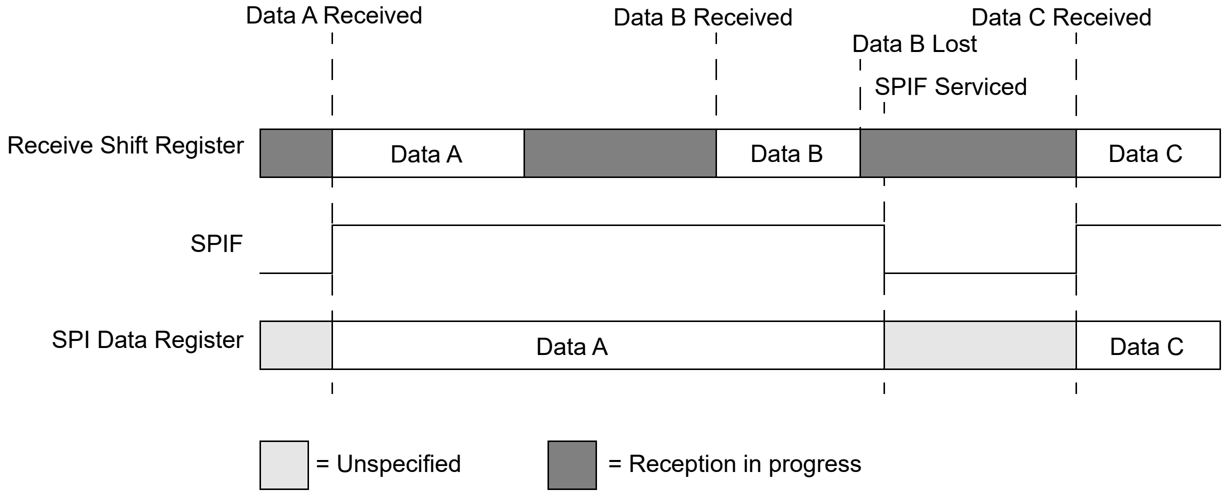


Figure 15-10. Reception with SPIF serviced too late

15.4 Functional Description

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI control register 1. While SPE is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select ($\overline{\text{SS}}$)
- Serial clock (SCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

The main element of the SPI system is the SPI data register. The n-bit[1] data register in the master and the n-bit[1] data register in the slave are linked by the MOSI and MISO pins to form a distributed 2n-bit[1] register. When a data transfer operation is performed, this 2n-bit[1] register is serially shifted n[1] bit positions by the S-clock from the master, so data is exchanged between the master and the slave. Data written to the master SPI data register becomes the output data for the slave, and data read from the master SPI data register after a transfer operation is the input data from the slave.

A read of SPISR with SPTEF = 1 followed by a write to SPIDR puts data into the transmit data register. When a transfer is complete and SPIF is cleared, received data is moved into the receive data register. This data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A common SPI data register address is shared for reading data from the read data buffer and for writing data to the transmit data register.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI control register 1 (SPICR1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by sampling data on odd numbered SCK edges or on even numbered SCK edges (see Section 15.4.3, “Transmission Formats”).

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI control register1 is set, master mode is selected, when the MSTR bit is clear, slave mode is selected.

NOTE

> A change of CPOL or MSTR bit while there is a received byte pending in the receive shift register will destroy the received byte and must be avoided.

15.4.1 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by writing to the master SPI data register. If the shift register is empty, data immediately transfers to the shift register. Data begins shifting out on the MOSI pin under the control of the serial clock.

- Serial clock

  The SPR2, SPR1, and SPR0 baud rate selection bits, in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI baud rate register, control the baud rate generator and determine the speed of the transmission. The SCK pin is the SPI clock output. Through the SCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.
- MOSI, MISO pin

  In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{\text{SS}}$ pin

  If MODFEN and SSOE are set, the $\overline{\text{SS}}$ pin is configured as slave select output. The $\overline{\text{SS}}$ output becomes low during each transmission and is high when the SPI is in idle state.

---

[1] n depends on the selected transfer width, please refer to Section 15.3.2.2, “SPI Control Register 2 (SPICR2)

If MODFEN is set and SSOE is cleared, the $\overline{\text{SS}}$ pin is configured as input for detecting mode fault error. If the $\overline{\text{SS}}$ input becomes low this indicates a mode fault error where another master tries to drive the MOSI and SCK lines. In this case, the SPI immediately switches to slave mode, by clearing the MSTR bit and also disables the slave output buffer MISO (or SISO in bidirectional mode). So the result is that all outputs are disabled and SCK, MOSI, and MISO are inputs. If a transmission is in progress when the mode fault occurs, the transmission is aborted and the SPI is forced into idle state.

This mode fault error also sets the mode fault (MODF) flag in the SPI status register (SPISR). If the SPI interrupt enable bit (SPIE) is set when the MODF flag becomes set, then an SPI interrupt sequence is also requested.

When a write to the SPI data register in the master occurs, there is a half SCK-cycle delay. After the delay, SCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI control register 1 (see Section 15.4.3, “Transmission Formats”).

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, XFRW, MODFEN, SPC0, or BIDIROE with SPC0 set, SPPR2-SPPR0 and SPR2-SPR0 in master mode will abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master must ensure that the remote slave is returned to idle state.

15.4.2 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI control register 1 is clear.

- Serial clock

  In slave mode, SCK is the SPI clock input from the master.

- MISO, MOSI pin

  In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI control register 2.

- $\overline{\text{SS}}$ pin

  The $\overline{\text{SS}}$ pin is the slave select input. Before a data transmission occurs, the $\overline{\text{SS}}$ pin of the slave SPI must be low. $\overline{\text{SS}}$ must remain low until the transmission is complete. If $\overline{\text{SS}}$ goes high, the SPI is forced into idle state.

  The $\overline{\text{SS}}$ input also controls the serial data output pin, if $\overline{\text{SS}}$ is high (not selected), the serial data output pin is high impedance, and, if $\overline{\text{SS}}$ is low, the first bit in the SPI data register is driven out of the serial data output pin. Also, if the slave is not selected ($\overline{\text{SS}}$ is high), then the SCK input is ignored and no internal shifting of the SPI shift register occurs.

  Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

NOTE

> When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI control register 1 is clear, odd numbered edges on the SCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

If the CPHA bit is set, even numbered edges on the SCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the $\overline{SS}$ input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the nth[1] shift, the transfer is considered complete and the received data is transferred into the SPI data register. To indicate transfer is complete, the SPIF flag in the SPI status register is set.

NOTE

> A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0, or BIDIROE with SPC0 set in slave mode will corrupt a transmission in progress and must be avoided.

15.4.3 Transmission Formats

During an SPI transmission, data is transmitted (shifted out serially) and received (shifted in serially) simultaneously. The serial clock (SCK) synchronizes shifting and sampling of the information on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. Optionally, on a master SPI device, the slave select line can be used to indicate multiple-master bus contention.

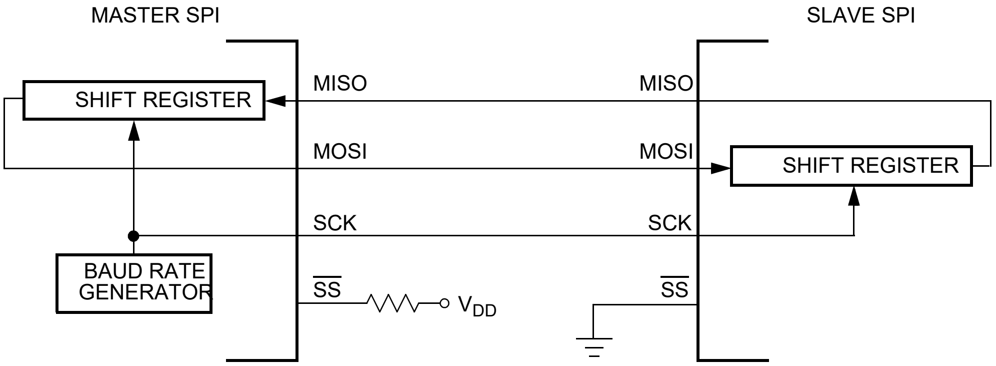


Figure 15-11. Master/Slave Transfer Block Diagram

---

[1] n depends on the selected transfer width, please refer to Section 15.3.2.2, "SPI Control Register 2 (SPICR2)

15.4.3.1 Clock Phase and Polarity Controls

Using two bits in the SPI control register 1, software selects one of four combinations of serial clock phase and polarity.

The CPOL clock polarity control bit specifies an active high or low clock and has no significant effect on the transmission format.

The CPHA clock phase control bit selects one of two fundamentally different transmission formats.

Clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

15.4.3.2 CPHA = 0 Transfer Format

The first edge on the SCK line is used to clock the first data bit of the slave into the master and the first data bit of the master into the slave. In some peripherals, the first bit of the slave's data is available at the slave's data out pin as soon as the slave is selected. In this format, the first SCK edge is issued a half cycle after $\overline{\text{SS}}$ has become low.

A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the shift register, depending on LSBFE bit.

After this second edge, the next bit of the SPI master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for a total of 16 edges on the SCK line, with data being latched on odd numbered edges and shifted on even numbered edges.

Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

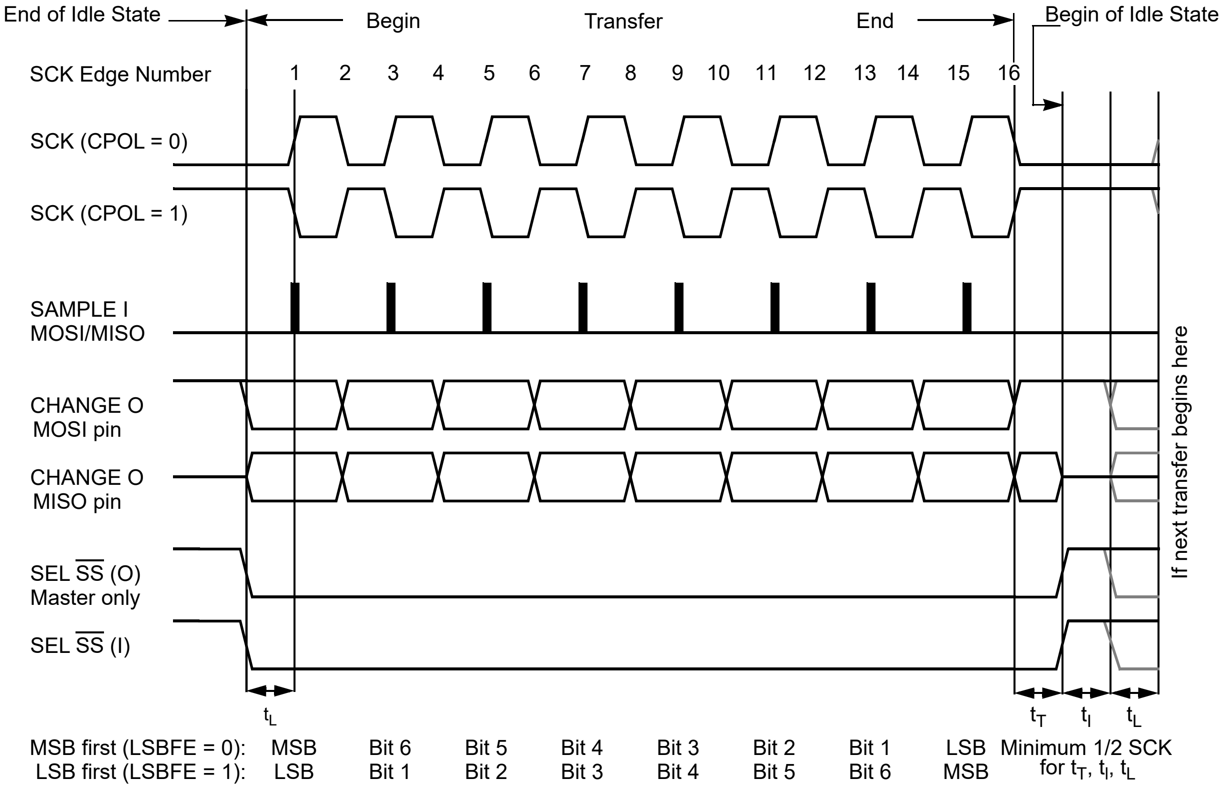
After 2n[1] (last) SCK edges:

- Data that was previously in the master SPI data register should now be in the slave data register and the data that was in the slave data register should be in the master.
- The SPIF flag in the SPI status register is set, indicating that the transfer is complete.

Figure 15-12 is a timing diagram of an SPI transfer where CPHA = 0. SCK waveforms are shown for CPOL = 0 and CPOL = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave and the MOSI signal is the output from the master. The $\overline{\text{SS}}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

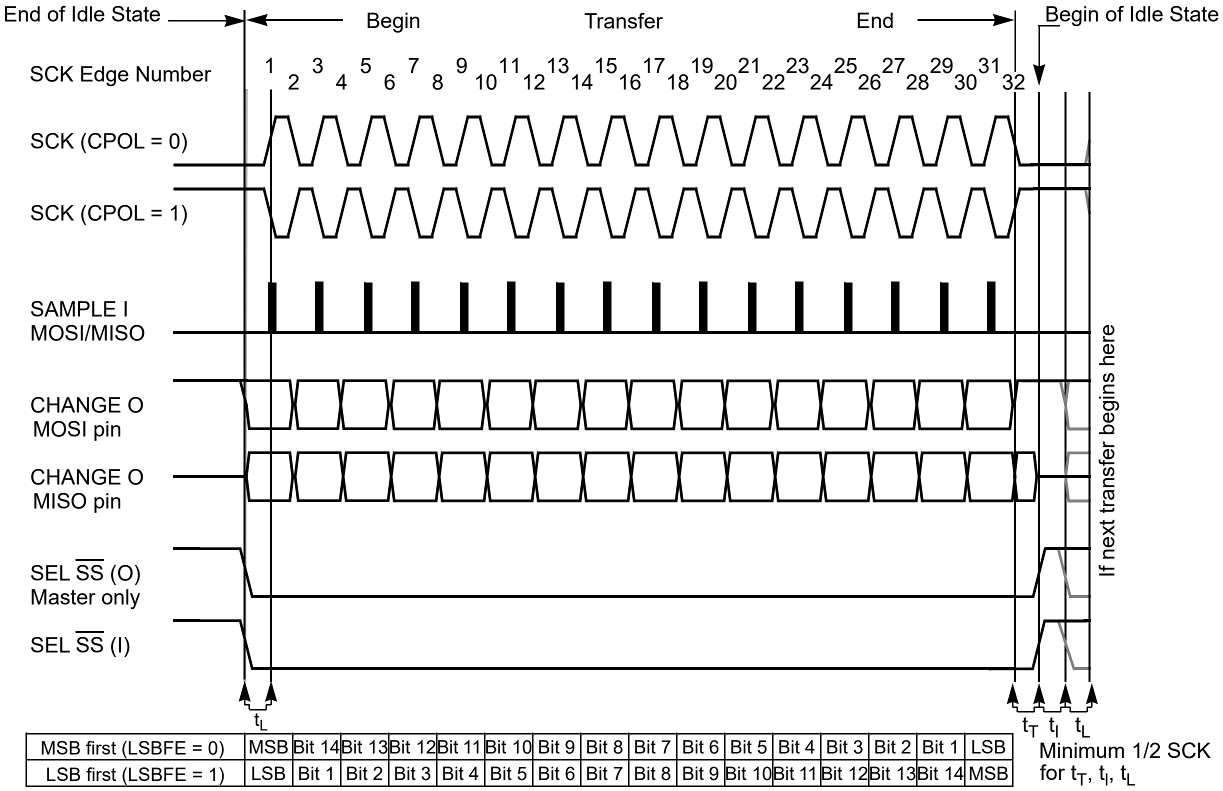
---

[1] n depends on the selected transfer width, please refer to Section 15.3.2.2, "SPI Control Register 2 (SPICR2)



t_L = Minimum leading time before the first SCK edge
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{\text{SS}}$ high time)
t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 15-12. SPI Clock Format 0 (CPHA = 0), with 8-bit Transfer Width selected (XFRW = 0)



MSB first (LSBFE = 0)	MSB	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	LSB
LSB first (LSBFE = 1)	LSB	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit 12	Bit 13	Bit 14	MSB

t_L = Minimum leading time before the first SCK edge
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time)
t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 15-13. SPI Clock Format 0 (CPHA = 0), with 16-Bit Transfer Width selected (XFRW = 1)

In slave mode, if the $\overline{SS}$ line is not deasserted between the successive transmissions then the content of the SPI data register is not transmitted; instead the last received data is transmitted. If the $\overline{SS}$ line is deasserted for at least minimum idle time (half SCK cycle) between successive transmissions, then the content of the SPI data register is transmitted.

In master mode, with slave select output enabled the $\overline{SS}$ line is always deasserted and reasserted between successive transfers for at least minimum idle time.

15.4.3.3 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin, the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the n[1]-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master.

A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

---

[1] n depends on the selected transfer width, please refer to Section 15.3.2.2, “SPI Control Register 2 (SPICR2)

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the SPI shift register, depending on LSBFE bit. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

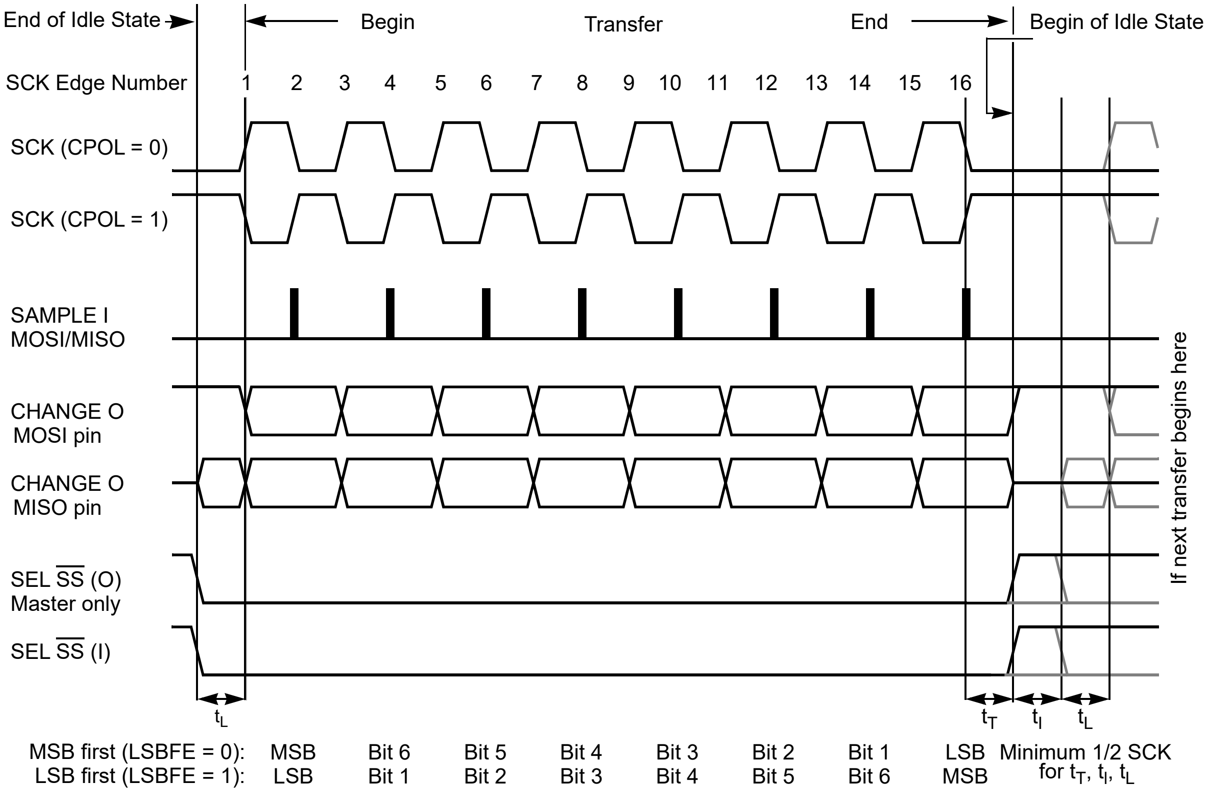
This process continues for a total of n[1] edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

Data reception is double buffered, data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After 2n[1] SCK edges:

- Data that was previously in the SPI data register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 15-14 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

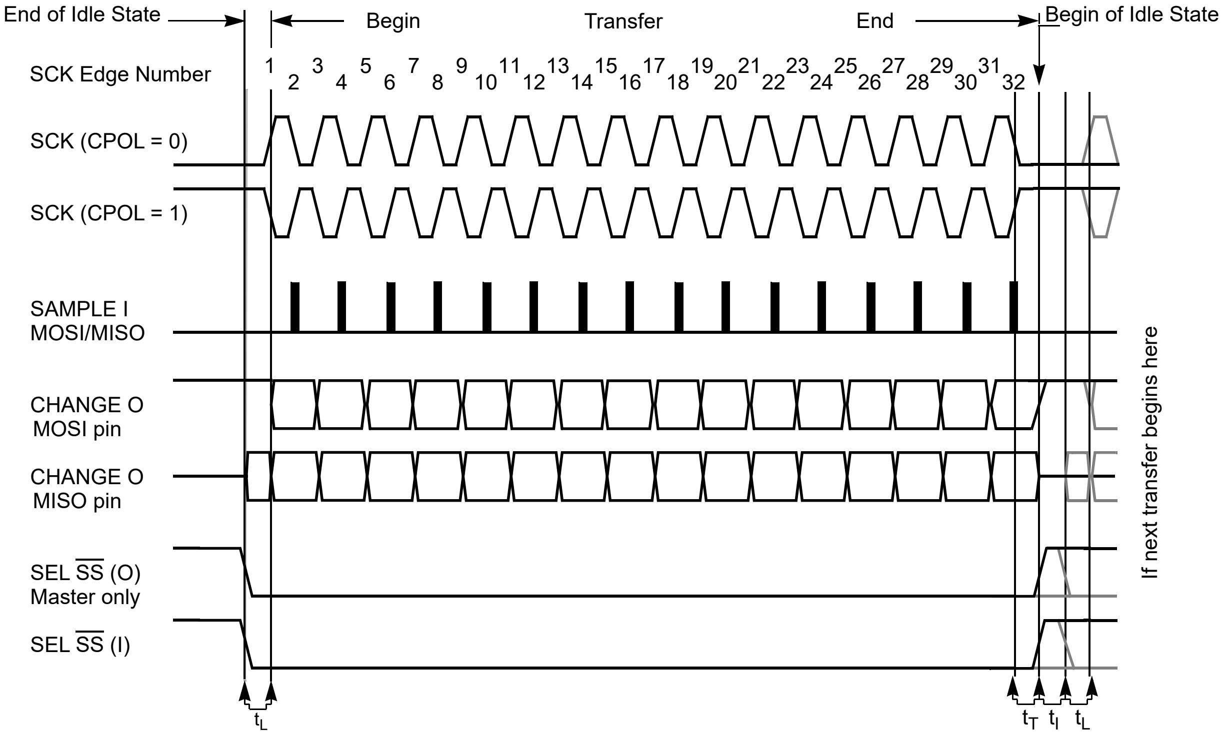


t_L = Minimum leading time before the first SCK edge, not required for back-to-back transfers

t_T = Minimum trailing time after the last SCK edge

t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time), not required for back-to-back transfers

Figure 15-14. SPI Clock Format 1 (CPHA = 1), with 8-Bit Transfer Width selected (XFRW = 0)



MSB first (LSBFE = 0)	MSB	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	LSB
LSB first (LSBFE = 1)	LSB	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit 12	Bit 13	Bit 14	MSB

Minimum 1/2 SCK for t_T, t_I, t_L

t_L = Minimum leading time before the first SCK edge, not required for back-to-back transfers
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{\text{SS}}$ high time), not required for back-to-back transfers

Figure 15-15. SPI Clock Format 1 (CPHA = 1), with 16-Bit Transfer Width selected (XFRW = 1)

The $\overline{\text{SS}}$ line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

- Back-to-back transfers in master mode

  In master mode, if a transmission has completed and new data is available in the SPI data register, this data is sent out immediately without a trailing and minimum idle time.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set one half SCK cycle after the last SCK edge.

15.4.4 SPI Baud Rate Generation

Baud rate generation consists of a series of divider stages. Six bits in the SPI baud rate register (SPPR2, SPPR1, SPPR0, SPR2, SPR1, and SPR0) determine the divisor to the SPI module clock which results in the SPI baud rate.

The SPI clock rate is determined by the product of the value in the baud rate preselection bits (SPPR2–SPPR0) and the value in the baud rate selection bits (SPR2–SPR0). The module clock divisor equation is shown in Equation 15-3.

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 15-3}$$

When all bits are clear (the default condition), the SPI module clock is divided by 2. When the selection bits (SPR2–SPR0) are 001 and the preselection bits (SPPR2–SPPR0) are 000, the module clock divisor becomes 4. When the selection bits are 010, the module clock divisor becomes 8, etc.

When the preselection bits are 001, the divisor determined by the selection bits is multiplied by 2. When the preselection bits are 010, the divisor is multiplied by 3, etc. See Table 15-6 for baud rate calculations for all bit conditions, based on a 25 MHz bus clock. The two sets of selects allows the clock to be divided by a non-power of two to achieve other baud rates such as divide by 6, divide by 10, etc.

The baud rate generator is activated only when the SPI is in master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease I_{DD} current.

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

15.4.5 Special Features

15.4.5.1 $\overline{\text{SS}}$ Output

The $\overline{\text{SS}}$ output feature automatically drives the $\overline{\text{SS}}$ pin low during transmission to select external devices and drives it high during idle to deselect external devices. When $\overline{\text{SS}}$ output is selected, the $\overline{\text{SS}}$ output pin is connected to the $\overline{\text{SS}}$ input pin of the external device.

The $\overline{\text{SS}}$ output is available only in master mode during normal SPI operation by asserting SSOE and MODFEN bit as shown in Table 15-2.

The mode fault feature is disabled while $\overline{\text{SS}}$ output is enabled.

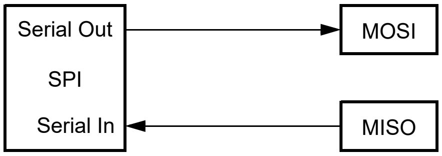
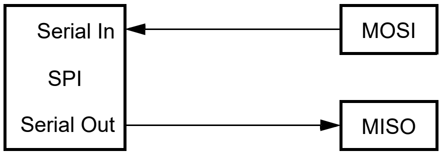
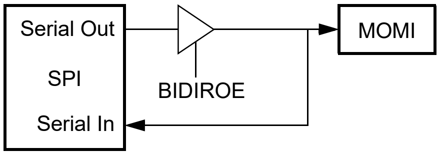
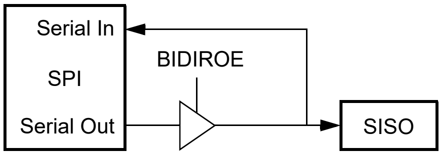
NOTE

Care must be taken when using the $\overline{\text{SS}}$ output feature in a multimaster system because the mode fault feature is not available for detecting system errors between masters.

15.4.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see Table 15-10). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 15-10. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode **SPC0 = 0**		
Bidirectional Mode **SPC0 = 1**		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

- The SCK is output for the master mode and input for the slave mode.
- The $\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.
- The bidirectional mode does not affect SCK and $\overline{SS}$ functions.

NOTE

In bidirectional master mode, with mode fault enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode. In this case MISO becomes occupied by the SPI and MOSI is not used. This must be considered, if the MISO pin is used for another purpose.

15.4.6 Error Conditions

The SPI has one error condition:

- Mode fault error

15.4.6.1 Mode Fault Error

If the $\overline{SS}$ input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. This condition is not permitted in normal operation, the MODF bit in the SPI status register is set automatically, provided the MODFEN bit is set.

In the special case where the SPI is in master mode and MODFEN bit is cleared, the $\overline{SS}$ pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. In case the SPI system is configured as a slave, the $\overline{\text{SS}}$ pin is a dedicated input pin. Mode fault error doesn't occur in slave mode.

If a mode fault error occurs, the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So SCK, MISO, and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for a SPI system configured in master mode, output enable of the MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to SPI control register 1. If the mode fault flag is cleared, the SPI becomes a normal master or slave again.

NOTE

If a mode fault error occurs and a received data byte is pending in the receive shift register, this data byte will be lost.

15.4.7 Low Power Mode Options

15.4.7.1 SPI in Run Mode

In run mode with the SPI system enable (SPE) bit in the SPI control register clear, the SPI system is in a low-power, disabled state. SPI registers remain accessible, but clocks to the core of this module are disabled.

15.4.7.2 SPI in Wait Mode

SPI operation in wait mode depends upon the state of the SPISWAI bit in SPI control register 2.

- If SPISWAI is clear, the SPI operates normally when the CPU is in wait mode
- If SPISWAI is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
  - If SPISWAI is set and the SPI is configured for master, any transmission and reception in progress stops at wait mode entry. The transmission and reception resumes when the SPI exits wait mode.
  - If SPISWAI is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SCK continues to be driven from the master. This keeps the slave synchronized to the master and the SCK.

    If the master transmits several bytes while the slave is in wait mode, the slave will continue to send out bytes consistent with the operation mode at the start of wait mode (i.e., if the slave is currently sending its SPIDR to the master, it will continue to send the same byte. Else if the slave is currently sending the last received byte from the master, it will continue to send each previous master byte).

NOTE

> Care must be taken when expecting data from a master while the slave is in wait or stop mode. Even though the shift register will continue to operate, the rest of the SPI is shut down (i.e., a SPIF interrupt will **not** be generated until exiting stop or wait mode). Also, the byte from the shift register will not be copied into the SPIDR register until after the slave SPI has exited wait or stop mode. In slave mode, a received byte pending in the receive shift register will be lost when entering wait or stop mode. An SPIF flag and SPIDR copy is generated only if wait mode is entered or exited during a tranmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither a SPIF nor a SPIDR copy will occur.

15.4.7.3 SPI in Stop Mode

Stop mode is dependent on the system. The SPI enters stop mode when the module clock is disabled (held high or low). If the SPI is in master mode and exchanging data when the CPU enters stop mode, the transmission is frozen until the CPU exits stop mode. After stop, data to and from the external SPI is exchanged correctly. In slave mode, the SPI will stay synchronized with the master.

The stop mode is not dependent on the SPISWAI bit.

15.4.7.4 Reset

The reset values of registers and signals are described in Section 15.3, “Memory Map and Register Definition”, which details the registers and their bit fields.

- If a data transmission occurs in slave mode after reset without a write to SPIDR, it will transmit garbage, or the data last received from the master before the reset.
- Reading from the SPIDR after reset will always read zeros.

15.4.7.5 Interrupts

The SPI only originates interrupt requests when SPI is enabled (SPE bit in SPICR1 set). The following is a description of how the SPI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

The interrupt flags MODF, SPIF, and SPTEF are logically ORed to generate an interrupt request.

15.4.7.5.1 MODF

MODF occurs when the master detects an error on the $\overline{\text{SS}}$ pin. The master SPI must be configured for the MODF feature (see Table 15-2). After MODF is set, the current transfer is aborted and the following bit is changed:

- MSTR = 0, The master bit in SPICR1 resets.

The MODF interrupt is reflected in the status register MODF flag. Clearing the flag will also clear the interrupt. This interrupt will stay active while the MODF flag is set. MODF has an automatic clearing process which is described in Section 15.3.2.4, “SPI Status Register (SPISR)”.

15.4.7.5.2 SPIF

SPIF occurs when new data has been received and copied to the SPI data register. After SPIF is set, it does not clear until it is serviced. SPIF has an automatic clearing process, which is described in Section 15.3.2.4, “SPI Status Register (SPISR)”.

15.4.7.5.3 SPTEF

SPTEF occurs when the SPI data register is ready to accept new data. After SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process, which is described in Section 15.3.2.4, “SPI Status Register (SPISR)”.

Chapter 16
Inter-Integrated Circuit (IICV3) Block Description

Table 16-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.03	28 Jul 2006	16.7.1.7/16-525	- Update flow-chart of interrupt routine for 10-bit address
V01.04	17 Nov 2006	16.3.1.2/16-505	- Revise Table1-5
V01.05	14 Aug 2007	16.3.1.1/16-505	- Backward compatible for IBAD bit name

16.1 Introduction

The inter-IC bus (IIC) is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange between devices. Being a two-wire device, the IIC bus minimizes the need for large numbers of connections between devices, and eliminates the need for an address decoder.

This bus is suitable for applications requiring occasional communications over a short distance between a number of devices. It also provides flexibility, allowing additional devices to be connected to the bus for further expansion and system development.

The interface is designed to operate up to 100 kbps with maximum bus loading and timing. The device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF.

16.1.1 Features

The IIC module has the following key features:

- Compatible with I2C bus standard
- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation

- Acknowledge bit generation/detection
- Bus busy detection
- General Call Address detection
- Compliant to ten-bit address

16.1.2 Modes of Operation

The IIC functions the same in normal, special, and emulation modes. It has two low power modes: wait and stop modes.

16.1.3 Block Diagram

The block diagram of the IIC module is shown in Figure 16-1.

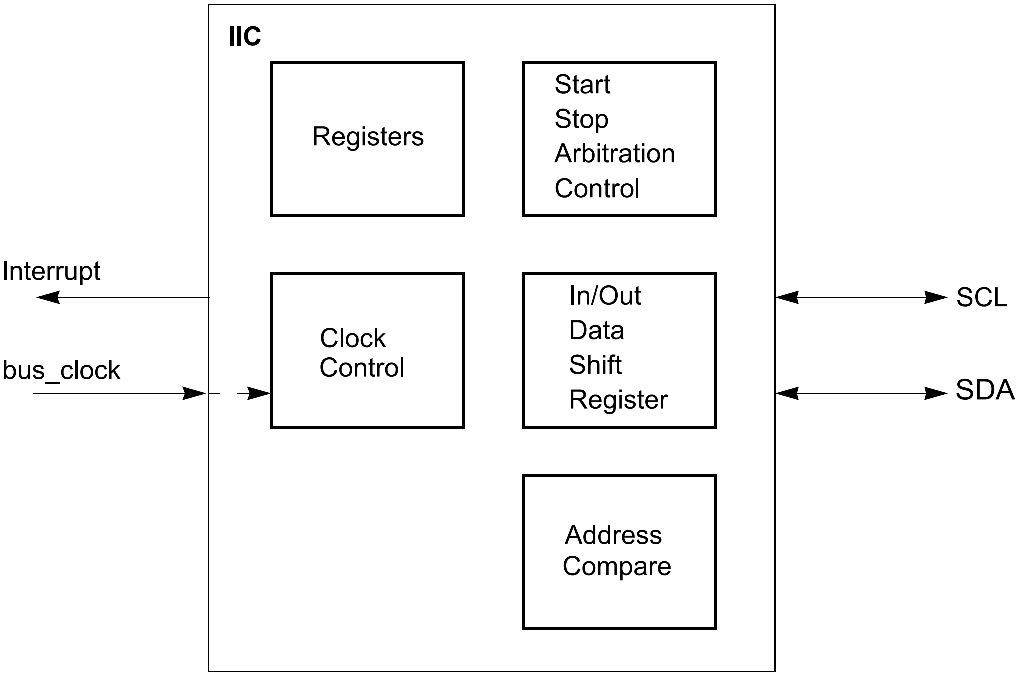


Figure 16-1. IIC Block Diagram

16.2 External Signal Description

The IICV3 module has two external pins.

16.2.1 IIC_SCL — Serial Clock Line Pin

This is the bidirectional serial clock line (SCL) of the module, compatible to the IIC bus specification.

16.2.2 IIC_SDA — Serial Data Line Pin

This is the bidirectional serial data line (SDA) of the module, compatible to the IIC bus specification.

16.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers for the IIC module.

16.3.1 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

<table>
<tr><th>Register Name</th><th></th><th>Bit 7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>Bit 0</th></tr>
<tr><td rowspan="2">0x0000
IBAD</td><td>R</td><td rowspan="2">ADR7</td><td rowspan="2">ADR6</td><td rowspan="2">ADR5</td><td rowspan="2">ADR4</td><td rowspan="2">ADR3</td><td rowspan="2">ADR2</td><td rowspan="2">ADR1</td><td>0</td></tr>
<tr><td>W</td><td>▨</td></tr>
<tr><td rowspan="2">0x0001
IBFD</td><td>R</td><td rowspan="2">IBC7</td><td rowspan="2">IBC6</td><td rowspan="2">IBC5</td><td rowspan="2">IBC4</td><td rowspan="2">IBC3</td><td rowspan="2">IBC2</td><td rowspan="2">IBC1</td><td rowspan="2">IBC0</td></tr>
<tr><td>W</td></tr>
<tr><td rowspan="2">0x0002
IBCR</td><td>R</td><td rowspan="2">IBEN</td><td rowspan="2">IBIE</td><td rowspan="2">MS/$\overline{\text{SL}}$</td><td rowspan="2">Tx/$\overline{\text{Rx}}$</td><td rowspan="2">TXAK</td><td>0</td><td>0</td><td rowspan="2">IBSWAI</td></tr>
<tr><td>W</td><td>RSTA</td><td>▨</td></tr>
<tr><td rowspan="2">0x0003
IBSR</td><td>R</td><td>TCF</td><td>IAAS</td><td>IBB</td><td rowspan="2">IBAL</td><td>0</td><td>SRW</td><td rowspan="2">IBIF</td><td>RXAK</td></tr>
<tr><td>W</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td rowspan="2">0x0004
IBDR</td><td>R</td><td rowspan="2">D7</td><td rowspan="2">D6</td><td rowspan="2">D5</td><td rowspan="2">D4</td><td rowspan="2">D3</td><td rowspan="2">D2</td><td rowspan="2">D1</td><td rowspan="2">D0</td></tr>
<tr><td>W</td></tr>
<tr><td rowspan="2">0x0005
IBCR2</td><td>R</td><td rowspan="2">GCEN</td><td rowspan="2">ADTYPE</td><td>0</td><td>0</td><td>0</td><td rowspan="2">ADR10</td><td rowspan="2">ADR9</td><td rowspan="2">ADR8</td></tr>
<tr><td>W</td><td>▨</td><td>▨</td><td>▨</td></tr>
</table>

▨ = Unimplemented or Reserved

Figure 16-2. IIC Register Summary

16.3.1.1 IIC Address Register (IBAD)

Module Base +0x0000

	7	6	5	4	3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
W								▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 16-3. IIC Bus Address Register (IBAD)

Read and write anytime

This register contains the address the IIC bus will respond to when addressed as a slave; note that it is not the address sent on the bus during the address transfer.

Table 16-2. IBAD Field Descriptions

Field	Description
7:1 ADR[7:1]	**Slave Address** — Bit 1 to bit 7 contain the specific slave address to be used by the IIC bus module.The default mode of IIC bus is slave mode for an address match on the bus.
0 Reserved	Reserved — Bit 0 of the IBAD is reserved for future compatibility. This bit will always read 0.

16.3.1.2 IIC Frequency Divider Register (IBFD)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
W								
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 16-4. IIC Bus Frequency Divider Register (IBFD)

Read and write anytime

Table 16-3. IBFD Field Descriptions

Field	Description
7:0 IBC[7:0]	**I Bus Clock Rate 7:0** — This field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider — IBC7:6, prescaled shift register — IBC5:3 select the prescaler divider and IBC2-0 select the shift register tap point. The IBC bits are decoded to give the tap and prescale values as shown in Table 16-4.

Table 16-4. I-Bus Tap and Prescale Values

IBC2-0 (bin)	SCL Tap (clocks)	SDA Tap (clocks)
000	5	1
001	6	1
010	7	2
011	8	2
100	9	3
101	10	3
110	12	4
111	15	4

Table 16-5. Prescale Divider Encoding

IBC5-3 (bin)	scl2start (clocks)	scl2stop (clocks)	scl2tap (clocks)	tap2tap (clocks)
000	2	7	4	1
001	2	7	4	2
010	2	9	6	4
011	6	9	6	8
100	14	17	14	16
101	30	33	30	32
110	62	65	62	64
111	126	129	126	128

Table 16-6. Multiplier Factor

IBC7-6	MUL
00	01
01	02
10	04
11	RESERVED

The number of clocks from the falling edge of SCL to the first tap (Tap[1]) is defined by the values shown in the scl2tap column of Table 16-4, all subsequent tap points are separated by 2^{IBC5-3} as shown in the tap2tap column in Table 16-5. The SCL Tap is used to generated the SCL period and the SDA Tap is used to determine the delay from the falling edge of SCL to SDA changing, the SDA hold time.

IBC7–6 defines the multiplier factor MUL. The values of MUL are shown in the Table 16-6.

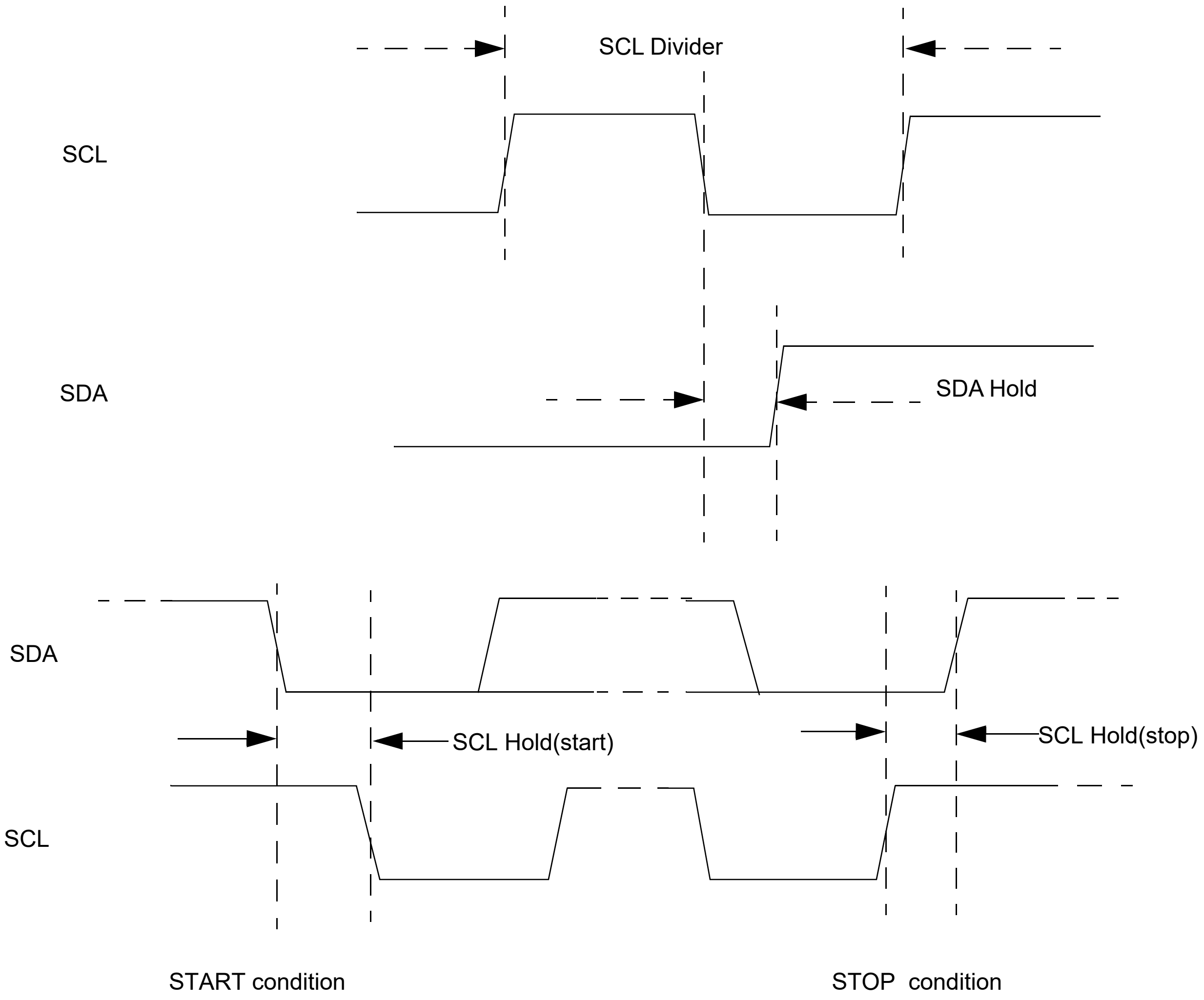


Figure 16-5. SCL Divider and SDA Hold

The equation used to generate the divider values from the IBFD bits is:

SCL Divider = MUL x {2 x (scl2tap + [(SCL_Tap -1) x tap2tap] + 2)}

The SDA hold delay is equal to the CPU clock period multiplied by the SDA Hold value shown in Table 16-7. The equation used to generate the SDA Hold value from the IBFD bits is:

SDA Hold = MUL x {scl2tap + [(SDA_Tap - 1) x tap2tap] + 3}

The equation for SCL Hold values to generate the start and stop conditions from the IBFD bits is:

SCL Hold(start) = MUL x [scl2start + (SCL_Tap - 1) x tap2tap]

SCL Hold(stop) = MUL x [scl2stop + (SCL_Tap - 1) x tap2tap]

Table 16-7. IIC Divider and Hold Values (Sheet 1 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
MUL=1				

Table 16-7. IIC Divider and Hold Values (Sheet 2 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
00	20/22	7	6	11
01	22/24	7	7	12
02	24/26	8	8	13
03	26/28	8	9	14
04	28/30	9	10	15
05	30/32	9	11	16
06	34/36	10	13	18
07	40/42	10	16	21
08	28/32	7	10	15
09	32/36	7	12	17
0A	36/40	9	14	19
0B	40/44	9	16	21
0C	44/48	11	18	23
0D	48/52	11	20	25
0E	56/60	13	24	29
0F	68/72	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289

Table 16-7. IIC Divider and Hold Values (Sheet 3 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
MUL=2				
40	40	14	12	22
41	44	14	14	24
42	48	16	16	26
43	52	16	18	28
44	56	18	20	30
45	60	18	22	32
46	68	20	26	36
47	80	20	32	42
48	56	14	20	30
49	64	14	24	34
4A	72	18	28	38
4B	80	18	32	42
4C	88	22	36	46
4D	96	22	40	50
4E	112	26	48	58
4F	136	26	60	70
50	96	18	36	50
51	112	18	44	58
52	128	26	52	66
53	144	26	60	74
54	160	34	68	82
55	176	34	76	90
56	208	42	92	106
57	256	42	116	130
58	160	18	76	82

Table 16-7. IIC Divider and Hold Values (Sheet 4 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
59	192	18	92	98
5A	224	34	108	114
5B	256	34	124	130
5C	288	50	140	146
5D	320	50	156	162
5E	384	66	188	194
5F	480	66	236	242
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258
64	576	98	284	290
65	640	98	316	322
66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074
7F	7680	1026	3836	3842
MUL=4				
80	72	28	24	44
81	80	28	28	48
82	88	32	32	52
83	96	32	36	56
84	104	36	40	60

Table 16-7. IIC Divider and Hold Values (Sheet 5 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
85	112	36	44	64
86	128	40	52	72
87	152	40	64	84
88	112	28	40	60
89	128	28	48	68
8A	144	36	56	76
8B	160	36	64	84
8C	176	44	72	92
8D	192	44	80	100
8E	224	52	96	116
8F	272	52	120	140
90	192	36	72	100
91	224	36	88	116
92	256	52	104	132
93	288	52	120	148
94	320	68	136	164
95	352	68	152	180
96	416	84	184	212
97	512	84	232	260
98	320	36	152	164
99	384	36	184	196
9A	448	68	216	228
9B	512	68	248	260
9C	576	100	280	292
9D	640	100	312	324
9E	768	132	376	388
9F	960	132	472	484
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964
A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284
B1	3072	260	1528	1540

Table 16-7. IIC Divider and Hold Values (Sheet 6 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

Note:Since the bus frequency is speeding up,the SCL Divider could be expanded by it.Therefore,in the table,when IBC[7:0] is from $00 to $0F,the SCL Divider is revised by the format value1/value2.Value1 is the divider under the low frequency.Value2 is the divider under the high frequency.How to select the divider depends on the bus frequency.When IBC[7:0] is from $10 to $BF,the divider is not changed.

16.3.1.3 IIC Control Register (IBCR)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	IBEN	IBIE	MS/SL	Tx/Rx	TXAK	0	0	IBSWAI
W						RSTA	▨	
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 16-6. IIC Bus Control Register (IBCR)

Read and write anytime

Table 16-8. IBCR Field Descriptions

Field	Description
7 IBEN	**I-Bus Enable** — This bit controls the software reset of the entire IIC bus module. 0 The module is reset and disabled. This is the power-on reset situation. When low the interface is held in reset but registers can be accessed 1 The IIC bus module is enabled.This bit must be set before any other IBCR bits have any effect If the IIC bus module is enabled in the middle of a byte transfer the interface behaves as follows: slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected. Master mode will not be aware that the bus is busy, hence if a start cycle is initiated then the current bus cycle may become corrupt. This would ultimately result in either the current bus master or the IIC bus module losing arbitration, after which bus operation would return to normal.
6 IBIE	**I-Bus Interrupt Enable** 0 Interrupts from the IIC bus module are disabled. Note that this does not clear any currently pending interrupt condition 1 Interrupts from the IIC bus module are enabled. An IIC bus interrupt occurs provided the IBIF bit in the status register is also set.
5 MS/SL	**Master/Slave Mode Select Bit** — Upon reset, this bit is cleared. When this bit is changed from 0 to 1, a START signal is generated on the bus, and the master mode is selected. When this bit is changed from 1 to 0, a STOP signal is generated and the operation mode changes from master to slave.A STOP signal should only be generated if the IBIF flag is set. MS/$\overline{\text{SL}}$ is cleared without generating a STOP signal when the master loses arbitration. 0 Slave Mode 1 Master Mode
4 Tx/Rx	**Transmit/Receive Mode Select Bit** — This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the status register. In master mode this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high. 0 Receive 1 Transmit
3 TXAK	**Transmit Acknowledge Enable** — This bit specifies the value driven onto SDA during data acknowledge cycles for both master and slave receivers. The IIC module will always acknowledge address matches, provided it is enabled, regardless of the value of TXAK. Note that values written to this bit are only used when the IIC bus is a receiver, not a transmitter. 0 An acknowledge signal will be sent out to the bus at the 9th clock bit after receiving one byte data 1 No acknowledge signal response is sent (i.e., acknowledge bit = 1)
2 RSTA	**Repeat Start** — Writing a 1 to this bit will generate a repeated START condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration. 1 Generate repeat start cycle
1 RESERVED	**Reserved** — Bit 1 of the IBCR is reserved for future compatibility. This bit will always read 0.
0 IBSWAI	**I Bus Interface Stop in Wait Mode** 0 IIC bus module clock operates normally 1 Halt IIC bus module clock generation in wait mode

Wait mode is entered via execution of a CPU WAI instruction. In the event that the IBSWAI bit is set, all clocks internal to the IIC will be stopped and any transmission currently in progress will halt.If the CPU were woken up by a source other than the IIC module, then clocks would restart and the IIC would resume from where was during the previous transmission. It is not possible for the IIC to wake up the CPU when its internal clocks are stopped.

If it were the case that the IBSWAI bit was cleared when the WAI instruction was executed, the IIC internal clocks and interface would remain alive, continuing the operation which was currently underway. It is also possible to configure the IIC such that it will wake up the CPU via an interrupt at the conclusion of the current operation. See the discussion on the IBIF and IBIE bits in the IBSR and IBCR, respectively.

16.3.1.4 IIC Status Register (IBSR)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
W	▨	▨	▨		▨	▨		▨
Reset	1	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 16-7. IIC Bus Status Register (IBSR)

This status register is read-only with exception of bit 1 (IBIF) and bit 4 (IBAL), which are software clearable.

Table 16-9. IBSR Field Descriptions

Field	Description
7 TCF	**Data Transferring Bit** — While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the 9th clock of a byte transfer. Note that this bit is only valid during or immediately following a transfer to the IIC module or from the IIC module. 0 Transfer in progress 1 Transfer complete
6 IAAS	**Addressed as a Slave Bit** — When its own specific address (I-bus address register) is matched with the calling address or it receives the general call address with GCEN== 1,this bit is set.The CPU is interrupted provided the IBIE is set.Then the CPU needs to check the SRW bit and set its Tx/$\overline{\text{Rx}}$ mode accordingly.Writing to the I-bus control register clears this bit. 0 Not addressed 1 Addressed as a slave
5 IBB	Bus Busy Bit 0 This bit indicates the status of the bus. When a START signal is detected, the IBB is set. If a STOP signal is detected, IBB is cleared and the bus enters idle state. 1 Bus is busy
4 IBAL	**Arbitration Lost** — The arbitration lost bit (IBAL) is set by hardware when the arbitration procedure is lost. Arbitration is lost in the following circumstances: 1. SDA sampled low when the master drives a high during an address or data transmit cycle. 2. SDA sampled low when the master drives a high during the acknowledge bit of a data receive cycle. 3. A start cycle is attempted when the bus is busy. 4. A repeated start cycle is requested in slave mode. 5. A stop condition is detected when the master did not request it. This bit must be cleared by software, by writing a one to it. A write of 0 has no effect on this bit.

Table 16-9. IBSR Field Descriptions (continued)

Field	Description
3 RESERVED	**Reserved** — Bit 3 of IBSR is reserved for future use. A read operation on this bit will return 0.
2 SRW	**Slave Read/Write** — When IAAS is set this bit indicates the value of the R/W command bit of the calling address sent from the master This bit is only valid when the I-bus is in slave mode, a complete address transfer has occurred with an address match and no other transfers have been initiated. Checking this bit, the CPU can select slave transmit/receive mode according to the command of the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IBIF	**I-Bus Interrupt** — The IBIF bit is set when one of the following conditions occurs: — Arbitration lost (IBAL bit set) — Data transfer complete (TCF bit set) — Addressed as slave (IAAS bit set) It will cause a processor interrupt request if the IBIE bit is set. This bit must be cleared by software, writing a one to it. A write of 0 has no effect on this bit.
0 RXAK	**Received Acknowledge** — The value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal has been received after the completion of 8 bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the 9th clock. 0 Acknowledge received 1 No acknowledge received

16.3.1.5 IIC Data I/O Register (IBDR)

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0

Figure 16-8. IIC Bus Data I/O Register (IBDR)

In master transmit mode, when data is written to the IBDR a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates next byte data receiving. In slave mode, the same functions are available after an address match has occurred.Note that the Tx/Rx bit in the IBCR must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the IIC is configured for master transmit but a master receive is desired, then reading the IBDR will not initiate the receive.

Reading the IBDR will return the last byte received while the IIC is configured in either master receive or slave receive modes. The IBDR does not reflect every byte that is transmitted on the IIC bus, nor can software verify that a byte has been written to the IBDR correctly by reading it back.

In master transmit mode, the first byte of data written to IBDR following assertion of MS/$\overline{\text{SL}}$ is used for the address transfer and should com.prise of the calling address (in position D7:D1) concatenated with the required R/$\overline{\text{W}}$ bit (in position D0).

16.3.1.6 IIC Control Register 2(IBCR2)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	GCEN	ADTYPE	0	0	0	ADR10	ADR9	ADR8
W								
Reset	0	0	0	0	0	0	0	0

Figure 16-9. IIC Bus Control Register 2(IBCR2)

This register contains the variables used in general call and in ten-bit address.

Read and write anytime

Table 16-10. IBCR2 Field Descriptions

Field	Description
7 GCEN	**General Call Enable**. 0 General call is disabled. The module dont receive any general call data and address. 1 enable general call. It indicates that the module can receive address and any data.
6 ADTYPE	**Address Type—** This bit selects the address length. The variable must be configured correctly before IIC enters slave mode. 0 7-bit address 1 10-bit address
5,4,3 RESERVED	**Reserved** — Bit 5,4 and 3 of the IBCR2 are reserved for future compatibility. These bits will always read 0.
2:0 ADR[10:8]	**Slave Address [10:8]** —These 3 bits represent the MSB of the 10-bit address when address type is asserted (ADTYPE = 1).

16.4 Functional Description

This section provides a complete functional description of the IICV3.

16.4.1 I-Bus Protocol

The IIC bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open drain or open collector outputs. Logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

Normally, a standard communication is composed of four parts: START signal, slave address transmission, data transfer and STOP signal. They are described briefly in the following sections and illustrated in Figure 16-10.

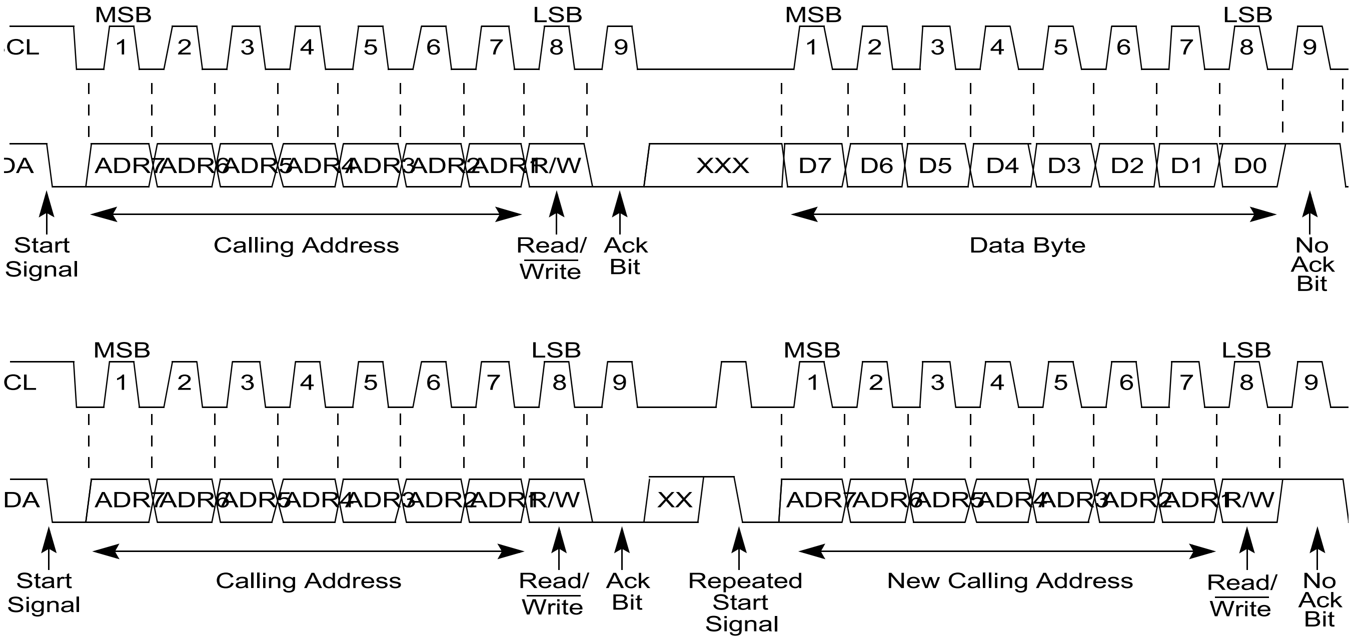


Figure 16-10. IIC-Bus Transmission Signals

16.4.1.1 START Signal

When the bus is free, i.e. no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START signal.As shown in Figure 16-10, a START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

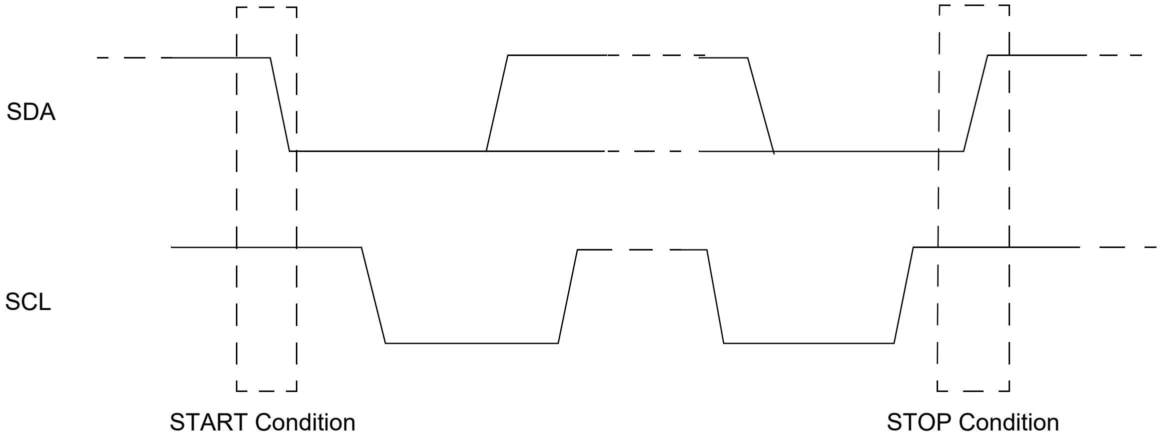


Figure 16-11. Start and Stop Conditions

16.4.1.2 Slave Address Transmission

The first byte of data transfer immediately after the START signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/W bit. The R/W bit tells the slave the desired direction of data transfer.

1 = Read transfer, the slave transmits data to the master.
0 = Write transfer, the master transmits data to the slave.

If the calling address is 10-bit, another byte is followed by the first byte.Only the slave with a calling address that matches the one transmitted by the master will respond by sending back an acknowledge bit. This is done by pulling the SDA low at the 9th clock (see Figure 16-10).

No two slaves in the system may have the same address. If the IIC bus is master, it must not transmit an address that is equal to its own slave address. The IIC bus cannot be master and slave at the same time.However, if arbitration is lost during an address cycle the IIC bus will revert to slave mode and operate correctly even if it is being addressed by another master.

16.4.1.3 Data Transfer

As soon as successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/W bit sent by the calling master

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in Figure 16-10. There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte has to be followed by an acknowledge bit, which is signalled from the receiving device by pulling the SDA low at the ninth clock. So one complete data byte transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a stop signal to abort the data transfer or a start signal (repeated start) to commence a new calling.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means 'end of data' to the slave, so the slave releases the SDA line for the master to generate STOP or START signal.Note in order to release the bus correctly,after no-acknowledge to the master,the slave must be immediately switched to receiver and a following dummy reading of the IBDR is necessary.

16.4.1.4 STOP Signal

The master can terminate the communication by generating a STOP signal to free the bus. However, the master may generate a START signal followed by a calling command without generating a STOP signal first. This is called repeated START. A STOP signal is defined as a low-to-high transition of SDA while SCL at logical 1 (see Figure 16-10).

The master can generate a STOP even if the slave has generated an acknowledge at which point the slave must release the bus.

16.4.1.5 Repeated START Signal

As shown in Figure 16-10, a repeated START signal is a START signal generated without first generating a STOP signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (transmit/receive mode) without releasing the bus.

16.4.1.6 Arbitration Procedure

The Inter-IC bus is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure, a bus master loses arbitration if it transmits logic 1 while another master transmits logic 0. The losing masters immediately switch over to slave receive mode and stop driving SDA output. In this case the transition from master to slave mode does not generate a STOP condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

16.4.1.7 Clock Synchronization

Because wire-AND logic is performed on SCL line, a high-to-low transition on SCL line affects all the devices connected on the bus. The devices start counting their low period and as soon as a device's clock has gone low, it holds the SCL line low until the clock high state is reached.However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see Figure 16-11). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods.The first device to complete its high period pulls the SCL line low again.

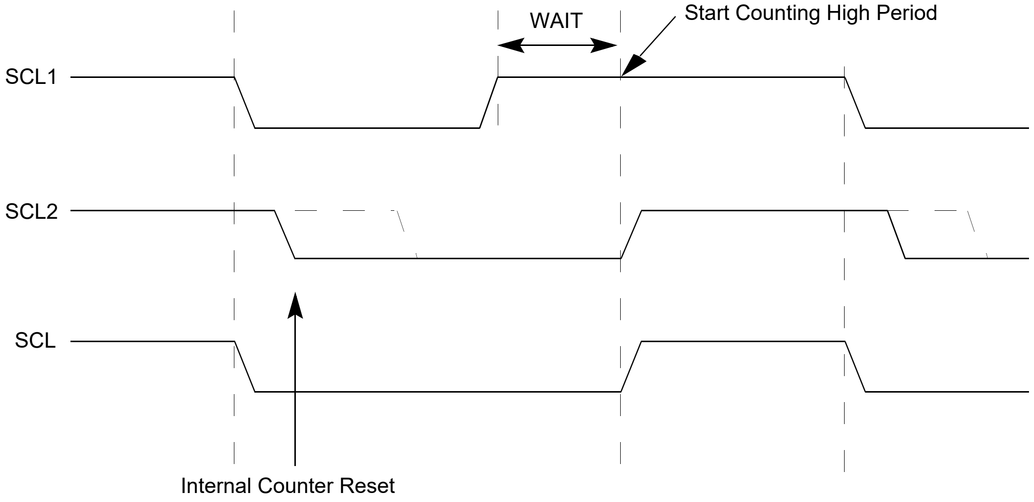


Figure 16-12. IIC-Bus Clock Synchronization

16.4.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (9 bits). In such case, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

16.4.1.9 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low the slave can drive SCL low for the required period and then release it.If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

16.4.1.10 Ten-bit Address

A ten-bit address is indicated if the first 5 bits of the first address byte are 0x11110. The following rules apply to the first address byte.

SLAVE ADDRESS	R/W BIT	DESCRIPTION
0000000	0	General call address
0000010	x	Reserved for different bus format
0000011	x	Reserved for future purposes
11111XX	x	Reserved for future purposes
11110XX	x	10-bit slave addressing

Figure 16-13. **Definition of bits in the first byte.**

The address type is identified by ADTYPE. When ADTYPE is 0, 7-bit address is applied. Reversely, the address is 10-bit address.Generally, there are two cases of 10-bit address.See the Figure 16-14 and Figure 16-15.

S	Slave Add1st 7bits 11110+ADR10+ADR9	R/W 0	A1	Slave Add 2nd byte ADR[8:1]	A2	Data	A3

Figure 16-14. A master-transmitter addresses a slave-receiver with a 10-bit address

S	Slave Add1st 7bits 11110+ADR10+ADR9	R/W 0	A1	Slave Add 2nd byte ADR[8:1]	A2	Sr	Slave Add 1st 7bits 11110+ADR10+ADR9	R/W 1	A3	Data	A4

Figure 16-15. A master-receiver addresses a slave-transmitter with a 10-bit address.

In the Figure 16-15, the first two bytes are the similar to Figure 16-14. After the repeated START(Sr),the first slave address is transmitted again, but the R/W is 1, meaning that the slave is acted as a transmitter.

16.4.1.11 General Call Address

To broadcast using a general call, a device must first generate the general call address($00), then after receiving acknowledge, it must transmit data.

In communication, as a slave device, provided the GCEN is asserted, a device acknowledges the broadcast and receives data until the GCEN is disabled or the master device releases the bus or generates a new transfer. In the broadcast, slaves always act as receivers. In general call, IAAS is also used to indicate the address match.

In order to distinguish whether the address match is the normal address match or the general call address match, IBDR should be read after the address byte has been received. If the data is $00, the match is general call address match. The meaning of the general call address is always specified in the first data byte and must be dealt with by S/W, the IIC hardware does not decode and process the first data byte.

When one byte transfer is done, the received data can be read from IBDR. The user can control the procedure by enabling or disabling GCEN.

16.4.2 Operation in Run Mode

This is the basic mode of operation.

16.4.3 Operation in Wait Mode

IIC operation in wait mode can be configured. Depending on the state of internal bits, the IIC can operate normally when the CPU is in wait mode or the IIC clock generation can be turned off and the IIC module enters a power conservation state during wait mode. In the later case, any transmission or reception in progress stops at wait mode entry.

16.4.4 Operation in Stop Mode

The IIC is inactive in stop mode for reduced power consumption. The STOP instruction does not affect IIC register states.

16.5 Resets

The reset state of each individual bit is listed in Section 16.3, “Memory Map and Register Definition,” which details the registers and their bit-fields.

16.6 Interrupts

IICV3 uses only one interrupt vector.

Table 16-11. Interrupt Summary

Interrupt	Offset	Vector	Priority	Source	Description

IIC Interrupt	—	—	—	IBAL, TCF, IAAS bits in IBSR register	When either of IBAL, TCF or IAAS bits is set may cause an interrupt based on arbitration lost, transfer complete or address detect conditions

Internally there are three types of interrupts in IIC. The interrupt service routine can determine the interrupt type by reading the status register.

IIC Interrupt can be generated on

1. Arbitration lost condition (IBAL bit set)
2. Byte transfer condition (TCF bit set)
3. Address detect condition (IAAS bit set)

The IIC interrupt is enabled by the IBIE bit in the IIC control register. It must be cleared by writing 0 to the IBF bit in the interrupt service routine.

16.7 Application Information

16.7.1 IIC Programming Examples

16.7.1.1 Initialization Sequence

Reset will put the IIC bus control register to its default status. Before the interface can be used to transfer serial data, an initialization procedure must be carried out, as follows:

1. Update the frequency divider register (IBFD) and select the required division ratio to obtain SCL frequency from system clock.
2. Update the ADTYPE of IBCR2 to define the address length, 7 bits or 10 bits.
3. Update the IIC bus address register (IBAD) to define its slave address. If 10-bit address is applied IBCR2 should be updated to define the rest bits of address.
4. Set the IBEN bit of the IIC bus control register (IBCR) to enable the IIC interface system.
5. Modify the bits of the IIC bus control register (IBCR) to select master/slave mode, transmit/receive mode and interrupt enable or not.
6. If supported general call, the GCEN in IBCR2 should be asserted.

16.7.1.2 Generation of START

After completion of the initialization procedure, serial data can be transmitted by selecting the 'master transmitter' mode. If the device is connected to a multi-master bus system, the state of the IIC bus busy bit (IBB) must be tested to check whether the serial bus is free.

If the bus is free (IBB=0), the start condition and the first byte (the slave address) can be sent. The data written to the data register comprises the slave calling address and the LSB set to indicate the direction of transfer required from the slave.

The bus free time (i.e., the time between a STOP condition and the following START condition) is built into the hardware that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period it may be necessary to wait until the IIC is busy after writing the calling address to the IBDR before proceeding with the following instructions. This is illustrated in the following example.

An example of a program which generates the START signal and transmits the first byte of data (slave address) is shown below:

```
CHFLAG     BRSET   IBSR,#$20,*       ;WAIT FOR IBB FLAG TO CLEAR
TXSTART    BSET    IBCR,#$30         ;SET TRANSMIT AND MASTER MODE;i.e. GENERATE START CONDITION
           MOVB    CALLING,IBDR      ;TRANSMIT THE CALLING ADDRESS, D0=R/W
IBFREE     BRCLR   IBSR,#$20,*       ;WAIT FOR IBB FLAG TO SET
```

16.7.1.3 Post-Transfer Software Response

Transmission or reception of a byte will set the data transferring bit (TCF) to 1, which indicates one byte communication is finished. The IIC bus interrupt bit (IBIF) is set also; an interrupt will be generated if the interrupt function is enabled during initialization by setting the IBIE bit. Software must clear the IBIF bit in the interrupt routine first. The TCF bit will be cleared by reading from the IIC bus data I/O register (IBDR) in receive mode or writing to IBDR in transmit mode.

Software may service the IIC I/O in the main program by monitoring the IBIF bit if the interrupt function is disabled. Note that polling should monitor the IBIF bit rather than the TCF bit because their operation is different when arbitration is lost.

Note that when an interrupt occurs at the end of the address cycle the master will always be in transmit mode, i.e. the address is transmitted. If master receive mode is required, indicated by R/W bit in IBDR, then the Tx/Rx bit should be toggled at this stage.

During slave mode address cycles (IAAS=1), the SRW bit in the status register is read to determine the direction of the subsequent transfer and the Tx/Rx bit is programmed accordingly.For slave mode data cycles (IAAS=0) the SRW bit is not valid, the Tx/Rx bit in the control register should be read to determine the direction of the current transfer.

The following is an example of a software response by a 'master transmitter' in the interrupt routine.

```
ISR        BCLR     IBSR,#$02             ;CLEAR THE IBIF FLAG
           BRCLR    IBCR,#$20,SLAVE       ;BRANCH IF IN SLAVE MODE
           BRCLR    IBCR,#$10,RECEIVE     ;BRANCH IF IN RECEIVE MODE
           BRSET    IBSR,#$01,END         ;IF NO ACK, END OF TRANSMISSION
TRANSMIT   MOVB     DATABUF,IBDR          ;TRANSMIT NEXT BYTE OF DATA
```

16.7.1.4 Generation of STOP

A data transfer ends with a STOP signal generated by the 'master' device. A master transmitter can simply generate a STOP signal after all the data has been transmitted. The following is an example showing how a stop condition is generated by a master transmitter.

```
MASTX     TST     TXCNT              ;GET VALUE FROM THE TRANSMITING COUNTER
          BEQ     END                ;END IF NO MORE DATA
          BRSET   IBSR,#$01,END      ;END IF NO ACK
          MOVB    DATABUF,IBDR       ;TRANSMIT NEXT BYTE OF DATA
          DEC     TXCNT              ;DECREASE THE TXCNT
          BRA     EMASTX             ;EXIT
END       BCLR    IBCR,#$20          ;GENERATE A STOP CONDITION
EMASTX    RTI                        ;RETURN FROM INTERRUPT
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data which can be done by setting the transmit acknowledge bit (TXAK) before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must be generated first. The following is an example showing how a STOP signal is generated by a master receiver.

```
MASR      DEC     RXCNT          ;DECREASE THE RXCNT
          BEQ     ENMASR         ;LAST BYTE TO BE READ
          MOVB    RXCNT,D1       ;CHECK SECOND LAST BYTE
          DEC     D1             ;TO BE READ
          BNE     NXMAR          ;NOT LAST OR SECOND LAST
LAMAR     BSET    IBCR,#$08      ;SECOND LAST, DISABLE ACK
                                 ;TRANSMITTING
          BRA     NXMAR
ENMASR    BCLR    IBCR,#$20      ;LAST ONE, GENERATE ‘STOP’ SIGNAL
NXMAR     MOVB    IBDR,RXBUF     ;READ DATA AND STORE
          RTI
```

16.7.1.5 Generation of Repeated START

At the end of data transfer, if the master continues to want to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

```
RESTART   BSET    IBCR,#$04      ;ANOTHER START (RESTART)
          MOVB    CALLING,IBDR   ;TRANSMIT THE CALLING ADDRESS;D0=R/W
```

16.7.1.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the transmit/receive mode select bit (Tx/Rx bit of IBCR) according to the R/W command bit (SRW). Writing to the IBCR clears the IAAS automatically. Note that the only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred, interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer may now be initiated by writing information to IBDR, for slave transmits, or dummy reading from IBDR, in slave receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the IBDR is accessed in the required mode.

In slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. Setting RXAK means an 'end of data' signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

16.7.1.7 Arbitration Lost

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices which lost arbitration are immediately switched to slave receive mode by the hardware. Their data output to the SDA line is stopped, but SCL continues to be generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with IBAL=1 and MS/SL=0. If one master attempts to start transmission while the bus is being engaged by another master, the hardware will inhibit the transmission; switch the MS/SL bit from 1 to 0 without generating STOP condition; generate an interrupt to CPU and set the IBAL to indicate that the attempt to engage the bus is failed. When considering these cases, the slave service routine should test the IBAL first and the software should clear the IBAL bit if it is set.

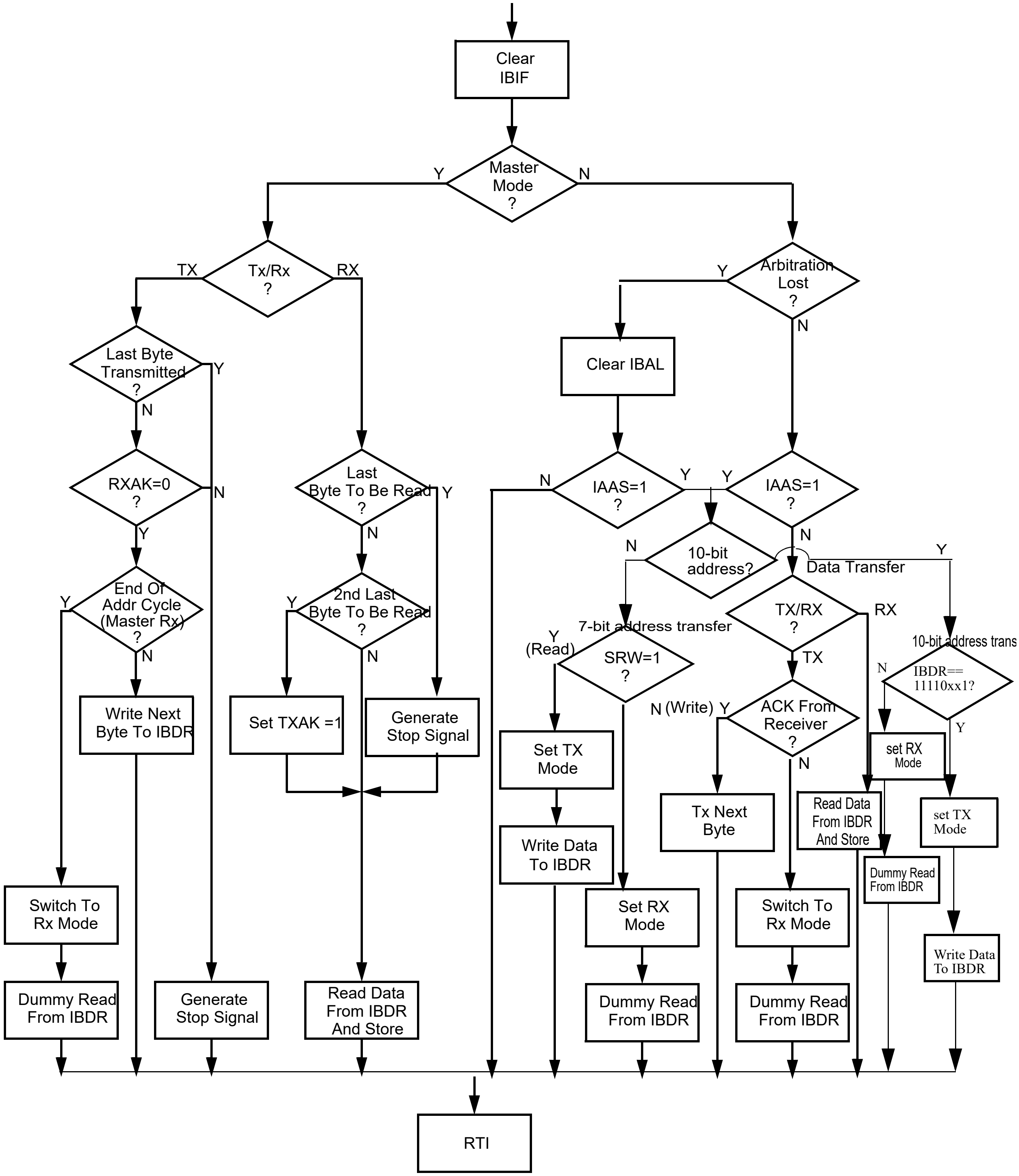


Figure 16-16. Flow-Chart of Typical IIC Interrupt Routine

Caution:When IIC is configured as 10-bit address,the point of the data array in interrupt routine must be reset after it’s addressed.

Chapter 17
CAN Physical Layer (S12CANPHYV3)

Table 17-1. Revision History Table

Revision Number	Revision Date	Sections Affected	Description of Changes
V02.00	05 Nov 2012		• Added CPTXD-dominant timeout feature
V03.00	15 Apr 2013		• Made transmit driver (CANH & CANL) independent of CPCHVL condition • Changed CPCLVL condition to disable CANL only • Added mode to cover separation of CANH and CANL drivers • Added configurable wake-up filter

17.1 Introduction

The CAN Physical Layer provides a physical layer for high speed CAN area network communication in automotive applications. It serves as an integrated interface to the CAN bus lines for the internally connected MSCAN controller through the pins CANH, CANL and SPLIT.

The CAN Physical Layer is designed to meet the CAN Physical Layer ISO 11898-2 and ISO 11898-5 standards.

17.1.1 Features

The CAN Physical Layer module includes these distinctive features:

- High speed CAN interface for baud rates of up to 1 Mbit/s
- ISO 11898-2 and ISO 11898-5 compliant for 12 V battery systems
- SPLIT pin driver for bus recessive level stabilization
- Low power mode with remote CAN wake-up handled by MSCAN module
- Configurable wake-up pulse filtering
- Over-current shutdown for CANH and CANL
- Voltage monitoring on CANH and CANL
- CPTXD-dominant timeout feature monitoring the CPTXD signal
- Fulfills the OEM “Hardware Requirements for (LIN,) CAN (and FlexRay) Interfaces in Automotive Applications” v1.3

17.1.2 Modes of Operation

There are five modes the CAN Physical Layer can take (refer to 17.5.2 for details):

1. Shutdown mode

   In shutdown mode the CAN Physical Layer is fully de-biased including the wake-up receiver.
2. Normal mode

   In normal mode the transceiver is fully biased and functional. The SPLIT pin drives 2.5 V if enabled.
3. Pseudo-normal mode
   Same as normal mode with CANL driver disabled.
4. Listen-only mode
   Same as normal mode with transmitter de-biased.
5. Standby mode with configurable wake-up feature

   In standby mode the transceiver is fully de-biased. The wake-up receiver is enabled out of reset.

- CPU Run Mode
  The CAN Physical Layer is able to operate normally in modes 1 to 4.
- CPU Wait Mode
  The CAN Physical Layer operation is the same as in CPU run mode.
- CPU Stop Mode
  The CAN Physical Layer enters standby mode when the device voltage regulator switches to reduced performance mode (“RPM”) after a CPU stop mode request.
  If enabled, the wake-up pulse filtering mechanism is activated immediately at CPU stop mode entry.

17.1.3 Block Diagram

Figure 17-1 shows a block diagram of the CAN Physical Layer. The module consists of a precision receiver, a low-power wake-up receiver, an output driver and diagnostics.

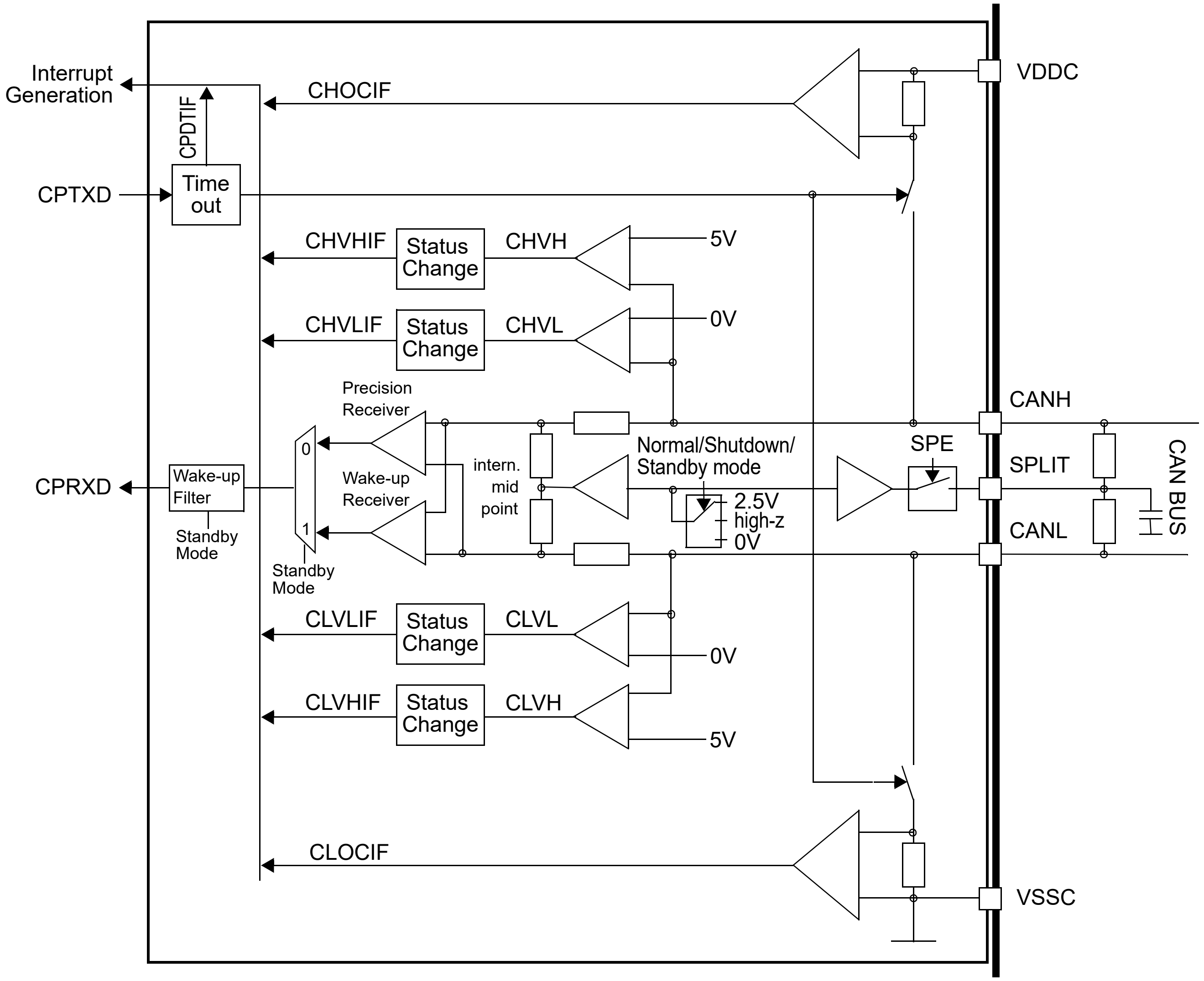


Figure 17-1. CAN Physical Layer Block Diagram

17.2 External Signal Description

Table 17-2 shows the external pins associated with the CAN Physical Layer.

Table 17-2. CAN Physical Layer Signal Properties

Name	Function
CANH	CAN Bus High Pin
SPLIT	2.5 V Termination Pin
CANL	CAN Bus Low Pin
VDDC	Supply Pin for CAN Physical Layer
VSSC	Ground Pin for CAN Physical Layer

17.2.1 CANH — CAN Bus High Pin

The CANH signal either connects directly to CAN bus high line or through an optional external common mode choke.

17.2.2 CANL — CAN Bus Low Pin

The CANL signal either connects directly to CAN bus low line or through an optional external common mode choke.

17.2.3 SPLIT — CAN Bus Termination Pin

The SPLIT pin can drive a 2.5 V bias for bus termination purpose (CAN bus middle point). Usage of this pin is optional and depends on bus termination strategy for a given bus network.

17.2.4 VDDC — Supply Pin for CAN Physical Layer

The VDDC pin is used to supply the CAN Physical Layer with 5 V from an external source.

17.2.5 VSSC — Ground Pin for CAN Physical Layer

The VSSC pin is the return path for the 5 V supply (VDDC).

17.3 Internal Signal Description

17.3.1 CPTXD — TXD Input to CAN Physical Layer

CPTXD is the input signal to the CAN Physical Layer. A logic 1 on this input is considered CAN recessive and a logic 0 as dominant level.

Per default, CPTXD is connected device-internally to the TXCAN transmitter output of the MSCAN module. For optional routing options consult the device level documentation.

17.3.2 CPRXD — RXD Output of CAN Physical Layer

CPRXD is the output signal of the CAN Physical Layer. A logic 1 on this output represents CAN recessive and a logic 0 a dominant level.

In stand-by mode the wake-up receiver is routed to this output. A dominant pulse filter can optionally be enabled to increase robustness against false wake-up pulses. In any other mode this signal defaults to the precision receiver without a pulse filter.

Per default, CPRXD is connected device-internally to the RXCAN receiver input of the MSCAN module. For optional routing options consult the device level documentation.

17.4 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the CAN Physical Layer.

17.4.1 Module Memory Map

A summary of the registers associated with the CAN Physical Layer sub-block is shown in Table 17-3. Detailed descriptions of the registers and bits are given in the following sections.

NOTE

> Register Address = Module Base Address + Address Offset, where the Module Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	CPDR	R	CPDR7	0	0	0	0	0	CPDR1	CPDR0
		W	▨	▨	▨	▨	▨	▨		▨
0x0001	CPCR	R	CPE	SPE	WUPE1-0		0	SLR2-0		
		W					▨			
0x0002	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0003	CPSR	R	CPCHVH	CPCHVL	CPCLVH	CPCLVL	CPDT	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0004	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0005	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0006	CPIE	R	0	0	0	CPVFIE	CPDTIE	0	0	CPOCIE
		W	▨	▨	▨			▨	▨	
0x0007	CPIF	R	CHVHIF	CHVLIF	CLVHIF	CLVLIF	CPDTIF	0	CHOCIF	CLOCIF
		W						▨		

▨ = Unimplemented or Reserved

Table 17-3. CAN Physical Layer Register Summary

17.4.2 Register Descriptions

This section describes all CAN Physical Layer registers and their individual bits.

17.4.2.1 Port CP Data Register (CPDR)

Module Base + 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	CPDR7	0	0	0	0	0	CPDR1	CPDR0
W								
Reset	1	0	0	0	0	0	1	1

Figure 17-2. Port CP Data Register (CPDR)

[1] Read: Anytime
Write: Anytime

Table 17-4. CPDR Register Field Descriptions

Field	Description
7 CPDR7	**Port CP Data Bit 7** Read-only bit. The synchronized CAN Physical Layer wake-up receiver output can be read at any time.
1 CPDR1	**Port CP Data Bit 1** The CAN Physical Layer CPTXD input can be directly controlled through this register bit if routed here (see device-level specification). In this case the register bit value is driven to the pin. 0 CPTXD is driven low (dominant) 1 CPTXD is driven high (recessive)
0 CPDR0	**Port CP Data Bit 0** Read-only bit. The synchronized CAN Physical Layer CPRXD output state can be read at any time.

17.4.2.2 CAN Physical Layer Control Register (CPCR)

Module Base + 0x0001

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	CPE	SPE	WUPE1-0		0	SLR2-0		
W								
Reset	0	0	1	0	0	0	0	0

Figure 17-3. CAN Physical Layer Control Register (CPCR)

[1] Read: Anytime
Write: Anytime except CPE which is set once

Table 17-5. CPCR Register Field Descriptions

Field	Description
7 CPE	**CAN Physical Layer Enable** Set once. If set to 1, the CAN Physical Layer exits shutdown mode and enters normal mode. 0 CAN Physical Layer is disabled (shutdown mode) 1 CAN Physical Layer is enabled
6 SPE	**Split Enable** If set to 1, the CAN Physical Layer SPLIT pin drives a 2.5 V bias in normal and listen-only mode. 0 SPLIT pin is high-impedance 1 SPLIT pin drives a 2.5 V bias
5-4 WUPE1-0	**Wake-Up Receiver Enable and Filter Select** If WUPE[1:0]≠0, the CAN Physical Layer wake-up receiver is enabled when not in shutdown mode. To save additional power, these bits should be set to 00, if the CAN bus is not used to wake up the device. For robustness against false wake-up an optional pulse filter can be enabled. 00 Wake-up receiver is disabled 10 Wake-up receiver is enabled, no filtering 01 Wake-up receiver is enabled, first wake-up event is masked 11 Wake-up receiver is enabled, first two wake-up events are masked
2-0 SLR2-0	**Slew Rate** The slew rate controls recessive to dominant and dominant to recessive transitions. This affects the delay time from CPTXD to the bus and from the bus to CPRXD. The loop time is thus affected by the slew rate selection. Six slew rates are available: 000 CAN Physical Layer slew rate 0 001 CAN Physical Layer slew rate 1 010 CAN Physical Layer slew rate 2 011 Reserved 100 CAN Physical Layer slew rate 4 101 CAN Physical Layer slew rate 5 110 CAN Physical Layer slew rate 6 111 Reserved

17.4.2.3 Reserved Register

Module Base + 0x0002 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 17-4. Reserved Register

[1] Read: Anytime
Write: Only in special mode

NOTE

This reserved register is designed for factory test purposes only and is not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

17.4.2.4 CAN Physical Layer Status Register (CPSR)

Module Base + 0x0003 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	CPCHVH	CPCHVL	CPCLVH	CPCLVL	CPDT	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-5. CAN Physical Layer Status Register (CPSR)

[1] Read: Anytime
Write: Never

Table 17-6. CPSR Register Field Descriptions

Field	Description
7 CPCHVH	**CANH Voltage Failure High Status Bit** This bit reflects the CANH voltage failure high monitor status. 0 Condition $V_{CANH} < V_{H5}$ 1 Condition $V_{CANH} \geq V_{H5}$
6 CPCHVL	**CANH Voltage Failure Low Status Bit** This bit reflects the CANH voltage failure low monitor status. 0 Condition $V_{CANH} > V_{H0}$ 1 Condition $V_{CANH} \leq V_{H0}$
5 CPCLVH	**CANL Voltage Failure High Status Bit** This bit reflects the CANL voltage failure high monitor status. 0 Condition $V_{CANL} < V_{L5}$ 1 Condition $V_{CANL} \geq V_{L5}$

Table 17-6. CPSR Register Field Descriptions

Field	Description
4 CPCLVL	**CANL Voltage Failure Low Status Bit** This bit reflects the CANL voltage failure low monitor status. 0 Condition $V_{CANL} > V_{L0}$ 1 Condition $V_{CANL} \leq V_{L0}$
3 CPDT	**CPTXD-Dominant Timeout Status Bit** This bit is set to 1, if CPTXD is dominant for longer than $t_{CPTXDDT}$. It signals a timeout event and remains set until CPTXD returns to recessive level for longer than 1 µs. 0 No CPTXD-timeout occurred or CPTXD has ceased to be dominant after timeout 1 CPTXD-dominant timeout occurred and CPTXD is still dominant

17.4.2.5 Reserved Register

Module Base + 0x0004 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 17-6. Reserved Register

[1] Read: Anytime
Write: Only in special mode

NOTE

This reserved register is designed for factory test purposes only and is not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

17.4.2.6 Reserved Register

Module Base + 0x0005 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	x	x	x	x	x	x	x	x

Figure 17-7. Reserved Register

[1] Read: Anytime
Write: Only in special mode

NOTE

This reserved register is designed for factory test purposes only and is not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

17.4.2.7 CAN Physical Layer Interrupt Enable Register (CPIE)

Module Base + 0x0006 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	CPVFIE	CPDTIE	0	0	CPOCIE
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-8. CAN Physical Layer Interrupt Enable Register (CPIE)

[1] Read: Anytime
Write: Anytime

Table 17-7. CPIE Register Field Descriptions

Field	Description
4 CPVFIE	**CAN Physical Layer Voltage-Failure Interrupt Enable** If enabled, the CAN Physical Layer generates an interrupt if any of the CAN Physical Layer voltage failure interrupt flags assert. 0 Voltage failure interrupt is disabled 1 Voltage failure interrupt is enabled
3 CPDTIE	**CPTXD-Dominant Timeout Interrupt Enable** If enabled, the CAN Physical Layer generates an interrupt if the CPTXD-dominant timeout interrupt flag asserts. 0 CPTXD-dominant timeout interrupt is disabled 1 CPTXD-dominant timeout interrupt is enabled
0 CPOCIE	**CAN Physical Layer Over-current Interrupt Enable** If enabled, the CAN Physical Layer generates an interrupt if any of the CAN Physical Layer over-current interrupt flags assert. 0 Over-current interrupt is disabled 1 Over-current interrupt is enabled

17.4.2.8 CAN Physical Layer Interrupt Flag Register (CPIF)

Module Base + 0x0007

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	CHVHIF	CHVLIF	CLVHIF	CLVLIF	CPDTIF	0	CHOCIF	CLOCIF
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-9. CAN Physical Layer Interrupt Flag Register (CPIF)

[1] Read: Anytime
Write: Anytime, write 1 to clear

If any of the flags is asserted an error interrupt is pending if enabled. A flag can be cleared by writing a logic level 1 to the corresponding bit location. Writing a 0 has no effect.

Table 17-8. CPIF Register Field Descriptions

Field	Description
7 CHVHIF	**CANH Voltage Failure High Interrupt Flag** This flag is set to 1 when the CPCHVH bit in the CAN Physical Layer Status Register (CPSR) changes. 0 No change in CPCHVH 1 CPCHVH has changed
6 CHVLIF	**CANH Voltage Failure Low Interrupt Flag** This flag is set to 1 when the CPCHVL bit in the CAN Physical Layer Status Register (CPSR) changes. 0 No change in CPCHVL 1 CPCHVL has changed

Table 17-8. CPIF Register Field Descriptions

Field	Description
5 CLVHIF	**CANL Voltage Failure High Interrupt Flag** This flag is set to 1 when the CPCLVH bit in the CAN Physical Layer Status Register (CPSR) changes. 0 No change in CPCLVH 1 CPCLVH has changed
4 CLVLIF	**CANL Voltage Failure Low Interrupt Flag** This flag is set to 1 when the CPCLVL bit in the CAN Physical Layer Status Register (CPSR) changes. 0 No change in CPCLVL 1 CPCLVL has changed
3 CPDTIF	**CAN CPTXD-Dominant Timeout Interrupt Flag** This flag is set to 1 when CPTXD is dominant longer than $t_{CPTXDDT}$. It signals a timeout event and entry of listen-only mode disabling the transmitter. Exit of listen-only mode which was entered at timeout is requested by clearing CPDTIF when CPDT is clear after setting CPTXD to recessive state. It takes 1 to 2 μs to return to normal mode. If CPTXD is dominant or dominant timeout status is still active (CPDT=1) when clearing the flag, the CAN Physical Layer remains in listen-only mode and this flag is set again after a delay (see 17.5.4.2, “CPTXD-Dominant Timeout Interrupt””). 0 No CPTXD-dominant timeout has occurred 1 CPTXD-dominant timeout has occurred
1 CHOCIF	**CANH Over-Current Interrupt Flag** This flag is set to 1 if an over current condition was detected on CANH when driving a dominant bit to the CAN bus. While this flag is asserted the CAN Physical Layer remains in listen-only mode. 0 Normal current level $I_{CANH} < I_{CANHOC}$ 1 Error event $I_{CANH} \geq I_{CANHOC}$ occurred
0 CLOCIF	**CANL Over-Current Interrupt Flag** This flag is set to 1 if an over current condition was detected on CANL when driving a dominant bit to the CAN bus. While this flag is asserted the CAN Physical Layer remains in listen-only mode. 0 Normal current level $I_{CANL} < I_{CANLOC}$ 1 Error event $I_{CANL} \geq I_{CANLOC}$ occurred

17.5 Functional Description

17.5.1 General

The CAN Physical Layer provides an interface for the SoC-integrated MSCAN controller.

17.5.2 Modes

Figure 17-10 shows the possible mode transitions depending on control bit CPE, device reduced performance mode (“RPM”; refer to “Low Power Modes” section in device overview) and bus error conditions.

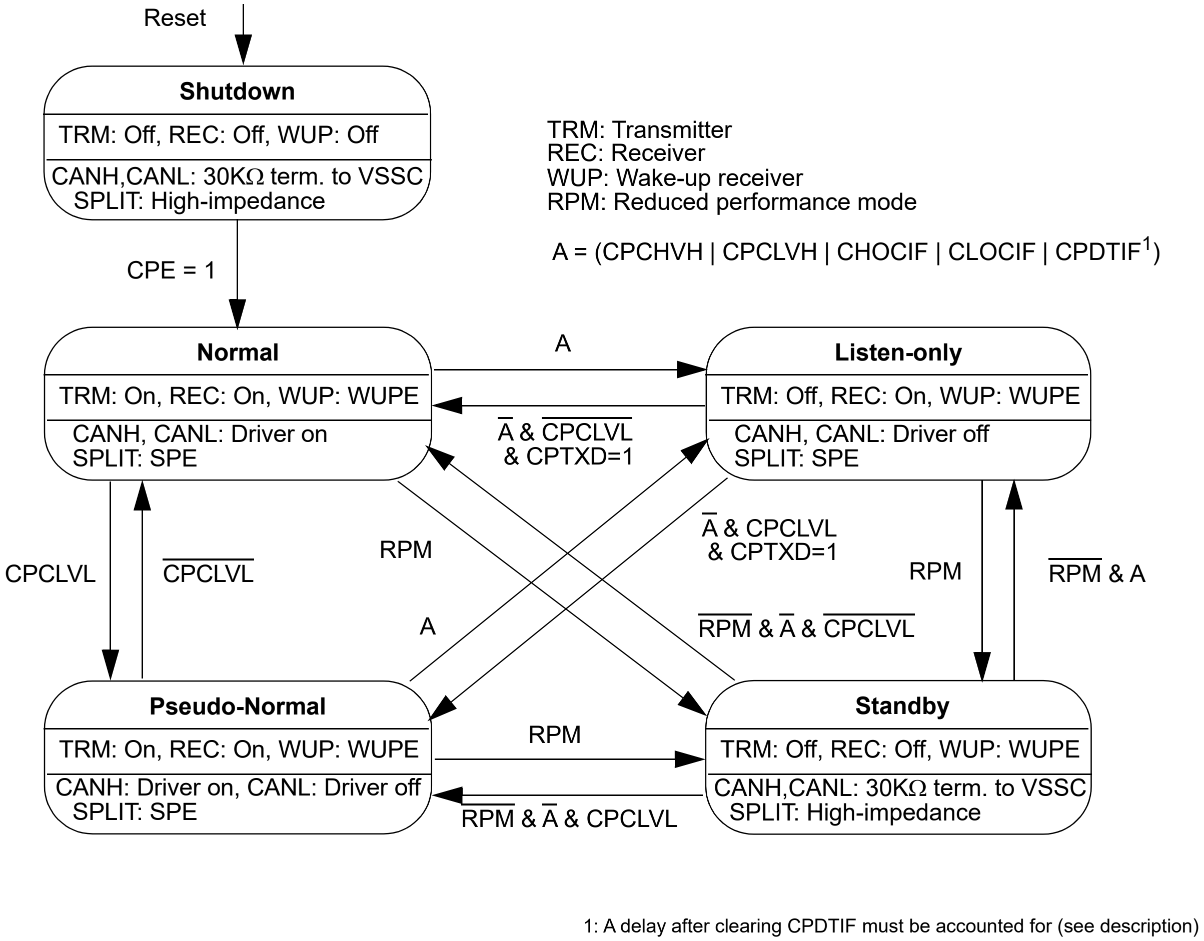


Figure 17-10. CAN Physical Layer Mode Transitions

17.5.2.1 Shutdown Mode

Shutdown mode is a low power mode and entered out of reset. The transceiver, wake-up, bus error diagnostic, dominant timeout and interrupt functionality are disabled. CANH and CANL lines are pulled to VSSC via high-ohmic input resistors of the receiver. The SPLIT pin as well as the internal mid-point reference are set to high-impedance.

Shutdown mode cannot be re-entered until reset.

17.5.2.2 Normal Mode

In normal mode the full transceiver functionality is available. In this mode, the CAN bus is controlled by the CPTXD input and the CAN bus state (recessive, dominant) is reported on the CPRXD output. The voltage failure, over-current and CPTXD-dominant timeout monitors are enabled. The SPLIT pin is driving a 2.5 V bias if enabled. The internal mid-point reference is set to 2.5 V.

If CPTXD is high, the transmit driver is set into recessive state, and CANH and CANL lines are biased to the voltage set at VDDC divided by 2, approx. 2.5 V.

If CPTXD is low, the transmit driver is set into dominant state, and CANH and CANL drivers are active. CANL is pulled low and CANH is pulled high.

The receiver reports the bus state on CPRXD. If the differential voltage V_{CANH} minus V_{CANL} at CANH and CANL is below the internal threshold, the bus is recessive and CPRXD is set high, otherwise a dominant bus is detected and CPRXD is set low.

When detecting a voltage high failure, over-current or CPTXD-dominant timeout event the CAN Physical Layer enters listen-only mode. A voltage low failure on CANL results in entering pseudo-normal mode. A voltage low failure on CANH maintains normal mode.

NOTE

After entering normal mode from shutdown or standby mode a settling time of t_{CP_set} must have passed until flags can be considered as valid.

17.5.2.3 Pseudo-Normal Mode

Pseudo-normal mode is identical to normal mode except for CANL driver being disabled as a result of a voltage low failure that has been detected on the CANL bus line (CPCLVL=1). CANH remains functional in this mode to allow transmission. Normal mode will automatically be re-entered after the error condition has ceased.

17.5.2.4 Listen-only Mode

Listen-only mode is entered upon detecting a CAN bus error condition (except for CPCLVL=1) or CPTXD-dominant timeout event. The entire transmitter is forced off. All other functions of the normal mode are maintained.

Application software action is required to re-enter normal mode by clearing the related flags if the bus error condition was caused by an over-current (refer to 17.6.3). In case of a voltage failure, normal mode will automatically be re-entered if the condition has passed. If the listen-only mode was caused by CPTXD-dominant timeout event, the related flag can only be cleared after the CPTXD has returned to recessive level.

17.5.2.5 Standby Mode

Standby is a reduced current consumption mode and is entered during RPM following a stop mode request. The transceiver and bus error diagnostics are disabled. The CPTXD-dominant timeout counter is stopped. CANH and CANL lines are pulled to VSSC via high-ohmic input resistors of the receiver. The SPLIT pin is set to high-impedance. The internal mid-point reference is set to 0V. All voltage failure and over-current monitors are disabled.

Standby is left as soon as the device returns from RPM.

17.5.3 Configurable Wake-Up

If the wake-up function is enabled, the CAN Physical Layer provides an asynchronous path through CPRXD to the MSCAN to support wake-up from stop mode. The CPRXD signal is switched from precision receiver to the low-power wake-up receiver as long as the device resides in RPM.

In order to avoid false wake-up after entering stop mode, a pulse filter can be enabled and configured to mask the first or first two wake-up events from the MSCAN input. The CPRXD output is held at recessive level until the selected number of wake-up events have been detected as shown in Figure 17-11.

A valid wakeup-event is defined as a dominant level with a length of min. t_{CPWUP} followed by a recessive level of length t_{CPWUP}.

The wake-up filter specification t_{WUP} of the MSCAN applies to wake-up the MSCAN from sleep mode. Refer to MSCAN chapter.After wake-up the CAN Physical Layer automatically returns to the mode where stop mode was requested.

Refer to 17.6.2, “Wake-up Mechanism”” for setup information.

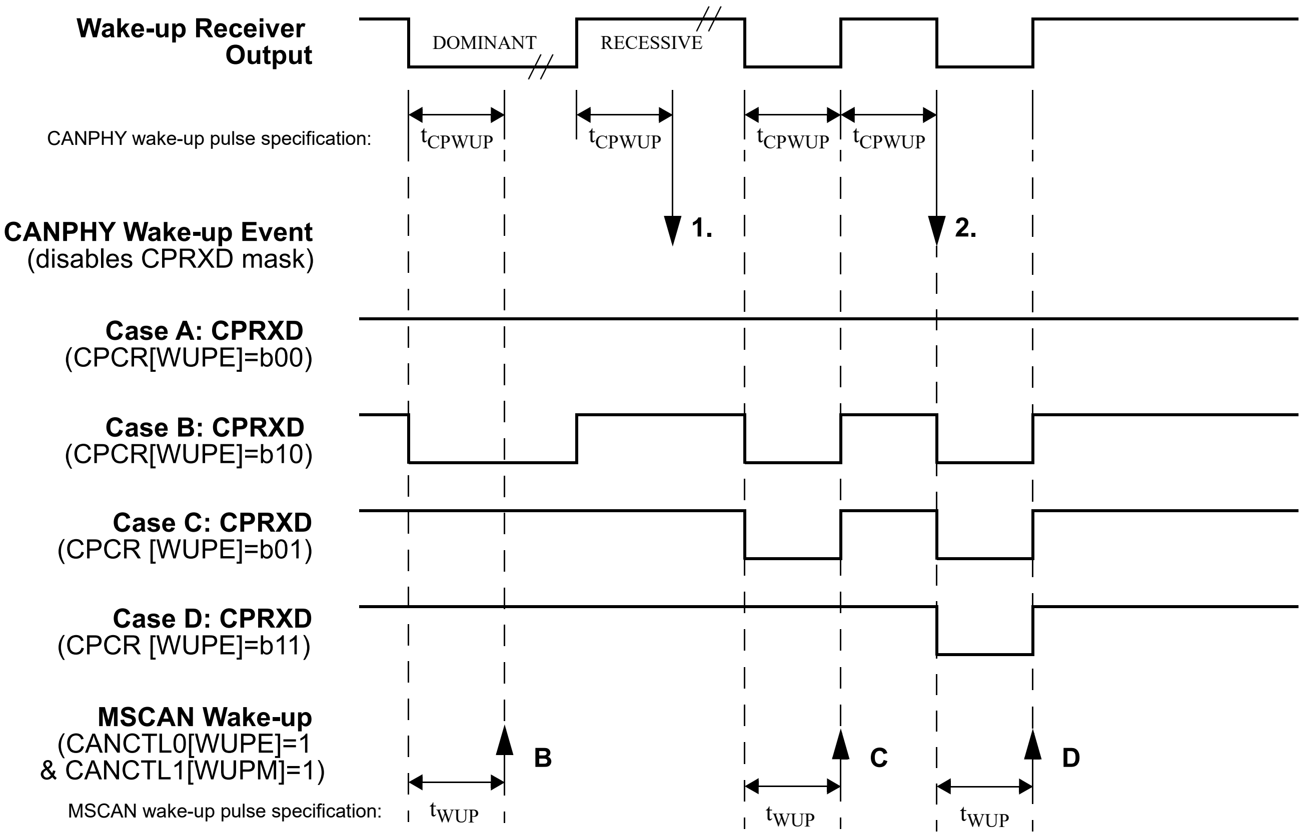


Figure 17-11. Wake-up Event Filtering

17.5.4 Interrupts

This section describes the interrupt generated by the CAN Physical Layer and its individual sources.

Vector addresses and interrupt priorities are defined at MCU level. The module internal interrupt sources are combined (OR-ed) into one module interrupt output CPI with a single local enable bit each for voltage failure and over-current errors.

Table 17-9. CAN Physical Layer Interrupt Sources

Module Interrupt Source	Module Internal Interrupt Source	Local Enable
CAN Physical Layer Interrupt (CPI)	CANH Voltage Failure High Interrupt (CHVHIF)	CPVFIE = 1
	CANH Voltage Failure Low Interrupt (CHVLIF)	
	CANL Voltage Failure High Interrupt (CLVHIF)	
	CANL Voltage Failure Low Interrupt (CLVLIF)	
	CPTXD-Dominant Timeout Interrupt (CPDTIF)	CPDTIE = 1
	CANH Over-Current Interrupt (CHOCIF)	CPOCIE = 1
	CANL Over-Current Interrupt (CLOCIF)	

17.5.4.1 Voltage Failure Interrupts

A voltage failure error is detected if voltage levels on the CAN bus lines exceed the specified limits.

The voltages on both lines CANH and CANL are monitored continuously for crossing the lower and higher thresholds, V_{H0}, V_{H5} and V_{L0}, V_{L5}, respectively.

A comparator output transition to error level results in setting the corresponding status bit in CAN Physical Layer Status Register (CPSR). A change of a status bit sets the related interrupt flag in CAN Physical Layer Interrupt Flag Register (CPIF).

The flags are used as interrupt sources of which either of the four can generate a CPI interrupt if the common enable bit CPVFIE in CAN Physical Layer Interrupt Enable Register (CPIE) is set.

17.5.4.2 CPTXD-Dominant Timeout Interrupt

For network lock-up protection of the CAN bus, the CAN physical layer features a permanent CPTXD-dominant timeout monitor. When the CPTXD signal has been dominant for more than $t_{CPTXDDT}$ the transmitter is disabled by entering listen-only mode and the bus is released to recessive state. The CPDT status and CPDTIF interrupt flags are both set.

To re-enable the transmitter, the CPDTIF flag must be cleared. If the CPTXD input is dominant or dominant timeout status is still active (CPDT=1), the CAN Physical Layer stays in listen-only mode and CPDTIF is set again after some microseconds to indicate that the attempt has failed. If CPTXD is recessive and CPDT=0 it takes 1 to 2 µs after clearing CPDTIF for returning to normal mode.

The flag is used as an interrupt source to generate a CPI interrupt if the enable bit CPDTIE in CAN Physical Layer Interrupt Enable Register (CPIE) is set.

17.5.4.3 Over-Current Interrupt

An over-current error is detected if current levels on the CAN bus lines exceed the specified limits while driving a dominant bit.

The current levels on both lines CANH and CANL are monitored continuously for crossing the thresholds I_{CANHOC} and I_{CANLOC}, respectively.

A comparator output transition to error level results in setting the corresponding interrupt flag in CAN Physical Layer Interrupt Flag Register (CPIF).

The flags are the direct interrupt sources of which either of the two can generate a CPI interrupt if the common enable bit CPOCIE in CAN Physical Layer Interrupt Enable Register (CPIE) is set.

17.6 Initialization/Application Information

17.6.1 Initialization Sequence

Setup for immediate CAN communication:

1. Enable and configure MSCAN

2. Configure CAN Physical Layer slew rate
3. Enable CAN Physical Layer interrupts
4. Optionally enable SPLIT pin
5. Configure wake-up filter or disable wake-up receiver in case of other wake-up sources
6. Enable CAN Physical Layer to enter normal mode
7. Start CAN communication

17.6.2 Wake-up Mechanism

In stop mode the CAN Physical Layer passes CAN bus states to CPRXD if the wake-up function is enabled (CPCR[WUPE1:WUPE0]≠0). In order to wake up the device from stop mode, the wake-up interrupt of the connected MSCAN module is used.

If CPCR[WUPE1:WUPE0]=b10 the CAN Physical Layer is transparent in stop mode and the MSCAN can be used with or without its integrated low-pass filter for wake-up. Refer to the MSCAN chapter for details on configuring and enabling the wake-up function.

For increased robustness against false wake-up, a CAN Physical Layer pulse filter can optionally be enabled to mask the first (CPCR[WUPE1:WUPE0]=b01) or first two (CPCR[WUPE1:WUPE0]=b11) wake-up events after entering stop mode. The appropriate number of masked pulses depends on the individual CAN bus network topology.

Note that the MSCAN can generate a wake-up interrupt immediately after it acknowledges sleep mode (CANCTL1[SPLAK]=1) whereas the CAN Physical Layer pulse filter takes effect only after entering stop mode. To avoid a false wake-up in between these two events, the MSCAN low-pass filter should also be activated (CANCTL1[WUPM]=1). After sleep mode acknowledge the CPU STOP instruction should be executed before the expiration of t_{WUP}(min) to enable the CAN Physical Layer pulse filter in time.

17.6.3 Bus Error Handling

Upon CAN bus error voltage high failures and over-current events listen-only is entered immediately and the transmitter is turned off. This mode is maintained as long as voltage failure conditions persist or, in case of over-current events, application software re-enables the transmit driver by clearing the related flags.

All high and low voltage levels for both CAN bus lines are continuously reflected in their related voltage failure status bits. A change in a status bit sets the corresponding flag and generates an interrupt if enabled. As long as any of the voltage failure high status bits is set, the transmit driver remains off. It will be turned on again automatically as soon as all voltage failure conditions have disappeared. In case of a voltage failure low condition on CANL only the CANL driver is disabled. A voltage failure low condition on CANH has no effect on the transmitter.

Voltage failure errors have informational purpose. If the application detects frequent CAN protocol errors it is advisable to take the appropriate action. No software action is need to re-enable the transmit driver.

An over-current event on either CAN bus line sets the related flag and turns off the transmit driver. This error can only be detected while driving the bus dominant. In contrast to the voltage failure the over-current condition instantaneously disappears as soon as the transmit driver is automatically being turned off. This state is locked and the application software must account for re-enabling the driver.

The recommended procedure to handle an over-current related bus error is:

1. On interrupt abort any scheduled transmissions
2. Read interrupt flag register to determine over-current source(s)
3. Clear related interrupt flag(s)
4. Retry CAN transmission
5. On interrupt abort any scheduled transmissions
6. Read interrupt flag register to determine over-current source(s)
7. If the same over-current error persists do not retry and run appropriate custom diagnostics

17.6.4 CPTXD-Dominant Timeout Recovery

Recovery from a CPTXD-dominant timeout error is attempted with the following sequence:

1. On CPTXD-dominant timeout interrupt set CPTXD input to recessive state
2. Wait until CPDT clear; exit loop if waiting for longer than 3 μs and report malfunction
3. Clear CPDTIF
4. Wait for min. 2 μs before attempting new transmission

Chapter 18
Scalable Controller Area Network (S12MSCANV3)

Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V03.14	12 Nov 2012	Table 18-10	• Corrected RxWRN and TxWRN threshold values
V03.15	12 Jan 2013	Table 18-2 Table 18-25 Figure 18-37 18.1/18-549 18.3.2.15/18-570	• Updated TIME bit description • Added register names to buffer map • Updated TSRH and TSRL read conditions • Updated introduction • Updated CANTXERR and CANRXERR register notes
V03.16	08 Aug 2013		• Corrected typos

18.1 Introduction

NXP's scalable controller area network (S12MSCANV3) definition is based on the MSCAN12 definition, which is the specific implementation of the MSCAN concept targeted for the S12, S12X and S12Z microcontroller families.

The module is a communication controller implementing the CAN 2.0A/B protocol as defined in the Bosch specification dated September 1991. For users to fully understand the MSCAN specification, it is recommended that the Bosch specification be read first to familiarize the reader with the terms and concepts contained within this document.

Though not exclusively intended for automotive applications, CAN protocol is designed to meet the specific requirements of a vehicle serial data bus: real-time processing, reliable operation in the EMI environment of a vehicle, cost-effectiveness, and required bandwidth.

MSCAN uses an advanced buffer arrangement resulting in predictable real-time behavior and simplified application software.

18.1.1 Glossary

Table 18-1. Terminology

ACK	Acknowledge of CAN message
CAN	Controller Area Network
CRC	Cyclic Redundancy Code
EOF	End of Frame
FIFO	First-In-First-Out Memory
IFS	Inter-Frame Sequence
SOF	Start of Frame
CPU bus	CPU related read/write data bus
CAN bus	CAN protocol related serial bus
oscillator clock	Direct clock from external oscillator
bus clock	CPU bus related clock
CAN clock	CAN protocol related clock

18.1.2 Block Diagram

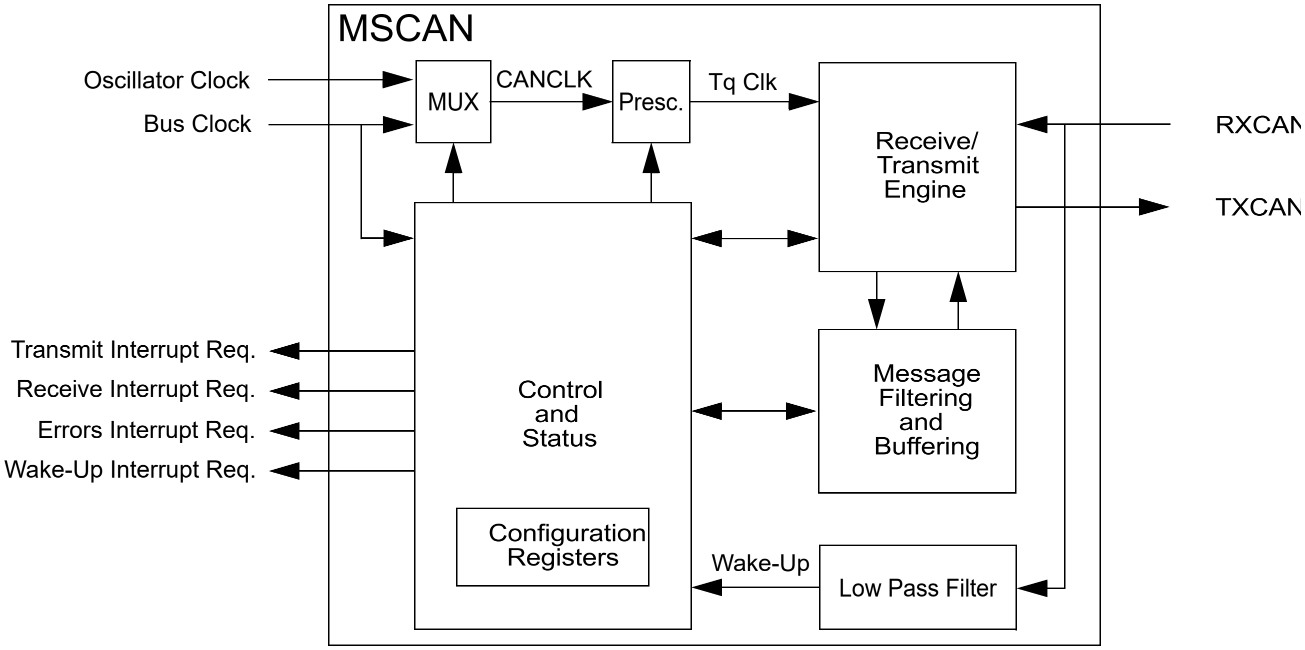


Figure 18-1. MSCAN Block Diagram

18.1.3 Features

The basic features of the MSCAN are as follows:

- Implementation of the CAN protocol — Version 2.0A/B
    - Standard and extended data frames
    - Zero to eight bytes data length
    - Programmable bit rate up to 1 Mbps[1]
    - Support for remote frames
- Five receive buffers with FIFO storage scheme
- Three transmit buffers with internal prioritization using a “local priority” concept
- Flexible maskable identifier filter supports two full-size (32-bit) extended identifier filters, or four 16-bit filters, or eight 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter
- Programmable loopback mode supports self-test operation
- Programmable listen-only mode for monitoring of CAN bus
- Programmable bus-off recovery functionality
- Separate signalling and interrupt capabilities for all CAN receiver and transmitter error states (warning, error passive, bus-off)
- Programmable MSCAN clock source either bus clock or oscillator clock
- Internal timer for time-stamping of received and transmitted messages
- Three low-power modes: sleep, power down, and MSCAN enable
- Global initialization of configuration registers

18.1.4 Modes of Operation

For a description of the specific MSCAN modes and the module operation related to the system operating modes refer to Section 18.4.4, “Modes of Operation”.

---

[1] Depending on the actual bit timing and the clock jitter of the PLL.

18.2 External Signal Description

The MSCAN uses two external pins.

NOTE

> On MCUs with an integrated CAN physical interface (transceiver) the MSCAN interface is connected internally to the transceiver interface. In these cases the external availability of signals TXCAN and RXCAN is optional.

18.2.1 RXCAN — CAN Receiver Input Pin

RXCAN is the MSCAN receiver input pin.

18.2.2 TXCAN — CAN Transmitter Output Pin

TXCAN is the MSCAN transmitter output pin. The TXCAN output pin represents the logic level on the CAN bus:

0 = Dominant state
1 = Recessive state

18.2.3 CAN System

A typical CAN system with MSCAN is shown in Figure 18-2. Each CAN station is connected physically to the CAN bus lines through a transceiver device. The transceiver is capable of driving the large current needed for the CAN bus and has current protection against defective CAN or defective stations.

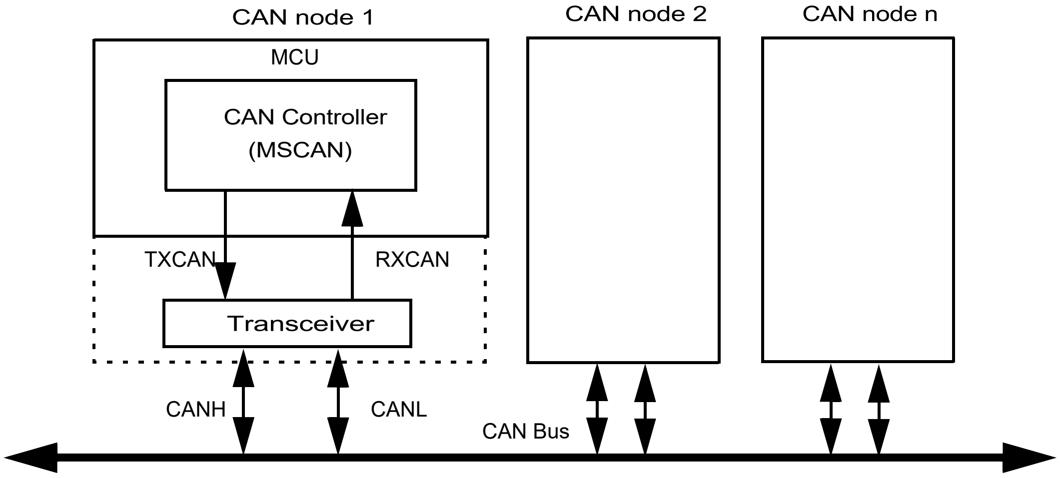


Figure 18-2. CAN System

18.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the MSCAN.

18.3.1 Module Memory Map

Figure 18-3 gives an overview on all registers and their individual bits in the MSCAN memory map. The *register address* results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level and can be found in the MCU memory map description. The *address offset* is defined at the module level.

The MSCAN occupies 64 bytes in the memory space. The base address of the MSCAN module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset.

The detailed register descriptions follow in the order they appear in the register map.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 CANCTL0	R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
	W		▨		▨				
0x0001 CANCTL1	R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
	W							▨	▨
0x0002 CANBTR0	R	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
	W								
0x0003 CANBTR1	R	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
	W								
0x0004 CANRFLG	R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
	W			▨	▨	▨	▨		
0x0005 CANRIER	R	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
	W								
0x0006 CANTFLG	R	0	0	0	0	0	TXE2	TXE1	TXE0
	W	▨	▨	▨	▨	▨			
0x0007 CANTIER	R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
	W	▨	▨	▨	▨	▨			
0x0008 CANTARQ	R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
	W	▨	▨	▨	▨	▨			
0x0009 CANTAAK	R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000A CANTBSEL	R	0	0	0	0	0	TX2	TX1	TX0
	W	▨	▨	▨	▨	▨			
0x000B CANIDAC	R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
	W	▨	▨			▨	▨	▨	▨
0x000C Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000D CANMISC	R	0	0	0	0	0	0	0	BOHOLD
	W	▨	▨	▨	▨	▨	▨	▨	
0x000E CANRXERR	R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 18-3. MSCAN Register Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000F CANTXERR	R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0010–0x0013 CANIDAR0–3	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x0014–0x0017 CANIDMRx	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0018–0x001B CANIDAR4–7	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x001C–0x001F CANIDMR4–7	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0020–0x002F CANRXFG	R	See Section 18.3.3, “Programmer’s Model of Message Storage”							
	W								
0x0030–0x003F CANTXFG	R	See Section 18.3.3, “Programmer’s Model of Message Storage”							
	W								

▨ = Unimplemented or Reserved

Figure 18-3. MSCAN Register Summary (continued)

18.3.2 Register Descriptions

This section describes in detail all the registers and register bits in the MSCAN module. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order. All bits of all registers in this module are completely synchronous to internal clocks during a register read.

18.3.2.1 MSCAN Control Register 0 (CANCTL0)

The CANCTL0 register provides various control bits of the MSCAN module as described below.

Module Base + 0x0000

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
W		▨		▨				
Reset:	0	0	0	0	0	0	0	1

▨ = Unimplemented

Figure 18-4. MSCAN Control Register 0 (CANCTL0)

[1] Read: Anytime
Write: Anytime when out of initialization mode; exceptions are read-only RXACT and SYNCH, RXFRM (which is set by the module only), and INITRQ (which is also writable in initialization mode)

NOTE

The CANCTL0 register, except WUPE, INITRQ, and SLPRQ, is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable again as soon as the initialization mode is exited (INITRQ = 0 and INITAK = 0).

Table 18-2. CANCTL0 Register Field Descriptions

Field	Description
7 RXFRM	**Received Frame Flag** — This bit is read and clear only. It is set when a receiver has received a valid message correctly, independently of the filter configuration. After it is set, it remains set until cleared by software or reset. Clearing is done by writing a 1. Writing a 0 is ignored. This bit is not valid in loopback mode. 0 No valid message was received since last clearing this flag 1 A valid message was received since last clearing of this flag
6 RXACT	**Receiver Active Status** — This read-only flag indicates the MSCAN is receiving a message[1]. The flag is controlled by the receiver front end. This bit is not valid in loopback mode. 0 MSCAN is transmitting or idle 1 MSCAN is receiving a message (including when arbitration is lost)
5 CSWAI[2]	**CAN Stops in Wait Mode** — Enabling this bit allows for lower power consumption in wait mode by disabling all the clocks at the CPU bus interface to the MSCAN module. 0 The module is not affected during wait mode 1 The module ceases to be clocked during wait mode
4 SYNCH	**Synchronized Status** — This read-only flag indicates whether the MSCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the MSCAN. 0 MSCAN is not synchronized to the CAN bus 1 MSCAN is synchronized to the CAN bus
3 TIME	**Timer Enable** — This bit activates an internal 16-bit wide free running timer which is clocked by the bit clock rate. If the timer is enabled, a 16-bit time stamp will be assigned to each transmitted/received message within the active TX/RX buffer. Right after the EOF of a valid message on the CAN bus, the time stamp is written to the highest bytes (0x000E, 0x000F) in the appropriate buffer (see Section 18.3.3, "Programmer's Model of Message Storage"). In loopback mode no receive timestamp is generated. The internal timer is reset (all bits set to 0) when disabled. This bit is held low in initialization mode. 0 Disable internal MSCAN timer 1 Enable internal MSCAN timer
2 WUPE[3]	**Wake-Up Enable** — This configuration bit allows the MSCAN to restart from sleep mode or from power down mode (entered from sleep) when traffic on CAN is detected (see Section 18.4.5.5, "MSCAN Sleep Mode"). This bit must be configured before sleep mode entry for the selected function to take effect. 0 Wake-up disabled — The MSCAN ignores traffic on CAN 1 Wake-up enabled — The MSCAN is able to restart

Table 18-2. CANCTL0 Register Field Descriptions (continued)

Field	Description
1 SLPRQ[4]	**Sleep Mode Request** — This bit requests the MSCAN to enter sleep mode, which is an internal power saving mode (see Section 18.4.5.5, "MSCAN Sleep Mode"). The sleep mode request is serviced when the CAN bus is idle, i.e., the module is not receiving a message and all transmit buffers are empty. The module indicates entry to sleep mode by setting SLPAK = 1 (see Section 18.3.2.2, "MSCAN Control Register 1 (CANCTL1)"). SLPRQ cannot be set while the WUPIF flag is set (see Section 18.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)"). Sleep mode will be active until SLPRQ is cleared by the CPU or, depending on the setting of WUPE, the MSCAN detects activity on the CAN bus and clears SLPRQ itself. 0 Running — The MSCAN functions normally 1 Sleep mode request — The MSCAN enters sleep mode when CAN bus idle
0 INITRQ[5,6]	**Initialization Mode Request** — When this bit is set by the CPU, the MSCAN skips to initialization mode (see Section 18.4.4.5, "MSCAN Initialization Mode"). Any ongoing transmission or reception is aborted and synchronization to the CAN bus is lost. The module indicates entry to initialization mode by setting INITAK = 1 (Section 18.3.2.2, "MSCAN Control Register 1 (CANCTL1)"). The following registers enter their hard reset state and restore their default values: CANCTL0[7], CANRFLG[8], CANRIER[9], CANTFLG, CANTIER, CANTARQ, CANTAAK, and CANTBSEL. The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, and CANIDMR0-7 can only be written by the CPU when the MSCAN is in initialization mode (INITRQ = 1 and INITAK = 1). The values of the error counters are not affected by initialization mode. When this bit is cleared by the CPU, the MSCAN restarts and then tries to synchronize to the CAN bus. If the MSCAN is not in bus-off state, it synchronizes after 11 consecutive recessive bits on the CAN bus; if the MSCAN is in bus-off state, it continues to wait for 128 occurrences of 11 consecutive recessive bits. Writing to other bits in CANCTL0, CANRFLG, CANRIER, CANTFLG, or CANTIER must be done only after initialization mode is exited, which is INITRQ = 0 and INITAK = 0. 0 Normal operation 1 MSCAN in initialization mode

[1] See the Bosch CAN 2.0A/B specification for a detailed definition of transmitter and receiver states.

[2] In order to protect from accidentally violating the CAN protocol, TXCAN is immediately forced to a recessive state when the CPU enters wait (CSWAI = 1) or stop mode (see Section 18.4.5.2, "Operation in Wait Mode" and Section 18.4.5.3, "Operation in Stop Mode").

[3] The CPU has to make sure that the WUPE register and the WUPIE wake-up interrupt enable register (see Section 18.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)) is enabled, if the recovery mechanism from stop or wait is required.

[4] The CPU cannot clear SLPRQ before the MSCAN has entered sleep mode (SLPRQ = 1 and SLPAK = 1).

[5] The CPU cannot clear INITRQ before the MSCAN has entered initialization mode (INITRQ = 1 and INITAK = 1).

[6] In order to protect from accidentally violating the CAN protocol, TXCAN is immediately forced to a recessive state when the initialization mode is requested by the CPU. Thus, the recommended procedure is to bring the MSCAN into sleep mode (SLPRQ = 1 and SLPAK = 1) before requesting initialization mode.

[7] Not including WUPE, INITRQ, and SLPRQ.

[8] TSTAT1 and TSTAT0 are not affected by initialization mode.

[9] RSTAT1 and RSTAT0 are not affected by initialization mode.

18.3.2.2 MSCAN Control Register 1 (CANCTL1)

The CANCTL1 register provides various control bits and handshake status information of the MSCAN module as described below.

Module Base + 0x0001 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
W							▨	▨
Reset:	0	0	0	1	0	0	0	1

▨ = Unimplemented

Figure 18-5. MSCAN Control Register 1 (CANCTL1)

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1), except CANE which is write once in normal and anytime in special system operation modes when the MSCAN is in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-3. CANCTL1 Register Field Descriptions

Field	Description
7 CANE	**MSCAN Enable** 0 MSCAN module is disabled 1 MSCAN module is enabled
6 CLKSRC	**MSCAN Clock Source** — This bit defines the clock source for the MSCAN module (only for systems with a clock generation module; Section 18.4.3.2, “Clock System,” and Section Figure 18-43., “MSCAN Clocking Scheme,”). 0 MSCAN clock source is the oscillator clock 1 MSCAN clock source is the bus clock
5 LOOPB	**Loopback Self Test Mode** — When this bit is set, the MSCAN performs an internal loopback which can be used for self test operation. The bit stream output of the transmitter is fed back to the receiver internally. The RXCAN input is ignored and the TXCAN output goes to the recessive state (logic 1). The MSCAN behaves as it does normally when transmitting and treats its own transmitted message as a message received from a remote node. In this state, the MSCAN 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. 0 Loopback self test disabled 1 Loopback self test enabled
4 LISTEN	**Listen Only Mode** — This bit configures the MSCAN as a CAN bus monitor. When LISTEN is set, all valid CAN messages with matching ID are received, but no acknowledgement or error frames are sent out (see Section 18.4.4.4, “Listen-Only Mode”). In addition, the error counters are frozen. Listen only mode supports applications which require “hot plugging” or throughput analysis. The MSCAN is unable to transmit any messages when listen only mode is active. 0 Normal operation 1 Listen only mode activated
3 BORM	**Bus-Off Recovery Mode** — This bit configures the bus-off state recovery mode of the MSCAN. Refer to Section 18.5.2, “Bus-Off Recovery,” for details. 0 Automatic bus-off recovery (see Bosch CAN 2.0A/B protocol specification) 1 Bus-off recovery upon user request
2 WUPM	**Wake-Up Mode** — If WUPE in CANCTL0 is enabled, this bit defines whether the integrated low-pass filter is applied to protect the MSCAN from spurious wake-up (see Section 18.4.5.5, “MSCAN Sleep Mode”). 0 MSCAN wakes up on any dominant level on the CAN bus 1 MSCAN wakes up only in case of a dominant pulse on the CAN bus that has a length of T_{wup}

Table 18-3. CANCTL1 Register Field Descriptions (continued)

Field	Description
1 SLPAK	**Sleep Mode Acknowledge** — This flag indicates whether the MSCAN module has entered sleep mode (see Section 18.4.5.5, “MSCAN Sleep Mode”). It is used as a handshake flag for the SLPRQ sleep mode request. Sleep mode is active when SLPRQ = 1 and SLPAK = 1. Depending on the setting of WUPE, the MSCAN will clear the flag if it detects activity on the CAN bus while in sleep mode. 0 Running — The MSCAN operates normally 1 Sleep mode active — The MSCAN has entered sleep mode
0 INITAK	**Initialization Mode Acknowledge** — This flag indicates whether the MSCAN module is in initialization mode (see Section 18.4.4.5, “MSCAN Initialization Mode”). It is used as a handshake flag for the INITRQ initialization mode request. Initialization mode is active when INITRQ = 1 and INITAK = 1. The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0–CANIDAR7, and CANIDMR0–CANIDMR7 can be written only by the CPU when the MSCAN is in initialization mode. 0 Running — The MSCAN operates normally 1 Initialization mode active — The MSCAN has entered initialization mode

18.3.2.3 MSCAN Bus Timing Register 0 (CANBTR0)

The CANBTR0 register configures various CAN bus timing parameters of the MSCAN module.

Module Base + 0x0002 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
Reset:	0	0	0	0	0	0	0	0

Figure 18-6. MSCAN Bus Timing Register 0 (CANBTR0)

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-4. CANBTR0 Register Field Descriptions

Field	Description
7-6 SJW[1:0]	**Synchronization Jump Width** — The synchronization jump width defines the maximum number of time quanta (Tq) clock cycles a bit can be shortened or lengthened to achieve resynchronization to data transitions on the CAN bus (see Table 18-5).
5-0 BRP[5:0]	**Baud Rate Prescaler** — These bits determine the time quanta (Tq) clock which is used to build up the bit timing (see Table 18-6).

Table 18-5. Synchronization Jump Width

SJW1	SJW0	Synchronization Jump Width
0	0	1 Tq clock cycle
0	1	2 Tq clock cycles
1	0	3 Tq clock cycles
1	1	4 Tq clock cycles

Table 18-6. Baud Rate Prescaler

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	Prescaler value (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	1	64

18.3.2.4 MSCAN Bus Timing Register 1 (CANBTR1)

The CANBTR1 register configures various CAN bus timing parameters of the MSCAN module.

Module Base + 0x0003 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
Reset:	0	0	0	0	0	0	0	0

Figure 18-7. MSCAN Bus Timing Register 1 (CANBTR1)

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-7. CANBTR1 Register Field Descriptions

Field	Description
7 SAMP	**Sampling** — This bit determines the number of CAN bus samples taken per bit time. 0 One sample per bit. 1 Three samples per bit[1]. If SAMP = 0, the resulting bit value is equal to the value of the single bit positioned at the sample point. If SAMP = 1, the resulting bit value is determined by using majority rule on the three total samples. For higher bit rates, it is recommended that only one sample is taken per bit time (SAMP = 0).
6-4 TSEG2[2:0]	**Time Segment 2** — Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see Figure 18-44). Time segment 2 (TSEG2) values are programmable as shown in Table 18-8.
3-0 TSEG1[3:0]	**Time Segment 1** — Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see Figure 18-44). Time segment 1 (TSEG1) values are programmable as shown in Table 18-9.

[1] In this case, PHASE_SEG1 must be at least 2 time quanta (Tq).

Table 18-8. Time Segment 2 Values

TSEG22	TSEG21	TSEG20	Time Segment 2
0	0	0	1 Tq clock cycle[1]
0	0	1	2 Tq clock cycles
:	:	:	:
1	1	0	7 Tq clock cycles
1	1	1	8 Tq clock cycles

[1] This setting is not valid. Please refer to Table 18-36 for valid settings.

Table 18-9. Time Segment 1 Values

TSEG13	TSEG12	TSEG11	TSEG10	Time segment 1
0	0	0	0	1 Tq clock cycle[1]
0	0	0	1	2 Tq clock cycles[1]
0	0	1	0	3 Tq clock cycles[1]
0	0	1	1	4 Tq clock cycles
:	:	:	:	:
1	1	1	0	15 Tq clock cycles
1	1	1	1	16 Tq clock cycles

[1] This setting is not valid. Please refer to Table 18-36 for valid settings.

The bit time is determined by the oscillator frequency, the baud rate prescaler, and the number of time quanta (Tq) clock cycles per bit (as shown in Table 18-8 and Table 18-9).

Eqn. 18-1

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{CANCLK}} \bullet (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

18.3.2.5 MSCAN Receiver Flag Register (CANRFLG)

A flag can be cleared only by software (writing a 1 to the corresponding bit position) when the condition which caused the setting is no longer valid. Every flag has an associated interrupt enable bit in the CANRIER register.

Module Base + 0x0004 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
W			▨	▨	▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-8. MSCAN Receiver Flag Register (CANRFLG)

[1] Read: Anytime
Write: Anytime when not in initialization mode, except RSTAT[1:0] and TSTAT[1:0] flags which are read-only; write of 1 clears flag; write of 0 is ignored

NOTE

The CANRFLG register is held in the reset state[1] when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable again as soon as the initialization mode is exited (INITRQ = 0 and INITAK = 0).

Table 18-10. CANRFLG Register Field Descriptions

Field	Description
7 WUPIF	**Wake-Up Interrupt Flag** — If the MSCAN detects CAN bus activity while in sleep mode (see Section 18.4.5.5, "MSCAN Sleep Mode,") and WUPE = 1 in CANTCTL0 (see Section 18.3.2.1, "MSCAN Control Register 0 (CANCTL0)"), the module will set WUPIF. If not masked, a wake-up interrupt is pending while this flag is set. 0 No wake-up activity observed while in sleep mode 1 MSCAN detected activity on the CAN bus and requested wake-up
6 CSCIF	**CAN Status Change Interrupt Flag** — This flag is set when the MSCAN changes its current CAN bus status due to the actual value of the transmit error counter (TEC) and the receive error counter (REC). An additional 4-bit (RSTAT[1:0], TSTAT[1:0]) status register, which is split into separate sections for TEC/REC, informs the system on the actual CAN bus status (see Section 18.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)"). If not masked, an error interrupt is pending while this flag is set. CSCIF provides a blocking interrupt. That guarantees that the receiver/transmitter status bits (RSTAT/TSTAT) are only updated when no CAN status change interrupt is pending. If the TECs/RECs change their current value after the CSCIF is asserted, which would cause an additional state change in the RSTAT/TSTAT bits, these bits keep their status until the current CSCIF interrupt is cleared again. 0 No change in CAN bus status occurred since last interrupt 1 MSCAN changed current CAN bus status
5-4 RSTAT[1:0]	**Receiver Status Bits** — The values of the error counters control the actual CAN bus status of the MSCAN. As soon as the status change interrupt flag (CSCIF) is set, these bits indicate the appropriate receiver related CAN bus status of the MSCAN. The coding for the bits RSTAT1, RSTAT0 is: 00 RxOK: $0 \leq$ receive error counter < 96 01 RxWRN: $96 \leq$ receive error counter < 128 10 RxERR: $128 \leq$ receive error counter *11 Bus-off*[1]*: 256 ≤ transmit error counter*
3-2 TSTAT[1:0]	**Transmitter Status Bits** — The values of the error counters control the actual CAN bus status of the MSCAN. As soon as the status change interrupt flag (CSCIF) is set, these bits indicate the appropriate transmitter related CAN bus status of the MSCAN. The coding for the bits TSTAT1, TSTAT0 is: 00 TxOK: $0 \leq$ transmit error counter < 96 01 TxWRN: $96 \leq$ transmit error counter < 128 10 TxERR: $128 \leq$ transmit error counter < 256 11 Bus-Off: $256 \leq$ transmit error counter

---

[1] The RSTAT[1:0], TSTAT[1:0] bits are not affected by initialization mode.

Table 18-10. CANRFLG Register Field Descriptions (continued)

Field	Description
1 OVRIF	**Overrun Interrupt Flag** — This flag is set when a data overrun condition occurs. If not masked, an error interrupt is pending while this flag is set. 0 No data overrun condition 1 A data overrun detected
0 RXF[2]	**Receive Buffer Full Flag** — RXF is set by the MSCAN when a new message is shifted in the receiver FIFO. This flag indicates whether the shifted buffer is loaded with a correctly received message (matching identifier, matching cyclic redundancy code (CRC) and no other errors detected). After the CPU has read that message from the RxFG buffer in the receiver FIFO, the RXF flag must be cleared to release the buffer. A set RXF flag prohibits the shifting of the next FIFO entry into the foreground buffer (RxFG). If not masked, a receive interrupt is pending while this flag is set. 0 No new message available within the RxFG 1 The receiver FIFO is not empty. A new message is available in the RxFG

[1] Redundant Information for the most critical CAN bus status which is “bus-off”. This only occurs if the Tx error counter exceeds a number of 255 errors. Bus-off affects the receiver state. As soon as the transmitter leaves its bus-off state the receiver state skips to RxOK too. Refer also to TSTAT[1:0] coding in this register.

[2] To ensure data integrity, do not read the receive buffer registers while the RXF flag is cleared. For MCUs with dual CPUs, reading the receive buffer registers while the RXF flag is cleared may result in a CPU fault condition.

18.3.2.6 MSCAN Receiver Interrupt Enable Register (CANRIER)

This register contains the interrupt enable bits for the interrupt flags described in the CANRFLG register.

Module Base + 0x0005 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
Reset:	0	0	0	0	0	0	0	0

Figure 18-9. MSCAN Receiver Interrupt Enable Register (CANRIER)

[1] Read: Anytime
Write: Anytime when not in initialization mode

NOTE

The CANRIER register is held in the reset state when the initialization mode is active (INITRQ=1 and INITAK=1). This register is writable when not in initialization mode (INITRQ=0 and INITAK=0).

The RSTATE[1:0], TSTATE[1:0] bits are not affected by initialization mode.

Table 18-11. CANRIER Register Field Descriptions

Field	Description
7 WUPIE[1]	**Wake-Up Interrupt Enable** 0 No interrupt request is generated from this event. 1 A wake-up event causes a Wake-Up interrupt request.
6 CSCIE	**CAN Status Change Interrupt Enable** 0 No interrupt request is generated from this event. 1 A CAN Status Change event causes an error interrupt request.
5-4 RSTATE[1:0]	**Receiver Status Change Enable** — These RSTAT enable bits control the sensitivity level in which receiver state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the RSTAT flags continue to indicate the actual receiver state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by receiver state changes. 01 Generate CSCIF interrupt only if the receiver enters or leaves “bus-off” state. Discard other receiver state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the receiver enters or leaves “RxErr” or “bus-off”[2] state. Discard other receiver state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
3-2 TSTATE[1:0]	**Transmitter Status Change Enable** — These TSTAT enable bits control the sensitivity level in which transmitter state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level, the TSTAT flags continue to indicate the actual transmitter state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by transmitter state changes. 01 Generate CSCIF interrupt only if the transmitter enters or leaves “bus-off” state. Discard other transmitter state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the transmitter enters or leaves “TxErr” or “bus-off” state. Discard other transmitter state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
1 OVRIE	**Overrun Interrupt Enable** 0 No interrupt request is generated from this event. 1 An overrun event causes an error interrupt request.
0 RXFIE	**Receiver Full Interrupt Enable** 0 No interrupt request is generated from this event. 1 A receive buffer full (successful message reception) event causes a receiver interrupt request.

[1] WUPIE and WUPE (see Section 18.3.2.1, “MSCAN Control Register 0 (CANCTL0)”) must both be enabled if the recovery mechanism from stop or wait is required.

[2] Bus-off state is only defined for transmitters by the CAN standard (see Bosch CAN 2.0A/B protocol specification). Because the only possible state change for the transmitter from bus-off to TxOK also forces the receiver to skip its current state to RxOK, the coding of the RXSTAT[1:0] flags define an additional bus-off state for the receiver (see Section 18.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)”).

18.3.2.7 MSCAN Transmitter Flag Register (CANTFLG)

The transmit buffer empty flags each have an associated interrupt enable bit in the CANTIER register.

Module Base + 0x0006 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TXE2	TXE1	TXE0
W	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	1	1	1

▨ = Unimplemented

Figure 18-10. MSCAN Transmitter Flag Register (CANTFLG)

[1] Read: Anytime
Write: Anytime when not in initialization mode; write of 1 clears flag, write of 0 is ignored

NOTE

The CANTFLG register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 18-12. CANTFLG Register Field Descriptions

Field	Description
2-0 TXE[2:0]	**Transmitter Buffer Empty** — This flag indicates that the associated transmit message buffer is empty, and thus not scheduled for transmission. The CPU must clear the flag after a message is set up in the transmit buffer and is due for transmission. The MSCAN sets the flag after the message is sent successfully. The flag is also set by the MSCAN when the transmission request is successfully aborted due to a pending abort request (see Section 18.3.2.9, “MSCAN Transmitter Message Abort Request Register (CANTARQ)”). If not masked, a transmit interrupt is pending while this flag is set. Clearing a TXEx flag also clears the corresponding ABTAKx (see Section 18.3.2.10, “MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)”). When a TXEx flag is set, the corresponding ABTRQx bit is cleared (see Section 18.3.2.9, “MSCAN Transmitter Message Abort Request Register (CANTARQ)”). When listen-mode is active (see Section 18.3.2.2, “MSCAN Control Register 1 (CANCTL1)”) the TXEx flags cannot be cleared and no transmission is started. Read and write accesses to the transmit buffer will be blocked, if the corresponding TXEx bit is cleared (TXEx = 0) and the buffer is scheduled for transmission. 0 The associated message buffer is full (loaded with a message due for transmission) 1 The associated message buffer is empty (not scheduled)

18.3.2.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

This register contains the interrupt enable bits for the transmit buffer empty interrupt flags.

Module Base + 0x0007 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
W	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-11. MSCAN Transmitter Interrupt Enable Register (CANTIER)

[1] Read: Anytime
Write: Anytime when not in initialization mode

NOTE

> The CANTIER register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 18-13. CANTIER Register Field Descriptions

Field	Description
2-0 TXEIE[2:0]	**Transmitter Empty Interrupt Enable** 0 No interrupt request is generated from this event. 1 A transmitter empty (transmit buffer available for transmission) event causes a transmitter empty interrupt request.

18.3.2.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)

The CANTARQ register allows abort request of queued messages as described below.

Module Base + 0x0008 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
W	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-12. MSCAN Transmitter Message Abort Request Register (CANTARQ)

[1] Read: Anytime
Write: Anytime when not in initialization mode

NOTE

> The CANTARQ register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 18-14. CANTARQ Register Field Descriptions

Field	Description
2-0 ABTRQ[2:0]	**Abort Request** — The CPU sets the ABTRQx bit to request that a scheduled message buffer (TXEx = 0) be aborted. The MSCAN grants the request if the message has not already started transmission, or if the transmission is not successful (lost arbitration or error). When a message is aborted, the associated TXE (see Section 18.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and abort acknowledge flags (ABTAK, see Section 18.3.2.10, "MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)") are set and a transmit interrupt occurs if enabled. The CPU cannot reset ABTRQx. ABTRQx is reset whenever the associated TXE flag is set. 0 No abort request 1 Abort request pending

18.3.2.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

The CANTAAK register indicates the successful abort of a queued message, if requested by the appropriate bits in the CANTARQ register.

Module Base + 0x0009

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-13. MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

[1] Read: Anytime
Write: Unimplemented

NOTE

The CANTAAK register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1).

Table 18-15. CANTAAK Register Field Descriptions

Field	Description
2-0 ABTAK[2:0]	**Abort Acknowledge** — This flag acknowledges that a message was aborted due to a pending abort request from the CPU. After a particular message buffer is flagged empty, this flag can be used by the application software to identify whether the message was aborted successfully or was sent anyway. The ABTAKx flag is cleared whenever the corresponding TXE flag is cleared. 0 The message was not aborted. 1 The message was aborted.

18.3.2.11 MSCAN Transmit Buffer Selection Register (CANTBSEL)

The CANTBSEL register allows the selection of the actual transmit message buffer, which then will be accessible in the CANTXFG register space.

Module Base + 0x000A

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TX2	TX1	TX0
W	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-14. MSCAN Transmit Buffer Selection Register (CANTBSEL)

[1] Read: Find the lowest ordered bit set to 1, all other bits will be read as 0
Write: Anytime when not in initialization mode

NOTE

> The CANTBSEL register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK=1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 18-16. CANTBSEL Register Field Descriptions

Field	Description
2-0 TX[2:0]	**Transmit Buffer Select** — The lowest numbered bit places the respective transmit buffer in the CANTXFG register space (e.g., TX1 = 1 and TX0 = 1 selects transmit buffer TX0; TX1 = 1 and TX0 = 0 selects transmit buffer TX1). Read and write accesses to the selected transmit buffer will be blocked, if the corresponding TXEx bit is cleared and the buffer is scheduled for transmission (see Section 18.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)"). 0 The associated message buffer is deselected 1 The associated message buffer is selected, if lowest numbered bit

The following gives a short programming example of the usage of the CANTBSEL register:

To get the next available transmit buffer, application software must read the CANTFLG register and write this value back into the CANTBSEL register. In this example Tx buffers TX1 and TX2 are available. The value read from CANTFLG is therefore 0b0000_0110. When writing this value back to CANTBSEL, the Tx buffer TX1 is selected in the CANTXFG because the lowest numbered bit set to 1 is at bit position 1. Reading back this value out of CANTBSEL results in 0b0000_0010, because only the lowest numbered bit position set to 1 is presented. This mechanism eases the application software's selection of the next available Tx buffer.

- LDAA CANTFLG; value read is 0b0000_0110
- STAA CANTBSEL; value written is 0b0000_0110
- LDAA CANTBSEL; value read is 0b0000_0010

If all transmit message buffers are deselected, no accesses are allowed to the CANTXFG registers.

18.3.2.12 MSCAN Identifier Acceptance Control Register (CANIDAC)

The CANIDAC register is used for identifier acceptance control as described below.

Module Base + 0x000B

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
W	▨	▨			▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-15. MSCAN Identifier Acceptance Control Register (CANIDAC)

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1), except bits IDHITx, which are read-only

Table 18-17. CANIDAC Register Field Descriptions

Field	Description
5-4 IDAM[1:0]	**Identifier Acceptance Mode** — The CPU sets these flags to define the identifier acceptance filter organization (see Section 18.4.3, “Identifier Acceptance Filter”). Table 18-18 summarizes the different settings. In filter closed mode, no message is accepted such that the foreground buffer is never reloaded.
2-0 IDHIT[2:0]	Identifier Acceptance Hit Indicator — The MSCAN sets these flags to indicate an identifier acceptance hit (see Section 18.4.3, “Identifier Acceptance Filter”). Table 18-19 summarizes the different settings.

Table 18-18. Identifier Acceptance Mode Settings

IDAM1	IDAM0	Identifier Acceptance Mode
0	0	Two 32-bit acceptance filters
0	1	Four 16-bit acceptance filters
1	0	Eight 8-bit acceptance filters
1	1	Filter closed

Table 18-19. Identifier Acceptance Hit Indication

IDHIT2	IDHIT1	IDHIT0	Identifier Acceptance Hit
0	0	0	Filter 0 hit
0	0	1	Filter 1 hit
0	1	0	Filter 2 hit
0	1	1	Filter 3 hit
1	0	0	Filter 4 hit
1	0	1	Filter 5 hit
1	1	0	Filter 6 hit
1	1	1	Filter 7 hit

The IDHITx indicators are always related to the message in the foreground buffer (RxFG). When a message gets shifted into the foreground buffer of the receiver FIFO the indicators are updated as well.

18.3.2.13 MSCAN Reserved Register

This register is reserved for factory testing of the MSCAN module and is not available in normal system operating modes.

Module Base + 0x000C

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-16. MSCAN Reserved Register

[1] Read: Always reads zero in normal system operation modes
Write: Unimplemented in normal system operation modes

NOTE

Writing to this register when in special system operating modes can alter the MSCAN functionality.

18.3.2.14 MSCAN Miscellaneous Register (CANMISC)

This register provides additional features.

Module Base + 0x000D

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	BOHOLD
W	▨	▨	▨	▨	▨	▨	▨	
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-17. MSCAN Miscellaneous Register (CANMISC)

[1] Read: Anytime
Write: Anytime; write of ‘1’ clears flag; write of ‘0’ ignored

Table 18-20. CANMISC Register Field Descriptions

Field	Description
0 BOHOLD	**Bus-off State Hold Until User Request** — If BORM is set in MSCAN Control Register 1 (CANCTL1), this bit indicates whether the module has entered the bus-off state. Clearing this bit requests the recovery from bus-off. Refer to Section 18.5.2, “Bus-Off Recovery,” for details. 0 Module is not bus-off or recovery has been requested by user in bus-off state 1 Module is bus-off and holds this state until user request

18.3.2.15 MSCAN Receive Error Counter (CANRXERR)

This register reflects the status of the MSCAN receive error counter.

Module Base + 0x000E

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-18. MSCAN Receive Error Counter (CANRXERR)

[1] Read: Only when in sleep mode (SLPRQ = 1 and SLPAK = 1) or initialization mode (INITRQ = 1 and INITAK = 1)
Write: Unimplemented

NOTE

Reading this register when in any other mode other than sleep or initialization mode may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

18.3.2.16 MSCAN Transmit Error Counter (CANTXERR)

This register reflects the status of the MSCAN transmit error counter.

Module Base + 0x000F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-19. MSCAN Transmit Error Counter (CANTXERR)

[1] Read: Only when in sleep mode (SLPRQ = 1 and SLPAK = 1) or initialization mode (INITRQ = 1 and INITAK = 1)
Write: Unimplemented

NOTE

Reading this register when in any other mode other than sleep or initialization mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

18.3.2.17 MSCAN Identifier Acceptance Registers (CANIDAR0-7)

On reception, each message is written into the background receive buffer. The CPU is only signalled to read the message if it passes the criteria in the identifier acceptance and identifier mask registers (accepted); otherwise, the message is overwritten by the next message (dropped).

The acceptance registers of the MSCAN are applied on the IDR0–IDR3 registers (see Section 18.3.3.1, “Identifier Registers (IDR0–IDR3)”) of incoming messages in a bit by bit manner (see Section 18.4.3, “Identifier Acceptance Filter”).

For extended identifiers, all four acceptance and mask registers are applied. For standard identifiers, only the first two (CANIDAR0/1, CANIDMR0/1) are applied.

Module Base + 0x0010 to Module Base + 0x0013 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset	0	0	0	0	0	0	0	0

Figure 18-20. MSCAN Identifier Acceptance Registers (First Bank) — CANIDAR0–CANIDAR3

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-21. CANIDAR0–CANIDAR3 Register Field Descriptions

Field	Description
7-0 AC[7:0]	**Acceptance Code Bits** — AC[7:0] comprise a user-defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

Module Base + 0x0018 to Module Base + 0x001B Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset	0	0	0	0	0	0	0	0

Figure 18-21. MSCAN Identifier Acceptance Registers (Second Bank) — CANIDAR4–CANIDAR7

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-22. CANIDAR4–CANIDAR7 Register Field Descriptions

Field	Description
7-0 AC[7:0]	**Acceptance Code Bits** — AC[7:0] comprise a user-defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

18.3.2.18 MSCAN Identifier Mask Registers (CANIDMR0–CANIDMR7)

The identifier mask register specifies which of the corresponding bits in the identifier acceptance register are relevant for acceptance filtering. To receive standard identifiers in 32 bit filter mode, it is required to program the last three bits (AM[2:0]) in the mask registers CANIDMR1 and CANIDMR5 to “don’t care.” To receive standard identifiers in 16 bit filter mode, it is required to program the last three bits (AM[2:0]) in the mask registers CANIDMR1, CANIDMR3, CANIDMR5, and CANIDMR7 to “don’t care.”

Module Base + 0x0014 to Module Base + 0x0017

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset	0	0	0	0	0	0	0	0

Figure 18-22. MSCAN Identifier Mask Registers (First Bank) — CANIDMR0–CANIDMR3

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-23. CANIDMR0–CANIDMR3 Register Field Descriptions

Field	Description
7-0 AM[7:0]	**Acceptance Mask Bits** — If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted. 0 Match corresponding acceptance code register and identifier bits 1 Ignore corresponding acceptance code register bit

Module Base + 0x001C to Module Base + 0x001F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset	0	0	0	0	0	0	0	0

Figure 18-23. MSCAN Identifier Mask Registers (Second Bank) — CANIDMR4–CANIDMR7

[1] Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 18-24. CANIDMR4–CANIDMR7 Register Field Descriptions

Field	Description
7-0 AM[7:0]	**Acceptance Mask Bits** — If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted. 0 Match corresponding acceptance code register and identifier bits 1 Ignore corresponding acceptance code register bit

18.3.3 Programmer’s Model of Message Storage

The following section details the organization of the receive and transmit message buffers and the associated control registers.

To simplify the programmer interface, the receive and transmit message buffers have the same outline. Each message buffer allocates 16 bytes in the memory map containing a 13 byte data structure.

An additional transmit buffer priority register (TBPR) is defined for the transmit buffers. Within the last two bytes of this memory map, the MSCAN stores a special 16-bit time stamp, which is sampled from an internal timer after successful transmission or reception of a message. This feature is only available for transmit and receiver buffers, if the TIME bit is set (see Section 18.3.2.1, “MSCAN Control Register 0 (CANCTL0)”).

The time stamp register is written by the MSCAN. The CPU can only read these registers.

Table 18-25. Message Buffer Organization

Offset Address	Register	Access
0x00X0	IDR0 — Identifier Register 0	R/W
0x00X1	IDR1 — Identifier Register 1	R/W
0x00X2	IDR2 — Identifier Register 2	R/W
0x00X3	IDR3 — Identifier Register 3	R/W
0x00X4	DSR0 — Data Segment Register 0	R/W
0x00X5	DSR1 — Data Segment Register 1	R/W
0x00X6	DSR2 — Data Segment Register 2	R/W
0x00X7	DSR3 — Data Segment Register 3	R/W
0x00X8	DSR4 — Data Segment Register 4	R/W
0x00X9	DSR5 — Data Segment Register 5	R/W
0x00XA	DSR6 — Data Segment Register 6	R/W
0x00XB	DSR7 — Data Segment Register 7	R/W
0x00XC	DLR — Data Length Register	R/W
0x00XD	TBPR — Transmit Buffer Priority Register[1]	R/W
0x00XE	TSRH — Time Stamp Register (High Byte)	R
0x00XF	TSRL — Time Stamp Register (Low Byte)	R

[1] Not applicable for receive buffers

Figure 18-24 shows the common 13-byte data structure of receive and transmit buffers for extended identifiers. The mapping of standard identifiers into the IDR registers is shown in Figure 18-25.

All bits of the receive and transmit buffers are ‘x’ out of reset because of RAM-based implementation[1]. All reserved or unused bits of the receive and transmit buffers always read ‘x’.

---

1. Exception: The transmit buffer priority registers are 0 out of reset.

Figure 18-24. Receive/Transmit Message Buffer — Extended Identifier Mapping

Register Name		Bit 7	6	5	4	3	2	1	Bit0
0x00X0 IDR0	R W	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
0x00X1 IDR1	R W	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15
0x00X2 IDR2	R W	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
0x00X3 IDR3	R W	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
0x00X4 DSR0	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00X5 DSR1	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00X6 DSR2	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00X7 DSR3	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00X8 DSR4	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00X9 DSR5	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00XA DSR6	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00XB DSR7	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x00XC DLR	R W					DLC3	DLC2	DLC1	DLC0

Figure 18-24. Receive/Transmit Message Buffer — Extended Identifier Mapping (continued)

Register Name		Bit 7	6	5	4	3	2	1	Bit0

▨ = Unused, always read ‘x’

Read:

- For transmit buffers, anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).
- For receive buffers, only when RXF flag is set (see Section 18.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)”).

Write:

- For transmit buffers, anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).
- Unimplemented for receive buffers.

Reset: Undefined because of RAM-based implementation

Figure 18-25. Receive/Transmit Message Buffer — Standard Identifier Mapping

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
IDR0 0x00X0	R W	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
IDR1 0x00X1	R W	ID2	ID1	ID0	RTR	IDE (=0)	▨	▨	▨
IDR2 0x00X2	R W	▨	▨	▨	▨	▨	▨	▨	▨
IDR3 0x00X3	R W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unused, always read ‘x’

18.3.3.1 Identifier Registers (IDR0–IDR3)

The identifier registers for an extended format identifier consist of a total of 32 bits: ID[28:0], SRR, IDE, and RTR. The identifier registers for a standard format identifier consist of a total of 13 bits: ID[10:0], RTR, and IDE.

18.3.3.1.1 IDR0–IDR3 for Extended Identifier Mapping

Module Base + 0x00X0

	7	6	5	4	3	2	1	0
R W	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
Reset:	x	x	x	x	x	x	x	x

Figure 18-26. Identifier Register 0 (IDR0) — Extended Identifier Mapping

Table 18-26. IDR0 Register Field Descriptions — Extended

Field	Description
7-0 ID[28:21]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X1

	7	6	5	4	3	2	1	0
R W	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15
Reset:	x	x	x	x	x	x	x	x

Figure 18-27. Identifier Register 1 (IDR1) — Extended Identifier Mapping

Table 18-27. IDR1 Register Field Descriptions — Extended

Field	Description
7-5 ID[20:18]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
4 SRR	**Substitute Remote Request** — This fixed recessive bit is used only in extended format. It must be set to 1 by the user for transmission buffers and is stored as received on the CAN bus for receive buffers.
3 IDE	**ID Extended** — This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send. 0 Standard format (11 bit) 1 Extended format (29 bit)
2-0 ID[17:15]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X2

	7	6	5	4	3	2	1	0
R W	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
Reset:	x	x	x	x	x	x	x	x

Figure 18-28. Identifier Register 2 (IDR2) — Extended Identifier Mapping

Table 18-28. IDR2 Register Field Descriptions — Extended

Field	Description
7-0 ID[14:7]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X3

	7	6	5	4	3	2	1	0
R W	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
Reset:	x	x	x	x	x	x	x	x

Figure 18-29. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 18-29. IDR3 Register Field Descriptions — Extended

Field	Description
7-1 ID[6:0]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
0 RTR	**Remote Transmission Request** — This flag reflects the status of the remote transmission request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame

18.3.3.1.2 IDR0–IDR3 for Standard Identifier Mapping

Module Base + 0x00X0

	7	6	5	4	3	2	1	0
R W	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
Reset:	x	x	x	x	x	x	x	x

Figure 18-30. Identifier Register 0 — Standard Mapping

Table 18-30. IDR0 Register Field Descriptions — Standard

Field	Description
7-0 ID[10:3]	**Standard Format Identifier —** The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 18-31.

Module Base + 0x00X1

	7	6	5	4	3	2	1	0
R W	ID2	ID1	ID0	RTR	IDE (=0)	▨	▨	▨
Reset:	x	x	x	x	x	x	x	x

▨ = Unused; always read ‘x’

Figure 18-31. Identifier Register 1 — Standard Mapping

Table 18-31. IDR1 Register Field Descriptions

Field	Description
7-5 ID[2:0]	**Standard Format Identifier —** The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 18-30.
4 RTR	**Remote Transmission Request** — This flag reflects the status of the Remote Transmission Request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame
3 IDE	**ID Extended** — This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send. 0 Standard format (11 bit) 1 Extended format (29 bit)

Module Base + 0x00X2

	7	6	5	4	3	2	1	0
R W								
Reset:	x	x	x	x	x	x	x	x

▨ = Unused; always read ‘x’

Figure 18-32. Identifier Register 2 — Standard Mapping

Module Base + 0x00X3

	7	6	5	4	3	2	1	0
R W								
Reset:	x	x	x	x	x	x	x	x

▨ = Unused; always read ‘x’

Figure 18-33. Identifier Register 3 — Standard Mapping

18.3.3.2 Data Segment Registers (DSR0-7)

The eight data segment registers, each with bits DB[7:0], contain the data to be transmitted or received. The number of bytes to be transmitted or received is determined by the data length code in the corresponding DLR register.

Module Base + 0x00X4 to Module Base + 0x00XB

	7	6	5	4	3	2	1	0
R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Reset:	x	x	x	x	x	x	x	x

Figure 18-34. Data Segment Registers (DSR0–DSR7) — Extended Identifier Mapping

Table 18-32. DSR0–DSR7 Register Field Descriptions

Field	Description
7-0 DB[7:0]	**Data bits 7-0**

18.3.3.3 Data Length Register (DLR)

This register keeps the data length field of the CAN frame.

Module Base + 0x00XC

	7	6	5	4	3	2	1	0
R W	▨	▨	▨	▨	DLC3	DLC2	DLC1	DLC0
Reset:	x	x	x	x	x	x	x	x

▨ = Unused; always read “x”

Figure 18-35. Data Length Register (DLR) — Extended Identifier Mapping

Table 18-33. DLR Register Field Descriptions

Field	Description
3-0 DLC[3:0]	**Data Length Code Bits** — The data length code contains the number of bytes (data byte count) of the respective message. During the transmission of a remote frame, the data length code is transmitted as programmed while the number of transmitted data bytes is always 0. The data byte count ranges from 0 to 8 for a data frame. Table 18-34 shows the effect of setting the DLC bits.

Table 18-34. Data Length Codes

Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

18.3.3.4 Transmit Buffer Priority Register (TBPR)

This register defines the local priority of the associated message buffer. The local priority is used for the internal prioritization process of the MSCAN and is defined to be highest for the smallest binary number. The MSCAN implements the following internal prioritization mechanisms:

- All transmission buffers with a cleared TXEx flag participate in the prioritization immediately before the SOF (start of frame) is sent.
- The transmission buffer with the lowest local priority field wins the prioritization.

In cases of more than one buffer having the same lowest priority, the message buffer with the lower index number wins.

Module Base + 0x00XD Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	PRIO7	PRIO6	PRIO5	PRIO4	PRIO3	PRIO2	PRIO1	PRIO0
Reset:	0	0	0	0	0	0	0	0

Figure 18-36. Transmit Buffer Priority Register (TBPR)

[1] Read: Anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”)
Write: Anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”)

18.3.3.5 Time Stamp Register (TSRH–TSRL)

If the TIME bit is enabled, the MSCAN will write a time stamp to the respective registers in the active transmit or receive buffer right after the EOF of a valid message on the CAN bus (see Section 18.3.2.1, “MSCAN Control Register 0 (CANCTL0)”). In case of a transmission, the CPU can only read the time stamp after the respective transmit buffer has been flagged empty.

The timer value, which is used for stamping, is taken from a free running internal CAN bit clock. A timer overrun is not indicated by the MSCAN. The timer is reset (all bits set to 0) during initialization mode. The CPU can only read the time stamp registers.

Module Base + 0x00XE Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
W								
Reset:	x	x	x	x	x	x	x	x

Figure 18-37. Time Stamp Register — High Byte (TSRH)

[1] Read: For transmit buffers: Anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”). For receive buffers: Anytime when RXF is set.
Write: Unimplemented

Module Base + 0x00XF

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
W								
Reset:	x	x	x	x	x	x	x	x

Figure 18-38. Time Stamp Register — Low Byte (TSRL)

[1] Read: or transmit buffers: Anytime when TXEx flag is set (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”). For receive buffers: Anytime when RXF is set.
Write: Unimplemented

18.4 Functional Description

18.4.1 General

This section provides a complete functional description of the MSCAN.

18.4.2 Message Storage

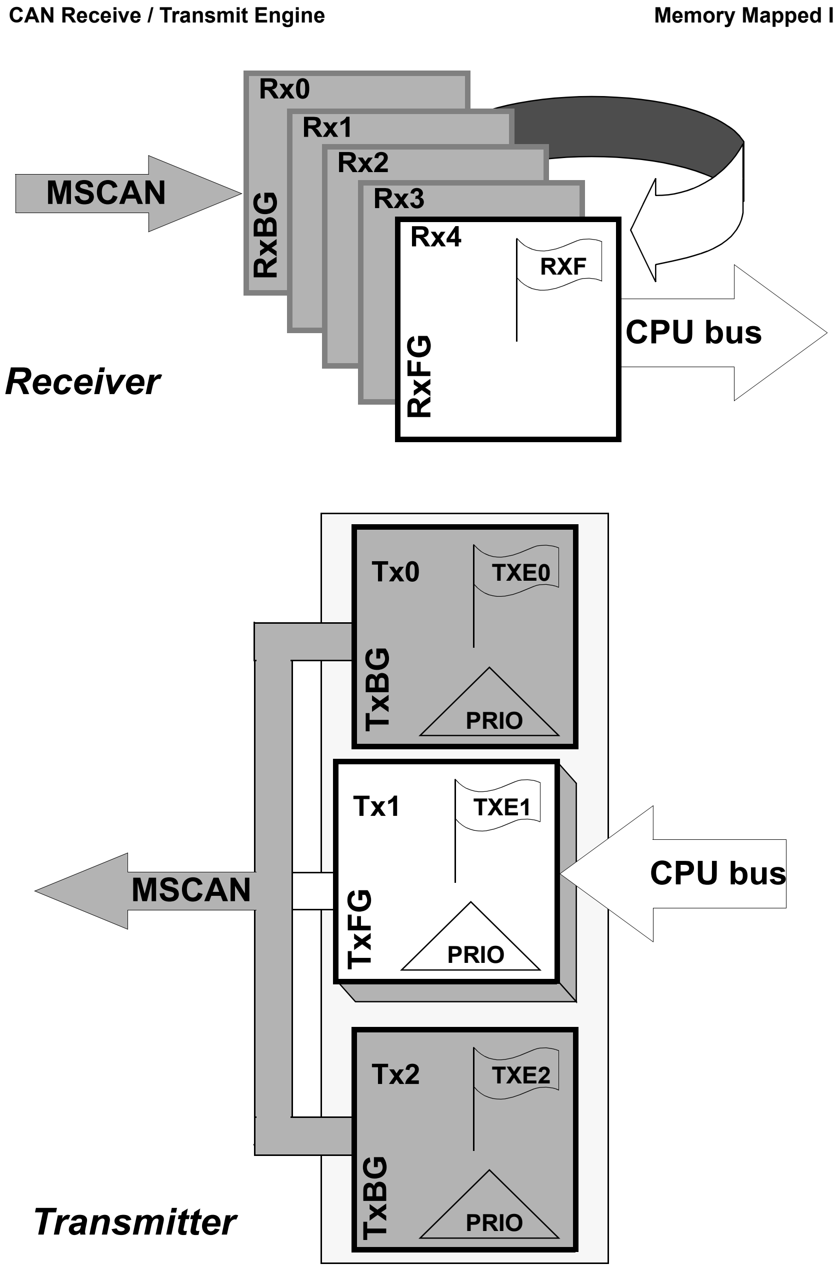


Figure 18-39. User Model for Message Buffer Organization

The MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

18.4.2.1 Message Transmit Background

Modern application layer software is built upon two fundamental assumptions:

- Any CAN node is able to send out a stream of scheduled messages without releasing the CAN bus between the two messages. Such nodes arbitrate for the CAN bus immediately after sending the previous message and only release the CAN bus in case of lost arbitration.
- The internal message queue within any CAN node is organized such that the highest priority message is sent out first, if more than one message is ready to be sent.

The behavior described in the bullets above cannot be achieved with a single transmit buffer. That buffer must be reloaded immediately after the previous message is sent. This loading process lasts a finite amount of time and must be completed within the inter-frame sequence (IFS) to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the CPU reacts with short latencies to the transmit interrupt.

A double buffer scheme de-couples the reloading of the transmit buffer from the actual message sending and, therefore, reduces the reactiveness requirements of the CPU. Problems can arise if the sending of a message is finished while the CPU re-loads the second buffer. No buffer would then be ready for transmission, and the CAN bus would be released.

At least three transmit buffers are required to meet the first of the above requirements under all circumstances. The MSCAN has three transmit buffers.

The second requirement calls for some sort of internal prioritization which the MSCAN implements with the “local priority” concept described in Section 18.4.2.2, “Transmit Structures.”

18.4.2.2 Transmit Structures

The MSCAN triple transmit buffer scheme optimizes real-time performance by allowing multiple messages to be set up in advance. The three buffers are arranged as shown in Figure 18-39.

All three buffers have a 13-byte data structure similar to the outline of the receive buffers (see Section 18.3.3, “Programmer’s Model of Message Storage”). An additional Transmit Buffer Priority Register (TBPR) contains an 8-bit local priority field (PRIO) (see Section 18.3.3.4, “Transmit Buffer Priority Register (TBPR)”). The remaining two bytes are used for time stamping of a message, if required (see Section 18.3.3.5, “Time Stamp Register (TSRH–TSRL)”).

To transmit a message, the CPU must identify an available transmit buffer, which is indicated by a set transmitter buffer empty (TXEx) flag (see Section 18.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”). If a transmit buffer is available, the CPU must set a pointer to this buffer by writing to the CANTBSEL register (see Section 18.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”). This makes the respective buffer accessible within the CANTXFG address space (see Section 18.3.3, “Programmer’s Model of Message Storage”). The algorithmic feature associated with the CANTBSEL register simplifies the transmit buffer selection. In addition, this scheme makes the handler software simpler because only one address area is applicable for the transmit process, and the required address space is minimized.

The CPU then stores the identifier, the control bits, and the data content into one of the transmit buffers. Finally, the buffer is flagged as ready for transmission by clearing the associated TXE flag.

The MSCAN then schedules the message for transmission and signals the successful transmission of the buffer by setting the associated TXE flag. A transmit interrupt (see Section 18.4.7.2, "Transmit Interrupt") is generated[1] when TXEx is set and can be used to drive the application software to re-load the buffer.

If more than one buffer is scheduled for transmission when the CAN bus becomes available for arbitration, the MSCAN uses the local priority setting of the three buffers to determine the prioritization. For this purpose, every transmit buffer has an 8-bit local priority field (PRIO). The application software programs this field when the message is set up. The local priority reflects the priority of this particular message relative to the set of messages being transmitted from this node. The lowest binary value of the PRIO field is defined to be the highest priority. The internal scheduling process takes place whenever the MSCAN arbitrates for the CAN bus. This is also the case after the occurrence of a transmission error.

When a high priority message is scheduled by the application software, it may become necessary to abort a lower priority message in one of the three transmit buffers. Because messages that are already in transmission cannot be aborted, the user must request the abort by setting the corresponding abort request bit (ABTRQ) (see Section 18.3.2.9, "MSCAN Transmitter Message Abort Request Register (CANTARQ)".) The MSCAN then grants the request, if possible, by:

1. Setting the corresponding abort acknowledge flag (ABTAK) in the CANTAAK register.
2. Setting the associated TXE flag to release the buffer.
3. Generating a transmit interrupt. The transmit interrupt handler software can determine from the setting of the ABTAK flag whether the message was aborted (ABTAK = 1) or sent (ABTAK = 0).

18.4.2.3 Receive Structures

The received messages are stored in a five stage input FIFO. The five message buffers are alternately mapped into a single memory area (see Figure 18-39). The background receive buffer (RxBG) is exclusively associated with the MSCAN, but the foreground receive buffer (RxFG) is addressable by the CPU (see Figure 18-39). This scheme simplifies the handler software because only one address area is applicable for the receive process.

All receive buffers have a size of 15 bytes to store the CAN control bits, the identifier (standard or extended), the data contents, and a time stamp, if enabled (see Section 18.3.3, "Programmer's Model of Message Storage").

The receiver full flag (RXF) (see Section 18.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)") signals the status of the foreground receive buffer. When the buffer contains a correctly received message with a matching identifier, this flag is set.

On reception, each message is checked to see whether it passes the filter (see Section 18.4.3, "Identifier Acceptance Filter") and simultaneously is written into the active RxBG. After successful reception of a valid message, the MSCAN shifts the content of RxBG into the receiver FIFO, sets the RXF flag, and

---

[1] The transmit interrupt occurs only if not masked. A polling scheme can be applied on TXEx also.

generates a receive interrupt[1] (see Section 18.4.7.3, “Receive Interrupt”) to the CPU. The user’s receive handler must read the received message from the RxFG and then reset the RXF flag to acknowledge the interrupt and to release the foreground buffer. A new message, which can follow immediately after the IFS field of the CAN frame, is received into the next available RxBG. If the MSCAN receives an invalid message in its RxBG (wrong identifier, transmission errors, etc.) the actual contents of the buffer will be over-written by the next message. The buffer will then not be shifted into the FIFO.

When the MSCAN module is transmitting, the MSCAN receives its own transmitted messages into the background receive buffer, RxBG, but does not shift it into the receiver FIFO, generate a receive interrupt, or acknowledge its own messages on the CAN bus. The exception to this rule is in loopback mode (see Section 18.3.2.2, “MSCAN Control Register 1 (CANCTL1)”) where the MSCAN treats its own messages exactly like all other incoming messages. The MSCAN receives its own transmitted messages in the event that it loses arbitration. If arbitration is lost, the MSCAN must be prepared to become a receiver.

An overrun condition occurs when all receive message buffers in the FIFO are filled with correctly received messages with accepted identifiers and another message is correctly received from the CAN bus with an accepted identifier. The latter message is discarded and an error interrupt with overrun indication is generated if enabled (see Section 18.4.7.5, “Error Interrupt”). The MSCAN remains able to transmit messages while the receiver FIFO is being filled, but all incoming messages are discarded. As soon as a receive buffer in the FIFO is available again, new valid messages will be accepted.

18.4.3 Identifier Acceptance Filter

The MSCAN identifier acceptance registers (see Section 18.3.2.12, “MSCAN Identifier Acceptance Control Register (CANIDAC)”) define the acceptable patterns of the standard or extended identifier (ID[10:0] or ID[28:0]). Any of these bits can be marked ‘don’t care’ in the MSCAN identifier mask registers (see Section 18.3.2.18, “MSCAN Identifier Mask Registers (CANIDMR0–CANIDMR7)”).

A filter hit is indicated to the application software by a set receive buffer full flag (RXF = 1) and three bits in the CANIDAC register (see Section 18.3.2.12, “MSCAN Identifier Acceptance Control Register (CANIDAC)”). These identifier hit flags (IDHIT[2:0]) clearly identify the filter section that caused the acceptance. They simplify the application software’s task to identify the cause of the receiver interrupt. If more than one hit occurs (two or more filters match), the lower hit has priority.

A very flexible programmable generic identifier acceptance filter has been introduced to reduce the CPU interrupt loading. The filter is programmable to operate in four different modes:

- Two identifier acceptance filters, each to be applied to:
  - The full 29 bits of the extended identifier and to the following bits of the CAN 2.0B frame:
    - Remote transmission request (RTR)
    - Identifier extension (IDE)
    - Substitute remote request (SRR)
  - The 11 bits of the standard identifier plus the RTR and IDE bits of the CAN 2.0A/B messages. This mode implements two filters for a full length CAN 2.0B compliant extended identifier. Although this mode can be used for standard identifiers, it is recommended to use the four or

---

[1] The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.

eight identifier acceptance filters.
Figure 18-40 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces a filter 0 hit. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces a filter 1 hit.

- Four identifier acceptance filters, each to be applied to:
  - The 14 most significant bits of the extended identifier plus the SRR and IDE bits of CAN 2.0B messages.
  - The 11 bits of the standard identifier, the RTR and IDE bits of CAN 2.0A/B messages.
    Figure 18-41 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces filter 0 and 1 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 2 and 3 hits.
- Eight identifier acceptance filters, each to be applied to the first 8 bits of the identifier. This mode implements eight independent filters for the first 8 bits of a CAN 2.0A/B compliant standard identifier or a CAN 2.0B compliant extended identifier.
  Figure 18-42 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces filter 0 to 3 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 4 to 7 hits.
- Closed filter. No CAN message is copied into the foreground buffer RxFG, and the RXF flag is never set.

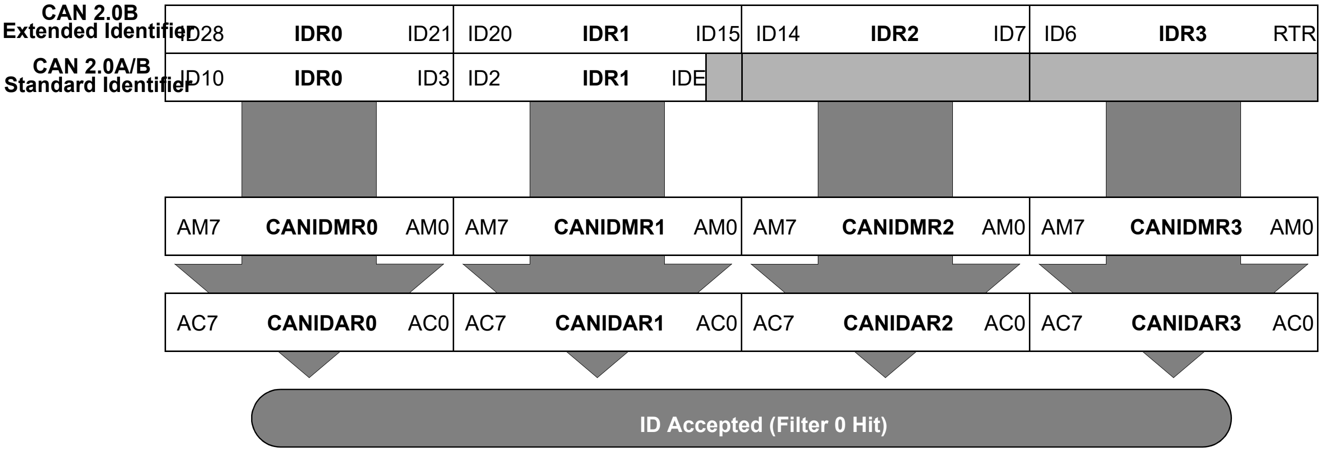


Figure 18-40. 32-bit Maskable Identifier Acceptance Filter

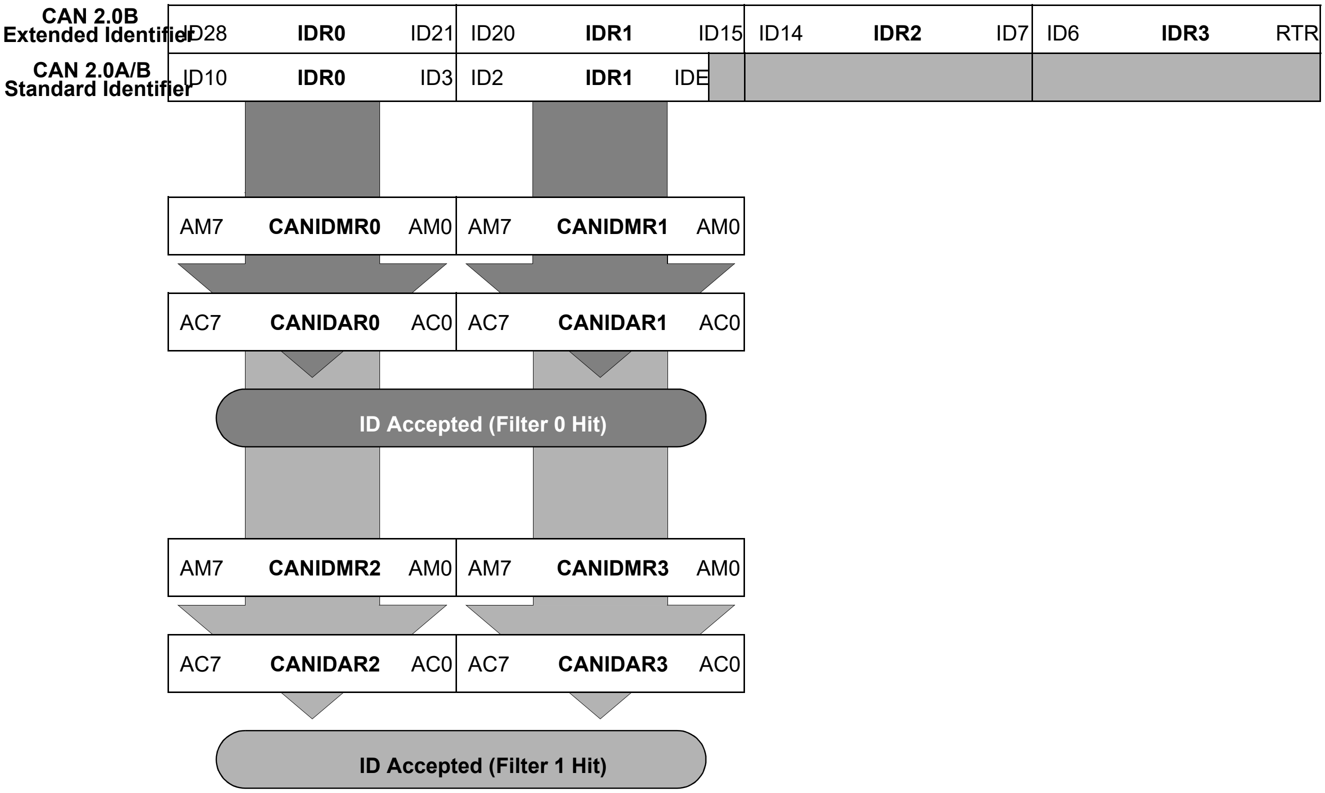


Figure 18-41. 16-bit Maskable Identifier Acceptance Filters

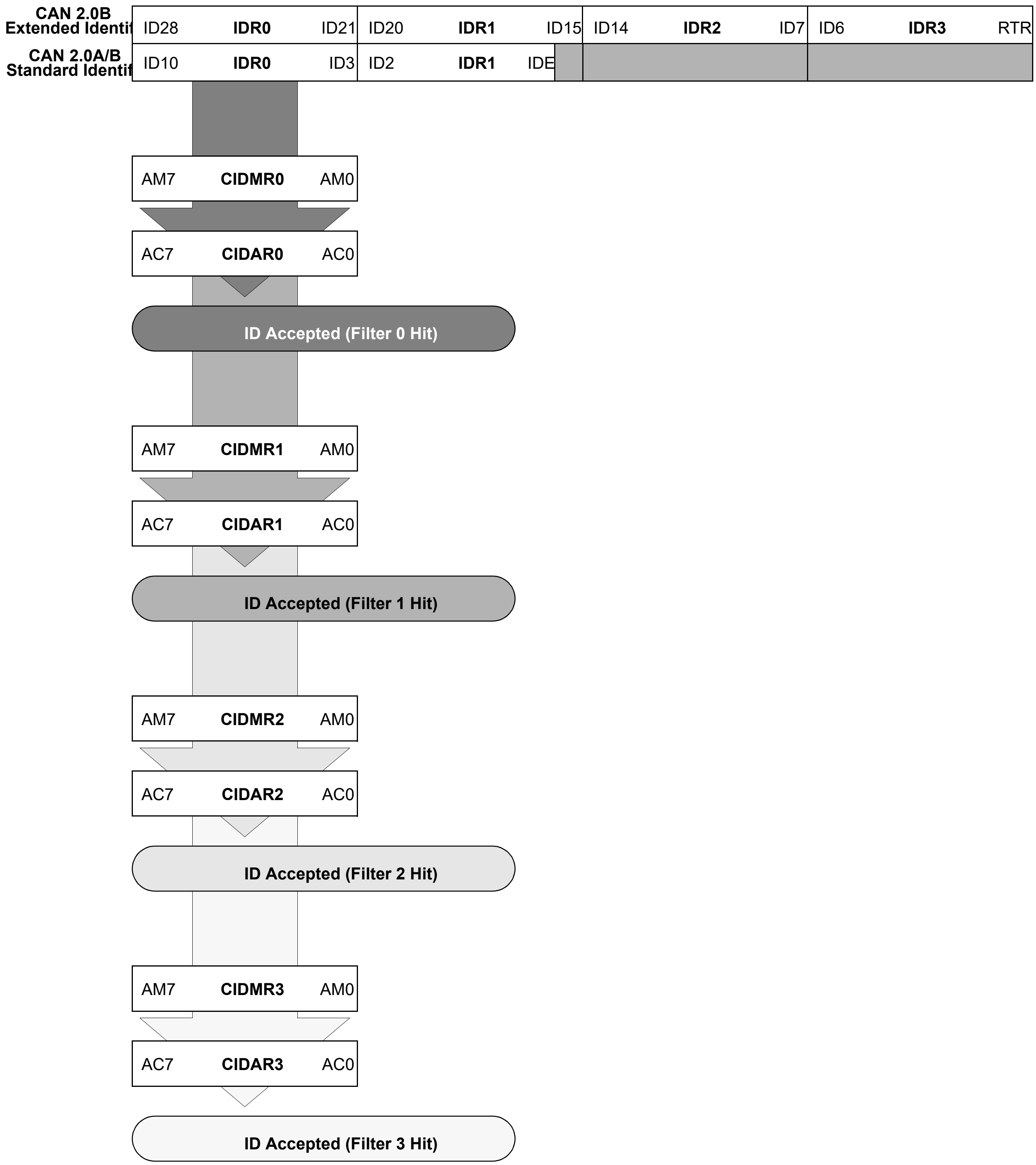


Figure 18-42. 8-bit Maskable Identifier Acceptance Filters

18.4.3.1 Protocol Violation Protection

The MSCAN protects the user from accidentally violating the CAN protocol through programming errors. The protection logic implements the following features:

- The receive and transmit error counters cannot be written or otherwise manipulated.
- All registers which control the configuration of the MSCAN cannot be modified while the MSCAN is on-line. The MSCAN has to be in Initialization Mode. The corresponding INITRQ/INITAK handshake bits in the CANCTL0/CANCTL1 registers (see Section 18.3.2.1, "MSCAN Control Register 0 (CANCTL0)") serve as a lock to protect the following registers:
  - — MSCAN control 1 register (CANCTL1)
  - — MSCAN bus timing registers 0 and 1 (CANBTR0, CANBTR1)
  - — MSCAN identifier acceptance control register (CANIDAC)
  - — MSCAN identifier acceptance registers (CANIDAR0–CANIDAR7)
  - — MSCAN identifier mask registers (CANIDMR0–CANIDMR7)
- The TXCAN is immediately forced to a recessive state when the MSCAN goes into the power down mode or initialization mode (see Section 18.4.5.6, "MSCAN Power Down Mode," and Section 18.4.4.5, "MSCAN Initialization Mode").
- The MSCAN enable bit (CANE) is writable only once in normal system operation modes, which provides further protection against inadvertently disabling the MSCAN.

18.4.3.2 Clock System

Figure 18-43 shows the structure of the MSCAN clock generation circuitry.

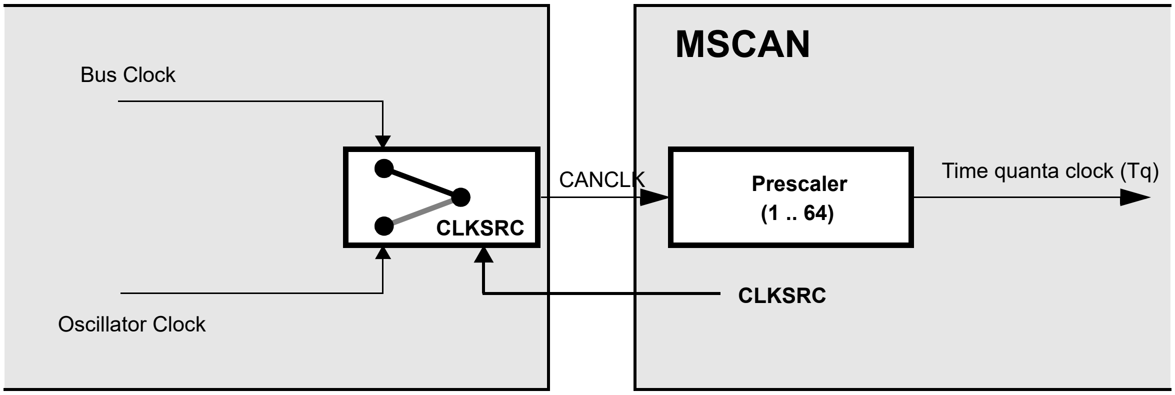


Figure 18-43. MSCAN Clocking Scheme

The clock source bit (CLKSRC) in the CANCTL1 register (18.3.2.2/18-557) defines whether the internal CANCLK is connected to the output of a crystal oscillator (oscillator clock) or to the bus clock.

The clock source has to be chosen such that the tight oscillator tolerance requirements (up to 0.4%) of the CAN protocol are met. Additionally, for high CAN bus rates (1 Mbps), a 45% to 55% duty cycle of the clock is required.

If the bus clock is generated from a PLL, it is recommended to select the oscillator clock rather than the bus clock due to jitter considerations, especially at the faster CAN bus rates.

For microcontrollers without a clock and reset generator (CRG), CANCLK is driven from the crystal oscillator (oscillator clock).

A programmable prescaler generates the time quanta (Tq) clock from CANCLK. A time quantum is the atomic unit of time handled by the MSCAN.

Eqn. 18-2

$$Tq = \frac{f_{CANCLK}}{(Prescaler\ \ value}$$

A bit time is subdivided into three segments as described in the Bosch CAN 2.0A/B specification. (see Figure 18-44):

- 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 PROP_SEG and the PHASE_SEG1 of the CAN standard. It can be programmed by setting the parameter TSEG1 to consist of 4 to 16 time quanta.
- Time Segment 2: This segment represents the PHASE_SEG2 of the CAN standard. It can be programmed by setting the TSEG2 parameter to be 2 to 8 time quanta long.

Eqn. 18-3

$$Bit\ \ Rate = \frac{f_{Tq}}{(number\ \ of\ \ Time\ \ Quanta)}$$

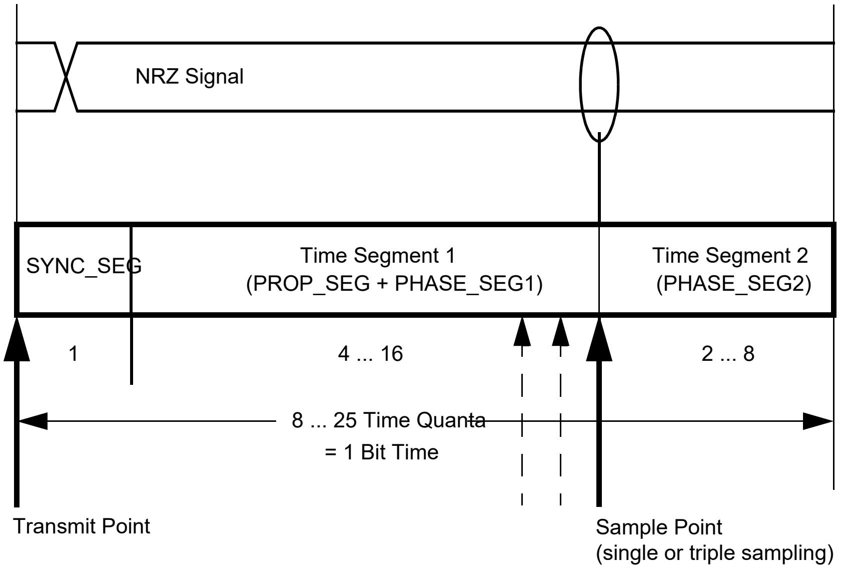


Figure 18-44. Segments within the Bit Time

Table 18-35. Time Segment Syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the CAN bus during this period.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node in receive mode samples the CAN bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The synchronization jump width (see the Bosch CAN 2.0A/B specification for details) can be programmed in a range of 1 to 4 time quanta by setting the SJW parameter.

The SYNC_SEG, TSEG1, TSEG2, and SJW parameters are set by programming the MSCAN bus timing registers (CANBTR0, CANBTR1) (see Section 18.3.2.3, "MSCAN Bus Timing Register 0 (CANBTR0)" and Section 18.3.2.4, "MSCAN Bus Timing Register 1 (CANBTR1)").

Table 18-36 gives an overview of the Bosch CAN 2.0A/B specification compliant segment settings and the related parameter values.

NOTE

It is the user's responsibility to ensure the bit time settings are in compliance with the CAN standard.

Table 18-36. Bosch CAN 2.0A/B Compliant Bit Time Segment Settings

Time Segment 1	TSEG1	Time Segment 2	TSEG2	Synchronization Jump Width	SJW
5 .. 10	4 .. 9	2	1	1 .. 2	0 .. 1
4 .. 11	3 .. 10	3	2	1 .. 3	0 .. 2
5 .. 12	4 .. 11	4	3	1 .. 4	0 .. 3
6 .. 13	5 .. 12	5	4	1 .. 4	0 .. 3
7 .. 14	6 .. 13	6	5	1 .. 4	0 .. 3
8 .. 15	7 .. 14	7	6	1 .. 4	0 .. 3
9 .. 16	8 .. 15	8	7	1 .. 4	0 .. 3

18.4.4 Modes of Operation

18.4.4.1 Normal System Operating Modes

The MSCAN module behaves as described within this specification in all normal system operating modes. Write restrictions exist for some registers.

18.4.4.2 Special System Operating Modes

The MSCAN module behaves as described within this specification in all special system operating modes. Write restrictions which exist on specific registers in normal modes are lifted for test purposes in special modes.

18.4.4.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like in normal system operating modes as described within this specification.

18.4.4.4 Listen-Only Mode

In an optional CAN bus monitoring mode (listen-only), the CAN node is able to receive valid data frames and valid remote frames, but it sends only “recessive” bits on the CAN bus. In addition, it cannot start a transmission.

If the MAC sub-layer is required to send a “dominant” bit (ACK bit, overload flag, or active error flag), the bit is rerouted internally so that the MAC sub-layer monitors this “dominant” bit, although the CAN bus may remain in recessive state externally.

18.4.4.5 MSCAN Initialization Mode

The MSCAN enters initialization mode when it is enabled (CANE=1).

When entering initialization mode during operation, any on-going transmission or reception is immediately aborted and synchronization to the CAN bus is lost, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations, the MSCAN immediately drives TXCAN into a recessive state.

NOTE

> The user is responsible for ensuring that the MSCAN is not active when initialization mode is entered. The recommended procedure is to bring the MSCAN into sleep mode (SLPRQ = 1 and SLPAK = 1) before setting the INITRQ bit in the CANCTL0 register. Otherwise, the abort of an on-going message can cause an error condition and can impact other CAN bus devices.

In initialization mode, the MSCAN is stopped. However, interface registers remain accessible. This mode is used to reset the CANCTL0, CANRFLG, CANRIER, CANTFLG, CANTIER, CANTARQ, CANTAAK, and CANTBSEL registers to their default values. In addition, the MSCAN enables the configuration of the CANBTR0, CANBTR1 bit timing registers; CANIDAC; and the CANIDAR, CANIDMR message filters. See Section 18.3.2.1, “MSCAN Control Register 0 (CANCTL0),” for a detailed description of the initialization mode.

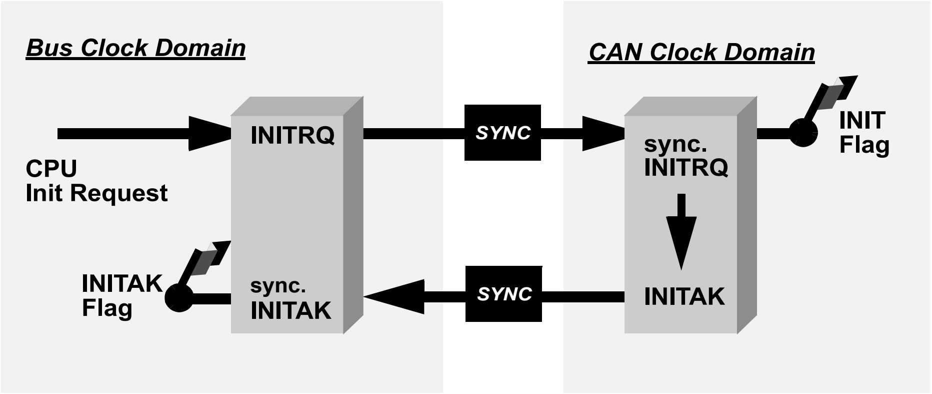


Figure 18-45. Initialization Request/Acknowledge Cycle

Due to independent clock domains within the MSCAN, INITRQ must be synchronized to all domains by using a special handshake mechanism. This handshake causes additional synchronization delay (see Figure 18-45).

If there is no message transfer ongoing on the CAN bus, the minimum delay will be two additional bus clocks and three additional CAN clocks. When all parts of the MSCAN are in initialization mode, the INITAK flag is set. The application software must use INITAK as a handshake indication for the request (INITRQ) to go into initialization mode.

NOTE

The CPU cannot clear INITRQ before initialization mode (INITRQ = 1 and INITAK = 1) is active.

18.4.5 Low-Power Options

If the MSCAN is disabled (CANE = 0), the MSCAN clocks are stopped for power saving.

If the MSCAN is enabled (CANE = 1), the MSCAN has two additional modes with reduced power consumption, compared to normal mode: sleep and power down mode. In sleep mode, power consumption is reduced by stopping all clocks except those to access the registers from the CPU side. In power down mode, all clocks are stopped and no power is consumed.

Table 18-37 summarizes the combinations of MSCAN and CPU modes. A particular combination of modes is entered by the given settings on the CSWAI and SLPRQ/SLPAK bits.

Table 18-37. CPU vs. MSCAN Operating Modes

CPU Mode	MSCAN Mode			
	Normal	Reduced Power Consumption		
		Sleep	Power Down	Disabled (CANE=0)
RUN	CSWAI = X[1] SLPRQ = 0 SLPAK = 0	CSWAI = X SLPRQ = 1 SLPAK = 1		CSWAI = X SLPRQ = X SLPAK = X
WAIT	CSWAI = 0 SLPRQ = 0 SLPAK = 0	CSWAI = 0 SLPRQ = 1 SLPAK = 1	CSWAI = 1 SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X
STOP			CSWAI = X SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X

[1] ‘X’ means don’t care.

18.4.5.1 Operation in Run Mode

As shown in Table 18-37, only MSCAN sleep mode is available as low power option when the CPU is in run mode.

18.4.5.2 Operation in Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode. If the CSWAI bit is set, additional power can be saved in power down mode because the CPU clocks are stopped. After leaving this power down mode, the MSCAN restarts and enters normal mode again.

While the CPU is in wait mode, the MSCAN can be operated in normal mode and generate interrupts (registers can be accessed via background debug mode).

18.4.5.3 Operation in Stop Mode

The STOP instruction puts the MCU in a low power consumption stand-by mode. In stop mode, the MSCAN is set in power down mode regardless of the value of the SLPRQ/SLPAK and CSWAI bits (Table 18-37).

18.4.5.4 MSCAN Normal Mode

This is a non-power-saving mode. Enabling the MSCAN puts the module from disabled mode into normal mode. In this mode the module can either be in initialization mode or out of initialization mode. See Section 18.4.4.5, “MSCAN Initialization Mode”.

18.4.5.5 MSCAN Sleep Mode

The CPU can request the MSCAN to enter this low power mode by asserting the SLPRQ bit in the CANCTL0 register. The time when the MSCAN enters sleep mode depends on a fixed synchronization delay and its current activity:

- If there are one or more message buffers scheduled for transmission (TXEx = 0), the MSCAN will continue to transmit until all transmit message buffers are empty (TXEx = 1, transmitted successfully or aborted) and then goes into sleep mode.
- If the MSCAN is receiving, it continues to receive and goes into sleep mode as soon as the CAN bus next becomes idle.
- If the MSCAN is neither transmitting nor receiving, it immediately goes into sleep mode.

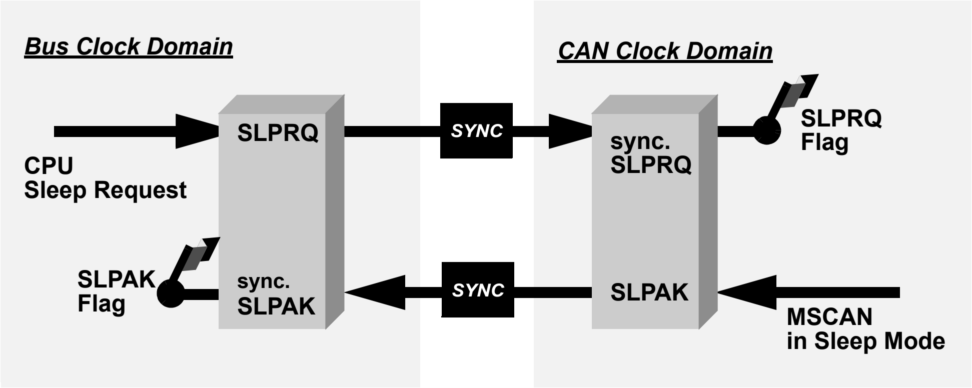


Figure 18-46. Sleep Request / Acknowledge Cycle

NOTE

The application software must avoid setting up a transmission (by clearing one or more TXEx flag(s)) and immediately request sleep mode (by setting SLPRQ). Whether the MSCAN starts transmitting or goes into sleep mode directly depends on the exact sequence of operations.

If sleep mode is active, the SLPRQ and SLPAK bits are set (Figure 18-46). The application software must use SLPAK as a handshake indication for the request (SLPRQ) to go into sleep mode.

When in sleep mode (SLPRQ = 1 and SLPAK = 1), the MSCAN stops its internal clocks. However, clocks that allow register accesses from the CPU side continue to run.

If the MSCAN is in bus-off state, it stops counting the 128 occurrences of 11 consecutive recessive bits due to the stopped clocks. TXCAN remains in a recessive state. If RXF = 1, the message can be read and RXF can be cleared. Shifting a new message into the foreground buffer of the receiver FIFO (RxFG) does not take place while in sleep mode.

It is possible to access the transmit buffers and to clear the associated TXE flags. No message abort takes place while in sleep mode.

If the WUPE bit in CANCTL0 is not asserted, the MSCAN will mask any activity it detects on CAN. RXCAN is therefore held internally in a recessive state. This locks the MSCAN in sleep mode. WUPE must be set before entering sleep mode to take effect.

The MSCAN is able to leave sleep mode (wake up) only when:

- CAN bus activity occurs and WUPE = 1

  or

- the CPU clears the SLPRQ bit

NOTE

The CPU cannot clear the SLPRQ bit before sleep mode (SLPRQ = 1 and SLPAK = 1) is active.

After wake-up, the MSCAN waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, if the MSCAN is woken-up by a CAN frame, this frame is not received.

The receive message buffers (RxFG and RxBG) contain messages if they were received before sleep mode was entered. All pending actions will be executed upon wake-up; copying of RxBG into RxFG, message aborts and message transmissions. If the MSCAN remains in bus-off state after sleep mode was exited, it continues counting the 128 occurrences of 11 consecutive recessive bits.

18.4.5.6 MSCAN Power Down Mode

The MSCAN is in power down mode (Table 18-37) when

- CPU is in stop mode

  or

- CPU is in wait mode and the CSWAI bit is set

When entering the power down mode, the MSCAN immediately stops all ongoing transmissions and receptions, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations to the above rule, the MSCAN immediately drives TXCAN into a recessive state.

NOTE

The user is responsible for ensuring that the MSCAN is not active when power down mode is entered. The recommended procedure is to bring the MSCAN into Sleep mode before the STOP or WAI instruction (if CSWAI is set) is executed. Otherwise, the abort of an ongoing message can cause an error condition and impact other CAN bus devices.

In power down mode, all clocks are stopped and no registers can be accessed. If the MSCAN was not in sleep mode before power down mode became active, the module performs an internal recovery cycle after powering up. This causes some fixed delay before the module enters normal mode again.

18.4.5.7 Disabled Mode

The MSCAN is in disabled mode out of reset (CANE=0). All module clocks are stopped for power saving, however the register map can still be accessed as specified.

18.4.5.8 Programmable Wake-Up Function

The MSCAN can be programmed to wake up from sleep or power down mode as soon as CAN bus activity is detected (see control bit WUPE in MSCAN Control Register 0 (CANCTL0). The sensitivity to existing CAN bus action can be modified by applying a low-pass filter function to the RXCAN input line (see control bit WUPM in Section 18.3.2.2, “MSCAN Control Register 1 (CANCTL1)”).

This feature can be used to protect the MSCAN from wake-up due to short glitches on the CAN bus lines. Such glitches can result from—for example—electromagnetic interference within noisy environments.

18.4.6 Reset Initialization

The reset state of each individual bit is listed in Section 18.3.2, “Register Descriptions,” which details all the registers and their bit-fields.

18.4.7 Interrupts

This section describes all interrupts originated by the MSCAN. It documents the enable bits and generated flags. Each interrupt is listed and described separately.

18.4.7.1 Description of Interrupt Operation

The MSCAN supports four interrupt vectors (see Table 18-38), any of which can be individually masked (for details see Section 18.3.2.6, “MSCAN Receiver Interrupt Enable Register (CANRIER)” to Section 18.3.2.8, “MSCAN Transmitter Interrupt Enable Register (CANTIER)”).

Refer to the device overview section to determine the dedicated interrupt vector addresses.

Table 18-38. Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Wake-Up Interrupt (WUPIF)	I bit	CANRIER (WUPIE)
Error Interrupts Interrupt (CSCIF, OVRIF)	I bit	CANRIER (CSCIE, OVRIE)
Receive Interrupt (RXF)	I bit	CANRIER (RXFIE)
Transmit Interrupts (TXE[2:0])	I bit	CANTIER (TXEIE[2:0])

18.4.7.2 Transmit Interrupt

At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. The TXEx flag of the empty message buffer is set.

18.4.7.3 Receive Interrupt

A message is successfully received and shifted into the foreground buffer (RxFG) of the receiver FIFO. This interrupt is generated immediately after receiving the EOF symbol. The RXF flag is set. If there are multiple messages in the receiver FIFO, the RXF flag is set as soon as the next message is shifted to the foreground buffer.

18.4.7.4 Wake-Up Interrupt

A wake-up interrupt is generated if activity on the CAN bus occurs during MSCAN sleep or power-down mode.

NOTE

This interrupt can only occur if the MSCAN was in sleep mode (SLPRQ = 1 and SLPAK = 1) before entering power down mode, the wake-up option is enabled (WUPE = 1), and the wake-up interrupt is enabled (WUPIE = 1).

18.4.7.5 Error Interrupt

An error interrupt is generated if an overrun of the receiver FIFO, error, warning, or bus-off condition occurs. MSCAN Receiver Flag Register (CANRFLG) indicates one of the following conditions:

- **Overrun** — An overrun condition of the receiver FIFO as described in Section 18.4.2.3, “Receive Structures,” occurred.
- **CAN Status Change** — The actual value of the transmit and receive error counters control the CAN bus state of the MSCAN. As soon as the error counters skip into a critical range (Tx/Rx-warning, Tx/Rx-error, bus-off) the MSCAN flags an error condition. The status change, which caused the error condition, is indicated by the TSTAT and RSTAT flags (see Section 18.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)” and Section 18.3.2.6, “MSCAN Receiver Interrupt Enable Register (CANRIER)”).

18.4.7.6 Interrupt Acknowledge

Interrupts are directly associated with one or more status flags in either the MSCAN Receiver Flag Register (CANRFLG) or the MSCAN Transmitter Flag Register (CANTFLG). Interrupts are pending as long as one of the corresponding flags is set. The flags in CANRFLG and CANTFLG must be reset within the interrupt handler to handshake the interrupt. The flags are reset by writing a 1 to the corresponding bit position. A flag cannot be cleared if the respective condition prevails.

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.

18.5 Initialization/Application Information

18.5.1 MSCAN initialization

The procedure to initially start up the MSCAN module out of reset is as follows:

1. Assert CANE
2. Write to the configuration registers in initialization mode
3. Clear INITRQ to leave initialization mode

If the configuration of registers which are only writable in initialization mode shall be changed:

1. Bring the module into sleep mode by setting SLPRQ and awaiting SLPAK to assert after the CAN bus becomes idle.
2. Enter initialization mode: assert INITRQ and await INITAK
3. Write to the configuration registers in initialization mode
4. Clear INITRQ to leave initialization mode and continue

18.5.2 Bus-Off Recovery

The bus-off recovery is user configurable. The bus-off state can either be left automatically or on user request.

For reasons of backwards compatibility, the MSCAN defaults to automatic recovery after reset. In this case, the MSCAN will become error active again after counting 128 occurrences of 11 consecutive recessive bits on the CAN bus (see the Bosch CAN 2.0 A/B specification for details).

If the MSCAN is configured for user request (BORM set in MSCAN Control Register 1 (CANCTL1)), the recovery from bus-off starts after both independent events have become true:

- 128 occurrences of 11 consecutive recessive bits on the CAN bus have been monitored
- BOHOLD in MSCAN Miscellaneous Register (CANMISC) has been cleared by the user

These two events may occur in any order.

Chapter 19
Digital Analog Converter (DAC_8B5V_V2)

19.1 Revision History

Table 19-1. Revision History Table

Rev. No. (Item No.)	Data	Sections Affected	Substantial Change(s)
1.4	17-Nov.-10	19.2.2	Update the behavior of the DACU pin during stop mode
1.5	29-Aug.-13	19.2.2, 19.3	added note about settling time added link to DACM register inside section 19.3
2.0	30-Jan.-14	19.2.3, 19.4.2.1, 19.5.4	added mode “Internal DAC only“
2.1	13-May.-15	Figure 19-5	correct read value of reserved register, Figure 19-5

Glossary

Table 19-2. Terminology

Term	Meaning
DAC	Digital to Analog Converter
VRL	Low Reference Voltage
VRH	High Reference Voltage
FVR	Full Voltage Range
SSC	Special Single Chip

19.2 Introduction

The DAC_8B5V module is a digital to analog converter. The converter works with a resolution of 8 bit and generates an output voltage between VRL and VRH.

The module consists of configuration registers and two analog functional units, a DAC resistor network and an operational amplifier.

The configuration registers provide all required control bits for the DAC resistor network and for the operational amplifier.

The DAC resistor network generates the desired analog output voltage. The unbuffered voltage from the DAC resistor network output can be routed to the external DACU pin. When enabled, the buffered voltage from the operational amplifier output is available on the external AMP pin.

The operational amplifier is also stand alone usable.

Figure 19-1 shows the block diagram of the DAC_8B5V module.

19.2.1 Features

The DAC_8B5V module includes these distinctive features:

- 1 digital-analog converter channel with:
  - 8 bit resolution
  - full and reduced output voltage range
  - buffered or unbuffered analog output voltage usable
- operational amplifier stand alone usable

19.2.2 Modes of Operation

The DAC_8B5V module behaves as follows in the system power modes:

1. CPU run mode

   The functionality of the DAC_8B5V module is available.

2. CPU stop mode

   Independent from the mode settings, the operational amplifier is disabled, switch S1 and S2 are open.

   If the “Unbuffered DAC” mode was used before entering stop mode, then the DACU pin will reach VRH voltage level during stop mode.

   The content of the configuration registers is unchanged.

NOTE

After enabling and after return from CPU stop mode, the DAC_8B5V module needs a settling time to get fully operational, see Settling time specification of dac_8b5V_analog_ll18.

19.2.3 Block Diagram

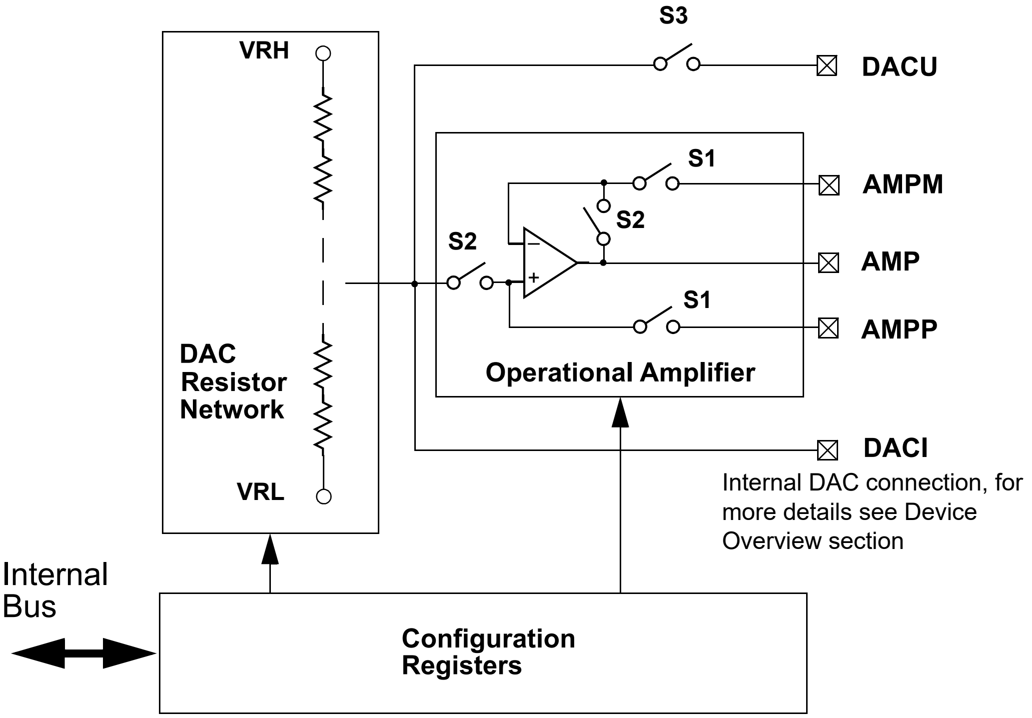


Figure 19-1. DAC_8B5V Block Diagram

19.3 External Signal Description

This section lists the name and description of all external ports.

19.3.1 DACU Output Pin

This analog pin drives the unbuffered analog output voltage from the DAC resistor network output, if the according mode is selected, see register bit DACM[2:0].

19.3.2 AMP Output Pin

This analog pin is used for the buffered analog output voltage from the operational amplifier output, if the according mode is selected, see register bit DACM[2:0].

19.3.3 AMPP Input Pin

This analog input pin is used as input signal for the operational amplifier positive input pin, if the according mode is selected, see register bit DACM[2:0].

NOTE

For connectivity of VRH and VRL please refer to Section 1.9.2, “DAC Connectivity

19.3.4 AMPM Input Pin

This analog pin is used as input for the operational amplifier negative input pin, if the according mode is selected, see register bit DACM[2:0].

19.4 Memory Map and Register Definition

This sections provides the detailed information of all registers for the DAC_8B5V module.

19.4.1 Register Summary

Figure 19-2 shows the summary of all implemented registers inside the DAC_8B5V module.

NOTE

> Register Address = Module Base Address + Address Offset, where the Module Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 DACCTL	R	FVR	DRIVE	0	0	0	DACM[2:0]		
	W			▨	▨	▨			
0x0001 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0002 DACVOL	R	VOLTAGE[7:0]							
	W								
0x0003 - 0x0006 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0007 Reserved	R	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
	W	▨							

▨ = Unimplemented

Figure 19-2. DAC_8B5V Register Summary

19.4.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

19.4.2.1 Control Register (DACCTL)

Module Base + 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	FVR	DRIVE	0	0	0	DACM[2:0]		
W			▨	▨	▨			
Reset	1	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 19-3. Control Register (DACCTL)

[1] Read: Anytime
Write: Anytime

Table 19-3. DACCTL Field Description

Field	Description
7 FVR	**Full Voltage Range** — This bit defines the voltage range of the DAC. 0 DAC resistor network operates with the reduced voltage range 1 DAC resistor network operates with the full voltage range **Note:** For more details see Section 19.5.8, “Analog output voltage calculation”.
6 DRIVE	**Drive Select** — This bit selects the output drive capability of the operational amplifier, see electrical Spec. for more details. 0 Low output drive for high resistive loads 1 High output drive for low resistive loads
2:0 DACM[2:0]	**Mode Select** — These bits define the mode of the DAC. A write access with an unsupported mode will be ignored. 000 Off 001 Operational Amplifier 010 Internal DAC only 100 Unbuffered DAC 101 Unbuffered DAC with Operational Amplifier 111 Buffered DAC other Reserved

19.4.2.2 Analog Output Voltage Level Register (DACVOL)

Module Base + 0x0002 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	VOLTAGE[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 19-4. Analog Output Voltage Level Register (DACVOL)

[1] Read: Anytime
Write: Anytime

Table 19-4. DACVOL Field Description

Field	Description
7:0 VOLTAGE[7:0]	**VOLTAGE** — This register defines (together with the FVR bit) the analog output voltage. For more detail see Equation 19-1 and Equation 19-2.

19.4.2.3 Reserved Register

Module Base + 0x0007 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W								
Reset	0	0	0	0	0	0	0	0

Figure 19-5. Reserved Registerfv_dac_8b5v_RESERVED

[1] Read: Anytime
Write: Only in special mode

NOTE

This reserved register bits are designed for factory test purposes only and are not intended for general user access. Writing to this register when in special modes can alter the modules functionality.

19.5 Functional Description

19.5.1 Functional Overview

The DAC resistor network and the operational amplifier can be used together or stand alone. Following modes are supported:

Table 19-5. DAC Modes of Operation

DACM[2:0]		Description			
		Submodules		Output	
		DAC resistor network	Operational Amplifier	DACU	AMP
Off	000	disabled	disabled	disconnected	disconnected
Operational amplifier	001	disabled	enabled	disabled	depend on AMPP and AMPM input
Internal DAC only	010	enabled	disabled	disconnected	disconnected
Unbuffered DAC	100	enabled	disabled	unbuffered resistor output voltage	disconnected
Unbuffered DAC with Operational amplifier	101	enabled	enabled	unbuffered resistor output voltage	depend on AMPP and AMPM input
Buffered DAC	111	enabled	enabled	disconnected	buffered resistor output voltage

The DAC resistor network itself can work on two different voltage ranges:

Table 19-6. DAC Resistor Network Voltage ranges

DAC Mode	Description
Full Voltage Range (FVR)	DAC resistor network provides a output voltage over the complete input voltage range, default after reset
Reduced Voltage Range	DAC resistor network provides a output voltage over a reduced input voltage range

Table 19-7 shows the control signal decoding for each mode. For more detailed mode description see the sections below.

Table 19-7. DAC Control Signals

DACM		DAC resistor network	Operational Amplifier	Switch S1	Switch S2	Switch S3
Off	000	disabled	disabled	open	open	open
Operational amplifier	001	disabled	enabled	closed	open	open
Internal DAC only	010	enabled	disabled	open	open	open
Unbuffered DAC	100	enabled	disabled	open	open	closed

Table 19-7. DAC Control Signals

DACM		DAC resistor network	Operational Amplifier	Switch S1	Switch S2	Switch S3
Unbuffered DAC with Operational amplifier	101	enabled	enabled	closed	open	closed
Buffered DAC	111	enabled	enabled	open	closed	open

19.5.2 Mode “Off”

The “Off” mode is the default mode after reset and is selected by DACCTL.DACM[2:0] = 0x0. During this mode the DAC resistor network and the operational amplifier are disabled and all switches are open. This mode provides the lowest power consumption. For decoding of the control signals see Table 19-7.

19.5.3 Mode “Operational Amplifier”

The “Operational Amplifier” mode is selected by DACCTL.DACM[2:0] = 0x1. During this mode the operational amplifier can be used independent from the DAC resister network. All required amplifier signals, AMP, AMPP and AMPM are available on the pins. The DAC resistor network output is disconnected from the DACU pin. The connection between the amplifier output and the negative amplifier input is open. For decoding of the control signals see Table 19-7.

19.5.4 Mode “Internal DAC only“

The “Internal DAC only” mode is selected by DACCNTL.DACM[2:0] = 0x2. During this mode the unbuffered analog voltage from the DAC resistor network is available on the internal connection DACI. The DACU output pin is disconnected. The operational amplifier is disabled and the operational amplifier signals are disconnected from the AMP pins. For decoding of the control signals see Table 19-7.

19.5.5 Mode “Unbuffered DAC”

The “Unbuffered DAC” mode is selected by DACCNTL.DACM[2:0] = 0x4. During this mode the unbuffered analog voltage from the DAC resistor network output is available on the DACU output pin. The operational amplifier is disabled and the operational amplifier signals are disconnected from the AMP pins. The unbuffered analog voltage from the DAC resistor network is available on the internal connection DACI. For decoding of the control signals see Table 19-7.

19.5.6 Mode “Unbuffered DAC with Operational Amplifier”

The “Unbuffered DAC with Operational Amplifier” mode is selected by DACCTL.DACM[2:0] = 0x5. During this mode the DAC resistor network and the operational amplifier are enabled and usable independent from each other. The unbuffered analog voltage from the DAC resistor network output is available on the DACU output pin.

The operational amplifier is disconnected from the DAC resistor network. All required amplifier signals, AMP, AMPP and AMPM are available on the pins. The connection between the amplifier output and the negative amplifier input is open. The unbuffered analog voltage from the DAC resistor network is available on the internal connection DACI. For decoding of the control signals see Table 19-7.

19.5.7 Mode “Buffered DAC”

The “Buffered DAC” mode is selected by DACCTL.DACM[2:0] = 0x7. During this is mode the DAC resistor network and the operational amplifier are enabled. The analog output voltage from the DAC resistor network output is buffered by the operational amplifier and is available on the AMP output pin.

The DAC resistor network output is disconnected from the DACU pin. The unbuffered analog voltage from the DAC resistor network is available on the internal connection DACI. For the decoding of the control signals see Table 19-7.

19.5.8 Analog output voltage calculation

The DAC can provide an analog output voltage in two different voltage ranges:

- FVR = 0, reduced voltage range

  The DAC generates an analog output voltage inside the range from 0.1 x (VRH - VRL) + VRL to 0.9 x (VRH-VRL) + VRL with a resolution ((VRH-VRL) x 0.8) / 256, see equation below:

$$\text{analog output voltage} = \text{VOLATGE}[7:0] \times ((\text{VRH-VRL}) \times 0.8) / 256) + 0.1 \times (\text{VRH-VRL}) + \text{VRL} \qquad \textit{Eqn. 19-1}$$

- FVR = 1, full voltage range

  The DAC generates an analog output voltage inside the range from VRL to VRH with a resolution (VRH-VRL) / 256, see equation below:

$$\text{analog output voltage} = \text{VOLTAGE}[7:0] \times (\text{VRH-VRL}) / 256 + \text{VRL} \qquad \textit{Eqn. 19-2}$$

See Table 19-8 for an example for VRL = 0.0 V and VRH = 5.0 V.

Table 19-8. Analog output voltage calculation

FVR	min. voltage	max. voltage	Resolution	Equation
0	0.5V	4.484V	15.625mV	**VOLTAGE[7:0] x (4.0V) / 256) + 0.5V**
1	0.0V	4.980V	19.531mV	**VOLTAGE[7:0] x (5.0V) / 256**

Chapter 20 5V Analog Comparator (ACMPV2)

Table 20-1. Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V02.01	22 Mar 2013		• Various changes based on shared review feedback • Separated ACIE and ACIF in two registers • Made ACPSEL and ACNSEL write only while module disabled • Changed initialization delay to 127 bus clock cycles •
V02.02	29 Feb 2013		• Minor corrections based on feedback from shared review
V02.03	25 Jun 2013		• Corrected ACMPO behavior in shutdown mode • Corrected ACMPC1 write restriction • Made input change during initialization delay description non-ambiguous

20.1 Introduction

The analog comparator (ACMP) provides a circuit for comparing two analog voltages. The comparator circuit is designed to operate across the full range between 0V and VDDA supply voltage (rail-to-rail operation).

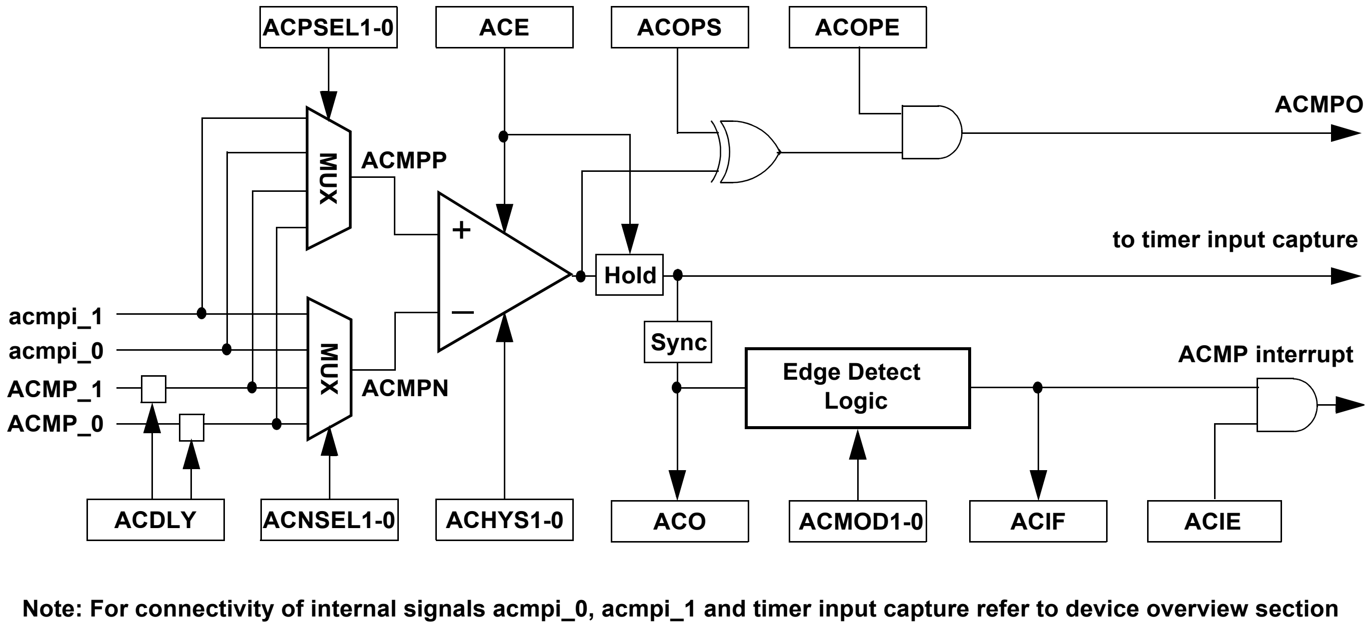
20.2 Features

The ACMP has the following features:

- 0V to VDDA supply rail-to-rail inputs
- Low offset
- Up to 4 inputs selectable as inverting and non-inverting comparator inputs:
  - 2 low-impedance inputs with selectable low pass filter for external pins
  - 2 high-impedance inputs with fixed filter for SoC-internal signals
- Selectable hysteresis
- Selectable interrupt on rising edge, falling edge, or rising and falling edges of comparator output
- Option to output comparator signal on an external pin with selectable polarity
- Support for triggering timer input capture events
- Operational over supply range from 3V-5% to 5V+10%
- Temperature range (T_J): -40°C to 175°C

20.3 Block Diagram

The block diagram of the ACMP is shown in Figure 20-1.



Note: For connectivity of internal signals acmpi_0, acmpi_1 and timer input capture refer to device overview section

Figure 20-1. ACMP Block Diagram

20.4 External Signals

The ACMP has two external analog input signals, ACMP_0 and ACMP_1 which can be configured to be used with the inverting and non-inverting input, and one digital output, ACMPO.

The inputs can accept a voltage that varies across the full 0V to VDDA voltage range. The ACMP monitors these inputs independent of any other functions which may be shared on these pins (GPIO, ADC).

The ACMP output signal can optionally be driven out on the external pin ACMPO.

20.4.1 Internal Signals

In addition to the external inputs there are two internal inputs acmpi_0 and acmpi_1 available for use with predefined SoC-internal analog signals. Refer to device-level section for connectivity.

20.5 Modes of Operation

1. Normal Mode
   The ACMP is operational when enabled (ACE=1) and not in STOP mode. It maintains operation throughout WAIT mode. An initialization delay of 127 bus clock cycles must be accounted for

when entering normal mode after leaving shutdown mode. During this time the comparator output path to all subsequent logic (ACO, ACIF, timer link) is held in its current state. Refer to I_{ACMP_run} for current consumption of ACMP during operation.

2. Shutdown Mode
   The ACMP is held in shutdown mode either when disabled (ACE=0) or during STOP mode. When leaving normal mode the current state of the comparator will be maintained. ACMPO drives zero in shutdown mode if not inverted (ACOPS=0). In this mode the current consumption is reduced to I_{ACMP_off}.

20.6 Memory Map and Register Definition

20.6.1 Register Map

Table 20-2 shows the ACMP register map.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Table 20-2. ACMP Register Map

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	ACMPC0	R	ACE	ACOPE	ACOPS	ACDLY	ACHYS1-0		ACMOD1-0	
		W								
0x0001	ACMPC1	R	0	0	ACPSEL1-0		0	0	ACNSEL1-0	
		W								
0x0002	ACMPC2	R	0	0	0	0	0	0	0	ACIE
		W								
0x0003	ACMPS	R	ACO	0	0	0	0	0	0	ACIF
		W								
0x0004– 0x0007	Reserved	R	0	0	0	0	0	0	0	0
		W								

20.6.2 Register Descriptions

20.6.2.1 ACMP Control Register 0 (ACMPC0)

Module Base + 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	ACE	ACOPE	ACOPS	ACDLY	ACHYS1-0		ACMOD1-0	
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-2. ACMP Control Register (ACMPC0)

[1] Read: Anytime
Write: Anytime

Table 20-3. ACMPC0 Register Field Descriptions

Field	Description
7 ACE	**ACMP Enable —** This bit enables the ACMP module. When set the related input pins are connected with the low pass input filters. **Note:** After setting ACE to 1 an initialization delay of 127 bus clock cycles must be accounted for. During this time the comparator output path to all subsequent logic (ACO, ACIF, timer link) is held at its current state. When setting ACE to 0 the current state of the comparator will be maintained.For ACMPO a delay of $t_{ACMP\_delay\_en}$ must be accounted for. 0 ACMP disabled 1 ACMP enabled
6 ACOPE	**ACMP Output Pin Enable —** This bit enables the ACMP output on external ACMPO pin. 0 ACMP output pin disabled 1 ACMP output is driven out to ACMPO
5 ACOPS	**ACMP Output Polarity Select** — This bit selects the output polarity on ACMPO. 0 ACMPO is ACMP output 1 ACMPO is ACMP output inverted
4 ACDLY	**ACMP Input Filter Select for Inputs ACMP_0 and ACMP_1** — This bit selects the analog input filter characteristics resulting in a signal propagation delay of t_{ACMP_delay}. 0 Select input filter with low speed characteristics 1 Select input filter with high speed characteristics

Table 20-3. ACMPC0 Register Field Descriptions (continued)

Field	Description
3-2 ACHYS1-0	**ACMP Hysteresis** — These bits select the ACMP hysteresis V_{ACMP_hyst}. 00 Select smallest hysteresis 01 Select small hysteresis 10 Select large hysteresis 11 Select largest hysteresis
1-0 ACMOD 1-0	**ACMP Mode** — These bits select the type of compare event to set ACIF. 00 Flag setting disabled 01 Flag setting on output rising edge 10 Flag setting on output falling edge 11 Flag setting on output rising or falling edge

20.6.2.2 ACMP Control Register 1 (ACMPC1)

Module Base + 0x0001 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	ACPSEL1-0		0	0	ACNSEL1-0	
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-3. ACMP Control Register (ACMPC1)

[1] Read: Anytime
Write: Only if ACE=0

Table 20-4. ACMPC1 Routing Register Field Descriptions

Field	Description
5-4 ACPSEL1-0	**ACMP Positive Input Select** — These bits select the ACMP non-inverting input connected to ACMPP. 11 acmpi_1 10 acmpi_0 01 ACMP_1 00 ACMP_0
1-0 ACNSEL1-0	**ACMP Negative Input Select** — These bits select the ACMP inverting input connected to ACMPN. 11 acmpi_1 10 acmpi_0 01 ACMP_1 00 ACMP_0

20.6.2.3 ACMP Control Register 2 (ACMPC2)

Module Base + 0x0002 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	ACIE
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-4. ACMP Control Register (ACMPC2)

[1] Read: Anytime
Write: Anytime

Table 20-5. ACMPC2 Register Field Descriptions

Field	Description
0 ACIE	**ACMP Interrupt Enable** — This bit enables the ACMP interrupt. 0 Interrupt disabled 1 Interrupt enabled

20.6.2.4 ACMP Status Register (ACMPS)

Module Base + 0x0003 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	ACO	0	0	0	0	0	0	ACIF
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-5. ACMP Status Register (ACMPS)

[1] Read: Anytime
Write:
ACIF: Anytime, write 1 to clear
ACO: Never

Table 20-6. ACMPS Register Field Descriptions

Field	Description
7 ACO	**ACMP Output** — Reading ACO returns the current value of the synchronized ACMP output. Refer to ACE description to account for initialization delay.
0 ACIF	**ACMP Interrupt Flag** — ACIF is set when a ACMP output has a valid edge defined by ACMPC0[ACMOD]. The setting of this flag lags the ACMPO output by two bus clocks. Writing 1 clears the flag. 0 Compare event has not occurred 1 Compare event has occurred

20.7 Functional Description

The ACMP compares the analog voltage between inverting and non-inverting inputs. It generates a digital output signal and a related interrupt if enabled. The comparator output is high when the voltage at the non-inverting input is greater than the voltage at the inverting input, and is low when the non-inverting input voltage is lower than the inverting input voltage. The size of the ACMP hysteresis can be adapted to the specific application to prevent unintended rapid switching.

Both the non-inverting and inverting input of the ACMP can be selected from four inputs: Two inputs from external signals ACMP_0 and ACMP_1 and two internal inputs acmpi_0 and acmpi_1. Refer to device-level section for connectivity. The positive input is selected by ACMPC1[ACPSEL] and the negative input is selected by ACMPC1[ACNSEL]. These bits can only be changed while the ACMP is disabled.

The ACMP is enabled with ACMPC0[ACE]. When this bit is set, the inputs are connected to low-pass filters while the comparator output is disconnected from the subsequent logic for 127 bus clock cycles. During this time the output state is preserved to mask potential glitches. This initialization delay must be accounted for before the first comparison result can be expected. The same delay must be accounted for after returning from STOP mode.

The initial hold state after reset is logic level 0. If input voltages are set to result in logic level 1 ($V_{ACMPP} > V_{ACMPM}$) before the initialization delay has passed, a flag will be set immediately after the delay if rising edge is selected as flag setting event.

Similarly the ACMPS[ACIF] flag will also be set when disabling the ACMP, then re-enabling it with the inputs changing to produce an opposite result to the hold state before the end of the initialization delay.

The unsynchronized comparator output can be connected to the synchronized timer input capture channel defined at SoC-level (see Figure 20-1). This feature can be used to generate time stamps and timer interrupts on ACMP events.

The comparator output signal can be read at register bit location ACMPS[ACO].

The condition causing the interrupt flag to assert is selected with ACMPC0[ACMOD]. This includes any edge configuration, that is rising, or falling, or rising and falling edges of the comparator output. Also flag setting can be disabled.

An interrupt will be generated if the interrupt enable bit (ACMPC2[ACIE]) and the interrupt flag (ACMPS[ACIF]) are both set. ACMPS[ACIF] is cleared by writing a 1.

The comparator output signal ACMPO can be driven out on an external pin by setting ACMPC0[ACOPE] and optionally inverted by setting ACMPC0[ACOPS].

One out of four hysteresis levels can be selected by setting ACMPC0[ACHYS].

The input delay of the ACMP_0 and ACMP_1 input depends on the selected filter characteristic by ACMPC0[ACDLY].

20.8 Interrupts

Table 20-7 shows the interrupt generated by the ACMP.

Table 20-7. ACMP Interrupt Sources

Module Interrupt Sources	Local Enable
ACMP interrupt	ACMPC2[ACIE]

Chapter 21
SENT Transmitter Module (SENTTXV1)

Table 21-1. Revision History Table

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
01.03	24-Jun-2013	All	Fixed typos in application section.
01.04	15-Oct-2013	All	Revision History table is now visible.
01.05	22-JAN-2014	All	Clarified module behavior in Stop mode.

21.1 Introduction

The Single Edge Nibble Transmission (SENT) module (SENTTX) is a transmitter for serial data frames which are implemented using the SENT encoding scheme. This module is based on the SAE J2716 information report titled "SENT - Single Edge Nibble Transmission for Automotive Applications" and released on January 27, 2010 (http://www.sae.org), Apr2007 and Feb2008. As per this standard, the SENT protocol is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10-bit A/Ds and PWM and as a simpler low-cost alternative to CAN or LIN.

The SENT encoding scheme is a unidirectional communications scheme from the sensor/transmitting device to the controller/receiving device which does not include a coordination signal from the controller/receiving device. The sensor signal is transmitted as a series of pulses with data encoded as falling to falling edge periods.

21.2 Glossary

The following terms and abbreviations are used in the document.

Table 21-2. Terminology

Term	Meaning
CRC	Cyclic Redundancy Check, an algorithm used to perform data integrity checks
idle level	Denotes the level of the SENT transmitter output signal (SENT_TX_OUT) when transmission is disabled. This is a logical one.
ISR	Interrupt Service Routine
MCU	Micro-Controller Unit
SENT	Single Edge Nibble Transfer

21.3 Features

- Features a 14 bit pre-scaler to derive a SENT-protocol compatible time unit of 3 to 90 µs from bus clock.
- Programmable number of transmitted data-nibbles (1 to 6).
- Provides hardware to support SAE J2716 2010 (SENT) Fast Channel communication[1].
- CRC nibble generation:
  - Supports SENT legacy method CRC generation in hardware.
  - Supports SENT recommended method CRC generation in hardware.
  - Optionally, the SENT status and communication nibble can be included in the automatic calculation of the CRC nibble.
  - Automatic CRC generation hardware can be bypassed to supply the CRC nibble directly from software.
- Supports optional pause-pulse generation. The optional pause pulse can have a fixed length or its length can be automatically adapted to get a fixed overall message period.
- Supports both continuous and software-triggered transmission.
- Interrupt-driven operation with five flags:
  - Transmit buffer empty
  - Transmission complete
  - Calibration pulse start
  - Transmitter under-run
  - Pause pulse rising-edge

21.4 Block Diagram

Figure 21-1 shows a block-diagram of the SENTTX module.

---

[1] Slow Channel communication can be implemented in software.

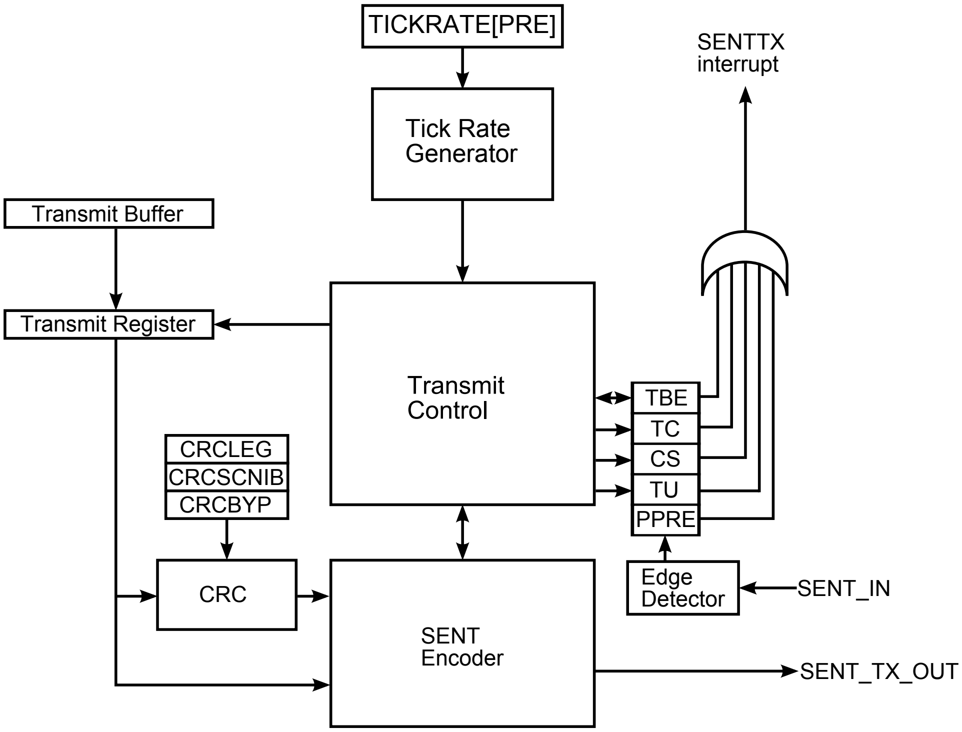


Figure 21-1. SENTTX Block Diagram

21.5 External Signals

The SENTTX module has two external signals.

21.5.1 SENT_TX_OUT - SENT Transmitter Output

The SENT_TX_OUT signal is the transmitter output signal.

21.5.2 SENT_IN - SENT Transmitter Input

The SENT_IN signal is used as a feedback input to compensate for the output delay on the transmitter output. This signal is used to detect when the Pause pulse rising-edge appears on the external pin.

21.6 Modes of Operation

- Run mode

  This is the basic mode of operation.
- Wait mode

  Wait mode does not affect the function of the SENTTX module. Operation is the same as in Run mode.
- Stop mode

In stop mode the SENTTX module aborts any transmission and resets interrupt flags INTFLG[PPRE,CS,TC,TBE][1]. This is not synchronized to an ongoing transmission sequence and can lead to the generation of an unexpected rising edge on the SENTTX module output and to the generation of unexpected interrupts (due to the reset of interrupt flags). To avoid this, software must take care to wait for message completion, disable transmission, and disable interrupt generation before Stop mode can be entered.

21.7 Memory Map and Register Definition

21.7.1 Register Map

Table 21-3 shows the SENTTX register map.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Table 21-3. SENTTX Register Map

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	TICKRATE	R	0	0	PRE[13:8]					
		W	▨	▨						
		R	PRE[7:0]							
		W								
0x0002	PPULSE	R	PPEN	PPFIXED	0	0	0	PPCOUNT[10:8]		
		W			▨	▨	▨			
		R	PPCOUNT[7:0]							
		W								
0x0004	CONFIG	R	TXINIT	TXEN	0	0	0	DNIBBLECOUNT[2:0]		
		W			▨	▨	▨			
		R	0	0	0	OPTEDGE	SINGLE	CRCSCN	CRCLEG	CRCBYP
		W	▨	▨	▨					
0x0006	INTEN	R	0	0	0	PPREIE	TUIE	CSIE	TCIE	TBEIE
		W	▨	▨	▨					
0x0007	INTFLG	R	0	0	0	PPRE	TU	CS	TC	TBE
		W	▨	▨	▨					

▨ = Unimplemented or Reserved

---

[1] Please refer to Section 21.7.2.5, “SENT Transmitter Interrupt Flag Register (INTFLG) for details.

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0008	TXBUF	R W	STATCONF[3:0]				CRC[3:0]			
		R W	DATA0[3:0]				DATA1[3:0]			
0x000A		R W	DATA2[3:0]				DATA3[3:0]			
		R W	DATA4[3:0]				DATA5[3:0]			
0x000C – 0x000F	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

21.7.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Writes to a reserved register location do not have any effect and reads of these locations return a zero.Details of register bit and field function follow the register diagrams, in bit order.

21.7.2.1 SENT Transmitter Tick Rate Register (TICKRATE)

Module Base + 0x0000

Access: User read/write[1]

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	PRE[13:0]													
W	▨	▨														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

[1] Read: Anytime.
Write: Anytime, if CONFIG[TXINIT] is one.

Table 21-4. SENT Transmitter Tick Rate Register (TICKRATE) Field Descriptions

Field	Description
13–0 PRE[13:0]	**SENTTX Tick Rate Bits** — The tick rate for the SENTTX module is determined by these 14 bits. The SENT transmitter Tick Rate Register can contain a value from 0 to 16383. It is used to derive the SENT unit-time (UT) from the SENTTX bus clock. The SENT unit-time has an allowable range of 3 to 90 μs. The formula for calculating the tick rate is: SENTTX tick rate = SENTTX bus clock rate / (PRE+1)

21.7.2.2 SENT Transmitter Pause-Pulse Register (PPULSE)

Module Base + 0x0002 Access: User read/write[1]

	15	14	13	12	11	10	9	8
R	PPEN	PPFIXED	0	0	0	PPCOUNT[10:8]		
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	PPCOUNT[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime.
Write: Anytime, if CONFIG[TXINIT] is one.

Table 21-5. SENT Transmitter Pause-Pulse Register (PPULSE) Field Descriptions

Field	Description
15 PPEN	**SENTTX Pause Pulse Enable** — If this bit is set, a pause pulse is inserted between the CRC nibble of the current SENT message and the start of the next SENT message. 1 - Pause pulse generation is enabled 0 - Pause pulse generation is disabled
14 PPFIXED	**SENTTX Fixed Length Pause Pulse** — If this bit is set, the optional pause pulse has a fixed length, as defined by the PPULSE[PPCOUNT] bits. If the value in PPULSE[PPCOUNT] is smaller than 12 (meaning smaller than a minimum pause pulse), a 12 tick pause pulse is used. Otherwise, if this bit is cleared, the PPULSE[PPCOUNT] bits define the overall message period. In this case the actual length of the pause pulse is adapted to the length of the remaining SENT message. Counting starts with the falling edge of the calibration pulse period at the start of the message. If the remaining message period after the falling edge of the CRC nibble is already lower than 12 ticks then a minimum pause-pulse of 12 ticks is appended. **Note:** For SENT a fixed or adapted pause pulse must not be longer than 768 ticks. 1 - Pause pulse length is fixed 0 - Pause pulse length is adapted to yield a fixed message period
10–0 PPCOUNT [10:0]	**SENTTX Pause Pulse Count Ticks** — If PPFIXED is one, these bits determine the length of the optional pause pulse inserted at the end of a SENT message. If PPFIXED is zero, these bits define the overall message period. The value in these bits is given in SENT ticks (UT).

21.7.2.3 SENT Transmitter Configuration Register (CONFIG)

Module Base + 0x0004 Access: User read/write[1]

	15	14	13	12	11	10	9	8
R	TXINIT	TXEN	0	0	0	DNIBBLECOUNT[2:0]		
W								
Reset	1	0	0	0	0	0	0	1

	7	6	5	4	3	2	1	0
R	0	0	0	OPTEDGE	SINGLE	CRCSCN	CRCLEG	CRCBYP
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime.
Write: TXINIT bit: Anytime. All other bits: Anytime, if TXINIT is one.

Table 21-6. SENT Transmitter Configuration Register (CONFIG) Field Descriptions

Field	Description
15 TXINIT	**SENTTX Initialization Enable** — If this bit is set, sending SENT messages is aborted and other configuration bits can be written. Setting this bit immediately aborts any ongoing transmission, resets interrupt flags INTFLG[PPRE,CS,TC,TBE] to their respective reset state and the SENTTX pin returns to idle level. Interrupt flags can only be cleared, if this bit is zero and CONFIG[TXEN] is one. 1 - Writing configuration bits is enabled. Data transmission is disabled. 0 - Writing configuration bits is disabled. Data transmission is enabled.
14 TXEN	**SENTTX Pin Enable** — This bit enables the SENTTX module function on the SENTTX output pin. This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - Output pin is controlled by the SENTTX module 0 - Output pin is not controlled by the SENTTX module
10–8 DNIBBLE-COUNT[2:0]	**SENTTX Data Nibble Count** — These bits represent the number of data nibbles to be transmitted. These bits can only be changed if the CONFIG[TXINIT] bit is one. Available range for the number of data nibbles in a SENT message is 1 to 6, encoded in the CONFIG[DNIBBLECOUNT] bits as follows: 000 - Reserved 001 - One data nibble is transmitted 010 - Two data nibbles are transmitted 011 - Three data nibbles are transmitted 100 - Four data nibbles are transmitted 101 - Five data nibbles are transmitted 110 - Six data nibbles are transmitted 111 - Reserved
4 OPTEDGE	**SENTTX Optimized Rising Edge Position** — If set, this bit causes the SENTTX module to place the rising edge at half the number of the pulse period ticks after the falling edge to yield a duty cycle of about 50%. If the pulse period is an odd number of ticks, the number of ticks for the low pulse is rounded down (meaning the low pulse is one tick shorter than the high pulse). Otherwise, if this bit is cleared, a rising edge is positioned 5 ticks (UT) after a falling edge. This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - Optimized positioning of rising edges enabled 0 - Rising edges are generated 5 UT ticks after falling-edges

Field	Description
3 SINGLE	**SENTTX Single Shot Operation** — If set, this bit causes the SENTTX module to stop after the current transmission is complete and wait for the Transmit-Buffer Empty bit to be cleared, before a new transmission is started. Transmitter Under-run is not flagged in this case. Otherwise, if this bit is cleared, messages are sent continuously back-to-back. This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - Single shot operation is enabled 0 - Single shot operation is disabled **Note:** Meaningful single shot operation requires pause pulse generation to be enabled (PPULSE[PPEN]=1). Otherwise the last nibble to be sent as part of the message (CRC) will have the terminating falling edge missing (rendering the whole transmission incomplete).
2 CRCSCN	**SENTTX CRC includes Status- and Communication Nibble** — If set, this bit causes the SENTTX module to include the status and communication nibble in the automatic generation of the CRC nibble. Otherwise, if this bit is cleared, the status and communication nibble is not included in the automatic CRC generation. This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - CRC generation includes Status- and Communication Nibble 0 - CRC generation excludes Status- and Communication Nibble
1 CRCLEG	**SENTTX CRC Legacy Algorithm Enable** — If set, this bit causes the SENTTX module to generate the automatic CRC nibble using the legacy CRC algorithm. Otherwise, if this bit is cleared, the CRC nibble is calculated using the recommended CRC algorithm. This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - CRC generation uses the legacy algorithm 0 - CRC generation uses the recommended algorithm
0 CRCBYP	**SENTTX Automatic CRC generation bypass** — If set, this bit causes the SENTTX module to bypass the automatic generation of the CRC nibble. The CRC information is taken from the Transmit Buffer (TXBUF[CRC]). This bit can only be changed if the CONFIG[TXINIT] bit is one. 1 - CRC generation bypass is enabled 0 - CRC generation bypass is disabled. Data in TXBUF[CRC] is ignored.

21.7.2.4 SENT Transmitter Interrupt Enable Register (INTEN)

Module Base + 0x0006

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	PPREIE	TUIE	CSIE	TCIE	TBEIE
W								
Reset	0	0	0	0	0	0	0	0

[1] Read: Anytime.
Write: Anytime.

Table 21-7. SENT Transmitter Interrupt Enable Register (INTEN) Field Descriptions

Field	Description
4 PPREIE	**SENTTX Pause Pulse Rising-Edge Interrupt Enable** — This bit enables the generation of a Pause Pulse Rising-Edge interrupt whenever the Pause Pulse Rising-Edge Flag is set in the SENTTX Interrupt Flag Register (INTFLG[PPRE]). 1 - Pause Pulse Rising-Edge interrupt is enabled 0 - Pause Pulse Rising-Edge interrupt is disabled
3 TUIE	**SENTTX Transmitter Under-run Interrupt Enable** — This bit enables the generation of a Transmitter Under-run Interrupt whenever the Transmitter Under-run Flag is set in the SENTTX Interrupt Flag Register (INTFLG[TU]). 1 - Transmitter Under-run Interrupt is enabled 0 - Transmitter Under-run Interrupt is disabled
2 CSIE	**SENTTX Calibration Pulse Start Interrupt Enable** — This bit enables the generation of a Calibration Pulse Start Interrupt whenever the Calibration Pulse Start Flag is set in the SENTTX Interrupt Flag Register (INTFLG[CS]). 1 - Calibration Pulse Start Interrupt is enabled 0 - Calibration Pulse Start Interrupt is disabled
1 TCIE	**SENTTX Transmission Complete Interrupt Enable** — This bit enables the generation of a Transmission Complete Interrupt whenever the Transmission Complete Flag is set in the SENTTX Interrupt Flag Register (INTFLG[TC]). 1 - Transmission Complete Interrupt is enabled 0 - Transmission Complete Interrupt is disabled
0 TBEIE	**SENTTX Transmit-Buffer Empty Interrupt Enable** — This bit enables the generation of a Transmit-Buffer Empty Interrupt whenever the Transmit-Buffer Empty Flag is set in the SENTTX Interrupt Flag Register (INTFLG[TBE]). 1 - Transmit-Buffer Empty Interrupt is enabled 0 - Transmit-Buffer Empty Interrupt is disabled

21.7.2.5 SENT Transmitter Interrupt Flag Register (INTFLG)

Module Base + 0x0007 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	PPRE	TU	CS	TC	TBE
W								
Reset	0	0	0	0	0	0	1	1

---

[1] Read: Anytime.
Write: Anytime, if CONFIG[TXINIT] is zero and CONFIG[TXEN] is one. Write one to clear.

Table 21-8. SENT Transmitter Interrupt Flag Register (INTFLG) Field Descriptions

Field	Description
4 PPRE	**SENTTX Pause Pulse Rising-Edge Flag** — If set, this bit indicates the rising-edge of a pause pulse has occurred (with sampling done using the bus clock). Write a one to this bit to clear. 1 - Pause Pulse Rising-edge detected 0 - Pause Pulse Rising-edge not detected
3 TU	**SENTTX Transmitter Under-run Flag** — If set, this bit indicates a Transmitter Under-run condition has occurred. A Transmitter Under-run condition means the transmit buffer was empty (INTFLG[TBE]=1) when the data from the transmit buffer is supposed to be copied to the transmit register. This copy is taken one bus-cycle before the ending falling edge of the calibration pulse is to occur, but only if the transmit buffer is marked as not empty (INTFLG[TBE]=0). Otherwise the falling edge at the end of the calibration pulse is suppressed, the transmission is aborted and an under-run condition is flagged. As long as this bit is set, the transmitter remains disabled and the SENT_TX_OUT signal remains at idle level. Write a one to this bit to clear. Clearing this bit resets INTFLG[PPRE,CS,TC,TBE] flags to their respective reset state. 1 - Transmitter Under-run condition has occurred 0 - Transmitter Under-run condition has not occurred
2 CS	**SENTTX Calibration Pulse Start Flag** — If set, this bit indicates a transmission has just started with the first falling edge of the calibration pulse. Write a one to this bit to clear. 1 - Calibration pulse has started 0 - Calibration pulse has not started
1 TC	**SENTTX Transmission Complete Flag** — If set, this bit indicates the current transmission has completed. It is set after the CRC nibble has been sent. Write a one to this bit to clear. 1 - Transmission has completed 0 - Transmission has not completed
0 TBE	**SENTTX Transmit-Buffer Empty Flag** — If set, this bit indicates the transmit-buffer is empty and can be written to. It is set by transferring the data from the transmit buffer to the transmit register. Write a one to this bit to clear. Clearing this bit declares the transmit buffer as full. 1 - Transmit-Buffer is empty 0 - Transmit-Buffer is full

21.7.2.6 SENT Transmit Buffer (TXBUF)

Module Base + 0x0008

Access: User read/write[1]

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STATCONF[3:0]				CRC[3:0]				DATA0[3:0]				DATA1[3:0]			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Module Base + 0x000A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA2[3:0]				DATA3[3:0]				DATA4[3:0]				DATA5[3:0]			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

[1] Read: Anytime.
Write: Anytime. While the Transmit-Buffer Empty bit in the INTFLG register is zero (INTFLG[TBE]=0) the data in the transmit buffer (TXBUF) should not be changed to avoid transmission of inconsistent data.

Table 21-9. SENT Transmit Buffer (TXBUF) Field Descriptions

Field	Description
31–28 STATCONF [3:0]	**SENTTX Status and Configuration Nibble** — These bits represent data which is sent as the SENT protocol status- and configuration-nibble.
27–24 CRC[3:0]	**SENTTX CRC Nibble** — These bits represent data which is sent as the SENT protocol CRC nibble, if the CRC bypass option bit in the SENTTX CONFIG register (CONFIG[CRCBYP]) is set. Otherwise these bits are ignored.
23–20 DATA0[3:0]	**SENTTX Data Nibble 0** — These bits represent data which is sent as the first SENT protocol data nibble.
19–16 DATA1[3:0]	**SENTTX Data Nibble 1** — These bits represent data which is sent as the second SENT protocol data nibble, if enabled by the data nibble count bits in the SENTTX CONFIG register (CONFIG[DNIBBLECOUNT]>=2).
15–12 DATA2[3:0]	**SENTTX Data Nibble 2** — These bits represent data which is sent as the third SENT protocol data nibble, if enabled by the data nibble count bits in the SENTTX CONFIG register (CONFIG[DNIBBLECOUNT]>=3).
11–8 DATA3[3:0]	**SENTTX Data Nibble 3** — These bits represent data which is sent as the fourth SENT protocol data nibble, if enabled by the data nibble count bits in the SENTTX CONFIG register (CONFIG[DNIBBLECOUNT]>=4).
7–4 DATA4[3:0]	**SENTTX Data Nibble 4** — These bits represent data which is sent as the fifth SENT protocol data nibble, if enabled by the data nibble count bits in the SENTTX CONFIG register (CONFIG[DNIBBLECOUNT]>=5).
3–0 DATA5[3:0]	**SENTTX Data Nibble 5** — These bits represent data which is sent as the sixth SENT protocol data nibble, if enabled by the data nibble count bits in the SENTTX CONFIG register (CONFIG[DNIBBLECOUNT]=6).

21.8 Functional Description

This section provides a complete functional description of the SENTTX module, describing the operation of the design in a number of subsections.

21.8.1 Message Format

The SENTTX module uses the standard SENT message format. The transmitted information is encoded into the distance between two consecutive falling edges. A message consists of:

1. One calibration/synchronization pulse period with a length of 56 unit-time (UT) ticks.
2. One status and serial communication nibble pulse period with a length of 12 to 27 UT ticks.
3. One to six data nibble pulse periods with a length of 12 to 27 UT ticks each.
4. One CRC nibble pulse period with a length of 12 to 27 UT ticks.
5. An optional pause pulse.

Figure 21-2 below shows an example of a SENT message (not drawn to scale).

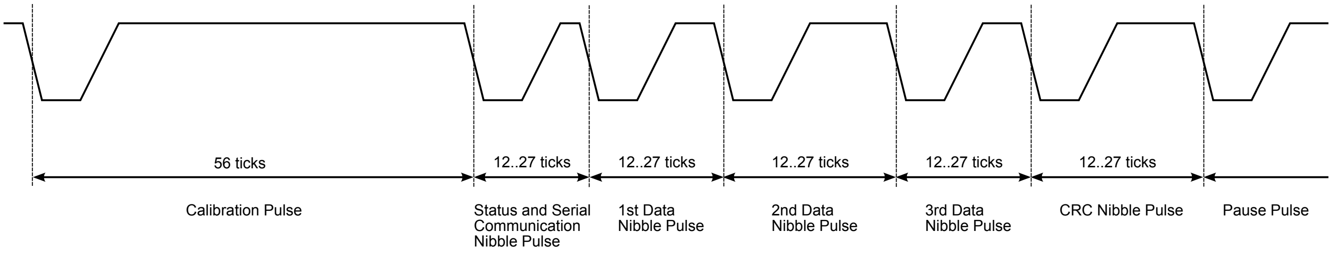


Figure 21-2. SENT Message Format

The number of transmitted data nibbles (1 to 6) is controlled by the data nibble count bits in the Configuration Register (CONFIG[DNIBBLECOUNT]).

The CRC nibble can be supplied using one of the following three methods:

1. Hardware calculates the CRC nibble from the transmitted data nibbles using the SENT recommended method (CONFIG[CRCLEG]=0, CONFIG[CRCBYP]=0).
2. Hardware calculates the CRC nibble from the transmitted data nibbles using the SENT legacy method (CONFIG[CRCLEG]=1, CONFIG[CRCBYP]=0).
3. The CRC nibble can be supplied by software, together with the status and serial communication nibble and the data nibbles in the Transmit Buffer (CONFIG[CRCBYP]=1).

An option exists to also include the status and serial communication nibble into the automatic CRC calculation (CONFIG[CRCSCN]=1).

21.8.2 Transmitter States

- Init State

  After system reset the SENTTX module starts up in Init State, which means the transmitter is initially disabled with all configuration registers (e.g. TICKRATE, PPULSE, CONFIG) writeable. Additionally, all interrupt flags are held in their respective reset states.

  The Init State can be left by clearing the CONFIG[TXINIT] bit with CONFIG[TXEN] being set (clearing CONFIG[TXINIT] and setting CONFIG[TXEN] to leave Init State can be done in the same write access).

- Idle State

  After leaving Init State, the SENTTX module enters Idle State. In this state, the registers TICKRATE, PPULSE and CONFIG (except CONFIG[TXINIT]) are read-only.

  The module waits for software to clear INTFLG[TBE] which announces valid data in the transmit buffer (TXBUF). Clearing INTFLG[TBE] causes the module to enter Transmit State.

- Transmit State

  In Transmit State the SENTTX module transmits the data written by software into the TXBUF register. As long as there is data available in TXBUF the SENTTX module remains in Transmit State. If single-shot mode is active (CONFIG[SINGLE]=1) the module enters Idle State after transmission completes and no new data is available in TXBUF. Otherwise, when not in single-shot mode and no new data is available at the end of the calibration pulse period of a new message, the

SENTTX module aborts the transmission, sets the Transmitter Under-run flag bit and enters Error State. The SENT_TX_OUT signal remains at idle level in case of a Transmitter Under-run condition.

If the system enters Stop mode while the SENTTX module is in Transmit State, the module aborts the current transmission, resets interrupt flags INTFLG[PPRE,CS,TC,TBE] and enters Idle State. The SENT_TX_OUT signal switches to idle level in this case.

- Error State

  In this state transmission of messages is disabled. The module waits for the Transmitter Under-run flag bit to be cleared.

  When software clears the Transmitter Under-run error flag bit, the module resets interrupt flags INTFLG[PPRE,CS,TC,TBE] and enters Idle State again.

Whenever software sets the CONFIG[TXINIT] bit, the SENTTX aborts any ongoing transmission, and resets interrupt flags INTFLG[PPRE,CS,TC,TBE]. The SENT_TX_OUT signal switches to idle level in this case. If the transmitter under-run error flag is set (INTFLG[TU]=1) when software sets the CONFIG[TXINIT] bit, the transmitter remains in Error State; else transmitter enters Init State.

21.8.3 Tick Rate Generation

A 14-bit modulus counter in the tick rate generator is used to derive a SENT-compliant tick period (3 to 90 µs) from the supplied bus clock. The value from 0 to 16383 written to the TICKRATE[PRE] bits determines the bus clock divisor.

SENTTX tick rate = SENTTX bus clock / (TICKRATE[PRE] + 1)

21.8.4 Transmission Modes

To transmit data, the MCU writes the data bits to the transmit buffer (TXBUF), which in turn are transferred to the transmit register. The transmit register then shifts out a message nibble-by-nibble to the SENT encoder. The SENT encoder controls the data passing through the SENT_TX_OUT signal. The transmit buffer (TXBUF) acts as a buffer between the MCU’s internal data bus and the transmit register.

The SENTTX module also sets a flag, the transmit buffer empty flag (INTFLG[TBE]), every time it transfers data from the transmit buffer (TXBUF) to the transmit register. This flag also acts as a transmit buffer state indicator. That means by clearing this flag software declares the content of the transmit buffer as valid to the hardware.

At the end of the data-dependent part of each message (that means at the end of the transmission of the CRC nibble) the SENTTX module sets the Transmission Complete flag (INTFLG[TC]).

At the start of the calibration pulse of each message the SENTTX module sets the Calibration-Pulse Start flag (INTFLG[CS]).

Each of these three flags can be used by software to drive the transfer of SENT messages by the SENTTX module. The following section provides an overview of available usage models.

21.8.4.1 Double-buffered Transmission without Pause Pulse

This option offers the time of an entire message for the software to prepare the data (typically about 200 unit-time ticks, depending on message length). The down-side is that the data is relatively old when it gets transmitted since it was prepared while the previous message was sent.

To use the double-buffered transmission option software uses the Transmit-Buffer Empty flag (TBE) to prepare new data for the next transmission. An example for this case is provided in Figure 21-3 below.

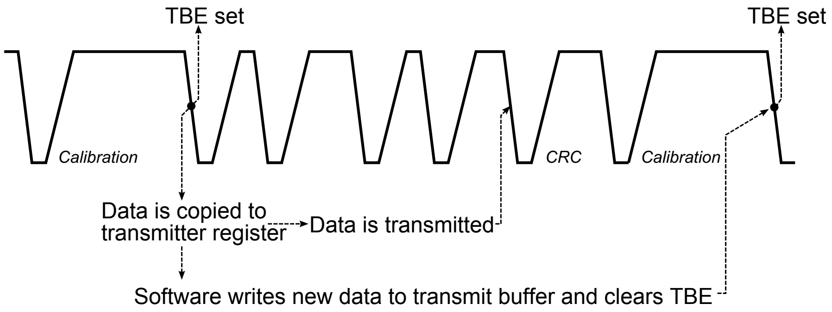


Figure 21-3. Transmit-Buffer Empty driven SENT transfer without Pause Pulse

21.8.4.2 Single-buffered Transmission without Pause Pulse

This option offers the most up-to-date transmit data. The disadvantage for software is that the preparation of the transmit data has to occur in much less time (less than the length of the calibration pulse, meaning less than 56 unit-time ticks) compared to the double-buffered transmission option.

The software uses the Transmission Complete interrupt (TC) to prepare new data for the next transmission. An example for this case is provided in Figure 21-4 below.

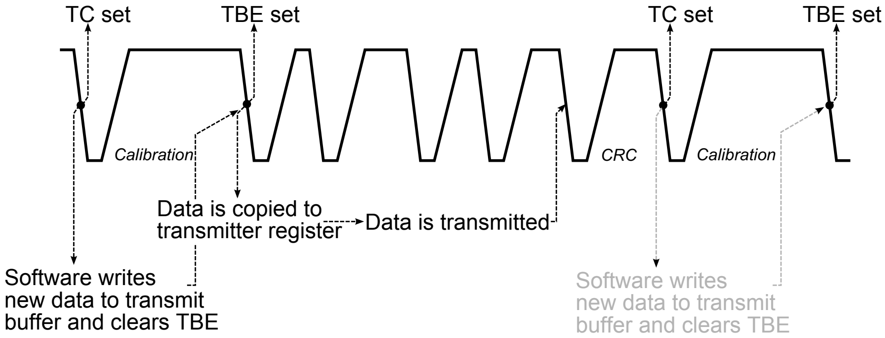


Figure 21-4. Transmission Complete driven SENT transfer without Pause Pulse

21.8.4.3 Double-buffered Transmission with Pause Pulse

This option is similar to the double-buffered transmission without pause-pulse (for details please refer to Section 21.8.4.1, “Double-buffered Transmission without Pause Pulse). The only difference is that due to the pause pulse the message periods become longer offering even more time for the software to prepare new data.

The software uses the Transmit Buffer Empty interrupt (TBE) to prepare new data for the next transmission. An example for this case is provided in Figure 21-5 below.

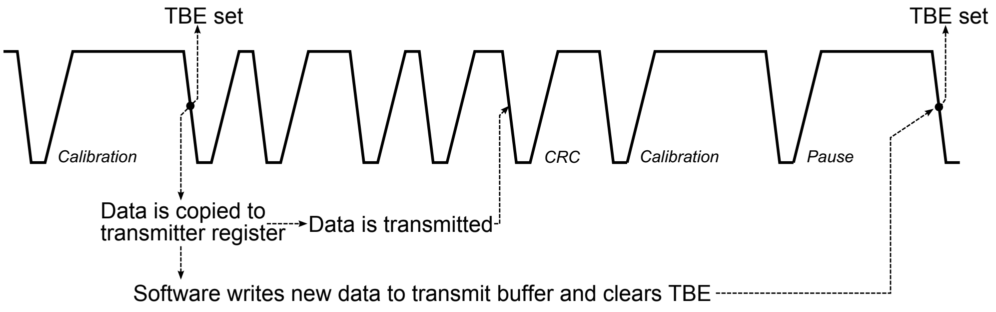


Figure 21-5. Transmit-Buffer Empty driven SENT transfer with Pause Pulse

21.8.4.4 Single-buffered Transmission with Pause Pulse

This option offers the most up-to-date transmit data. The disadvantage for software is that the preparation of the transmit data has to occur in much less time (less than the length of the calibration pulse, meaning less than 56 unit-time ticks) compared to the double-buffered transmission option. This is similar to the single-buffered transmission without pause pulse case (please refer to Section 21.8.4.2, “Single-buffered Transmission without Pause Pulse for comparison).

The software uses the Calibration Pulse Start interrupt (CS) to prepare new data for the next transmission. An example for this case is provided in Figure 21-6 below.

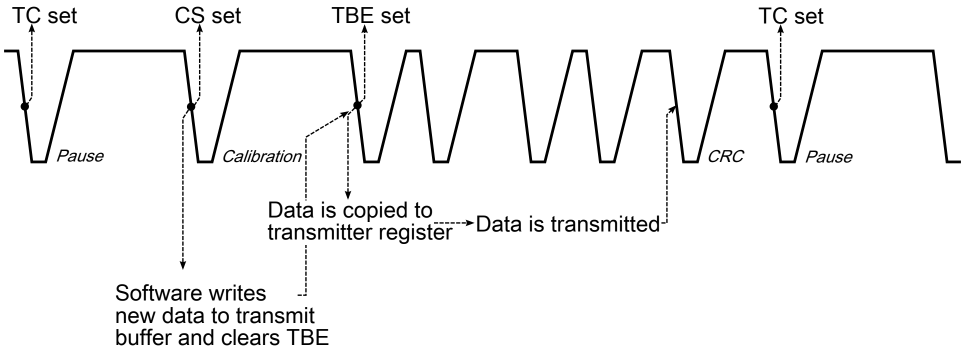


Figure 21-6. Calibration Pulse Start driven SENT transfer with Pause Pulse

21.8.4.5 Early Single-buffered Transmission with Pause Pulse

This option is similar to the single-buffered transmission case, but offers additional time for the software to prepare new data by adding the duration of the pause pulse to the calibration pulse of the next message.

Transmitted data is slightly older than with the single-buffered transmission with pause pulse case (please refer to Section 21.8.4.4, “Single-buffered Transmission with Pause Pulse for comparison).

The software uses the Transmission Complete interrupt (TC) to prepare new data for the next transmission. An example for this case is provided in Figure 21-7 below.

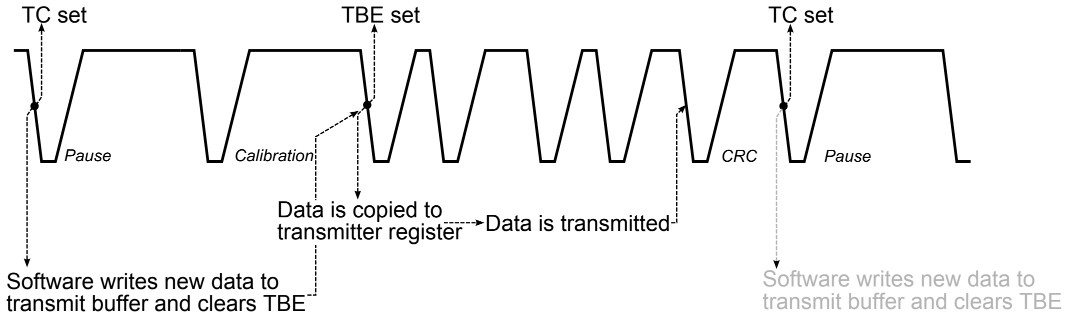


Figure 21-7. Transmission Complete driven SENT transfer with Pause Pulse

21.8.4.6 Software Triggered Transmission

In addition to the automatic triggering of message transmission, the SENTTX module also features a software controlled way of triggering a transmission of a message (CONFIG[SINGLE]=1). This enables the possibility to synchronize message transmission to external events (like, for example, a periodic timer interrupt, or an external trigger pulse from a pin). Also, this transmission mode eliminates the possible occurrence of a transmitter underflow condition because the actual transmission does not start before software declares the data in the transmit buffer as valid by clearing the transmit-buffer empty flag (INTFLG[TBE]=0).

Since the start of transmission is controlled by software, this feature requires the optional pause pulse to be enabled: without pause pulse the terminating edge of a message is the start of the next calibration pulse which in this case is software controlled and potentially appears much later leading to an incomplete CRC nibble of the current message.

21.9 Interrupts

Table 21-10 shows the interrupts generated by the SENTTX module.

Table 21-10. SENTTX Interrupt Sources

Module Interrupt Source	Local Enable
SENTTX interrupt	INTEN[PPREIE] INTEN[TUIE] INTEN[CSIE] INTEN[TCIE] INTEN[TBEIE]

21.10 Application Information

This section provides basic information useful for implementing software for the SENTTX module.

21.10.1 Initialization

All configuration options need to be defined before enabling the transmitter. This is the recommended initialization sequence:

1. Configure the SENT tick rate in the TICKRATE register.
2. Set required Pause-Pulse options in the PPULSE register.
3. Define all protocol related configuration options in the CONFIG register (such as number of data nibbles, CRC method to be used, etc.). Enable the transmitter and lock all static configuration bits in the TICKRATE, PPULSE and CONFIG registers by clearing the CONFIG[TXINIT] bit. Writing configuration bits in CONFIG and clearing CONFIG[TXINIT] can be done in a single write access to the CONFIG register.
4. Set interrupt enable bits in the interrupt enable register (INTEN) as required by the chosen transfer mode (see Section 21.8.4, “Transmission Modes for details).
5. Write data to the transmit buffer register (TXBUF) and clear the transmit buffer empty bit (INTFLG[TBE]) to start the transmission.

21.10.2 Stop Mode

Since the SENTTX module needs the bus-clock to be running for transmission, Stop mode causes any ongoing transmission to be aborted. Interrupt flags INTFLG[PPRE,CS,TC,TBE] are reset. The SENT_TX_OUT pin returns to idle level. For a SENT receiver this looks like a transmitter reset. After wake-up the transmitter waits for new data to be provided before transmission is started again.

If Stop mode is really required, the following options exist to synchronize Stop mode entry to the transmission sequence:

- Software Triggered Transmission (please refer to Section 21.8.4, “Transmission Modes for details)

  In this mode software is responsible for triggering the transmission of each individual message. Simply wait for message completion, for example by waiting for the Pause-Pulse Rising-Edge flag to be set (INTFLG[PPRE]=1), before entering Stop mode to avoid the generation of incomplete messages.
- Single- and Double-Buffered Transmission with optional Pause-Pulse (please refer to Section 21.8.4, “Transmission Modes for details)

  In this mode, the optional Pause-Pulse can be used as an indicator for an inter-message gap to enter Stop mode to avoid incomplete messages. In this mode the Pause-Pulse Rising-Edge flag (INTFLG[PPRE]) can be used as a message completion indicator before the transmitter can be disabled to cleanly enter Stop mode.
- Single- and Double-Buffered Transmission without optional Pause-Pulse (please refer to Section 21.8.4, “Transmission Modes for details)

Due to the structure of the SENT protocol in this mode, no real inter-message gap exists to avoid incomplete messages when entering Stop mode. However, it is (for example) possible to let the transmitter run into a transmitter under-run error condition by not providing new data. This stops the transmission at a defined state (at the expected end of the calibration pulse of a new message). Software must take care of the transmitter under-run condition before transmission can be started again after wake-up.

While the SENTTX module is not capable of generating new interrupts if the bus-clock is not running, entering Stop mode causes interrupt flags INTFLG[PPRE,CS,TC,TBE] to enter their respective reset states before the bus-clock is shut down. This means the SENTTX module interrupt asserts if interrupt enables for interrupt flags getting set by Stop mode entry (INTFLG[TC,TBE]) are left in active state before entering Stop mode.

Wake-up from Stop mode must be done by means outside of the SENTTX module (please refer to the Device Overview chapter in the Device Reference Manual for more details).

Chapter 22
192 KB Flash Module (S12ZFTMRZ192K2KV2)

22.1 Introduction

The FTMRZ192K2K module implements the following:

- 192 KB of P-Flash (Program Flash) memory
- 2 KB of EEPROM memory

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

The Flash memory may be read as bytes and aligned words. Read access time is one bus cycle for bytes and aligned words. For misaligned words access, the CPU has to perform twice the byte read access command. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is possible to read from P-Flash memory while some commands are executing on EEPROM memory. It is not possible to read from EEPROM memory while a command is executing on P-Flash memory. Simultaneous P-Flash and EEPROM operations are discussed in <st-blue>Section 22.4.6 Allowed Simultaneous P-Flash and EEPROM Operations.

Both P-Flash and EEPROM memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by half-phrase, only one single bit fault in an aligned 4 byte half-phrase containing the byte or word accessed will be corrected.

22.1.1 Glossary

Command Write Sequence — An MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

EEPROM Memory — The EEPROM memory constitutes the nonvolatile memory store for data.

EEPROM Sector — The EEPROM sector is the smallest portion of the EEPROM memory that can be erased. The EEPROM sector consists of 4 bytes.

NVM Command Mode — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

Phrase — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes two sets of aligned double words with each set including 7 ECC bits for single bit fault correction and double bit fault detection within each double word.

P-Flash Memory — The P-Flash memory constitutes the main nonvolatile memory store for applications.

P-Flash Sector — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 512 bytes.

Program IFR — Nonvolatile information register located in the P-Flash block that contains the Version ID, and the Program Once field.

22.1.2 Features

22.1.2.1 P-Flash Features

- 192 KB of P-Flash divided into 384 sectors of 512 bytes
- Single bit fault correction and double bit fault detection within a 32-bit double word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Ability to read the P-Flash memory while programming a word in the EEPROM memory
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

22.1.2.2 EEPROM Features

- 2 KB of EEPROM memory composed of one 2 KB Flash block divided into 512 sectors of 4 bytes
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation

- Protection scheme to prevent accidental program or erase of EEPROM memory
- Ability to program up to four words in a burst sequence

22.1.2.3 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

22.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 22-1.

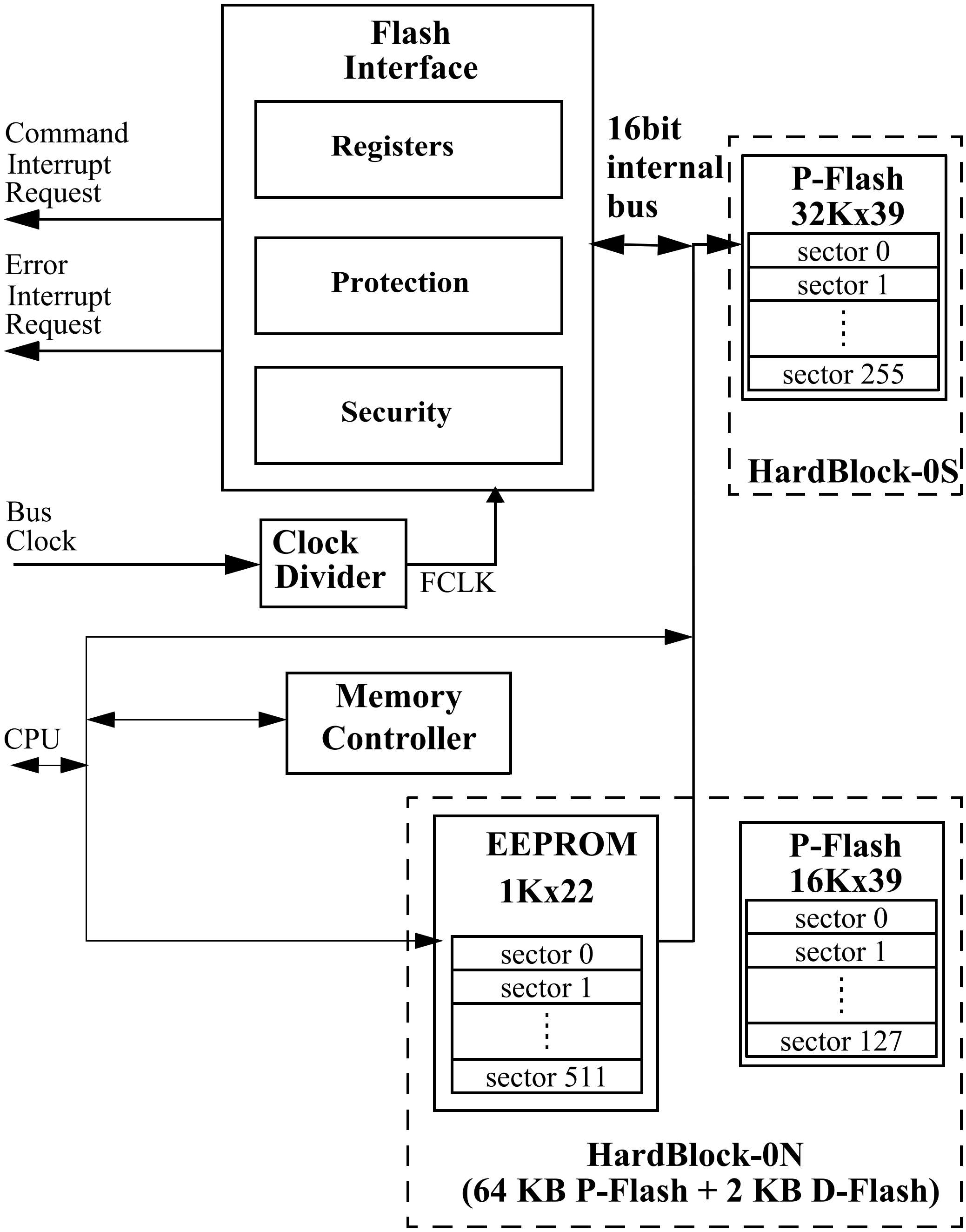


Figure 22-1. FTMRZ192K2K Block Diagram

22.2 External Signal Description

The Flash module contains no signals that connect off-chip.

22.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

CAUTION

> Writing to the Flash registers while a Flash command is executing (that is indicated when the value of flag CCIF reads as '0') is not allowed. If such action is attempted, the result of the write operation will be unpredictable.
>
> Writing to the Flash registers is allowed when the Flash is not busy executing commands (CCIF = 1) and during initialization right after reset, despite the value of flag CCIF in that case (refer to <st-blue>Section 22.6 Initialization for a complete description of the reset sequence).

.

Table 22-1. FTMRZ Memory Map

Global Address (in Bytes)	Size (Bytes)	Description
0x0_0000 – 0x0_0FFF	4,096	Register Space
0x10_0000 – 0x10_07FF	2,048	EEPROM memory
0x1F_4000 – 0x1F_FFFF	49,152	NVM Resource Area[1] (see <f-helvetica><st-bold>Figure 22-3.)
0xFD_0000 – 0xFF_FFFF	196,608	P-Flash Memory

[1] See NVM Resource area description in <st-blue>Section 22.4.4 Internal NVM resource

22.3.1 Module Memory Map

The S12Z architecture places the P-Flash memory between global addresses 0xFD_0000 and 0xFF_FFFF as shown in Table 22-2.

The P-Flash memory map is shown in <f-helvetica><st-bold>Figure 22-2..

Table 22-2. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0xFD_0000 – 0xFF_FFFF	192 K	P-Flash Block Contains Flash Configuration Field (see Table 22-3)

The FPROT register, described in <st-blue>Section 22.3.2.9 P-Flash Protection Register (FPROT), can be set to protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0xFF_8000 in the Flash memory (called the lower region), one growing downward from global address 0xFF_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protection. The Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 22-3.

Table 22-3. Flash Configuration Field

Global Address	Size (Bytes)	Description
0xFF_FE00-0xFF_FE07	8	Backdoor Comparison Key Refer to Section 22.4.7.11, "Verify Backdoor Access Key Command," and Section 22.5.1, "Unsecuring the MCU using Backdoor Key Access"
0xFF_FE08-0xFF_FE09[1]	2	Protection Override Comparison Key. Refer to Section 22.4.7.17, "Protection Override Command"
0xFF_FE0A-0xFF_FE0B[1]	2	Reserved
0xFF_FE0C[1]	1	P-Flash Protection byte. Refer to Section 22.3.2.9, "P-Flash Protection Register (FPROT)"
0xFF_FE0D[1]	1	EEPROM Protection byte. Refer to Section 22.3.2.10, "EEPROM Protection Register (DFPROT)"
0xFF_FE0E[1]	1	Flash Nonvolatile byte Refer to Section 22.3.2.11, "Flash Option Register (FOPT)"
0xFF_FE0F[1]	1	Flash Security byte Refer to Section 22.3.2.2, "Flash Security Register (FSEC)"

[1] 0xFF_FE08-0xFF_FE0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0xFF_FE0A - 0xFF_FE0B reserved field should be programmed to 0xFF.

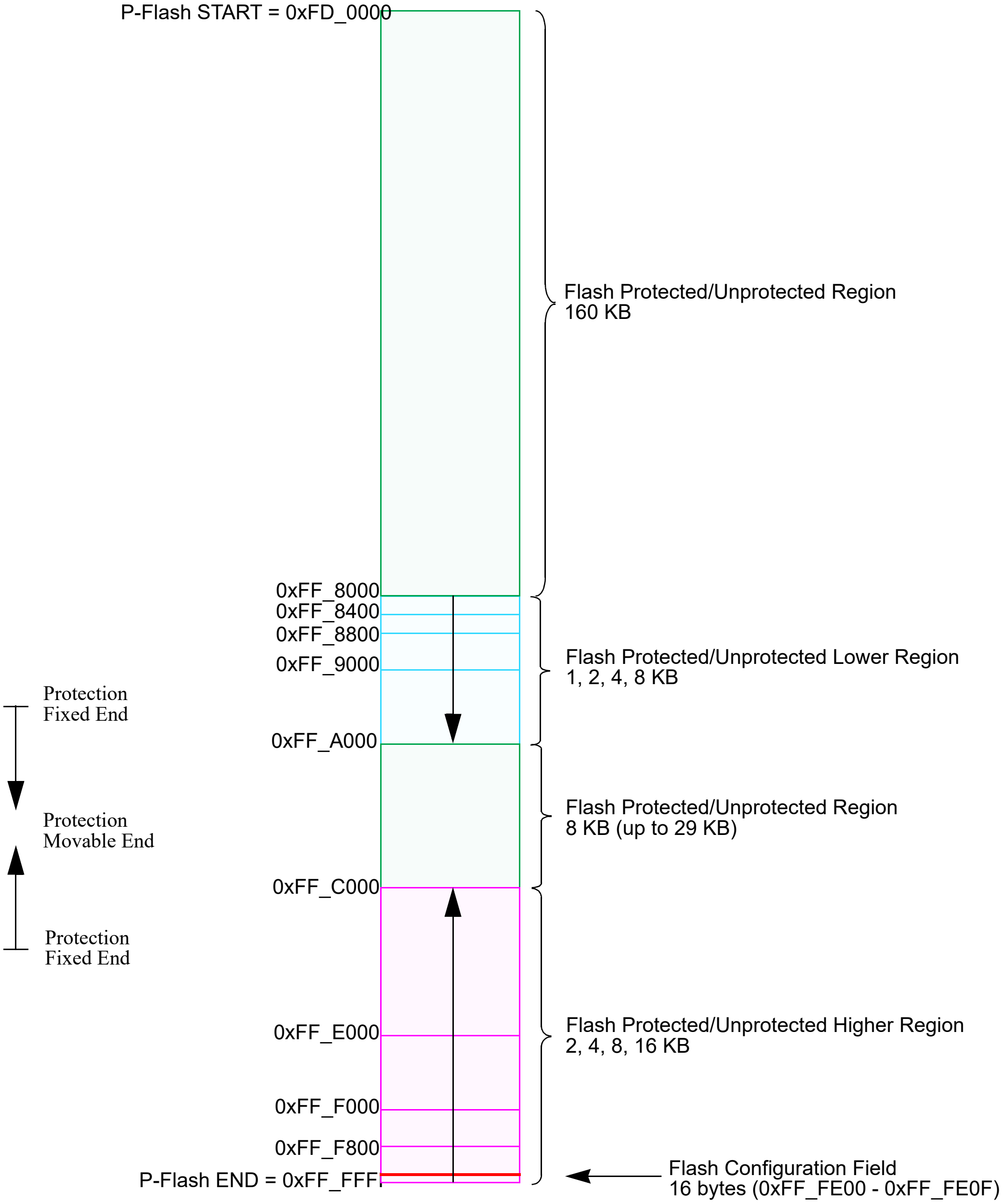


Figure 22-2. P-Flash Memory Map

Table 22-4. Program IFR Fields

Global Address	Size (Bytes)	Field Description
0x1F_C000 – 0x1F_C007	8	Reserved
0x1F_C008 – 0x1F_C0B5	174	Reserved
0x1F_C0B6 – 0x1F_C0B7	2	Version ID[1]
0x1F_C0B8 – 0x1F_C0BF	8	Reserved
0x1F_C0C0 – 0x1F_C0FF	64	Program Once Field Refer to Section 22.4.7.6, "Program Once Command"

[1] Used to track firmware patch versions, see <st-blue>Section 22.4.2 IFR Version ID Word

Table 22-5. Memory Controller Resource Fields (NVM Resource Area[1])

Global Address	Size (Bytes)	Description
0x1F_4000 – 0x1F_41FF	512	Reserved
0x1F_4200 – 0x1F_7FFF	15,872	Reserved
0x1F_8000 – 0x1F_97FF	6,144	Reserved
0x1F_9800 – 0x1F_BFFF	10,240	Reserved
0x1F_C000 – 0x1F_C0FF	256	P-Flash IFR (see Table 22-4)
0x1F_C100 – 0x1F_C1FF	256	Reserved.
0x1F_C200 – 0x1F_FFFF	15,872	Reserved.

[1] See <st-blue>Section 22.4.4 Internal NVM resource for NVM Resources Area description.

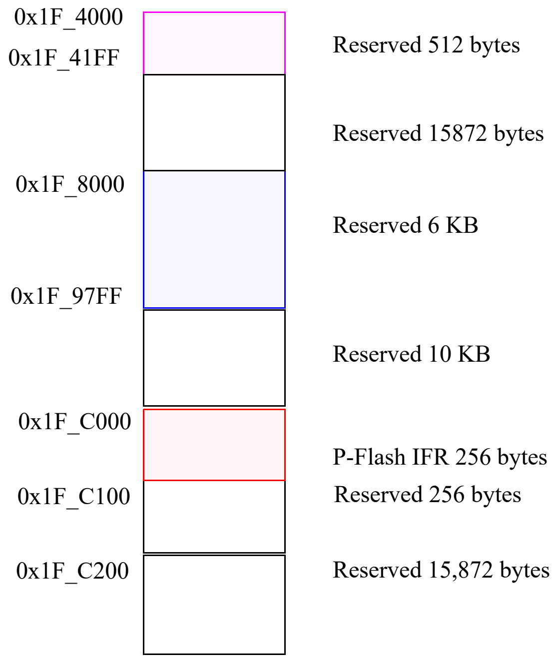


Figure 22-3. Memory Controller Resource Memory Map (NVM Resources Area)

22.3.2 Register Descriptions

The Flash module contains a set of 24 control and status registers located between Flash module base + 0x0000 and 0x0017.

In the case of the writable registers, the write accesses are forbidden during Flash command execution (for more detail, see Caution note in <st-blue>Section 22.3 Memory Map and Registers).

A summary of the Flash module registers is given in Figure 22-4 with detailed descriptions in the following subsections.

Address & Name		7	6	5	4	3	2	1	0
0x0000 FCLKDIV	R	FDIVLD	FDIVLCK	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
0x0001 FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
	W								

Figure 22-4. FTMRZ192K2K Register Summary

Address & Name		7	6	5	4	3	2	1	0
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								
0x0003 FPSTAT	R	FPOVRD	0	0	0	0	0	0	WSTATACK
	W								
0x0004 FCNFG	R	CCIE	0	ERSAREQ	IGNSF	WSTAT[1:0]		FDFD	FSFD
	W								
0x0005 FERCNFG	R	0	0	0	0	0	0	0	SFDIE
	W								
0x0006 FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
	W								
0x0007 FERSTAT	R	0	0	0	0	0	0	DFDF	SFDIF
	W								
0x0008 FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
	W								
0x0009 DFPROT	R	DPOPEN	0	DPS5	DPS4	DPS3	DPS2	DPS1	DPS0
	W								
0x000A FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
0x000B FRSV1	R	0	0	0	0	0	0	0	0
	W								
0x000C FCCOB0HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000D FCCOB0LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								
0x000E FCCOB1HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000F FCCOB1LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								

Figure 22-4. FTMRZ192K2K Register Summary (continued)

Address & Name		7	6	5	4	3	2	1	0
0x0010 FCCOB2HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0011 FCCOB2LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0012 FCCOB3HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0013 FCCOB3LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0014 FCCOB4HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0015 FCCOB4LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0016 FCCOB5HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0017 FCCOB5LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0

▨ = Unimplemented or Reserved

Figure 22-4. FTMRZ192K2K Register Summary (continued)

22.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

Offset Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	FDIVLD	FDIVLCK	FDIV[5:0]					
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-5. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bit 7 is not writable, bit 6 is write-once-hi and controls the writability of the FDIV field in normal mode. In special mode, bits 6-0 are writable any number of times but bit 7 remains unwritable.

CAUTION

> The FCLKDIV register should never be written while a Flash command is executing (CCIF=0).

Table 22-6. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded** 0 FCLKDIV register has not been written since the last reset 1 FCLKDIV register has been written since the last reset
6 FDIVLCK	**Clock Divider Locked** 0 FDIV field is open for writing 1 FDIV value is locked and cannot be changed. Once the lock bit is set high, only reset can clear this bit and restore writability to the FDIV field in normal mode.
5–0 FDIV[5:0]	**Clock Divider Bits** — FDIV[5:0] must be set to effectively divide BUSCLK down to 1 MHz to control timed events during Flash program and erase algorithms. Table 22-7 shows recommended values for FDIV[5:0] based on the BUSCLK frequency. Please refer to Section 22.4.5, “Flash Command Operations,” for more information.

Table 22-7. FDIV values for various BUSCLK Frequencies

BUSCLK Frequency (MHz)		FDIV[5:0]
MIN[1]	MAX[2]	
1.0	1.6	0x00
1.6	2.6	0x01
2.6	3.6	0x02
3.6	4.6	0x03
4.6	5.6	0x04
5.6	6.6	0x05
6.6	7.6	0x06
7.6	8.6	0x07
8.6	9.6	0x08
9.6	10.6	0x09
10.6	11.6	0x0A
11.6	12.6	0x0B
12.6	13.6	0x0C
13.6	14.6	0x0D
14.6	15.6	0x0E
15.6	16.6	0x0F
16.6	17.6	0x10
17.6	18.6	0x11
18.6	19.6	0x12
19.6	20.6	0x13
20.6	21.6	0x14
21.6	22.6	0x15
22.6	23.6	0x16
23.6	24.6	0x17
24.6	25.6	0x18
25.6	26.6	0x19
26.6	27.6	0x1A
27.6	28.6	0x1B
28.6	29.6	0x1C
29.6	30.6	0x1D
30.6	31.6	0x1E
31.6	32.6	0x1F
32.6	33.6	0x20
33.6	34.6	0x21
34.6	35.6	0x22
35.6	36.6	0x23
36.6	37.6	0x24
37.6	38.6	0x25
38.6	39.6	0x26
39.6	40.6	0x27
40.6	41.6	0x28
41.6	42.6	0x29
42.6	43.6	0x2A
43.6	44.6	0x2B
44.6	45.6	0x2C
45.6	46.6	0x2D
46.6	47.6	0x2E
47.6	48.6	0x2F
48.6	49.6	0x30
49.6	50.6	0x31

[1] BUSCLK is Greater Than this value.

[2] BUSCLK is Less Than or Equal to this value.

22.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

Offset Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	KEYEN[1:0]		RNV[5:2]				SEC[1:0]	
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

▨ = Unimplemented or Reserved

Figure 22-6. Flash Security Register (FSEC)

[1] Loaded from Flash configuration field, during reset sequence.

All bits in the FSEC register are readable but not writable.

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0xFF_FE0F located in P-Flash memory (see Table 22-3) as indicated by reset condition F in Figure 22-6. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

Table 22-8. FSEC Field Descriptions

Field	Description
7–6 KEYEN[1:0]	**Backdoor Key Security Enable Bits** — The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 22-9.
5–2 RNV[5:2]	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements.
1–0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 22-10. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 22-9. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED[1]
10	ENABLED
11	DISABLED

[1] Preferred KEYEN state to disable backdoor key access.

Table 22-10. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED[1]
10	UNSECURED
11	SECURED

[1] Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in <st-blue>Section 22.5 Security.

22.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to indicate the amount of parameters loaded into the FCCOB registers for Flash memory operations.

Offset Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 22-11. FCCOBIX Field Descriptions

Field	Description
2–0 CCOBIX[1:0]	**Common Command Register Index**— The CCOBIX bits are used to indicate how many words of the FCCOB register array are being read or written to. See 22.3.2.13 Flash Common Command Object Registers (FCCOB),” for more details.

22.3.2.4 Flash Protection Status Register (FPSTAT)

This Flash register holds the status of the Protection Override feature.

Offset Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	FPOVRD	0	0	0	0	0	0	WSTATACK
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	1

▨ = Unimplemented or Reserved

Figure 22-8. Flash Protection Status Register (FPSTAT)

All bits in the FPSTAT register are readable but are not writable.

Table 22-12. FPSTAT Field Descriptions

Field	Description
7 FPOVRD	**Flash Protection Override Status** — The FPOVRD bit indicates if the Protection Override feature is currently enabled. See Section 22.4.7.17, "Protection Override Command" for more details. 0 Protection is not overridden 1 Protection is overridden, contents of registers FPROT and/or DFPROT (and effective protection limits determined by their current contents) were determined during execution of command Protection Override
0 WSTATACK	**Wait-State Switch Acknowledge** — The WSTATACK bit indicates that the wait-state configuration is effectively set according to the value configured on bits FCNFG[WSTAT] (see Section 22.3.2.5, "Flash Configuration Register (FCNFG)"). WSTATACK bit is cleared when a change in FCNFG[WSTAT] is requested by writing to those bits, and is set when the Flash has effectively switched to the new wait-state configuration. The application must check the status of WSTATACK bit to make sure it reads as 1 before changing the frequency setup (see Section 22.4.3, "Flash Block Read Access"). 0 Wait-State switch is pending, Flash reads are still happening according to the previous value of FCNFG[WSTAT] 1 Wait-State switch is complete, Flash reads are already working according to the value set on FCNFG[WSTAT]

22.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt, control generation of wait-states and forces ECC faults on Flash array read access from the CPU.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	ERSAREQ	IGNSF	WSTAT[1:0]		FDFD	FSFD
W		▨	▨					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, WSTAT, FDFD, and FSFD bits are readable and writable, ERSAREQ bit is read only, and remaining bits read 0 and are not writable.

Table 22-13. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see <st-blue>Section 22.3.2.7 Flash Status Register (FSTAT))
5 ERSAREQ	**Erase All Request** — Requests 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 Reference Manual for assertion of the *soc_erase_all_req* input to the FTMRZ module. 0 No request or request complete 1 Request to: a) run the Erase All Blocks command b) verify the erased state c) program the security byte in the Flash Configuration Field to the unsecure state d) release MCU security by setting the SEC field of the FSEC register to the unsecure state as defined in Table 22-8. of <st-blue>Section 22.3.2.2 Flash Security Register (FSEC). The ERSAREQ bit sets to 1 when *soc_erase_all_req* is asserted, CCIF=1 and the Memory Controller starts executing the sequence. ERSAREQ will be reset to 0 by the Memory Controller when the operation is completed (see <st-blue>Section 22.4.7.7.1 Erase All Pin).
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see <st-blue>Section 22.3.2.8 Flash Error Status Register (FERSTAT)). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated
3–2 WSTAT[1:0]	**Wait State control bits** — The WSTAT[1:0] bits define how many wait-states are inserted on each read access to the Flash as shown on Table 22-14..Right after reset the maximum amount of wait-states is set, to be later re-configured by the application if needed. Depending on the system operating frequency being used the number of wait-states can be reduced or disabled, please refer to the Data Sheet for details. For additional information regarding the procedure to change this configuration please see <st-blue>Section 22.4.3 Flash Block Read Access. The WSTAT[1:0] bits should not be updated while the Flash is executing a command (CCIF=0); if that happens the value of this field will not change and no action will take place.
1 FDFD	**Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations. The FDFD bit is cleared by writing a 0 to FDFD. 0 Flash array read operations will set the DFDF flag in the FERSTAT register only if a double bit fault is detected 1 Any Flash array read operation will force the DFDF flag in the FERSTAT register to be set (see <st-blue>Section 22.3.2.7 Flash Status Register (FSTAT))
0 FSFD	**Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD. 0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected 1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see <st-blue>Section 22.3.2.7 Flash Status Register (FSTAT)) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see <st-blue>Section 22.3.2.6 Flash Error Configuration Register (FERCNFG))

Table 22-14. Flash Wait-States control

WSTAT[1:0]	Wait-State configuration
00	ENABLED, maximum number of cycles[1]
01	reserved[2]
10	reserved[2]
11	DISABLED

[1] Reset condition. For a target of 100MHz core frequency / 50MHz bus frequency the maximum number required is 1 cycle.

[2] Value will read as 01 or 10, as written. In the current implementation the Flash will behave the same as 00 (wait-states enabled, maximum number of cycles).

22.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

Offset Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	SFDIE
W	▨	▨	▨	▨	▨	▨	▨	
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 22-15. FERCNFG Field Descriptions

Field	Description
0 SFDIE	**Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation. 0 SFDIF interrupt disabled whenever the SFDIF flag is set (see <st-blue>Section 22.3.2.8 Flash Error Status Register (FERSTAT)) 1 An interrupt will be requested whenever the SFDIF flag is set (see <st-blue>Section 22.3.2.8 Flash Error Status Register (FERSTAT))

22.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

Offset Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT[1:0]	
W		▨			▨	▨	▨	▨
Reset	1	0	0	0	0	0	0[1]	0[1]

▨ = Unimplemented or Reserved

Figure 22-11. Flash Status Register (FSTAT)

[1] Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see <st-blue>Section 22.6 Initialization).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

Table 22-16. FSTAT Field Descriptions

Field	Description
7 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command in progress 1 Flash command has completed
5 ACCERR	**Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see <st-blue>Section 22.4.5.2 Command Write Sequence) or issuing an illegal Flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR. 0 No access error detected 1 Access error detected
4 FPVIOL	**Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash or EEPROM memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected 1 Protection violation detected
3 MGBUSY	**Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller**.** 0 Memory Controller is idle 1 Memory Controller is busy executing a Flash command (CCIF = 0)
2 RSVD	**Reserved Bit** — This bit is reserved and always reads 0**.**
1–0 MGSTAT[1:0]	**Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. The MGSTAT bits are cleared automatically at the start of the execution of a Flash command. See Section 22.4.7, “Flash Command Description,” and Section 22.6, “Initialization” for details.

22.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

Offset Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DFDF	SFDIF
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 22-17. FERSTAT Field Descriptions

Field	Description
1 DFDF	**Double Bit Fault Detect Flag** — The setting of the DFDF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation returning invalid data was attempted on a Flash block that was under a Flash command operation.[1] The DFDF flag is cleared by writing a 1 to DFDF. Writing a 0 to DFDF has no effect on DFDF.[2] 0 No double bit fault detected 1 Double bit fault detected or a Flash array read operation returning invalid data was attempted while command running. See Section 22.4.3, “Flash Block Read Access” for details
0 SFDIF	**Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation returning invalid data was attempted on a Flash block that was under a Flash command operation. The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF. 0 No single bit fault detected 1 Single bit fault detected and corrected or a Flash array read operation returning invalid data was attempted while command running

[1] In case of ECC errors the corresponding flag must be cleared for the proper setting of any further error, i.e. any new error will only be indicated properly when DFDF and/or SFDIF are clear at the time the error condition is detected.

[2] There is a one cycle delay in storing the ECC DFDF and SFDIF fault flags in this register. At least one NOP is required after a flash memory read before checking FERSTAT for the occurrence of ECC errors.

22.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

Offset Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS[1:0]		FPLDIS	FPLS[1:0]	
W		▨						
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

▨ = Unimplemented or Reserved

Figure 22-13. Flash Protection Register (FPROT)

[1] Loaded from Flash configuration field, during reset sequence.

The (unreserved) bits of the FPROT register are writable in Normal Single Chip Mode with the restriction that the size of the protected region can only be increased(see Section 22.3.2.9.1, "P-Flash Protection Restrictions," and Table 22-22.). All (unreserved) bits of the FPROT register are writable without restriction in Special Single Chip Mode.

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0xFF_FE0C located in P-Flash memory (see Table 22-3) as indicated by reset condition 'F' in Figure 22-13. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

Table 22-18. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 22-19 for the P-Flash block. 0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits 1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0xFF_FFFF. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4–3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 22-20. The FPHS bits can only be written to while the FPHDIS bit is set.
2 FPLDIS	**Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0xFF_8000. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1–0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 22-21. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 22-19. P-Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No P-Flash Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full P-Flash Memory Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

[1] For range sizes, refer to Table 22-20 and Table 22-21.

Table 22-20. P-Flash Protection Higher Address Range

FPHS[1:0]	Global Address Range	Protected Size
00	0xFF_F800–0xFF_FFFF	2 KB
01	0xFF_F000–0xFF_FFFF	4 KB
10	0xFF_E000–0xFF_FFFF	8 KB
11	0xFF_C000–0xFF_FFFF	16 KB

Table 22-21. P-Flash Protection Lower Address Range

FPLS[1:0]	Global Address Range	Protected Size
00	0xFF_8000–0xFF_83FF	1 KB
01	0xFF_8000–0xFF_87FF	2 KB
10	0xFF_8000–0xFF_8FFF	4 KB
11	0xFF_8000–0xFF_9FFF	8 KB

All possible P-Flash protection scenarios are shown in Figure 22-14 . Although the protection scheme is loaded from the Flash memory at global address 0xFF_FE0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in Normal Single Chip Mode while providing as much protection as possible if reprogramming is not required.

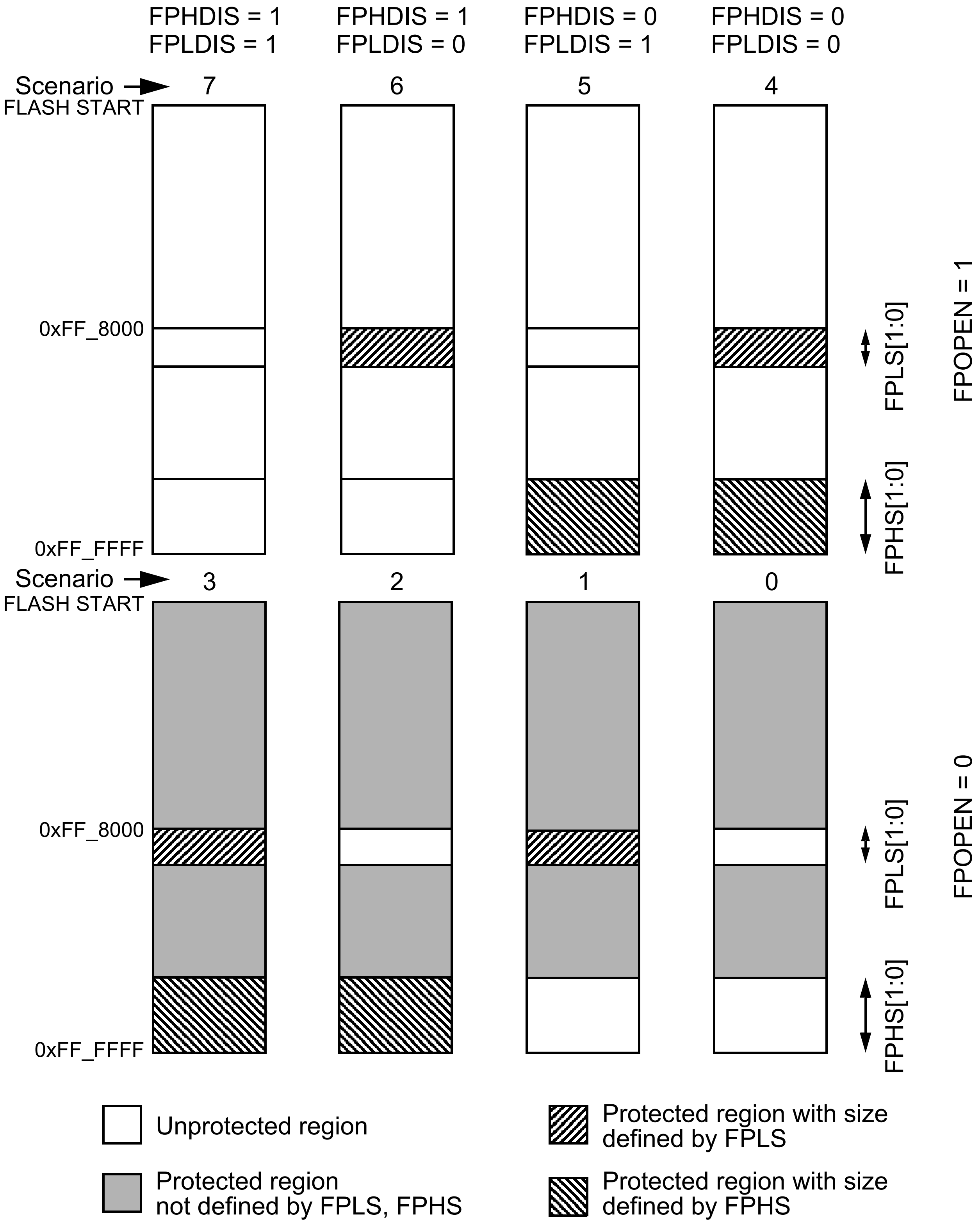


Figure 22-14. P-Flash Protection Scenarios

22.3.2.9.1 P-Flash Protection Restrictions

In Normal Single Chip Mode the general guideline is that P-Flash protection can only be added and not removed. Table 22-22 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

Table 22-22. P-Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

[1] Allowed transitions marked with X, see Figure 22-14 for a definition of the scenarios.

22.3.2.10 EEPROM Protection Register (DFPROT)

The DFPROT register defines which EEPROM sectors are protected against program and erase operations.

Offset Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	DPOPEN	0	DPS[5:0]					
W								
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

Figure 22-15. EEPROM Protection Register (DFPROT)

[1] Loaded from Flash configuration field, during reset sequence.

The (unreserved) bits of the DFPROT register are writable in Normal Single Chip Mode with the restriction that protection can be added but not removed. Writes in Normal Single Chip Mode must increase the DPS value and the DPOPEN bit can only be written from 1 (protection disabled) to 0 (protection enabled). If the DPOPEN bit is set, the state of the DPS bits is irrelevant.All DPOPEN/DPS bit registers are writable without restriction in Special Single Chip Mode.

During the reset sequence, fields DPOPEN and DPS of the DFPROT register are loaded with the contents of the EEPROM protection byte in the Flash configuration field at global address 0xFF_FE0D located in

P-Flash memory (see Table 22-3) as indicated by reset condition F in Table 22-24.. To change the EEPROM protection that will be loaded during the reset sequence, the P-Flash sector containing the EEPROM protection byte must be unprotected, then the EEPROM protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase containing the EEPROM protection byte during the reset sequence, the DPOPEN bit will be cleared and DPS bits will be set to leave the EEPROM memory fully protected.

Trying to alter data in any protected area in the EEPROM memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. Block erase of the EEPROM memory is not possible if any of the EEPROM sectors are protected.

Table 22-23. DFPROT Field Descriptions

Field	Description
7 DPOPEN	**EEPROM Protection Control** 0 Enables EEPROM memory protection from program and erase with protected address range defined by DPS bits 1 Disables EEPROM memory protection from program and erase
5–0 DPS[5:0]	**EEPROM Protection Size** — The DPS[5:0] bits determine the size of the protected area in the EEPROM memory, this size increase in step of 32 bytes, as shown in Table 22-24. .

Table 22-24. EEPROM Protection Address Range

DPS[5:0]	Global Address Range	Protected Size
000000	0x10_0000 – 0x10_001F	32 bytes
000001	0x10_0000 – 0x10_003F	64 bytes
000010	0x10_0000 – 0x10_005F	96 bytes
000011	0x10_0000 – 0x10_007F	128 bytes
000100	0x10_0000 – 0x10_009F	160 bytes
000101	0x10_0000 – 0x10_00BF	192 bytes
The Protection Size goes on enlarging in step of 32 bytes, for each DPS value increasing of one. . . .		
111111	0x10_0000 – 0x10_07FF	2,048 bytes

22.3.2.11 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R	NV[7:0]							
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

▨ = Unimplemented or Reserved

Figure 22-16. Flash Option Register (FOPT)

[1] Loaded from Flash configuration field, during reset sequence.

All bits in the FOPT register are readable but are not writable.

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0xFF_FE0E located in P-Flash memory (see Table 22-3) as indicated by reset condition F in Figure 22-16. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 22-25. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the Device Overview for proper use of the NV bits.

22.3.2.12 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

Offset Module Base + 0x000B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 22-17. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

22.3.2.13 Flash Common Command Object Registers (FCCOB)

The FCCOB is an array of six words. Byte wide reads and writes are allowed to the FCCOB registers.

Offset Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	CCOB[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-18. Flash Common Command Object 0 High Register (FCCOB0HI)

Offset Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	CCOB[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-19. Flash Common Command Object 0 Low Register (FCCOB0LO)

Offset Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	CCOB[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-20. Flash Common Command Object 1 High Register (FCCOB1HI)

Offset Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	CCOB[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-21. Flash Common Command Object 1 Low Register (FCCOB1LO)

Offset Module Base + 0x0010

	7	6	5	4	3	2	1	0
R	CCOB[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-22. Flash Common Command Object 2 High Register (FCCOB2HI)

Offset Module Base + 0x0011

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-23. Flash Common Command Object 2 Low Register (FCCOB2LO)

Offset Module Base + 0x0012

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 22-24. Flash Common Command Object 3 High Register (FCCOB3HI)

Offset Module Base + 0x0013

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-25. Flash Common Command Object 3 Low Register (FCCOB3LO)

Offset Module Base + 0x0014

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 22-26. Flash Common Command Object 4 High Register (FCCOB4HI)

Offset Module Base + 0x0015

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-27. Flash Common Command Object 4 Low Register (FCCOB4LO)

Offset Module Base + 0x0016

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 22-28. Flash Common Command Object 5 High Register (FCCOB5HI)

Offset Module Base + 0x0017

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-29. Flash Common Command Object 5 Low Register (FCCOB5LO)

22.3.2.13.1 FCCOB - NVM Command Mode

NVM command mode uses the FCCOB registers to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 22-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. The value written to the FCCOBIX field must reflect the amount of CCOB words loaded for command execution.

Table 22-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in <st-blue>Section 22.4.7 Flash Command Description.

Table 22-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Register	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	FCCOB0	HI	FCMD[7:0] defining Flash command
		LO	Global address [23:16]
001	FCCOB1	HI	Global address [15:8]
		LO	Global address [7:0]

Table 22-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Register	Byte	FCCOB Parameter Fields (NVM Command Mode)
010	FCCOB2	HI	Data 0 [15:8]
		LO	Data 0 [7:0]
011	FCCOB3	HI	Data 1 [15:8]
		LO	Data 1 [7:0]
100	FCCOB4	HI	Data 2 [15:8]
		LO	Data 2 [7:0]
101	FCCOB5	HI	Data 3 [15:8]
		LO	Data 3 [7:0]

22.4 Functional Description

22.4.1 Modes of Operation

The FTMRZ192K2K module provides the modes of operation normal and special . The operating mode is determined by module-level inputs and affects the FCLKDIV, FCNFG, and DFPROT registers (see Table 22-28.).

22.4.2 IFR Version ID Word

The version ID word is stored in the IFR at address 0x1F_C0B6. The contents of the word are defined in Table 22-27.

Table 22-27. IFR Version ID Fields

[15:4]	[3:0]
Reserved	VERNUM

- VERNUM: Version number. The first version is number 0b_0001 with both 0b_0000 and 0b_1111 meaning 'none'.

22.4.3 Flash Block Read Access

If data read from the Flash block results in a double-bit fault ECC error (meaning that data is detected to be in error and cannot be corrected), the read data will be tagged as invalid during that access (please look into the Reference Manual for details). Forcing the DFDF status bit by setting FDFD (see <st-blue>Section

22.3.2.5 Flash Configuration Register (FCNFG)) has effect only on the DFDF status bit value and does not result in an invalid access.

To guarantee the proper read timing from the Flash array, the FTMRZ192K2K FMU will control (i.e. pause) the S12Z core accesses, considering that the MCU can be configured to fetch data at a faster frequency than the Flash block can support. Right after reset the FTMRZ192K2K FMU will be configured to run with the maximum amount of wait-states enabled; if the user application is setup to run at a slower frequency the control bits FCNFG[WSTAT] (see <st-blue>Section 22.3.2.5 Flash Configuration Register (FCNFG)) can be configured by the user to disable the generation of wait-states, so it does not impose a performance penalty to the system if the read timing of the S12Z core is setup to be within the margins of the Flash block. For a definition of the frequency values where wait-states can be disabled please look into the Reference Manual.

The following sequence must be followed when the transition from a higher frequency to a lower frequency is going to happen:

- Flash resets with wait-states enabled;
- system frequency must be configured to the lower target;
- user writes to FNCNF[WSTAT] to disable wait-states;
- user reads the value of FPSTAT[WSTATACK], the new wait-state configuration will be effective when it reads as 1;
- user must re-write FCLKDIV to set a new value based on the lower frequency.

The following sequence must be followed on the contrary direction, going from a lower frequency to a higher frequency:

- user writes to FCNFG[WSTAT] to enable wait-states;
- user reads the value of FPSTAT[WSTATACK], the new wait-state configuration will be effective when it reads as 1;
- user must re-write FCLKDIV to set a new value based on the higher frequency;
- system frequency must be set to the upper target.

CAUTION

> If the application is going to require the frequency setup to change, the value to be loaded on register FCLKDIV will have to be updated according to the new frequency value. In this scenario the application must take care to avoid locking the value of the FCLKDIV register: bit FDIVLCK must not be set if the value to be loaded on FDIV is going to be re-written, otherwise a reset is going to be required. Please refer to Section 22.3.2.1, “Flash Clock Divider Register (FCLKDIV) and Section 22.4.5.1, “Writing the FCLKDIV Register.

22.4.4 Internal NVM resource

IFR is an internal NVM resource readable by CPU . The IFR fields are shown in Table 22-4..

The NVM Resource Area global address map is shown in Table 22-5..

22.4.5 Flash Command Operations

Flash command operations are used to modify Flash memory contents.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from BUSCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution, according to MCU functional mode and MCU security state.

22.4.5.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide BUSCLK down to a target FCLK of 1 MHz. Table 22-7. shows recommended values for the FDIV field based on BUSCLK frequency.

NOTE

> Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 0.8 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

22.4.5.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see <st-blue>Section 22.3.2.7 Flash Status Register (FSTAT)) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

22.4.5.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. The CCOBIX bits in the FCCOBIX register must reflect the amount of words loaded into the FCCOB registers (see <st-blue>Section 22.3.2.3 Flash CCOB Index Register (FCCOBIX)).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 22-30.

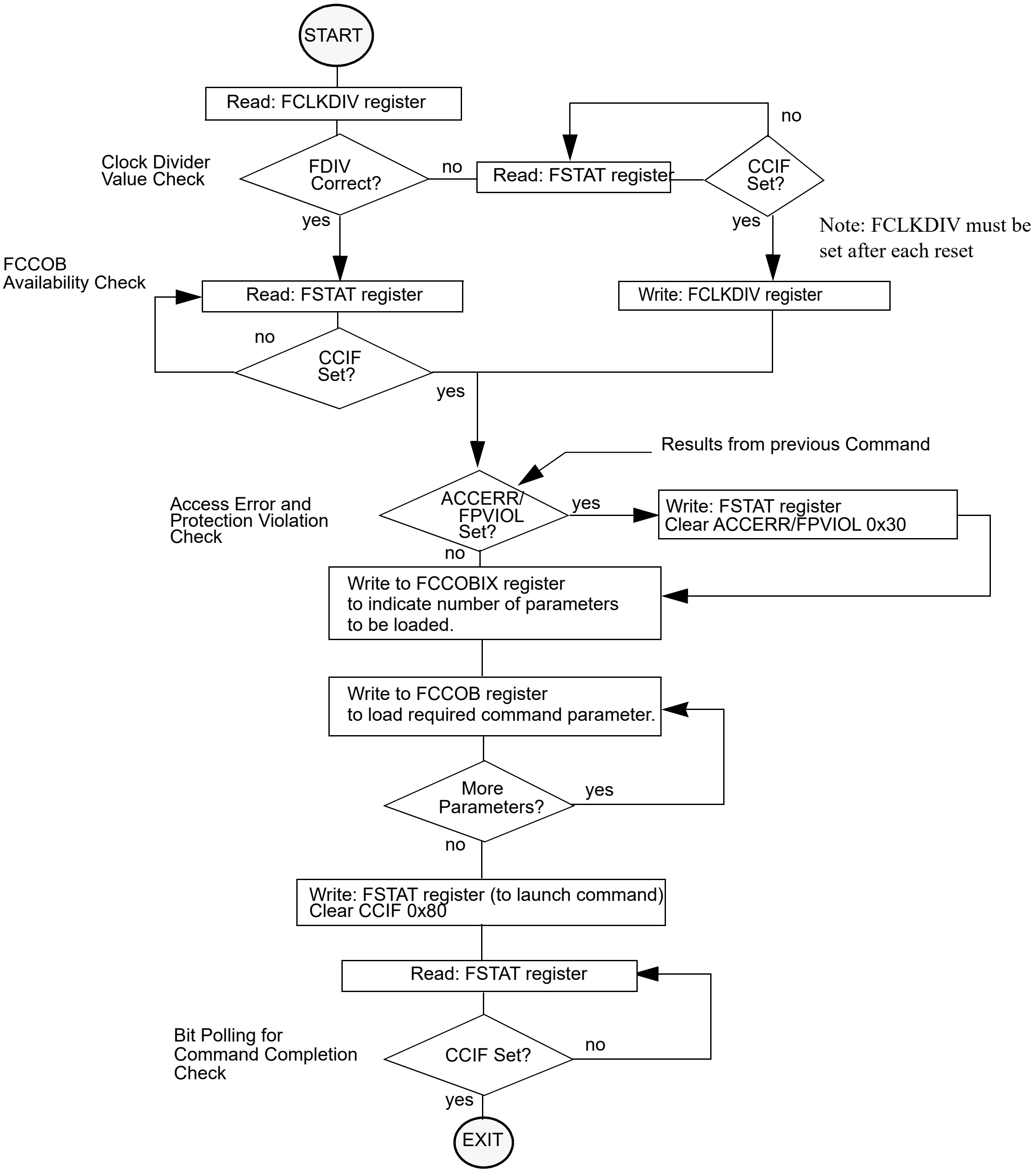


Figure 22-30. Generic Flash Command Write Sequence Flowchart

22.4.5.3 Valid Flash Module Commands

Table 22-28. present the valid Flash commands, as enabled by the combination of the functional MCU mode (Normal SingleChip NS, Special Singlechip SS) with the MCU security state (Unsecured, Secured).

Table 22-28. Flash Commands by Mode and Security State

FCMD	Command	Unsecured		Secured	
		NS[1]	SS[2]	NS[3]	SS[4]
0x01	Erase Verify All Blocks	*	*	*	
0x02	Erase Verify Block	*	*	*	
0x03	Erase Verify P-Flash Section	*	*	*	
0x04	Read Once	*	*	*	
0x06	Program P-Flash	*	*	*	
0x07	Program Once	*	*	*	
0x08	Erase All Blocks		*		
0x09	Erase Flash Block	*	*	*	
0x0A	Erase P-Flash Sector	*	*	*	
0x0B	Unsecure Flash		*		
0x0C	Verify Backdoor Access Key	*		*	
0x0D	Set User Margin Level	*	*	*	
0x0E	Set Field Margin Level		*		
0x10	Erase Verify EEPROM Section	*	*	*	
0x11	Program EEPROM	*	*	*	
0x12	Erase EEPROM Sector	*	*	*	
0x13	Protection Override	*	*	*	

[1] Unsecured Normal Single Chip mode

[2] Unsecured Special Single Chip mode.

[3] Secured Normal Single Chip mode.

[4] Secured Special Single Chip mode.Please refer to <st-blue>Section 22.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM.

22.4.5.4 P-Flash Commands

Table 22-29 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

Table 22-29. P-Flash Commands

FCMD	Command	Function on P-Flash Memory
0x01	Erase Verify All Blocks	Verify that all P-Flash (and EEPROM) blocks are erased.
0x02	Erase Verify Block	Verify that a P-Flash block is erased.
0x03	Erase Verify P-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x04	Read Once	Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block that was previously programmed using the Program Once command.
0x06	Program P-Flash	Program a phrase in a P-Flash block.
0x07	Program Once	Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block that is allowed to be programmed only once.
0x08	Erase All Blocks	Erase all P-Flash (and EEPROM) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the DPOPEN bit in the DFPROT register are set prior to launching the command.
0x09	Erase Flash Block	Erase a P-Flash (or EEPROM) block. An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
0x0A	Erase P-Flash Sector	Erase all bytes in a P-Flash sector.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and EEPROM) blocks and verifying that all P-Flash (and EEPROM) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).
0x13	Protection Override	Supports a mode to temporarily override Protection configuration (for P-Flash and/or EEPROM) by verifying a key.

22.4.5.5 EEPROM Commands

Table 22-30 summarizes the valid EEPROM commands along with the effects of the commands on the EEPROM block.

Table 22-30. EEPROM Commands

FCMD	Command	Function on EEPROM Memory
0x01	Erase Verify All Blocks	Verify that all EEPROM (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the EEPROM block is erased.
0x08	Erase All Blocks	Erase all EEPROM (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the DPOPEN bit in the DFPROT register are set prior to launching the command.
0x09	Erase Flash Block	Erase a EEPROM (or P-Flash) block. An erase of the full EEPROM block is only possible when DPOPEN bit in the DFPROT register is set prior to launching the command.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all EEPROM (and P-Flash) blocks and verifying that all EEPROM (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the EEPROM block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the EEPROM block (special modes only).
0x10	Erase Verify EEPROM Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program EEPROM	Program up to four words in the EEPROM block.
0x12	Erase EEPROM Sector	Erase all bytes in a sector of the EEPROM block.
0x13	Protection Override	Supports a mode to temporarily override Protection configuration (for P-Flash and/or EEPROM) by verifying a key.

22.4.6 Allowed Simultaneous P-Flash and EEPROM Operations

Only the operations marked ‘OK’ in Table 22-31. are permitted to be run simultaneously on the Program Flash and EEPROM blocks. Some operations cannot be executed simultaneously because certain hardware resources are shared by the two memories. The priority has been placed on permitting Program Flash reads while program and erase operations execute on the EEPROM, providing read (P-Flash) while write (EEPROM) functionality. Any attempt to access P-Flash and EEPROM simultaneously when it is not allowed will result in an illegal access that will trigger a machine exception in the CPU (please look into the Reference Manual for details). Please note that during the execution of each command there is a period, before the operation in the Flash array actually starts, where reading is allowed and valid data is returned. Even if the simultaneous operation is marked as not allowed the Flash will report an illegal access only in the cycle the read collision actually happens, maximizing the time the array is available for reading.

Table 22-31. Allowed P-Flash and EEPROM Simultaneous Operations

	EEPROM				
Program Flash	**Read**	**Margin Read**[2]	**Program**	**Sector Erase**	**Mass Erase**[2]
Read	OK[1]	OK	OK	OK	▨
Margin Read[2]	▨	▨	▨	▨	▨
Program	▨	▨	▨	▨	▨
Sector Erase	▨	▨	▨	▨	▨
Mass Erase[3]	▨	▨	▨	▨	OK

[1] Strictly speaking, only one read of either the P-Flash or EEPROM can occur at any given instant, but the memory controller will transparently arbitrate P-Flash and EEPROM accesses giving uninterrupted read access whenever possible.

[2] A ‘Margin Read’ is any read after executing the margin setting commands ‘Set User Margin Level’ or ‘Set Field Margin Level’ with anything but the ‘normal’ level specified. See the Note on margin settings in <st-blue>Section 22.4.7.12 Set User Margin Level Command and <st-blue>Section 22.4.7.13 Set Field Margin Level Command.

[3] The ‘Mass Erase’ operations are commands ‘Erase All Blocks’ and ‘Erase Flash Block’

22.4.7 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation may return invalid data resulting in an illegal access (as described on <st-blue>Section 22.4.6 Allowed Simultaneous P-Flash and EEPROM Operations).

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see <st-blue>Section 22.3.2.7 Flash Status Register (FSTAT)).

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

22.4.7.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and EEPROM blocks have been erased.

Table 22-32. Erase Verify All Blocks Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x01	Not required

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed. If all blocks are not erased, it means blank check failed, both MGSTAT bits will be set.

Table 22-33. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read约 if blank check failed .
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or if blank check failed.

22.4.7.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or EEPROM block has been erased.

Table 22-34. Erase Verify Block Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x02	Global address [23:16] to identify Flash block
FCCOB1	Global address [15:0] to identify Flash block	

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or EEPROM block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.If the block is not erased, it means blank check failed, both MGSTAT bits will be set.

Table 22-35. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if an invalid global address [23:0] is supplied see Table 22-2)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read or if blank check failed.
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or if blank check failed.

22.4.7.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases.

Table 22-36. Erase Verify P-Flash Section Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x03	Global address [23:16] of a P-Flash block
FCCOB1	Global address [15:0] of the first phrase to be verified	
FCCOB2	Number of phrases to be verified	

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed. If the section is not erased, it means blank check failed, both MGSTAT bits will be set.

Table 22-37. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied see Table 22-2)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if the requested section crosses a the P-Flash address boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read or if blank check failed.
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or if blank check failed.

22.4.7.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash. The Read Once field is programmed using the Program Once command described in <st-blue>Section 22.4.7.6 Program Once Command. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 22-38. Read Once Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x04	Not Required
FCCOB1	Read Once phrase index (0x0000 - 0x0007)	
FCCOB2	Read Once word 0 value	
FCCOB3	Read Once word 1 value	
FCCOB4	Read Once word 2 value	
FCCOB5	Read Once word 3 value	

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block will return invalid data.

Table 22-39. Read Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read

22.4.7.5 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

A P-Flash phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash phrase is not allowed.

Table 22-40. Program P-Flash Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x06	Global address [23:16] to identify P-Flash block
FCCOB1	Global address [15:0] of phrase location to be programmed[1]	
FCCOB2	Word 0 program value	
FCCOB3	Word 1 program value	
FCCOB4	Word 2 program value	
FCCOB5	Word 3 program value	

[1] Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

Table 22-41. Program P-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied see Table 22-2)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the global address [17:0] points to a protected area
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.6 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash. The Program Once reserved field can be read using the Read Once command as described in <st-blue>Section 22.4.7.4 Read Once Command. The Program Once command must only be issued once since the nonvolatile information register in P-Flash cannot be erased. The Program Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 22-42. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
FCCOB0	0x07	Not Required
FCCOB1	Program Once phrase index (0x0000 - 0x0007)	
FCCOB2	Program Once word 0 value	

Table 22-42. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters
FCCOB3	Program Once word 1 value
FCCOB4	Program Once word 2 value
FCCOB5	Program Once word 3 value

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash will return invalid data.

Table 22-43. Program Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed[1]
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

[1] If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

22.4.7.7 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and EEPROM memory space.

Table 22-44. Erase All Blocks Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 22-45. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 22-28)
	FPVIOL	Set if any area of the P-Flash or EEPROM memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.7.1 Erase All Pin

The functionality of the Erase All Blocks command is also available in an uncommanded fashion from the *soc_erase_all_req* input pin on the Flash module. Refer to the Reference Manual for information on control of *soc_erase_all_req*.

The erase-all function requires the clock divider register FCLKDIV (see <st-blue>Section 22.3.2.1 Flash Clock Divider Register (FCLKDIV)) to be loaded before invoking this function using *soc_erase_all_req* input pin. Please refer to the Reference Manual for information about the default value of FCLKDIV in case direct writes to register FCLKDIV are not allowed by the time this feature is invoked. If FCLKDIV is not properly set the erase-all operation will not execute and the ACCERR flag in FSTAT register will set. After the execution of the erase-all function the FCLKDIV register will be reset and the value of register FCLKDIV must be loaded before launching any other command afterwards.

Before invoking the erase-all function using the *soc_erase_all_req* pin, the ACCERR and FPVIOL flags in the FSTAT register must be clear. When invoked from *soc_erase_all_req* the erase-all function will erase all P-Flash memory and EEPROM memory space regardless of the protection settings. If the post-erase verify passes, the routine will then release security by setting the SEC field of the FSEC register to the unsecure state (see <st-blue>Section 22.3.2.2 Flash Security Register (FSEC)). The security byte in the Flash Configuration Field will be programmed to the unsecure state (see Table 22-8.). The status of the erase-all request is reflected in the ERSAREQ bit in the FCNFG register (see <st-blue>Section 22.3.2.5 Flash Configuration Register (FCNFG)). The ERSAREQ bit in FCNFG will be cleared once the operation has completed and the normal FSTAT error reporting will be available as described inTable 22-46..

At the end of the erase-all sequence Protection will remain configured as it was before executing the erase-all function. If the application requires programming P-Flash and/or EEPROM after the erase-all function completes, the existing protection limits must be taken into account. If protection needs to be disabled the user may need to reset the system right after completing the erase-all function.

Table 22-46. Erase All Pin Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if command not available in current mode (see Table 22-28)
	MGSTAT1	Set if any errors have been encountered during the erase verify operation, or during the program verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the erase verify operation, or during the program verify operation

22.4.7.8 Erase Flash Block Command

The Erase Flash Block operation will erase all addresses in a P-Flash or EEPROM block.

Table 22-47. Erase Flash Block Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x09	Global address [23:16] to identify Flash block
FCCOB1	Global address [15:0] in Flash block to be erased	

Upon clearing CCIF to launch the Erase Flash Block command, the Memory Controller will erase the selected Flash block and verify that it is erased. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 22-48. Erase Flash Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied
		Set if the supplied P-Flash address is not phrase-aligned or if the EEPROM address is not word-aligned
	FPVIOL	Set if an area of the selected Flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.9 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 22-49. Erase P-Flash Sector Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x0A	Global address [23:16] to identify P-Flash block to be erased
FCCOB1	Global address [15:0] anywhere within the sector to be erased. Refer to <st-blue>Section 22.1.2.1 P-Flash Features for the P-Flash sector size.	

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

Table 22-50. Erase P-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied see Table 22-2)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the selected P-Flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.10 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and EEPROM memory space and, if the erase is successful, will release security.

Table 22-51. Unsecure Flash Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x0B	Not required

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and EEPROM memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

Table 22-52. Unsecure Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 22-28)
	FPVIOL	Set if any area of the P-Flash or EEPROM memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.11 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 22-9.). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see

Table 22-3.). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 22-53. Verify Backdoor Access Key Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x0C	Not required
FCCOB1	Key 0	
FCCOB2	Key 1	
FCCOB3	Key 2	
FCCOB4	Key 3	

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0xFF_FE00, etc. If the backdoor keys match, security will be 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 aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

Table 22-54. Verify Backdoor Access Key Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 100 at command launch
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see <st-blue>Section 22.3.2.2 Flash Security Register (FSEC))
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

22.4.7.12 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of the P-Flash or EEPROM block.

Table 22-55. Set User Margin Level Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x0D	Global address [23:16] to identify Flash block
FCCOB1	Global address [15:0] to identify Flash block	

Table 22-55. Set User Margin Level Command FCCOB Requirements

Register	FCCOB Parameters
FCCOB2	Margin level setting.

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

NOTE

When the EEPROM block is targeted, the EEPROM user margin levels are applied only to the EEPROM reads. However, when the P-Flash block is targeted, the P-Flash user margin levels are applied to both P-Flash and EEPROM reads. It is not possible to apply user margin levels to the P-Flash block only.

Valid margin level settings for the Set User Margin Level command are defined in Table 22-56..

Table 22-56. Valid Set User Margin Level Settings

FCCOB2	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level[1]
0x0002	User Margin-0 Level[2]

[1] Read margin to the erased state

[2] Read margin to the programmed state

Table 22-57. Set User Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied see Table 22-2)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

NOTE

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

22.4.7.13 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of the P-Flash or EEPROM block.

Table 22-58. Set Field Margin Level Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x0E	Global address [23:16] to identify Flash block
FCCOB1	Global address [15:0] to identify Flash block	
FCCOB2	Margin level setting.	

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the field margin level for the targeted block and then set the CCIF flag.

NOTE

When the EEPROM block is targeted, the EEPROM field margin levels are applied only to the EEPROM reads. However, when the P-Flash block is targeted, the P-Flash field margin levels are applied to both P-Flash and EEPROM reads. It is not possible to apply field margin levels to the P-Flash block only.

Valid margin level settings for the Set Field Margin Level command are defined in Table 22-59.

Table 22-59. Valid Set Field Margin Level Settings

FCCOB2	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level[1]
0x0002	User Margin-0 Level[2]
0x0003	Field Margin-1 Level[1]
0x0004	Field Margin-0 Level[2]

[1] Read margin to the erased state

[2] Read margin to the programmed state

Table 22-60. Set Field Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied see Table 22-2)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

CAUTION

Field margin levels must only be used during verify of the initial factory programming.

NOTE

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

22.4.7.14 Erase Verify EEPROM Section Command

The Erase Verify EEPROM Section command will verify that a section of code in the EEPROM is erased. The Erase Verify EEPROM Section command defines the starting point of the data to be verified and the number of words.

Table 22-61. Erase Verify EEPROM Section Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x10	Global address [23:16] to identify the EEPROM block
FCCOB1	Global address [15:0] of the first word to be verified	
FCCOB2	Number of words to be verified	

Upon clearing CCIF to launch the Erase Verify EEPROM Section command, the Memory Controller will verify the selected section of EEPROM memory is erased. The CCIF flag will set after the Erase Verify EEPROM Section operation has completed. If the section is not erased, it means blank check failed, both MGSTAT bits will be set.

Table 22-62. Erase Verify EEPROM Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the requested section breaches the end of the EEPROM block
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read or if blank check failed.
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or if blank check failed.

22.4.7.15 Program EEPROM Command

The Program EEPROM operation programs one to four previously erased words in the EEPROM block. The Program EEPROM operation will confirm that the targeted location(s) were successfully programmed upon completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

Table 22-63. Program EEPROM Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x11	Global address [23:16] to identify the EEPROM block
FCCOB1	Global address [15:0] of word to be programmed	
FCCOB2	Word 0 program value	
FCCOB3	Word 1 program value, if desired	
FCCOB4	Word 2 program value, if desired	
FCCOB5	Word 3 program value, if desired	

Upon clearing CCIF to launch the Program EEPROM command, the user-supplied words will be transferred to the Memory Controller and be programmed if the area is unprotected. The CCOBIX index value at Program EEPROM command launch determines how many words will be programmed in the EEPROM block. The CCIF flag is set when the operation has completed.

Table 22-64. Program EEPROM Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] > 101 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the requested group of words breaches the end of the EEPROM block
	FPVIOL	Set if the selected area of the EEPROM memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.16 Erase EEPROM Sector Command

The Erase EEPROM Sector operation will erase all addresses in a sector of the EEPROM block.

Table 22-65. Erase EEPROM Sector Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x12	Global address [23:16] to identify EEPROM block
FCCOB1	Global address [15:0] anywhere within the sector to be erased. See <st-blue>Section 22.1.2.2 EEPROM Features for EEPROM sector size.	

Upon clearing CCIF to launch the Erase EEPROM Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase EEPROM Sector operation has completed.

Table 22-66. Erase EEPROM Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 22-28)
		Set if an invalid global address [23:0] is suppliedsee Table 22-2
		Set if a misaligned word address is supplied (global address [0] != 0)
	FPVIOL	Set if the selected area of the EEPROM memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

22.4.7.17 Protection Override Command

The Protection Override command allows the user to temporarily override the protection limits, either decreasing, increasing or disabling protection limits, on P-Flash and/or EEPROM, if the comparison key provided as a parameter loaded on FCCOB matches the value of the key previously programmed on the Flash Configuration Field (see Table 22-3.). The value of the Protection Override Comparison Key must not be 16’hFFFF, that is considered invalid and if used as argument will cause the Protection Override feature to be disabled. Any valid key value that does not match the value programmed in the Flash Configuration Field will cause the Protection Override feature to be disabled. Current status of the Protection Override feature can be observed on FPSTAT FPOVRD bit (see Section 22.3.2.4, “Flash Protection Status Register (FPSTAT)).

Table 22-67. Protection Override Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x13	Protection Update Selection [1:0] See Table 22-68.
FCCOB1	Comparison Key	
FCCOB2	reserved	New FPROT value
FCCOB3	reserved	New DFPROT value

Table 22-68. Protection Override selection description

Protection Update Selection code [1:0]	Protection register selection
bit 0	Update P-Flash protection 0 - keep unchanged (do not update) 1 - update P-Flash protection with new FPROT value loaded on FCCOB
bit 1	Update EEPROM protection 0 - keep unchanged (do not update) 1 - update EEPROM protection with new DFPROT value loaded on FCCOB

If the comparison key successfully matches the key programmed in the Flash Configuration Field the Protection Override command will preserve the current values of registers FPROT and DFPROT stored in an internal area and will override these registers as selected by the Protection Update Selection field with the value(s) loaded on FCCOB parameters. The new values loaded into FPROT and/or DFPROT can reconfigure protection without any restriction (by increasing, decreasing or disabling protection limits). If the command executes successfully the FPSTAT FPOVRD bit will set.

If the comparison key does not match the key programmed in the Flash Configuration Field, or if the key loaded on FCCOB is 16’hFFFF, the value of registers FPROT and DFPROT will be restored to their original contents before executing the Protection Override command and the FPSTAT FPOVRD bit will be cleared. If the contents of the Protection Override Comparison Key in the Flash Configuration Field is left in the erased state (i.e. 16’hFFFF) the Protection Override feature is permanently disabled. If the command execution is flagged as an error (ACCERR being set for incorrect command launch) the values of FPROT and DFPROT will not be modified.

The Protection Override command can be called multiple times and every time it is launched it will preserve the current values of registers FPROT and DFPROT in a single-entry buffer to be restored later; when the Protection Override command is launched to restore FPROT and DFPROT these registers will assume the values they had before executing the Protection Override command on the last time. If contents of FPROT and/or DFPROT registers were modified by direct register writes while protection is overridden these modifications will be lost. Running Protection Override command to restore the contents of registers FPROT and DFPROT will not force them to the reset values.

Table 22-69. Protection Override Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != (001, 010 or 011) at command launch.
		Set if command not available in current mode (see Table 22-28).
		Set if protection is supposed to be restored (if key does not match or is invalid) and Protection Override command was not run previously (bit FPSTAT FPOVRD is 0), so there are no previous valid values of FPROT and DFPROT to be re-loaded.
		Set if Protection Update Selection[1:0] = 00 (in case of CCOBIX[2:0] = 010 or 011)
		Set if Protection Update Selection[1:0] = 00, CCOBIX[2:0] = 001 and a valid comparison key is loaded as a command parameter.
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

22.4.8 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an ECC fault.

Table 22-70. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Command Complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit
ECC Single Bit Fault on Flash Read	SFDIF (FERSTAT register)	SFDIE (FERCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

22.4.8.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the SFDIF flag in combination with the SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 22.3.2.5, “Flash Configuration Register (FCNFG)”, Section 22.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 22.3.2.7, “Flash Status Register (FSTAT)”, and Section 22.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 22-31.

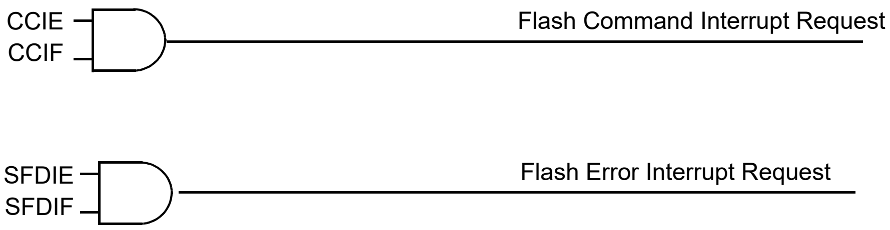


Figure 22-31. Flash Module Interrupts Implementation

22.4.9 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 22.4.8, “Interrupts”).

22.4.10 Stop Mode

If a Flash command is active (CCIF = 0) when the MCU requests stop mode, the current Flash operation will be completed before the MCU is allowed to enter stop mode.

22.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 22-10). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0xFF_FE0F. The security state out of reset can be permanently changed by programming the security byte assuming that the MCU is starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM

- .Mode and Security Effects on Flash Command Availability

22.5.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0xFF_FE00-0xFF_FE07). If the KEYEN[1:0] bits are in the enabled state (see <st-blue>Section 22.3.2.2 Flash Security Register (FSEC)), the Verify Backdoor Access Key command (see <st-blue>Section 22.4.7.11 Verify Backdoor Access Key Command) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 22-10) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash memory and EEPROM memory will not be available for read access and will return invalid data.

The user code stored in the P-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[1:0] bits are in the enabled state (see <st-blue>Section 22.3.2.2 Flash Security Register (FSEC)), 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 <st-blue>Section 22.4.7.11 Verify Backdoor Access Key Command
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit 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. The security as defined in the Flash security byte (0xFF_FE0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0xFF_FE00-0xFF_FE07 are unaffected by the Verify Backdoor Access Key command sequence. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed to the unsecure state, if desired. In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0xFF_FE00-0xFF_FE07 in the Flash configuration field.

22.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

A secured MCU can be unsecured in special single chip mode using an automated procedure described in Section 22.4.7.7.1, "Erase All Pin", For a complete description about how to activate that procedure please look into the Reference Manual.

22.5.3 .Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 22-28.

22.6 Initialization

On each system reset the flash module executes an initialization sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and DFPROT protection registers, and the FOPT and FSEC registers. The initialization routine reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set.

CCIF is cleared throughout the initialization sequence. The Flash module holds off all CPU access for a portion of the initialization sequence. Flash reads are allowed once the hold is removed. Completion of the initialization sequence is marked by setting CCIF high which enables user commands.

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

Appendix A
MCU Electrical Specifications

Revision History

Rev. No. (Item No.)	Date (Submitted By)	Substantial Change(s)
Rev 1.0	19-January-2015	• Initial Release for Publication on nxp.com
Rev 1.1	26-March-2018	• Changed footnote 4 of Appendix Table E-1., “Voltage Regulator Electrical Characteristics (Junction Temperature From –40°C To +175°C) • All S12ZVC material with 2018 Datecode for work week 23 (1823) and beyond have a fully trimmed ACLK. − Shipping label marking is 1823 and beyond − Component top side marking is week 23 and beyond • See Appendix , “Example Top Side Marking
Rev 1.2	19-March-2019	• Corrected Item 5 bus frequency in Table A-5 • Corrected Note 1 in Table A-5

A.1 General

This supplement contains the most accurate electrical information for the MC9S12ZVC-Family available at the time of publication.

A.1.1 Power Pins

Table A-1. Power Supplies

Mnemonic	Nominal Voltage	Description
VSS	0 V	Ground pin for 1.8V core supply voltage generated by on chip voltage regulator
VDDX1/[1]	5.0 V	5V power supply output for I/O drivers generated by on chip voltage regulator
VSSX1	0 V	Ground pin for I/O drivers
VDDX2	5.0 V	5V power supply output for I/O drivers generated by on chip voltage regulator
VSSX2	0 V	Ground pin for I/O drivers
VDDA	5.0 V	5V Power supply for the analog-to-digital converter and for the reference circuit of the internal voltage regulator
VSSA	0 V	Ground pin for VDDA analog supply
VSUP	12 V/18 V	External power supply for voltage regulator
VDDC	5 V	Power supply for CANPHY
VSSC	0 V	Ground pin for CANPHY

[1] All VDDX pins are internally connected by metal

VDDA is connected to VDDX1 and VDDX2 pins by diodes for ESD protection such that VDDX must not exceed VDDA by more than a diode voltage drop. VSSA and VSSX1 and VSSX2 are connected by anti-parallel diodes for ESD protection.

A.1.2 Pins

There are 4 groups of functional pins.

A.1.2.1 General Purpose I/O Pins (GPIO)

The I/O pins have a level in the VDDX/VDDA range. This class of pins is comprised of all port I/O pins, BKGD and the $\overline{\text{RESET}}$ pins.

A.1.2.2 High Voltage Pins

These consist of the VSUP, BCTL, BCTLC, CANH, CANL, SPLIT and PL[1:0] pins. These pins are intended to interface to external components operating in the automotive battery range. They have nominal voltages above the standard 5V I/O voltage range.

A.1.2.3 Oscillator

If the designated EXTAL and XTAL pins are configured for external oscillator operation then these pins have a nominal voltage of 1.8 V.

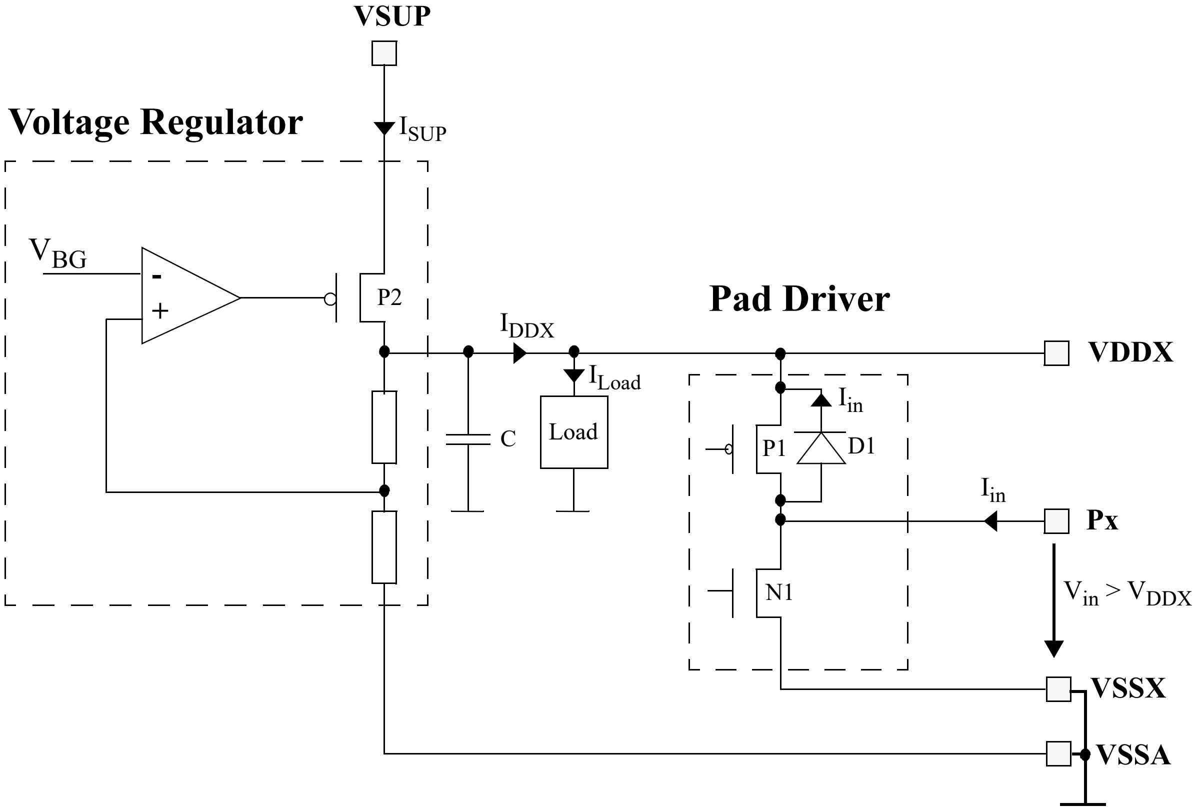
A.1.2.4 TEST

This pin is used for production testing only. The TEST pin must be tied to ground in all applications.

A.1.3 Current Injection

Power supply must maintain regulation within operating V_{DDX} or V_{DD} range during instantaneous and operating maximum current conditions. **Figure A-1.** shows a 5 V GPIO pad driver and the on chip voltage regulator with VDDX output. It shows also the power and ground pins VSUP, VDDX, VSSX and VSSA. Px represents any 5 V GPIO pin. Assume Px is configured as an input. The pad driver transistors P1 and N1 are switched off (high impedance). If the voltage V_{in} on Px is greater than V_{DDX} a positive injection current I_{in} will flow through diode D1 into VDDX node. If this injection current I_{in} is greater than I_{Load}, the internal power supply VDDX may go out of regulation. Ensure the external V_{DDX} load will shunt current greater than maximum injection current. This is the greatest risk when the MCU is not consuming power; e.g., if no system clock is present, or if the clock rate is very low which would reduce overall power consumption.

Figure A-1. Current Injection on GPIO Port if $V_{in} > V_{DDX}$



A.1.4 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only. A functional operation outside these ranges is not guaranteed. Stress beyond these limits may affect the reliability or cause permanent damage of the device.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level.

Table A-2. Absolute Maximum Ratings

Num	Rating	Symbol	Min	Max	Unit
1	Voltage regulator supply voltage	V_{SUP}	-0.3	42	V
2	High voltage inputs PL[1:0] input voltage	V_{Lx}	-27	42	V
3	Voltage Regulator Ballast Connection	V_{BCTL}	-0.3	42	V
4	Supplies VDDA, VDDC, VDDX	V_{VDDACX}	-0.3	6	V
5	Base connection of bipolar for CANPHY supply	V_{BCTLC}	-0.3	42	V
6	Voltage difference V_{DDX} to V_{DDA}[1]	Δ_{VDDX}	–0.1	0.1	V
7	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.3	0.3	V
8	Digital I/O input voltage	V_{IN}	–0.3	6.0	V
9	EXTAL, XTAL [2]	V_{ILV}	–0.3	2.16	V
10	TEST input	V_{TEST}	–0.3	10.0	V
11	Instantaneous current. Single pin limit for all digital I/O pins[3]	I_D	–25	+25	mA
12	Instantaneous maximum current on PP0,PP4,PP5 & PP6	I_{PP0456}	-30	+80	mA
13	Instantaneous maximum current on PP2	I_{PP2}	-80	+25	mA
14	Instantaneous maximum current. Single pin limit for EXTAL, XTAL	I_{DL}	–25	+25	mA
15	Storage temperature range	T_{stg}	–65	155	°C

[1] VDDX and VDDA must be shorted

[2] EXTAL, XTAL pins configured for external oscillator operation only

[3] All digital I/O pins are internally clamped to V_{SSX} and V_{DDX}, or V_{SSA} and V_{DDA}.

A.1.5 ESD Protection and Latch-up Immunity

All ESD testing is in conformity with CDF-AEC-Q100 stress test qualification for automotive grade integrated circuits. During the device qualification ESD stresses were performed for the Human Body Model (HBM) and the Charged-Device Model.

A device will be defined as a failure if after exposure to ESD pulses the device no longer meets the device specification. Complete DC parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-3. ESD and Latch-up Test Conditions

Model	Spec	Description	Symbol	Value	Unit
Human Body	JESD22-A114	Series Resistance	R	1500	Ω
		Storage Capacitance	C	100	pF
		Number of Pulse per pin positive negative	-	- 1 1	
Charged-Device	JESD22-C101	Series Resistance	R	0	Ω
		Storage Capacitance	C	4	pF
Latch-up for 5V GPIOs		Minimum Input Voltage Limit		-2.5	V
		Maximum Input Voltage Limit		+7.5	V
Latch-up for BCTL/BCTLC/CANH/CANL/SPLIT		Minimum Input Voltage Limit		-7	V
		Maximum Input Voltage Limit		+21	V

Table A-4. ESD Protection and Latch-up Characteristics

Num	Rating	Symbol	Min	Max	Unit
1	Human Body Model (HBM): -all pins except CANH/CANL/SPLIT/HVI -CANH/CANL/SPLIT/HVI	V_{HBM}	+/-2 +/-4	-	KV
2	Charged-Device Model (CDM): Corner Pins	V_{CDM}	+/-750	-	V
3	Charged-Device Model (CDM): all other pins	V_{CDM}	+/-500	-	V
4	Direct Contact Discharge IEC61000-4-2 with and without 220nF capacitor (R=330, C=150pF) CANL and CANH	V_{ESDIEC}	+/-6		KV
5	Latch-up Current of 5V GPIOs at T=125°C positive negative	I_{LAT}	+100 -100	-	mA
6	Latch-up Current of 5V GPIOs at 27°C positive negative	I_{LAT}	+200 -200	-	mA

A.1.6 Operating Conditions

This section describes the operating conditions of the device. Unless otherwise noted these conditions apply to the following electrical parameters.

NOTE

Please refer to the temperature rating of the device with regards to the ambient temperature T_A and the junction temperature T_J. For power dissipation calculations refer to Section A.1.7, “Power Dissipation and Thermal Characteristics”.

Table A-5. Operating Conditions

Num	Rating	Symbol	Min	Typ	Max	Unit
1	Voltage regulator supply voltage[1]	V_{SUP}	3.5	12	40	V
2	Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	–0.1	—	0.1	V
3	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.3	—	0.3	V
4	Oscillator	f_{osc}	4	—	20	MHz
5	Bus frequency[2] T_J < 150°C 150°C < T_J< 175°C (Temp option W only)	f_{bus}	[4]	— —	32 25	MHz
6	Bus frequency without wait states	f_{WSTAT}	—	—	25	MHz
7a	Operating junction temperature range Operating ambient temperature range[3] (option C)	T_J T_A	–40 –40	— —	105 85	°C
7b	Operating junction temperature range Operating ambient temperature range[3] (option V)	T_J T_A	–40 –40	— —	125 105	°C
7c	Operating junction temperature range Operating ambient temperature range[3] (option M)	T_J T_A	–40 –40	— —	150 125	°C
7d	Operating junction temperature range Operating ambient temperature range[3] (option W)	T_J T_A	–40 –40	— —	175 150	°C

[1] Normal operating range is 5.5 V - 18 V. Continuous operation at 40 V is not allowed. Only Transient Conditions (Load Dump) single pulse t_{max}<400 ms. Operation down to 3.5V is guaranteed without reset, however some electrical parameters are specified only in the range above 4.5 V. Operation up to 28.5V is limited to 1 hour over lifetime of the device. In this range the device continues to function but electrical parameters are degraded.

[2] The flash program and erase operations must configure f_{NVMOP} as specified in the NVM electrical section.

[3] Please refer to Section A.1.7, “Power Dissipation and Thermal Characteristics” for more details about the relation between ambient temperature T_A and device junction temperature T_J.

[4] Refer to f_{ATDCLK} for minimum ADC operating frequency. This is derived from the bus clock.

NOTE

Operation is guaranteed when powering down until low voltage reset assertion.

A.1.7 Power Dissipation and Thermal Characteristics

Power dissipation and thermal characteristics are closely related. The user must assure that the maximum operating junction temperature is not exceeded. The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \Theta_{JA})$$

T_J = Junction Temperature, [°C]

T_A = Ambient Temperature, [°C]

P_D = Total Chip Power Dissipation, [W]

Θ_{JA} = Package Thermal Resistance, [°C/W]

The total power dissipation P_D can be calculated from the equation below. Table A-6 below lists the power dissipation components. Table A-6 gives an overview of the supply currents.

$$P_D = P_{VSUP} + P_{BCTL} + P_{INT} - P_{GPIO} + P_{CANPHY}$$

Table A-6. Power Dissipation Components

Power Component	Description
$P_{VSUP} = V_{SUP}\ I_{SUP}$	Internal Power through VSUP pin
$P_{BCTL} = V_{BCTL}\ I_{BCTL}$	Internal Power through BCTL pin
$P_{INT} = V_{DDX}\ I_{VDDX} + V_{DDA}\ I_{VDDA}$	Internal Power through VDDX/A pins.
$P_{GPIO} = V_{I/O}\ I_{I/O}$	Power dissipation of external load driven by GPIO Port. Assuming the load is connected between GPIO and ground. This power component is included in P_{INT} and is subtracted from overall MCU power dissipation P_D
$P_{CANPHY} = [V_{DDC} -(V_{CANH} - V_{CANL})]\ I_{VDDC}$	Power dissipation of CANPHY

Figure A-2. Supply Currents Overview

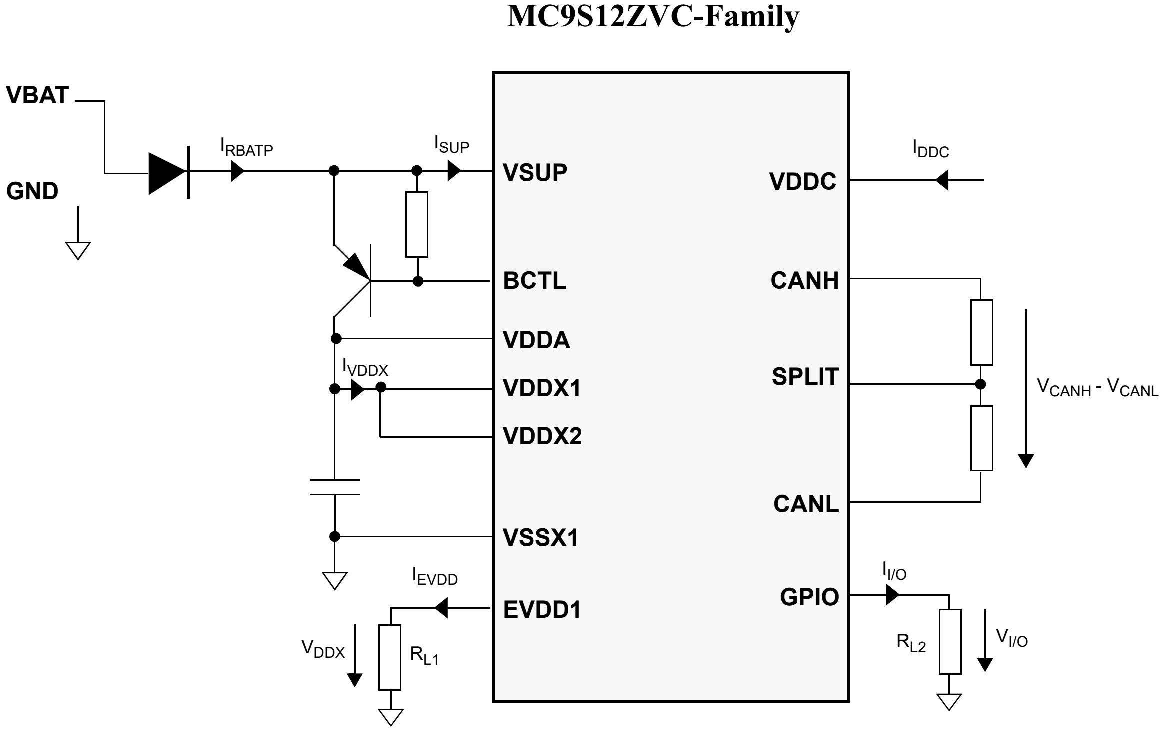


Table A-7. Thermal Package Characteristics

Num	Rating	Symbol	Min	Typ	Max	Unit
	64 LQFP-EP					
1	Thermal resistance 64LQFP-EP, single sided PCB[1] Natural Convection	θ_{JA}	—	64	—	°C/W
2	Thermal resistance 64LQFP-EP, double sided PCB[2] with 2 internal planes. Natural Convection.	θ_{JA}	—	30	—	°C/W
3	Thermal resistance 64LQFP-EP, single sided PCB[2] (@200 ft./min)	θ_{JA}	—	51	—	°C/W
4	Thermal resistance 64LQFP-EP, double sided PCB[2] with 2 internal planes (@200 ft./min).	θ_{JA}	—	24	—	°C/W
5	Junction to Board 64LQFP-EP[3]	θ_{JB}	—	13	—	°C/W
6	Junction to Case Top 64LQFP-EP[4]	θ_{JCtop}	—	16	—	°C/W
7	Junction to Case Bottom 64LQFP-EP[5]	$\theta_{JCbottom}$	—	1.6	—	°C/W
8	Junction to Package Top 64LQFP-EP[6]	Ψ_{JT}	—	4	—	°C/W
	48 LQFP					
9	Thermal resistance 48LQFP, single sided PCB[1] Natural Convection	θ_{JA}	—	70	—	°C/W
10	Thermal resistance 48LQFP, double sided PCB[2] with 2 internal planes. Natural Convection.	θ_{JA}	—	46	—	°C/W
11	Thermal resistance 48LQFP, single sided PCB[2] (@200 ft./min)	θ_{JA}	—	58	—	°C/W
12	Thermal resistance 48LQFP, double sided PCB[2] with 2 internal planes (@200 ft./min).	θ_{JA}	—	40	—	°C/W
13	Junction to Board 48LQFP[3]	θ_{JB}	—	23	—	°C/W
14	Junction to Case Top 48LQFP[4]	θ_{JCtop}	—	16	—	°C/W
15	Junction to Package Top 48LQFP[6]	Ψ_{JT}	—	2	—	°C/W

[1] Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to JEDEC JESD51-2 with the single layer board (JESD51-3) horizontal.

[2] Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to the JEDEC specification JESD51-6 with the board (JESD51-7) horizontal.

[3] Thermal resistance between the die and the printed circuit board per JEDEC JESD51-8. Board temperature is measured on the top surface of the board near the package.

[4] Thermal resistance between the die and the case top surface as measured by the cold plate method (MIL SPEC-883 Method 1012.1).

[5] Thermal resistance between the die and the solder pad on the bottom of the package based on simulation without any interface resistance

[6] Thermal characterization parameter indicating the temperature difference between package top and the junction temperature per JEDEC JESD51-2

A.1.8 I/O Characteristics

This section describes the characteristics of I/O pins.

Table A-8. 5V I/O Characteristics (Junction Temperature From –40°C To +175°C)

Conditions are 4.5 V < V_{DDX}< 5.5 V , unless otherwise noted. I/O Characteristics for all GPIO pins (defined in A.1.2.1/A-698).						
Num	**Rating /Classification**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Input high voltage	V_{IH}	0.65*V_{DDX}	—	—	V
2	Input high voltage	V_{IH}	—	—	V_{DDX}+0.3	V
3	Input low voltage	V_{IL}	—	—	0.35*V_{DDX}	V
4	Input low voltage	V_{IL}	V_{SSX}–0.3	—	—	V
5	Input hysteresis	V_{HYS}	—	250	—	mV
6	Input leakage current (All GPIO except PAD3,PP0,PP2,PP4,PP5 and PP6, Pins in high impedance input mode)[1] V_{in} = V_{DDX} or V_{SSX}	I_{in}	-1	—	1	μA
7	Input leakage current PAD3,PP0,PP2,PP4,PP5 and PP6, Pins in high impedance input mode)[2] V_{in} = V_{DDX} or V_{SSX}	I_{in}	-2.5	—	2.5	μA

Table A-8. 5V I/O Characteristics (Junction Temperature From –40°C To +175°C)

Conditions are 4.5 V < V_{DDX}< 5.5 V , unless otherwise noted. I/O Characteristics for all GPIO pins (defined in A.1.2.1/A-698).						
8	Output high voltage (All GPIO except PP0,PP2,PP4,PP5 and PP6) I_{OH} = –4 mA	V_{OH}	V_{DDX} – 0.8	—	—	V
9	Output low voltage (All GPIO except PP0, PP2, PP4, PP5 and PP6) I_{OL} = +4mA	V_{OL}	—	—	0.8	V
I/O Characteristic PP2						
10	Output high voltage Partial Drive I_{OH} = –2mA IOH = -15mA IOH = -10mA	V_{OH}	V_{DDX} – 0.8 V_{DDX} – 0.2 V_{DDX} – 0.1	— — —	— — —	V
11	Output low voltage Partial drive I_{OL} = +2mA Full drive I_{OL} = +20mA	V_{OL}	—	—	0.8	V
12	Input leakage current (pin in high impedance input mode)[3] V_{in} = V_{DDX} or V_{SSX}	I_{in}	-2.5	—	2.5	µA
13	Over-current Detect Threshold	I_{OCD}	-80	—	-40	mA
14	Maximum allowed continous current	I_{PP}	-20	—	10	mA
I/O Characteristic PP0, PP4, PP5, PP6						
15	Output high voltage Partial Drive I_{OH} = –2mA Full Drive IOH = –18mA	V_{OH}	V_{DDX} –0.8	—	—	V
16	Output low voltage Partial drive I_{OL} = +2mA Full drive I_{OL} = +20mA	V_{OL}	— —	— —	0.8 0.25	V
17	Input leakage current (pin in high impedance input mode) V_{in} = V_{DDX} or V_{SSX}	I_{in}	-2.5	—	2.5	µA
18	Over-current Detect Threshold	I_{OCD}	40	—	80	mA
19	Maximum allowed continous current	I_{PP}	-10	—	25	mA
20	Internal pull up current (All GPIO except RESET) V_{IH} min > input voltage > V_{IL} max	I_{PUL}	-10	—	-130	µA
21	Internal pull up resistance (RESET pin)	R_{PUL}	2.5	5	10	KΩ
22	Internal pull down current V_{IH} min > input voltage > V_{IL} max	I_{PDH}	10	—	130	µA
23	Input capacitance	C_{in}	—	7	—	pF
24	Injection current[4] Single pin limit Total device limit, sum of all injected currents	I_{ICS} I_{ICP}	–2.5 –25	—	2.5 25	mA

---

[1] Maximum leakage current occurs at maximum operating temperature. Current decreases by approximately one-half for each 8°°C to 12°°C in the temperature range from 50°C to 125°C.

[2] Maximum leakage current occurs at maximum operating temperature. Current decreases by approximately one-half for each 8°°C to 12°°C in the temperature range from 50°C to 125°C.

[3] Maximum leakage current occurs at maximum operating temperature. Current decreases by approximately one-half for each 8°°C to 12°°C in the temperature range from 50°C to 125°C.

[4] For sake of ADC conversion accuracy, the application should avoid to inject any current into pins PAD3/VREFH. Refer to Section A.1.3, “Current Injection” for more details.

This following tables describe the timing characteristics of I/O pins.

Table A-9. Pin Timing Characteristics (Junction Temperature From –40°C To +175°C)

Num	Rating	Symbol	Min	Typ	Max	Unit
Conditions are 4.5 V < V_{DDX}< 5.5 V unless otherwise noted. I/O Characteristics for all GPIO pins (defined in A.1.2.1/A-698).						
1	Port P, S, AD, L interrupt input pulse filtered [1]	t_{P_MASK}	—	—	3	µs
2	Port P, S, AD, L interrupt input pulse passed [1]	t_{P_PASS}	10	—	—	µs
3	Port P, S, AD, L interrupt input pulse filtered in number of bus clock cycles of period $1/f_{bus}$	n_{P_MASK}	—	—	3	
4	Port P, S, AD, L interrupt input pulse passed in number of bus clock cycles of period $1/f_{bus}$	n_{P_PASS}	4	—	—	
5	$\overline{\text{IRQ}}$ pulse width, edge-sensitive mode in number of bus clock cycles of period $1/f_{bus}$	n_{IRQ}	1	—	—	
6	$\overline{\text{RESET}}$ pin input pulse filtered	R_{P_MASK}	—	—	12	ns
7	$\overline{\text{RESET}}$ pin input pulse passed	R_{P_PASS}	22	—	—	ns

[1] Parameter only applies in stop or pseudo stop mode.

A.1.9 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

A.1.9.1 Measurement Conditions

Current is measured on VSUP. VDDX is connected to VDDA. It does not include the current to drive external loads. Unless otherwise noted the currents are measured in special single chip mode and the CPU code is executed from RAM. For Run and Wait current measurements PLL is on and the reference clock is the IRC1M trimmed to 1MHz. For the junction temperature range from -40°C to +150°C the bus frequency is 32MHz. Table A-10, Table A-11 and Table A-12 show the configuration of the CPMU module and the peripherals for Run, Wait and Stop current measurement.

Table A-10. CPMU Configuration for Pseudo Stop Current Measurement

CPMU REGISTER	Bit settings/Conditions
CPMUCLKS	PLLSEL=0, PSTP=1, CSAD=0, PRE=PCE=RTIOSCSEL=1 COPOSCSEL[1:0]=01
CPMUOSC	OSCE=1, Quartz oscillator f_{EXTAL}=4MHz
CPMURTI	RTDEC=0, RTR[6:4]=111, RTR[3:0]=1111
CPMUCOP	WCOP=1, CR[2:0]=111

Table A-11. CPMU Configuration for Run/Wait and Full Stop Current Measurement

CPMU REGISTER	Bit settings/Conditions
CPMUSYNR	VCOFRQ[1:0]= 1,SYNDIV[5:0] = 31
CPMUPOSTDIV	POSTDIV[4:0]=0
CPMUCLKS	PLLSEL=1, CSAD=0
CPMUOSC	OSCE=0, Reference clock for PLL is f_{ref}=f_{irc1m} trimmed to 1MHz
API settings for STOP current measurement	
CPMUAPICTL	APIEA=0, APIFE=1, APIE=0
CPMUACLKTR	trimmed to >=20Khz
CPMUAPIRH/RL	set to 0xFFFF

Table A-12. Peripheral Configurations for Run and Wait Current Measurement

Peripheral	Configuration
SCI	Continuously transmit data (0x55) at speed of 19200 baud
SPI	Configured to master mode, continuously transmit data (0x55) at 1Mbit/s
ACMP0 & 1	The module is enabled and the plus & minus inputs are toggling with 0-1 and 1-0 at 1MHz
DAC	The module is enabled in buffered mode at full voltage range
ADC	The peripheral is configured to operate at its maximum specified frequency and to continuously convert voltages on a single input channel
MSCAN	The module is connected to CANPHY and continously transmit data (0x55 or 0xAA) with a bit rate of 500kbit/s.
CANPHY	The module is enabled and connect to MSCAN module

Table A-12. Peripheral Configurations for Run and Wait Current Measurement

Peripheral	Configuration
IIC	Operate in master mode and contiously transmit data (0x55 or 0xAA) at the bit rate of 100Kbit/s
PWM0	The module is configured with a modulus rate of 10 kHz
PWM1	The module is configured with a modulus rate of 10 kHz
TIM0	Channel 7:4 are configured to output compare mode, channel 3:0 are configured to input capture mode channel 1:0 connected to SENTTX, channel 2 connected to ACLK and channel 3 connected to RXD0
TIM1	Channel 3:0 are configured to output compare mode
SENT	Continously transmit data 0 at maximum rate
COP & RTI	Enabled
BATS	Enabled

Table A-13. Run and Wait Current Characteristics

Conditions see Table A-11 and Table A-12, V_{SUP}=18 V						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Run Current, -40°C < T_J < 150°C, f_{bus}= 32MHz	I_{SUPR}	—	25	35	mA
2	Wait Current, -40°C < T_J < 150°C, f_{bus}= 32MHz	I_{SUPW}	—	18	25	mA

Table A-14. Stop Current Characteristics

Conditions are: V_{SUP}=18 V						
Num	**Rating**[1]	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
Stop Current all modules off						
1	$T_A = T_J$= -40°C	I_{SUPS}	—	16	30	µA
2	$T_A = T_J$= 150°C	I_{SUPS}	—	200	1600	µA
3	$T_A = T_J$= 175°C	I_{SUPS}	—	—	2200	µA
4	$T_A = T_J$ = 25°C	I_{SUPS}	—	18	—	µA
5	$T_A = T_J$ = 85°C	I_{SUPS}	—	41	120	µA
6	$T_A = T_J$ = 105°C	I_{SUPS}	—	73	210	µA
Stop Current API enabled						
7	$T_A = T_J$ = 25°C	I_{SUPS}	—	23	—	µA

[1] If MCU is in STOP long enough then TA = TJ . Die self heating due to stop current can be ignored.

Table A-15. Pseudo Stop Current Characteristics

Conditions are: V_{SUP}=12V, API, COP & RTI enabled						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	T_J = 25°C	I_{SUPPS}	—	270	—	µA

A.1.10 ADC Calibration Configuration

The reference voltage V_{BG} is measured under the conditions shown in Table A-16. The values stored in the IFR are the average of eight consecutive conversions at Tj=150 °C and eight consecutive conversions at Tj=-40 °C. The code is executed from RAM. The result is programmed to the IFR, otherwise there is no flash activity.

Table A-16. Measurement Conditions

Description	Symbol	Max	Unit
Regulator Supply Voltage at VSUP	V_{SUP}	5	V
Supply Voltage at VDDX and VDDA	$V_{DDX,A}$	5	V
ADC reference voltage high	V_{RH}	5	V
ADC reference voltage low	V_{RL}	0	V
ADC clock	f_{ATDCLK}	2	MHz
ADC sample time	t_{SMP}	4	ADC clock cycles
Bus clock frequency	f_{bus}	48	MHz
Junction temperature	T_j	-40 and 150	°C

Appendix B
ADC Electrical Specifications

This section describes the characteristics of the analog-to-digital converter.

B.1 ADC Operating Characteristics

The Table B-1 shows conditions under which the ADC operates.

The following constraints exist to obtain full-scale, full range results:

$V_{SSA} \leq V_{RL} \leq V_{IN} \leq V_{RH} \leq V_{DDA}$.

This constraint exists since the sample buffer amplifier can not drive beyond the power supply levels that it ties to. If the input level goes outside of this range it will effectively be clipped.

Table B-1. ADC Operating Characteristics

Supply voltage 4.5 V < V_{DDA} < 5.5 V, Junction Temperature From –40°°C To +175°C						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Reference potential Low High	 V_{RL} V_{RH}	 V_{SSA} $V_{DDA}/2$	 — —	 $V_{DDA}/2$ V_{DDA}	 V V
2	Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	-0.1	0	0.1	V
3	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.1	0	0.1	V
4	Differential reference voltage[1]	V_{RH}-V_{RL}	3.13	5.0	5.5	V
5a	ADC Clock Frequency (derived from bus clock via the prescaler). Junction temperature from –40°°C to +150°C	f_{ATDCLK}	0.25	—	8.33	MHz
5b	ADC Clock Frequency (derived from bus clock via the prescaler). Junction temperature from 150°°C to +175°C	f_{ATDCLK}	0.25	—	6.67	MHz
6	Buffer amplifier turn on time (delay after module start/recovery from Stop mode)	t_{REC}	—	—	1	µs
7	ADC disable time	$t_{DISABLE}$	—	—	3	bus clock cycles
8	ADC Conversion Period [2] 12 bit resolution: 10 bit resolution: 8 bit resolution:	 N_{CONV12} N_{CONV10} N_{CONV8}	 19 18 16	 — — —	 39 38 36	ADC clock cycles

[1] Full accuracy is not guaranteed when differential voltage is less than 4.50 V

[2] The minimum time assumes a sample time of 4 ATD clock cycles. The maximum time assumes a sample time of 24 ATD clock

cycles.

B.1.1 Factors Influencing Accuracy

Source resistance, source capacitance and current injection have an influence on the accuracy of the ADC .**Figure B-1.** A further factor is that PortAD pins that are configured as output drivers switching.

B.1.1.1 Port AD Output Drivers Switching

PortAD output drivers switching can adversely affect the ADC accuracy whilst converting the analog voltage on other PortAD pins because the output drivers are supplied from the VDDA/VSSA ADC supply pins. Although internal design measures are implemented to minimize the effect of output driver noise, it is recommended to configure PortAD pins as outputs only for low frequency, low load outputs. The impact on ADC accuracy is load dependent and not specified. The values specified are valid under condition that no PortAD output drivers switch during conversion.

B.1.1.2 Source Resistance

Due to the input pin leakage current as specified in conjunction with the source resistance there will be a voltage drop from the signal source to the ADC input. The maximum source resistance R_S specifies results in an error (10-bit resolution) of less than 1/2 LSB (2.5 mV) at the maximum leakage current. If device or operating conditions are less than worst case or leakage induced error is acceptable, larger values of source resistance of up to 10Kohm are allowed.

B.1.1.3 Source Capacitance

When sampling an additional internal capacitor is switched to the input. This can cause a voltage drop due to charge sharing with the external capacitance and the pin capacitance. For a maximum sampling error of the input voltage ≤ 1LSB (10-bit resolution), then the external filter capacitor, $C_f \geq 1024 * (C_{INS} – C_{INN})$.

B.1.1.4 Current Injection

There are two cases to consider.

1. A current is injected into the channel being converted. The channel being stressed has conversion values of 0x3FF (in 10-bit mode) for analog inputs greater than V_{RH} and 0x000 for values less than V_{RL} unless the current is higher than specified as a disruptive condition.
2. Current is injected into pins in the neighborhood of the channel being converted. A portion of this current is picked up by the channel (coupling ratio K), This additional current impacts the accuracy of the conversion depending on the source resistance.

   The additional input voltage error on the converted channel can be calculated as:

   $$V_{ERR} = K * R_S * I_{INJ}$$

   with I_{INJ} being the sum of the currents injected into the two pins adjacent to the converted channel.

Table B-2. ADC Electrical Characteristics (Junction Temperature From –40°C To +175°C)

Supply voltage 3.13 V < V_{DDA} < 5.5 V						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Max input source resistance	R_S	—	—	1	KΩ
2	Total input capacitance Non sampling Total input capacitance Sampling	C_{INN} C_{INS}	— —	— —	10 16	pF
3	Input internal Resistance	R_{INA}	-	5	15	kΩ
4	Disruptive analog input current	I_{NA}	-2.5	—	2.5	mA
5	Coupling ratio positive current injection	K_p	—	—	1E-4	A/A
6	Coupling ratio negative current injection	K_n	—	—	5E-3	A/A

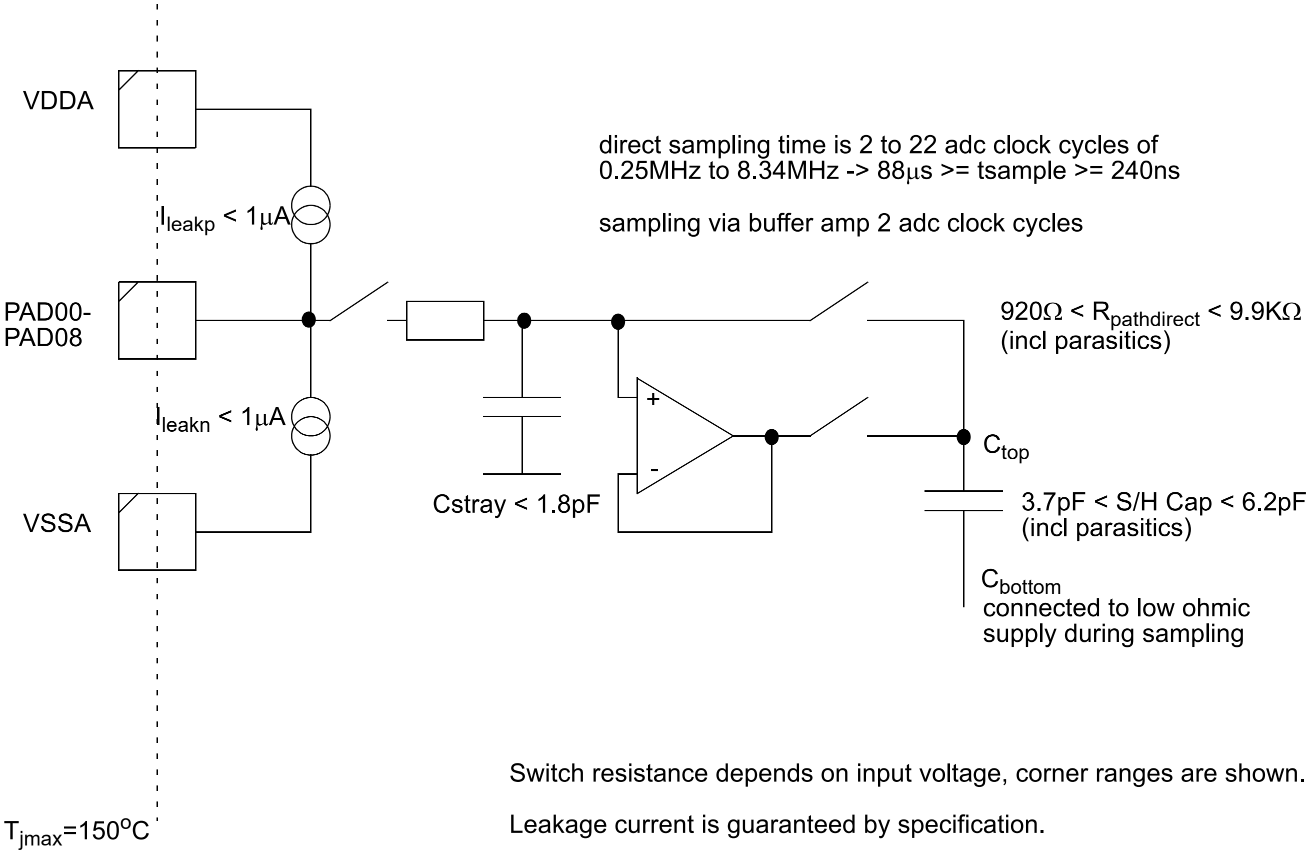


Figure B-1.

B.1.2 ADC Accuracy

Table B-3. specifies the ADC conversion performance excluding any errors due to current injection, input capacitance and source resistance.

B.1.2.1 ADC Accuracy Definitions

For the following definitions see also **Figure B-2.**.
Differential non-linearity (DNL) is defined as the difference between two adjacent switching steps.

$$\mathrm{DNL}(i) = \frac{V_i - V_{i-1}}{1\mathrm{LSB}} - 1$$

The integral non-linearity (INL) is defined as the sum of all DNLs:

$$\mathrm{INL}(n) = \sum_{i=1}^{n} \mathrm{DNL}(i) = \frac{V_n - V_0}{1\mathrm{LSB}} - n$$

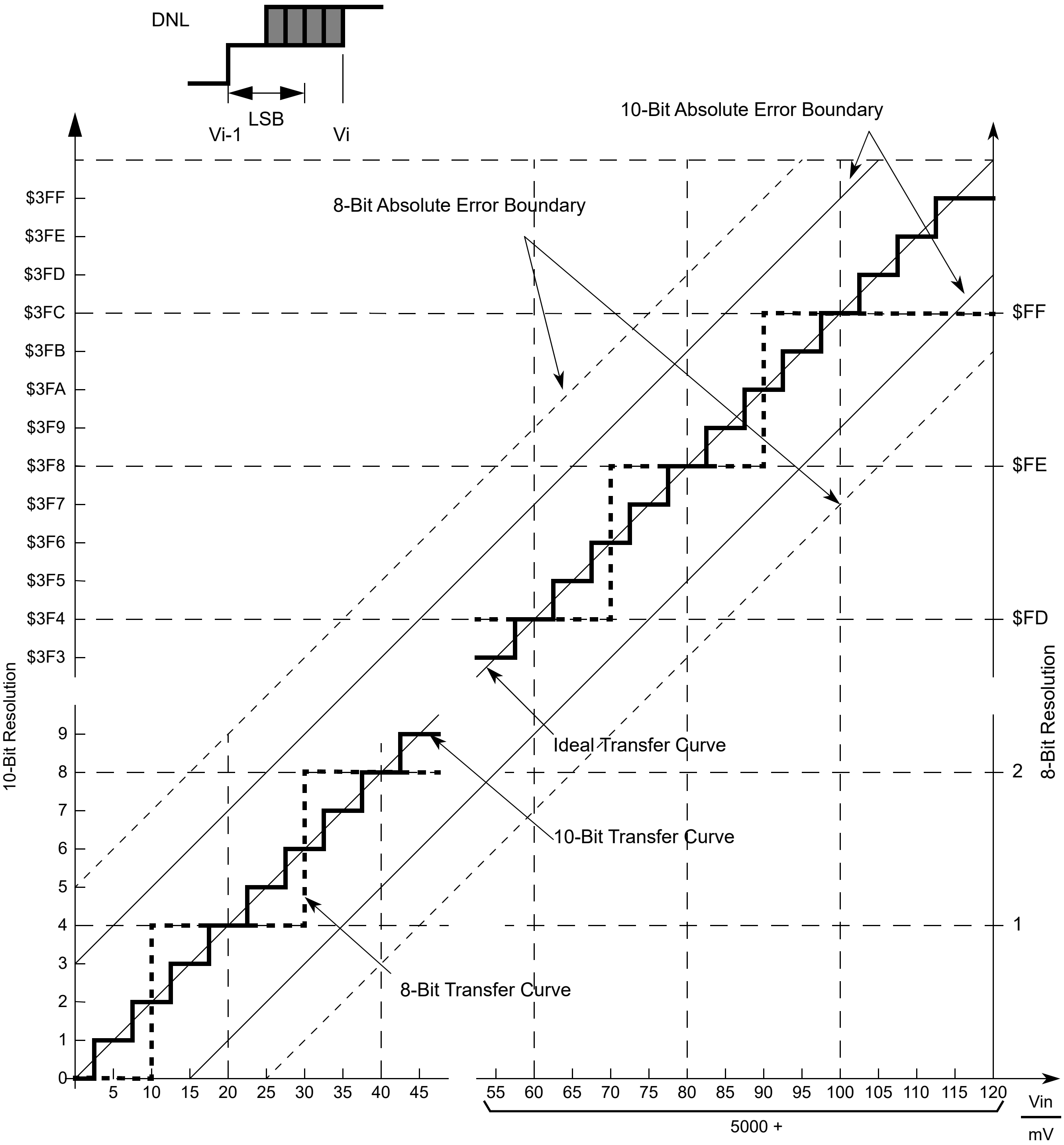


Figure B-2. ADC Accuracy Definitions

Table B-3. ADC Conversion Performance 5 V range (Junction Temperature From –40°C To +175°C)

Supply voltage 4.5 < V_{DDA} < 5.5V, 4.5V < V_{REF} < 5.5 V. f_{ADCCLK} = 8.0 MHz The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.							
Num	**Rating**[1]		**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Resolution (V_{REF}=5.12V)	12-Bit	LSB	—	1.25	—	mV
2	Differential Nonlinearity	12-Bit	DNL	-4	±2	4	counts
3	Integral Nonlinearity	12-Bit	INL	-5	±2.5	5	counts
4	Absolute Error[2]	12-Bit	AE	-7	±4	7	counts
5	Resolution (V_{REF}=5.12V)	10-Bit	LSB	—	5	—	mV
6	Differential Nonlinearity	10-Bit	DNL	-1	±0.5	1	counts
7	Integral Nonlinearity	10-Bit	INL	-2	±1	2	counts
8	Absolute Error<st-blue>2	10-Bit	AE	-3	±2	3	counts
9	Resolution (V_{REF}=5.12V)	8-Bit	LSB	—	20	—	mV
10	Differential Nonlinearity	8-Bit	DNL	-0.5	±0.3	0.5	counts
11	Integral Nonlinearity	8-Bit	INL	-1	±0.5	1	counts
12	Absolute Error<st-blue>2	8-Bit	AE	-1.5	±1	1.5	counts

[1] The 8-bit and 10-bit mode operation is structurally tested in production test. Absolute values are tested in 12-bit mode.

[2] These values include the quantization error which is inherently 1/2 count for any A/D converter.

Appendix C
MSCAN Electrical Specifications

Table C-1. MSCAN Wake-up Pulse Characteristics (Junction Temperature From –40°C To +175°C)

Conditions are 4.5 V < V_{DDX}< 5.5 V, unless otherwise noted.						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	MSCAN wake-up dominant pulse filtered	t_{WUP}	—	—	1.5	µs
2	MSCAN wake-up dominant pulse pass	t_{WUP}	5	—	—	µs

Appendix D
SPI Electrical Specifications

This section provides electrical parametrics and ratings for the SPI.

In **Figure D-1.** the measurement conditions are listed.

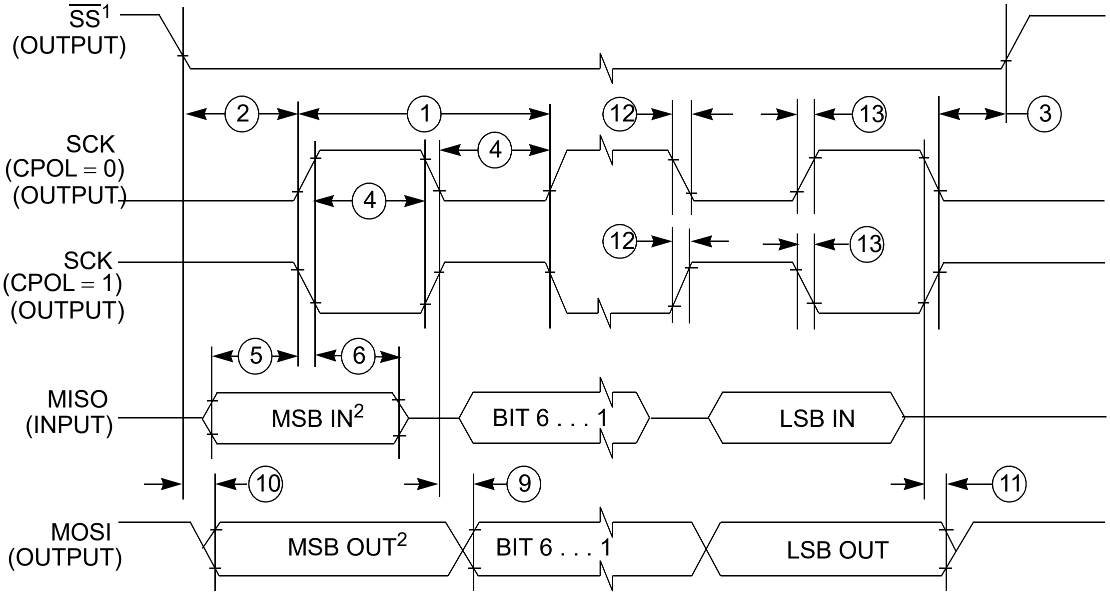
Figure D-1. Measurement Conditions

Description	Value	Unit
Drive mode	full drive mode	—
Load capacitance C_{LOAD}[1], on all outputs	50	pF
Thresholds for delay measurement points	(35% / 65%) VDDX	V

[1] Timing specified for equal load on all SPI output pins. Avoid asymmetric load.

D.0.1 Master Mode

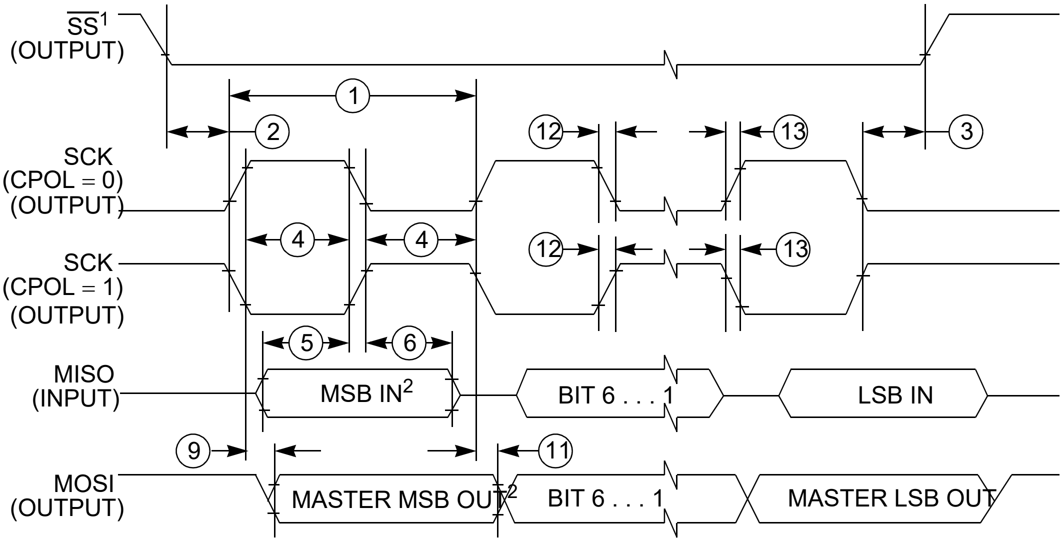
In **Figure D-2.** the timing diagram for master mode with transmission format CPHA=0 is depicted.



1. If enabled.
2. LSBFE = 0. For LSBFE = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure D-2. SPI Master Timing (CPHA=0)

In **Figure D-3.** the timing diagram for master mode with transmission format CPHA=1 is depicted.



1. If enabled.

2. LSBFE = 0. For LSBFE = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure D-3. SPI Master Timing (CPHA=1)

In **Table D-1.** the timing characteristics for master mode are listed.

Table D-1. SPI Master Mode Timing Characteristics (Junction Temperature From –40°C To +175°C)

Num	Characteristic	Symbol				Unit
			Min	Typ	Max	
1	SCK Frequency	f_{sck}	1/2048	—	1/2	f_{bus}
1	SCK Period	t_{sck}	2	—	2048	t_{bus}
2	Enable Lead Time	t_{lead}	—	1/2	—	t_{sck}
3	Enable Lag Time	t_{lag}	—	1/2	—	t_{sck}
4	Clock (SCK) High or Low Time	t_{wsck}	—	1/2	—	t_{sck}
5	Data Setup Time (Inputs)	t_{su}	8	—	—	ns
6	Data Hold Time (Inputs)	t_{hi}	8	—	—	ns
9	Data Valid after SCK Edge	t_{vsck}	—	—	15	ns
10	Data Valid after $\overline{SS}$ fall (CPHA=0)	t_{vss}	—	—	15	ns
11	Data Hold Time (Outputs)	t_{ho}	0	—	—	ns
12	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

D.0.2 Slave Mode

In **Figure D-4.** the timing diagram for slave mode with transmission format CPHA=0 is depicted.

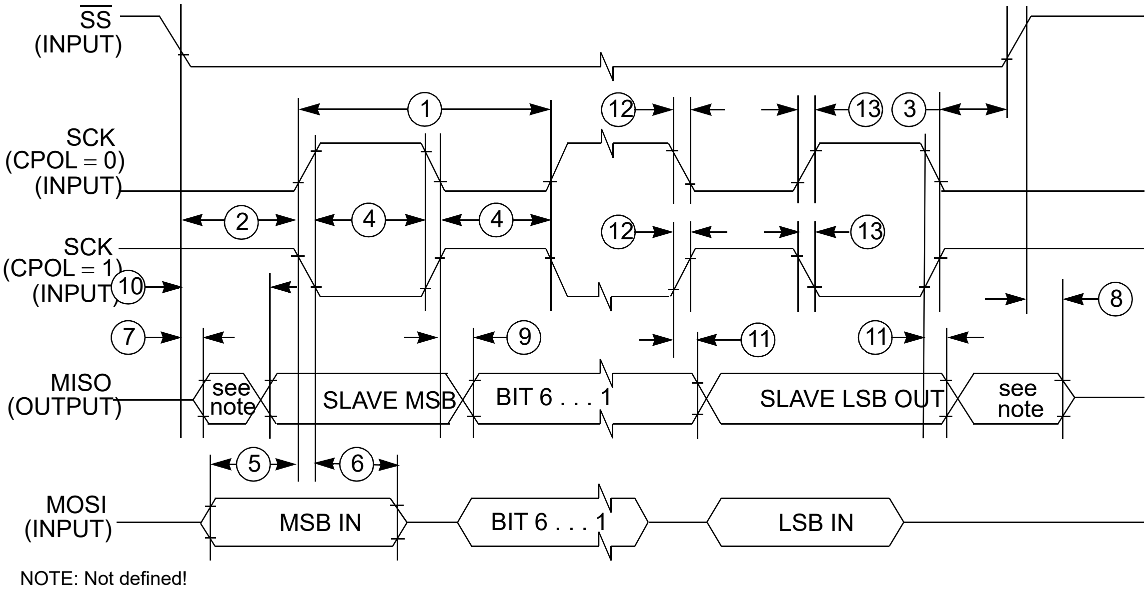


Figure D-4. SPI Slave Timing (CPHA=0)

In **Figure D-5.** the timing diagram for slave mode with transmission format CPHA=1 is depicted.

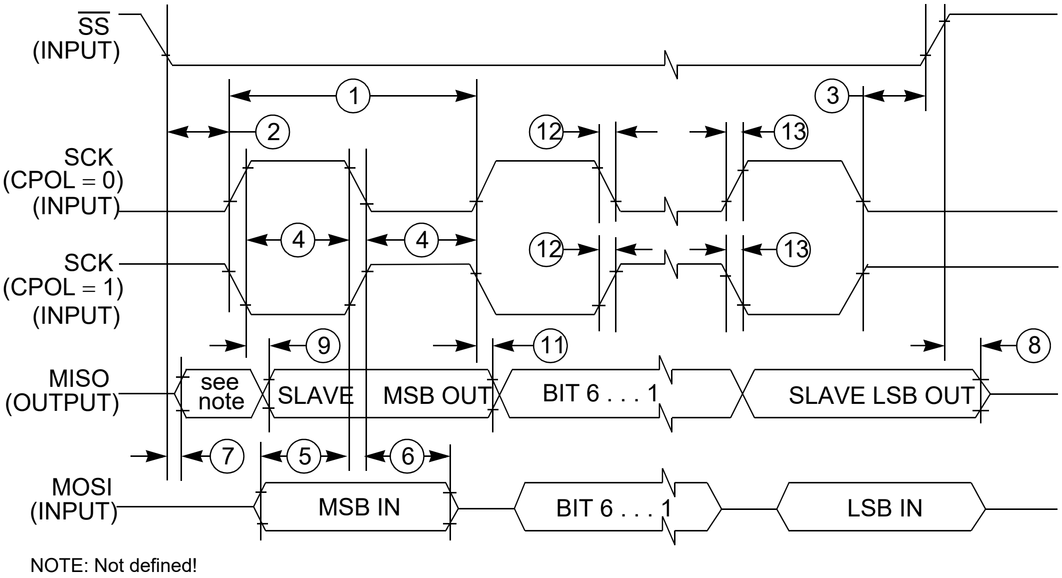


Figure D-5. SPI Slave Timing (CPHA=1)

In **Table D-2.** the timing characteristics for slave mode are listed.

Table D-2. SPI Slave Mode Timing Characteristics (Junction Temperature From –40°C To +175°C)

Num	Characteristic	Symbol	Min	Typ	Max	Unit
1	SCK Frequency	f_{sck}	DC	—	1/4	f_{bus}
1	SCK Period	t_{sck}	4	—	∞	t_{bus}
2	Enable Lead Time	t_{lead}	4	—	—	t_{bus}
3	Enable Lag Time	t_{lag}	4	—	—	t_{bus}
4	Clock (SCK) High or Low Time	t_{wsck}	4	—	—	t_{bus}
5	Data Setup Time (Inputs)	t_{su}	8	—	—	ns
6	Data Hold Time (Inputs)	t_{hi}	8	—	—	ns
7	Slave Access Time (time to data active)	t_a	—	—	20	ns
8	Slave MISO Disable Time	t_{dis}	—	—	22	ns
9	Data Valid after SCK Edge	t_{vsck}	—	—	30 + t_{bus} [1]	ns
10	Data Valid after $\overline{\text{SS}}$ fall	t_{vss}	—	—	30 + t_{bus} [1]	ns
11	Data Hold Time (Outputs)	t_{ho}	20	—	—	ns
12	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

[1] t_{bus} added due to internal synchronization delay

Appendix E
CPMU Electrical Specifications (VREG, OSC, IRC, PLL)

E.1 VREG Electrical Specifications

Table E-1. Voltage Regulator Electrical Characteristics (Junction Temperature From –40°C To +175°C)

VDDA and VDDX must be shorted on the application board.						
Num	**Characteristic**	**Symbol**	**Min**	**Typical**	**Max**	**Unit**
1	Input Voltages	V_{SUP}	3.5	—	40	V
2	Output Voltage VDDX (with external PNP)					
	Full Performance Mode V_{SUP} > =6V	V_{DDX}	4.85	5.0	5.15	V
	Full Performance Mode 5.5V <= V_{SUP} <=6V		4.50	5.0	5.15	V
	Full Performance Mode 3.5V <= V_{SUP} <=5.5V		3.13	—	5.15	V
	Reduced Performance Mode (stop mode)		2.5	5.5	5.75	V
	V_{SUP} > =3.5V					
3	Output Voltage VDDX (without external PNP)					
	Full Performance Mode V_{SUP} > =6V	V_{DDX}	4.80	4.95	5.10	V
	Full Performance Mode 5.5V <= V_{SUP} <=6V		4.50	4.95	5.10	V
	Full Performance Mode 3.5V <= V_{SUP} <=5.5V		3.13	—	5.10	V
	Reduced Performance Mode (stop mode)		2.5	5.5	5.70	V
	V_{SUP} > =3.5V					
4	Load Current VDDX[1] (without external PNP)					
	(-40°C < T_J < 150°C)	I_{DDX}				mA
	Full Performance Mode V_{SUP} > 6V		0	—	70	
	Full Performance Mode 3.5V <= V_{SUP} <=6V		0	—	25	
5	Load Current VDDX(1) (without external PNP)					
	Full Performance Mode V_{SUP} > 6V	I_{DDX}	0	—	55	mA
	Full Performance Mode 3.5V <= V_{SUP} <=6V		0	—	20	mA
	Reduced Performance Mode (stop mode)		0	—	5	mA
6	Short circuit VDDX fall back current V_{DDX}<=0.5V	I_{DDX}	—	100	—	mA
7	Output Voltage VDDC with external PNP					
	Full Performance Mode V_{SUP} > =6V	V_{DDC}	4.85	5.0	5.15	V
	Full Performance Mode 5.5V <= V_{SUP} <=6V		4.50	5.0	5.15	V
	Full Performance Mode 3.5V <= V_{SUP} <=5.5V		3.13	—	5.15	V
	Reduced Performance Mode (stop mode)		2.5	5.5	5.75	V
	V_{SUP} > =3.5V					
8	Load Current VDDC					
	Reduced Performance Mode (stop mode)	I_{DDC}	0	—	2.5	mA
9	Low Voltage Interrupt Assert Level[2]	V_{LVIA}	4.04	4.23	4.40	V
	Low Voltage Interrupt Deassert Level	V_{LVID}	4.19	4.38	4.49	V
10a	VDDX Low Voltage Reset deassert[3]	V_{LVRXD}	—	3.05	3.13	V
10b	VDDX Low Voltage Reset assert	V_{LVRXA}	2.95	3.02	—	V

Table E-1. Voltage Regulator Electrical Characteristics (Junction Temperature From –40°C To +175°C)

VDDA and VDDX must be shorted on the application board.						
Num	**Characteristic**	**Symbol**	**Min**	**Typical**	**Max**	**Unit**
11	Trimmed ACLK output frequency[4]	f_{ACLK}	—	20	—	KHz
12	Trimmed ACLK internal clock Δf / $f_{nominal}$ [4]	df_{ACLK}	- 6%	—	+ 6%	—
13	The first period after enabling the counter by APIFE might be reduced by API start up delay	t_{sdel}	—	—	100	µs
14	Temperature Sensor Slope	dV_{HT}	5.05	5.25	5.45	mV/°C
15	Temperature Sensor Output Voltage T_J=150°C untrimmed	V_{HT}	—	2.4	—	V
16	High Temperature Interrupt Assert[5] High Temperature Interrupt Deassert	T_{HTIA} T_{HTID}	120 110	132 122	144 134	°C °C
17	Bandgap output voltage	V_{BG}	1.14	1.20	1.28	V
18	Bandgap output voltage V_{SUP} dependency T_J =150°C, 3.5V < V_{SUP}<18V	Δ_{VBGV}	-5	—	5	mV
19	Bandgap output voltage temperature dependency V_{SUP}<18V, -40°C < T_J < 150°C	Δ_{VBGT}	-20	—	20	mV
20	Max. Base Current For External PNP (VDDX)[6] -40°C < T_J < 150°C	$I_{BCTLMAX}$	2.3	—	—	mA
21	Max. Base Current For External PNP (VDDX) 150°C < T_J < 175°C	$I_{BCTLMAX}$	1.5	—	—	mA
22	Max. Base Current For External PNP (VDDC) -40°C < T_J < 150°C	$I_{BCTLCMAX}$	2.3	—	—	mA
23	Max. Base Current For External PNP (VDDC) 150°C < T_J < 175°C	$I_{BCTLCMAX}$	1.5	—	—	mA
24	Recovery time from STOP	t_{STP_REC}	—	23	—	µs

[1] Please note that the core current is derived from VDDX

[2] LVI is monitored on the VDDA supply domain

[3] LVRX is monitored on the VDDX supply domain only active during full performance mode. During reduced performance mode (stop mode) voltage supervision is solely performed by the POR block monitoring core VDD.

[4] Nominal condition is T_a=25°C and VDDA=VDDX=5V

[5] VREGHTTR=0x88

[6] This is the minimum base current that can be guaranteed when the external PNP is delivering maximum current.

E.2 IRC and OSC Electrical Specifications

Table E-2. IRC electrical characteristics

Num	Rating	Symbol	Min	Typ	Max	Unit
1a	Internal Reference Frequency, factory trimmed -40°C < T_J < 150°C	f_{IRC1M_TRIM}	0.9895	1.002	1.0145	MHz
1b	Internal Reference Frequency, factory trimmed 150°C < T_J < 175°C	f_{IRC1M_TRIM}	0.9855		1.0145	MHz

Table E-3. OSC electrical characteristics (Junction Temperature From –40°C To +175°C)

Num	Rating	Symbol	Min	Typ	Max	Unit
1	Nominal crystal or resonator frequency	f_{OSC}	4.0	—	20	MHz
2	Startup Current	i_{OSC}	100	—	—	μA
3a	Oscillator start-up time (4MHz)[1]	t_{UPOSC}	—	2	10	ms
3b	Oscillator start-up time (8MHz)[1]	t_{UPOSC}	—	1.6	8	ms
3c	Oscillator start-up time (16MHz)[1]	t_{UPOSC}	—	1	5	ms
3d	Oscillator start-up time (20MHz)[1]	t_{UPOSC}	—	1	4	ms
4	Clock Monitor Failure Assert Frequency	f_{CMFA}	200	450	1200	KHz
5	Input Capacitance (EXTAL, XTAL pins)	C_{IN}	—	7	—	pF
6	EXTAL Pin Input Hysteresis	$V_{HYS,EXTAL}$	—	120	—	mV
7	EXTAL Pin oscillation amplitude (loop controlled Pierce)	$V_{PP,EXTAL}$	—	1	—	V
8	EXTAL Pin oscillation required amplitude[2]	$V_{PP,EXTAL}$	0.8	—	1.5	V

[1] These values apply for carefully designed PCB layouts with capacitors that match the crystal/resonator requirements.

[2] Needs to be measured at room temperature on the application board using a probe with very low (<=5pF) input capacitance.

E.3 Phase Locked Loop

E.3.1 Jitter Information

With each transition of the feedback clock, the deviation from the reference clock is measured and the input voltage to the VCO is adjusted accordingly.The adjustment is done continuously with no abrupt changes in the VCOCLK frequency. Noise, voltage, temperature and other factors cause slight variations in the control loop resulting in a clock jitter. This jitter affects the real minimum and maximum clock periods as illustrated in **Figure E-1.**.

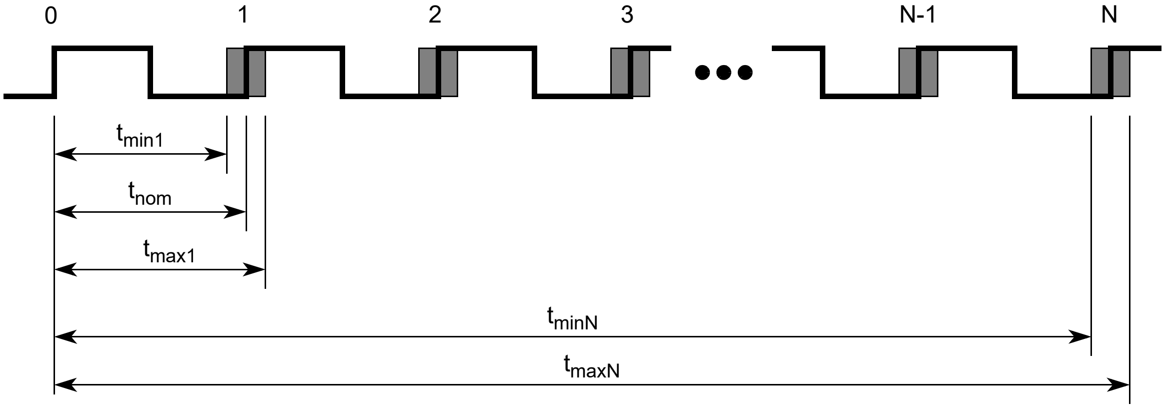


Figure E-1. Jitter Definitions

The relative deviation of t_{nom} is at its maximum for one clock period, and decreases towards zero for larger number of clock periods (N).

Defining the jitter as:

$$J(N) = \max\left(\left|1 - \frac{t_{max}(N)}{N \cdot t_{nom}}\right|, \left|1 - \frac{t_{min}(N)}{N \cdot t_{nom}}\right|\right)$$

The following equation is a good fit for the maximum jitter:

$$J(N) = \frac{j_1}{\sqrt{N(POSTDIV + 1)}}$$

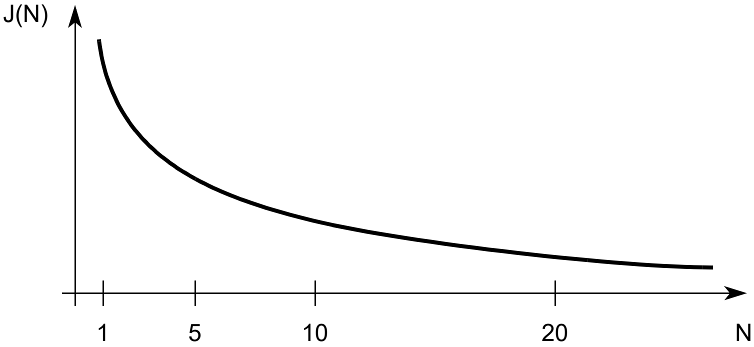


Figure E-2. Maximum Bus Clock Jitter Approximation (N = Number of Bus Cycles)

NOTE

Peripheral module prescalers eliminate the effect of jitter to a large extent.

Table E-4. PLL Characteristics (Junction Temperature From –40°C To +175°C)

Conditions are 4.5 V < V_{DDX}< 5.5 V unless otherwise noted						
Num	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	VCO frequency during system reset	f_{VCORST}	8	—	32	MHz
2	VCO locking range	f_{VCO}	32	—	64	MHz
3	Reference Clock	f_{REF}	1	—	—	MHz
4	Lock Detection	$\|\Delta_{Lock}\|$	0	—	1.5	%[1]
5	Un-Lock Detection Threshold	$\|\Delta_{unl}\|$	0.5	—	2.5	%[1]
7	Time to lock	t_{lock}	—	—	150 + 256/f_{REF}	µs
8	Jitter fit parameter 1[2] 40°C < T_J < 150°C	j_1	—	—	2	%
9a	PLL Clock Monitor Failure assert frequency	f_{PMFA}	0.45	1.1	1.6	MHz

[1] % deviation from target frequency

[2] f_{REF} = 1MHz, f_{BUS} = 32MHz

Appendix F
BATS Electrical Specifications

F.1 Static Electrical Characteristics

Table F-1. Static Electrical Characteristics - BATS (Junction Temperature From -40°C To +150°C)

Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[1] under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	Low Voltage Warning (LBI 1) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI1_A} V_{LBI1_D} V_{LBI1_H}	4.75 – –	5.5 – 0.4	6 6.5 –	V V V
2	Low Voltage Warning (LBI 2) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI2_A} V_{LBI2_D} V_{LBI2_H}	6 – –	6.75 – 0.4	7.25 7.75 –	V V V
3	Low Voltage Warning (LBI 3) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI3_A} V_{LBI3_D} V_{LBI3_H}	7 – –	7.75 – 0.4	8.5 9 –	V V V
4	Low Voltage Warning (LBI 4) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI4_A} V_{LBI4_D} V_{LBI4_H}	8 – –	9 – 0.4	10 10.5 –	V V V
5	High Voltage Warning (HBI 1) Assert (Measured on VSUP pin, rising edge) Deassert (Measured on VSUP pin, falling edge) Hysteresis (measured on VSUP pin)	V_{HBI1_A} V_{HBI1_D} V_{HBI1_H}	14.5 14.0 –	16.5 – 1.0	18 – –	V V V
6	High Voltage Warning (HBI 2) Assert (Measured on VSUP pin, rising edge) Deassert (Measured on VSUP pin, falling edge) Hysteresis (measured on VSUP pin)	V_{HBI2_A} V_{HBI2_D} V_{HBI2_H}	25 24 –	27.5 – 1.0	30 – –	V V V
7	Pin Input Divider Ratio[2] $Ratio_{VSUP} = V_{SUP} / V_{ADC}$ 5.5V < VSUP < 29 V	$Ratio_{VSUP}$	–	9	–	–
8	Analog Input Matching Absolute Error on V_{ADC} - compared to $V_{SUP} / Ratio_{VSUP}$	$AI_{Matching}$	–	+-2%	+-5%	–

[1] T_A: Ambient Temperature

[2] V_{ADC}: Voltage accessible at the ADC input channel

Table F-2. Static Electrical Characteristics - BATS (Junction Temperature From 150°C To +175°C)

Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Low Voltage Warning (LBI 1) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI1_A} V_{LBI1_D} V_{LBI1_H}	4.75 – –	5.5 – 0.4	6 6.5 –	V V V
2	Low Voltage Warning (LBI 2) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI2_A} V_{LBI2_D} V_{LBI2_H}	6 – –	6.75 – 0.4	7.25 7.75 –	V V V
3	Low Voltage Warning (LBI 3) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI3_A} V_{LBI3_D} V_{LBI3_H}	7 – –	7.75 – 0.4	8.5 9 –	V V V
4	Low Voltage Warning (LBI 4) Assert (Measured on VSUP pin, falling edge) Deassert (Measured on VSUP pin, rising edge) Hysteresis (measured on VSUP pin)	V_{LBI4_A} V_{LBI4_D} V_{LBI4_H}	8 – –	9 – 0.4	10 10.5 –	V V V
5	High Voltage Warning (HBI 1) Assert (Measured on VSUP pin, rising edge) Deassert (Measured on VSUP pin, falling edge) Hysteresis (measured on VSUP pin)	V_{HBI1_A} V_{HBI1_D} V_{HBI1_H}	14.5 14 –	16.5 – 1.0	18 – –	V V V
6	High Voltage Warning (HBI 2) Assert (Measured on VSUP pin, rising edge) Deassert (Measured on VSUP pin, falling edge) Hysteresis (measured on VSUP pin)	V_{HBI2_A} V_{HBI2_D} V_{HBI2_H}	25 24 –	27.5 – 1.0	**30** – –	V V V
7	Pin Input Divider Ratio[1] $Ratio_{VSUP} = V_{SUP} / V_{ADC}$ 5.5V < VSUP < 29 V	$Ratio_{VSUP}$	–	9	–	–
8	Analog Input Matching Absolute Error on V_{ADC} - compared to $V_{SUP} / Ratio_{VSUP}$	$AI_{Matching}$	–	+-2%	+-5%	–

[1] V_{ADC}: Voltage accessible at the ADC input channel

F.2 Dynamic Electrical Characteristics

Table F-3. Dynamic Electrical Characteristics - (BATS).

Characteristics noted under conditions 5.5V ≤ VSUP ≤ 18 V, unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[1] under nominal conditions..

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	Enable Uncertainty Time	T_{EN_UNC}	–	1	–	us
2	Voltage Warning Low Pass Filter	f_{VWLP_filter}	–	0.5	–	Mhz

[1] T_A: Ambient Temperature

Appendix G PIM Electrical Specifications

G.1 High-Voltage Inputs (HVI) Electrical Characteristics

Table G-1. Static Electrical Characteristics - High Voltage Input Pins - Port L

Characteristics are 5.5V ≤ V_{SUP} ≤ 18 V, -40°C ≤ T_J ≤ 175°C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[2] under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
25	Digital Input Threshold • $V_{SUP} > 6.5V$ • $5.5V \le V_{SUP} \le 6.5V$	V_{TH_HVI}	 2.8 2.0	 3.5 2.5	 4.5 3.8	 V V
26	Input Hysteresis	V_{HYS_HVI}	_	250	_	mV
27	Pin Input Divider Ratio with external series R_{EXT_HVI} Ratio = V_{HVI} / $V_{Internal(ADC)}$	$Ratio_{L\_HVI}$ $Ratio_{H\_HVI}$	– –	2 6	– –	
28	Analog Input Matching Absolute Error on V_{ADC} • Compared to V_{HVI} / $Ratio_{L\_HVI}$ ($1V < V_{HVI} < 7V$) • Compared to V_{HVI} / $Ratio_{H\_HVI}$ ($3V < V_{HVI} < 21V$) • Direct Mode (PTADIRL=1) (0.5V < VHVI < 3.5V)	 AIM_{L_HVI} AIM_{H_HVI} AIM_{D_HVI}	 – – –	 ± 2 ± 2 ± 2	 ± 5 ± 5 ± 5	 % % %
29	High Voltage Input Series Resistor **Note:** Always required externally at HVI pins.	R_{EXT_HVI}	–	10	–	kΩ
30	Enable Uncertainty Time	t_{UNC_HVI}	–	1	–	µs
31	Input capacitance	C_{IN_HVI}	–	8	–	pF

[1] T_J: Junction Temperature

[2] T_A: Ambient Temperature

Table G-2. Absolute Maximum Ratings - High Voltage Input Pins - Port L

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	V_{HVI} Voltage Range	V_{HVI}	-27	–	42	V

Appendix H ACMP Electrical Specifications

This section describe the electrical characteristics of the analog comparator module.

H.1 Maximum Ratings

Table H-1. Maximum Ratings of the analog comparator - (ACMP).

Characteristics noted under conditions 3.13V <= VDDA <= 5.5V>, -40°C ≤ T_J ≤ 175°C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[2] under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	Max Rating (relative to supply)	V_{ACMP_MAX}	-0.3	–	VDDA +0.3	V
	Max Rating (absolute) V_{ACMP_0} V_{ACMP_1} V_{acmpi_0} V_{acmpi_1}	V_{ACMP_MAXA}	-0.3	–	6	V

[1] T_J: Junction Temperature

[2] T_A: Ambient Temperature

H.2 Static Electrical Characteristics

Table H-2. Static Electrical Characteristics of the analog comparator - (ACMP).

Characteristics noted under conditions 3.13V <= VDDA <= 5.5V>, -40°C ≤ T_J ≤ 150°C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[2] under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
2	Supply Current of ACMP T_J ≤ 150°C • Module disabled • Module enabled $\Delta V_{in} > 5^*V_{hyst}$	 I_{ACMP_off} I_{ACMP_run}	 - 80	 - -	 3 160	 µA µA
3	Supply Current of ACMP T_J ≤ 175°C • Module disabled • Module enabled $\Delta V_{in} > 5^*V_{hyst}$	 I_{ACMP_off} I_{ACMP_run}	 - 80	 -	 5 160	 µA µA

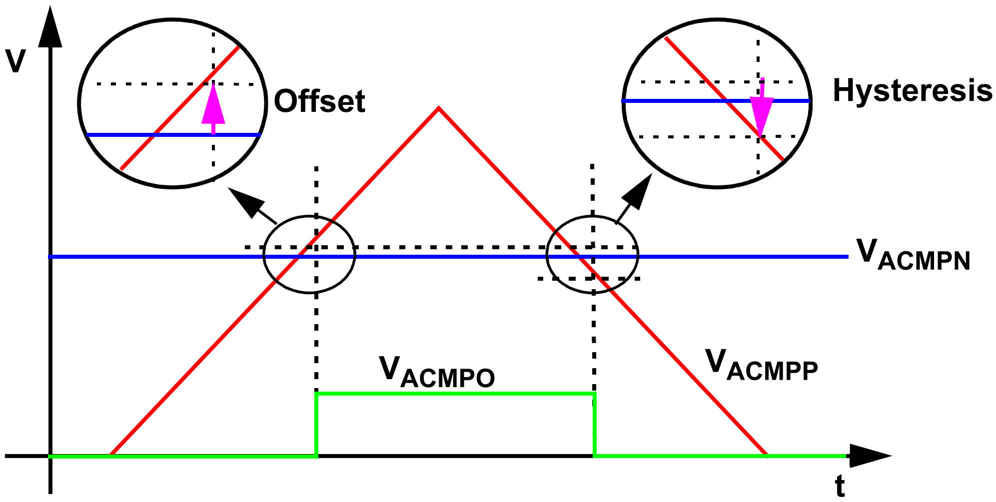
Table H-2. Static Electrical Characteristics of the analog comparator - (ACMP).

Characteristics noted under conditions 3.13V <= VDDA <= 5.5V>, -40°C ≤ T_J ≤ 150°C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[2] under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
4	Pad Input Current in V_{ACMP_in} range • -40°C ≤ T_J ≤ 80°C • -40°C ≤ T_J ≤ 150°C • -40°C ≤ T_J ≤ 175°C For 0V < Vpad_in < VDDA	$I_{ACMP\_pad\_in}$	-1 -2 -3	- - -	1 2 3	µA µA µA
5	Input Offset • -40°C ≤ T_J ≤ 150°C • -40°C ≤ T_J ≤ 175°C	V_{ACMP_offset}	-25 -25	0 0	25 25	mV mV
6	Input Hysteresis in run mode • [ACHYS] = 00 • [ACHYS] = 01 • [ACHYS] = 10 • [ACHYS] = 11	V_{ACMP_hyst}	-3 -10 -30 -50	-12 -24 -60 -125	-22 -40 -100 -200	mV mV mV mV
7	Common Mode Input range • V_{ACMP_0} • V_{ACMP_1} • V_{acmpi_0} • V_{acmpi_1}	V_{ACMP_in}	0	$V_{DDA}/2$	V_{DDA}	V
8	Common Mode Input range 150°C ≤ T_J ≤ 175°C • V_{ACMP_0} • V_{ACMP_1} • V_{acmpi_0} • V_{acmpi_1}	V_{ACMP_in}	0	$V_{DDA}/2$	V_{DDA}	V

[1] T_J: Junction Temperature

[2] T_A: Ambient Temperature

Input Offset and Hysteresis



H.3 Dynamic Electrical Characteristics

Table H-3. Dynamic Electrical Characteristics of the analog comparator - (ACMP).

Characteristics noted under conditions 3.13V <= VDDA <= 5.5V>, -40°C $\leq T_J \leq$ 150°C[1] unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C[2] under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	Output uncertain time after module enable	$t_{ACMP\_dly\_en}$	-	1	2	µs
2	ACMP Propagation Delay of Inputs ACMP0 and ACMP1 for -2*$V_{hyst(typ)}$ to +2*$V_{hyst(typ)}$ input step (w/o synchronize delay) • ACDLY=0 Low speed mode • ACDLY=1 High speed mode ACMPP is crossing ACMPN in positive direction	t_{ACMP_delay}	130 20	300 70	750 400	ns ns
3	ACMP Propagation Delay of Inputs ACMP0 and ACMP1 for -2*$V_{hyst(typ)}$ to +2*$V_{hyst(typ)}$ input step (w/o synchronize delay) 150°C $\leq T_J \leq$ 175°C • ACDLY=0 Low speed mode • ACDLY=1 High speed mode ACMPP is crossing ACMPN in positive direction	t_{ACMP_delay}	- -	- -	800 450	ns ns

[1] T_J: Junction Temperature

[2] T_A: Ambient Temperature

Appendix I
S12CANPHY Electrical Specifications

I.1 Maximum Ratings

Table I-1. Maximum Ratings

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.				
Num	**Ratings**	**Symbol**	**Value**	**Unit**
1	DC voltage on CANL, CANH, SPLIT	V_{BUS}	-32 to +40	V
2	Continuous current on CANH and CANL	I_{LH}	200	mA
3	ESD on CANH, CANL and SPLIT (HBM)	V_{ESDCH}	± 4000	V
4	ESD on CANH, CANL (IEC61000-4, Czap = 150 pF, Rzap = 330 Ω)	V_{ESDIEC}	± 6000	V

I.2 Static Electrical Characteristics

Table I-2. Static Electrical Characteristics

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
CAN TRANSCEIVER CURRENT						
1	Supply Current of canphy_ll18uhv Normal mode, Bus Recessive State Normal mode, Bus Dominant State without Bus Load Standby mode Shutdown mode	 I_{RES} I_{DOM} I_{STB} I_{SDN}		 1.7 3.8 0.022 0		mA
PINS (CANH AND CANL)						
2	Bus Pin Common Mode Voltage	V_{COM}	-12	-	12	V
3a	Differential Input Voltage (Normal mode) Recessive State at RXD Dominant State at RXD	 V_{CANH} - V_{CANL}	 -1.0 0.9	 - -	 0.5 5.0	 V V
3b	Differential Input Voltage (Standby mode) Recessive State at RXD Dominant State at RXD	 V_{CANH} - V_{CANL}	 -1.0 1.1	 - -	 0.4 5.0	 V V
4	Differential Input Hysteresis	V_{HYS}		175		mV

Table I-2. Static Electrical Characteristics

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
5	Input Resistance	R_{IN}	5	32.5	50	kΩ
6	Differential Input Resistance	R_{IND}	10	65	100	kΩ
7	Common mode input resistance matching	R_{INM}	-3	0	+3	%
8	CANH Output Voltage (R_L = 60Ω), (Normal mode) TXD Dominant State TXD Recessive State	V_{CANH}	2.75 2.0	3.5 2.5	4.5 3.0	V V
9	CANL Output Voltage (R_L = 60Ω), (Normal mode) TXD Dominant State TXD Recessive State	V_{CANL}	0.5 2.0	1.5 2.5	2.25 3.0	V V
10	Differential Output Voltage (R_L = 60Ω), (Normal mode) TXD Dominant State TXD Recessive State	V_{OH} - V_{OL}	1.5 -0.5	2.0 0	3.0 0.05	V V
11	CANH, CANL driver symmetry (Normal mode) (V_{CANH} + V_{CANL}) / VDDC	V_{SYM}	0.9	1	1.1	-
12	Output Current Capability (Dominant State) CANH CANL	I_{CANH} I_{CANL}		55 55		mA mA
13	CANH, CANL Overcurrent Detection (Tj >=25^{o}C) CANH CANL	I_{CANHOC} I_{CANLOC}	70 70	85 85	100 100	mA mA
14	CANH, CANL Output Voltage (no load, Standby mode) CANH CANL	V_{CANH} V_{CANL}	-0.1 -0.1	0 0	0.1 0.1	V V
15	CANH and CANL Input Current (Standby mode) V_{CANH}, V_{CANL} from 0 V to 5.0 V V_{CANH}, V_{CANL} = - 2.0 V V_{CANH}, V_{CANL} = 7.0 V	I_{CAN1}			20 -75 250	uA uA uA
16	CANH and CANL Input Current (Device unsupplied) (VSUP tied to ground or left open) V_{CANH}, V_{CANL} from 0V to 5 V V_{CANH}, V_{CANL} = - 2.0 V V_{CANH}, V_{CANL} = 7.0 V	I_{CAN2}			10 -75 250	uA uA uA
17	CANH, CANL Input capacitance (Normal mode) CANH CANL	C_{CANH} C_{CANL}		14 16		pF pF
18	CANH to CANL differential capacitance (Normal mode)	C_{HLDIFF}		6		pF
	DIAGNOSTIC INFORMATION (CANH AND CANL)					
19	CANL to 0 V Threshold	V_{L0}	-0.75	-0.15	0	V

Table I-2. Static Electrical Characteristics

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
20	CANH to 0 V Threshold	V_{H0}	-0.75	-0.15	0	V
21	CANL to 5.0 V Threshold	V_{L5}	VDDC	VDDC +0.15	VDDC + 0.75	V
22	CANH to 5.0 V Threshold	V_{H5}	VDDC	VDDC +0.15	VDDC +0.75	V
SPLIT						
23	Output voltage Loaded condition I_{SPLIT}= +/- 500 uA Unloaded condition Rmeasure > 1 MΩ	V_{SPLIT}	0.3 0.45	0.5 0.5	0.7 0.55	VDDC VDDC
24	Leakage current -12 V < V_{SPLIT} < +12 V -22 V < V_{SPLIT} < +35 V	I_{LSPLIT}	- -	0 -	5 25	µA µA

I.3 Dynamic Electrical Characteristics

Table I-3. Dynamic Electrical Characteristics

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
SIGNAL EDGE RISE AND FALL TIMES (CANH, CANL)						
1	Propagation Loop Delay TXD to RXD (Recessive to Dominant) Slew Rate 6 Slew Rate 5 Slew Rate 4 Slew Rate 2 Slew Rate 1 Slew Rate 0	t_{LRD}		 146 112 89 83 72 64	 (255)	ns
2	Propagation Delay TXD to CAN (Recessive to Dominant) Slew Rate 6 Slew Rate 5 Slew Rate 4 Slew Rate 2 Slew Rate 1 Slew Rate 0	t_{TRD}		 98 63 43 38 28 23		ns
3	Propagation Delay CAN to RXD (Recessive to Dominant, using slew rate 0)	t_{RRD}		42		ns
4	Propagation Loop Delay TXD to RXD (Dominant to Recessive) Slew Rate 6 Slew Rate 5 Slew Rate 4 Slew Rate 2 Slew Rate 1 Slew Rate 0	t_{LDR}		 366 224 153 139 114 102	 (255)	ns
5	Propagation Delay TXD to CAN (Dominant to Recessive) Slew Rate 6 Slew Rate 5 Slew Rate 4 Slew Rate 2 Slew Rate 1 Slew Rate 0	t_{TDR}		 280 152 90 81 56 46		ns
6	Propagation Delay CAN to RXD (Dominant to Recessive, using slew rate 0)	t_{RDR}		56		ns

Table I-3. Dynamic Electrical Characteristics

Characteristics noted under conditions 5.5V <= VSUP <=18 V >, -40°C <= Tj <= 150°C > unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
7	Non-Differential Slew Rate (CANL or CANH) Slew Rate 6 Slew Rate 5 Slew Rate 4 Slew Rate 2 Slew Rate 1 Slew Rate 0	 t_{SL6} t_{SL5} t_{SL4} t_{SL2} t_{SL1} t_{SL0}		 6 10 19 23 35 55		V/µs
8	Bus Communication Rate	t_{BUS}			1.0 M	bps
9	Settling time after entering Normal mode	t_{CP_set}			10	µs
10	CPTXD-dominant timeout	$t_{CPTXDDT}$		2		ms
11	CANPHY wake-up dominant pulse filtered	t_{CPWUP}			1.5	µs
12	CANPHY wake-up dominant pulse pass	t_{CPWUP}	5			µs

Appendix J
DAC8B5V Electrical Specifications

Table J-1. Static Electrical Characteristics - DAC8B5V

Characteristics noted under conditions 4.75V <= VDDA <= 5.25V>, -40°C < T_j < 175°C , VRH=VDDA, VRL=VSSA unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
1	Supply Current of DAC8B5V buffer disabled buffer enabled FVR=0 DRIVE=1 buffer enabled FVR=1 DRIVE=0	I_{buf}	- - -	- 365 215	5 800 800	µA
2	Reference current (–40°C <T_J< +150°C) reference disabled reference enabled	I_{ref}	-	- 50	1 150	µA
3	Resolution		8			bit
4	Relative Accuracy measured at AMP	INL	-0.5		+0.5	LSB
5	Differential Nonlinearity measured at AMP	DNL	-0.5		+0.5	LSB
6	Relative Accuracy measured at AMP 150°C < T_j < 175°C	INL	-0.75		+0.75	LSB
7	Differential Nonlinearity measured at AMP 150°C < T_j < 175°C	DNL	-0.75		+0.75	LSB
8	DAC Range A (FVR bit = 1)	V_{out}	0...255/256(VRH-VRL)+VRL			V
9	DAC Range B (FVR bit = 0)	V_{out}	32...287/320(VRH-VRL)+VRL			V
10	Output Voltage unbuffered range A or B (load >= 50MΩ)	V_{out}	full DAC Range A or B			V
11	Output Voltage (DRIVE bit = 0) *) buffered range A (load >= 100KΩ to VSSA) buffered range A (load >= 100KΩ to VDDA)	V_{out}	0 0.15	- -	VDDA-0.15 VDDA	V
	buffered range B (load >= 100KΩ to VSSA) buffered range B (load >= 100KΩ to VDDA)		full DAC Range B			
12	Output Voltage (DRIVE bit = 1) **) buffered range B with 6.4KΩ load into resistor divider of 800Ω /6.56KΩ between VDDA and VSSA. (equivalent load is >= 65KΩ to VSSA) or (equivalent load is >= 7.5KΩ to VDDA)	V_{out}	full DAC Range B			V

Table J-1. Static Electrical Characteristics - DAC8B5V

Characteristics noted under conditions 4.75V <= VDDA <= 5.25V>, -40°C < T_j < 175°C , VRH=VDDA, VRL=VSSA unless otherwise noted. Typical values noted reflect the approximate parameter mean at T_A = 25°C under nominal conditions unless otherwise noted.

Num	Ratings	Symbol	Min	Typ	Max	Unit
13	Buffer Output Capacitive load	C_{load}	0	-	100	pF
14	Buffer Output Offset	V_{offset}	-30	-	+30	mV
15	Settling time	t_{delay}	-	3	5	µs
16	Reverence voltage high	V_{refh}	VDDA-0.1V	VDDA	VDDA+0.1 V	V

*) DRIVE bit = 1 is not recomended in this case.

**) DRIVE bit = 0 is not allowed with this high load.

Appendix K
NVM Electrical Parameters

K.1 NVM Timing Parameters

The time base for all NVM program or erase operations is derived from the bus clock using the FCLKDIV register. The frequency of this derived clock must be set within the limits specified as f_{NVMOP}. The NVM module does not have any means to monitor the frequency and will not prevent program or erase operation at frequencies above or below the specified minimum. When attempting to program or erase the NVM module at a lower frequency, a full program or erase transition is not assured.

The device bus frequency, below which the flash wait states can be disabled, is specified in the device operating condition table in Table A-5.

The following sections provide equations which can be used to determine the time required to execute specific flash commands. All timing parameters are a function of the bus clock frequency, f_{NVMBUS}. All program and erase times are also a function of the NVM operating frequency, f_{NVMOP}. A summary of key timing parameters can be found in Table K.

K.2 NVM Reliability Parameters

The reliability of the NVM blocks is guaranteed by stress test during qualification, constant process monitors and burn-in to screen early life failures.

The data retention and program/erase cycling failure rates are specified at the operating conditions noted. The program/erase cycle count on the sector is incremented every time a sector or mass erase event is executed.

Table K-1. NVM Clock Timing Characteristics

Num	Rating	Symbol	Min	Typ	Max	Unit
1	Bus frequency	f_{NVMBUS}	1	32	32	MHz
2	Operating frequency	f_{NVMOP}	0.8	1.0	1.05	MHz

Table K-2. NVM Timing Characteristics 32 MHz (Junction Temperature From –40°C To +175°C)

Num	Command	f_{NVMOP} cycle	f_{NVMBUS} cycle	Symbol	Min[1]	Typ[2]	Max[3]	Lfmax[4]	Unit
1	Erase Verify All Blocks[5,6]	0	51101	t_{RD1ALL}	1.60	1.60	3.19	102.20	ms
2	Erase Verify Block (Pflash)[5]	0	49894	t_{RD1BLK_P}	1.56	1.56	3.12	99.79	ms
3	Erase Verify Block (EEPROM)[6]	0	1608	t_{RD1BLK_D}	0.05	0.05	0.10	3.22	ms
4	Erase Verify P-Flash Section	0	624	t_{RD1SEC}	0.02	0.02	0.04	1.25	ms
5	Read Once	0	512	t_{RDONCE}	16.00	16.00	16.00	512.00	us

Table K-2. NVM Timing Characteristics 32 MHz (Junction Temperature From –40°C To +175°C)

Num	Command	f_{NVMOP} cycle	f_{NVMBUS} cycle	Symbol	Min[1]	Typ[2]	Max[3]	Lfmax[4]	Unit
6	Program P-Flash (4 Word)	164	3215	t_{PGM_4}	0.26	0.26	0.57	13.07	ms
7	Program Once	164	3138	$t_{PGMONCE}$	0.25	0.26	0.26	3.34	ms
8	Erase All Blocks[5,6]	200126	51914	t_{ERSALL}	192.22	201.75	203.37	353.99	ms
9	Erase Flash Block (Pflash)[5]	200120	50464	t_{ERSBLK_P}	192.17	201.70	203.27	351.08	ms
10	Erase Flash Block (EEPROM)[6]	100060	1905	t_{ERSBLK_D}	95.35	100.12	100.18	128.89	ms
11	Erase P-Flash Sector	20015	1016	t_{ERSPG}	19.09	20.05	20.08	27.05	ms
12	Unsecure Flash	200126	51992	t_{UNSECU}	192.22	201.75	203.38	354.14	ms
13	Verify Backdoor Access Key	0	524	t_{VFYKEY}	16.38	16.38	16.38	524.00	us
14	Set User Margin Level	0	465	t_{MLOADU}	14.53	14.53	14.53	465.00	us
15	Set Factory Margin Level	0	474	t_{MLOADF}	14.81	14.81	14.81	474.00	us
16	Erase Verify EEPROM Section	0	608	$t_{DRD1SEC}$	0.02	0.02	0.04	1.22	ms
17	Program EEPROM (1 Word)	68	1689	t_{DPGM_1}	0.12	0.12	0.28	6.84	ms
18	Program EEPROM (2 Word)	136	2713	t_{DPGM_2}	0.21	0.22	0.48	11.02	ms
19	Program EEPROM (3 Word)	204	3737	t_{DPGM_3}	0.31	0.32	0.67	15.20	ms
20	Program EEPROM (4 Word)	272	4761	t_{DPGM_4}	0.41	0.42	0.87	19.38	ms
21	Erase EEPROM Sector	5015	834	t_{DERSPG}	4.80	5.04	20.50	39.25	ms
22	Protection Override	0	506	$t_{PRTOVRD}$	15.81	15.81	15.81	506.00	us

[1] Minimum times are based on maximum f_{NVMOP} and maximum f_{NVMBUS}

[2] Typical times are based on typical f_{NVMOP} and typical f_{NVMBUS}

[3] Maximum times are based on typical f_{NVMOP} and typical f_{NVMBUS} plus aging

[4] Lowest-frequency max times are based on minimum f_{NVMOP} and minimum f_{NVMBUS} plus aging

[5] Affected by Pflash size

[6] Affected by EEPROM size

Table K-3. NVM Reliability Characteristics (Junction Temperature From –40°C To +175°C)

NUM	C	Rating	Symbol	Min	Typ	Max	Unit
	Program Flash Arrays						
1	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after up to 10,000 program/erase cycles	t_{NVMRET}	20	100[2]	—	Years
2	C	Program Flash number of program/erase cycles (-40°C ≤ Tj ≤ 150°C)	n_{FLPE}	10K	100K[3]	—	Cycles
	EEPROM Array						
3	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after up to 100,000 program/erase cycles	t_{NVMRET}	5	100[2]	—	Years
4	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after up to 10,000 program/erase cycles	t_{NVMRET}	10	100[2]	—	Years
5	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after less than 100 program/erase cycles	t_{NVMRET}	20	100[2]	—	Years
6	C	EEPROM number of program/erase cycles (-40°C ≤ Tj ≤ 150°C)	n_{FLPE}	100K	500K[3]	—	Cycles

[1] T_{Javg} does not exceed 85°C in a typical temperature profile over the lifetime of a consumer, industrial or automotive application.

[2] Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how NXP defines Typical Data Retention, please refer to Engineering Bulletin EB618

[3] Spec table quotes typical endurance evaluated at 25°C for this product family. For additional information on how NXP defines Typical Endurance, please refer to Engineering Bulletin EB619.

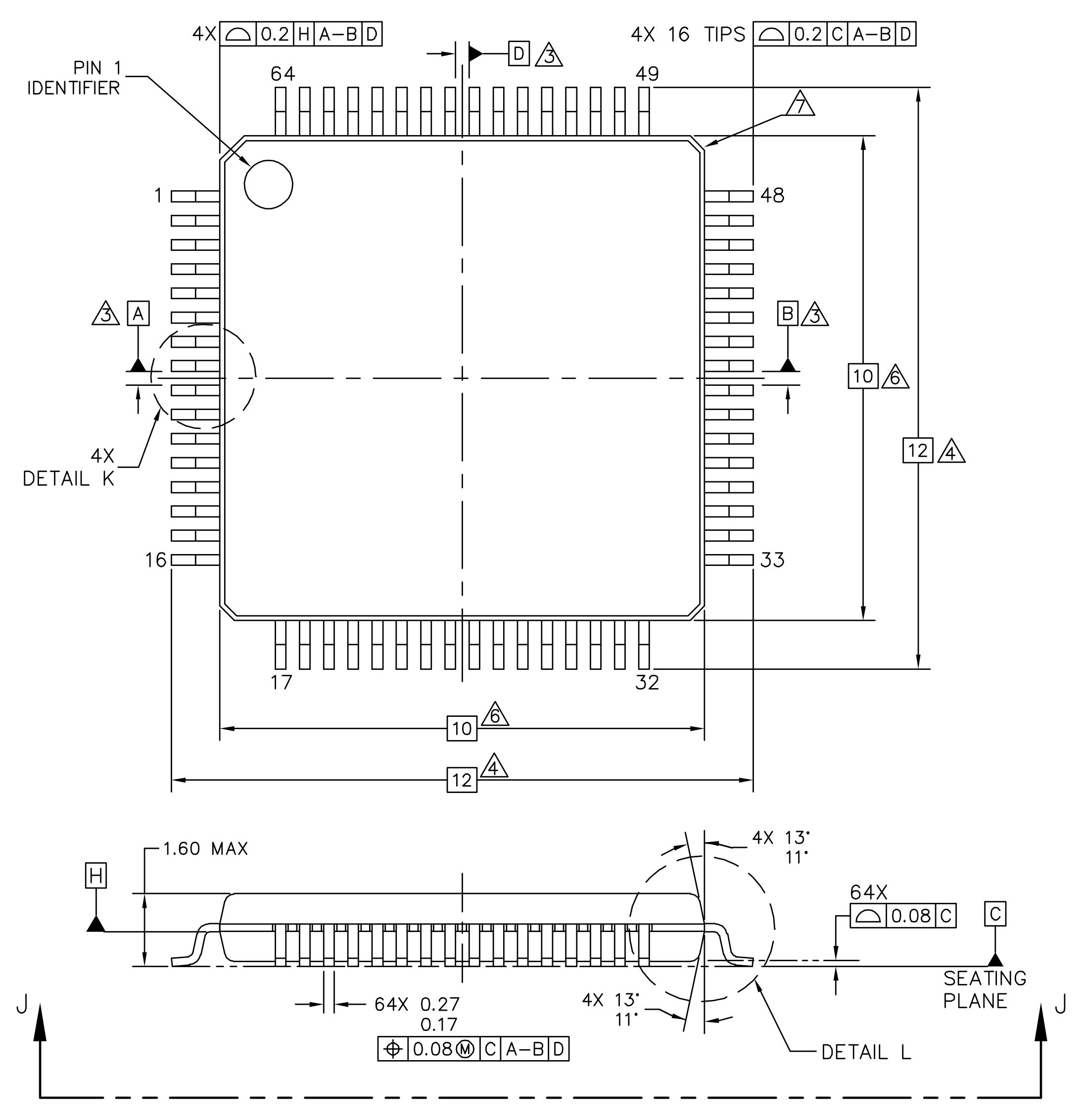
K.3 NVM Factory Shipping Condition

Devices are shipped from the factory with flash and EEPROM in the erased state. Data retention specifications begin at time of this erase operation. For additional information on how NXP defines Typical Data Retention, please refer to Engineering Bulletin EB618.

Appendix L
Package Information

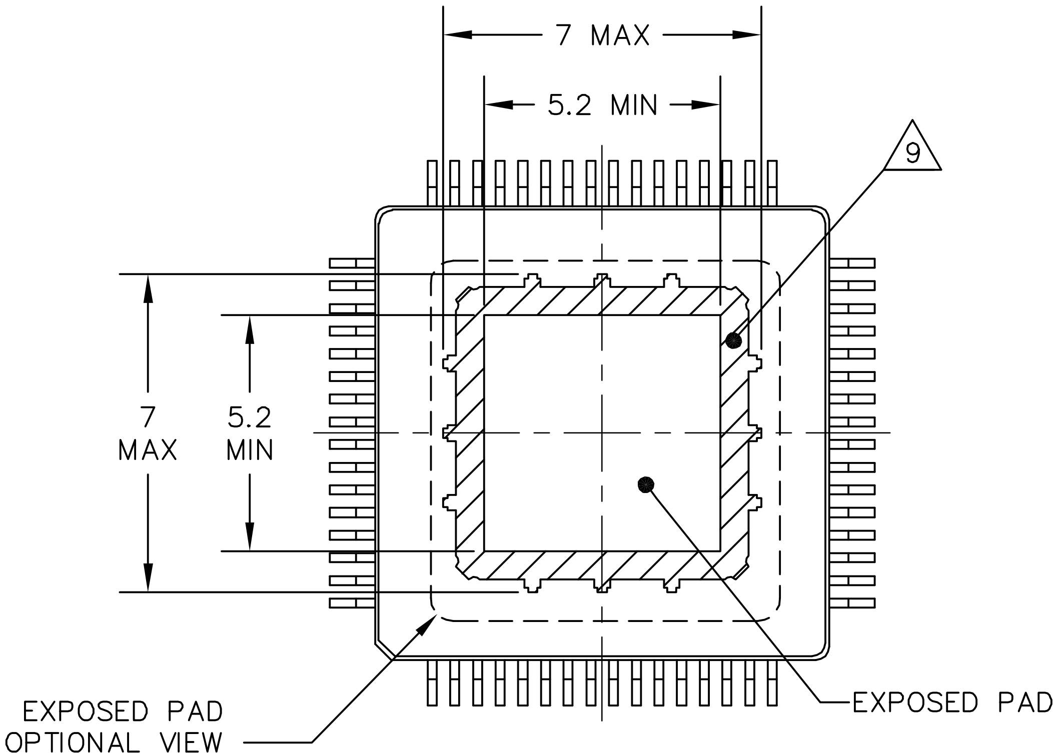
Figure L-1. 64 LQFP Exposed Pad Package

freescale semiconductor	MECHANICAL OUTLINES DICTIONARY	DOCUMENT NO: 98ASA10763D
		PAGE: 1899
	DO NOT SCALE THIS DRAWING	REV: B

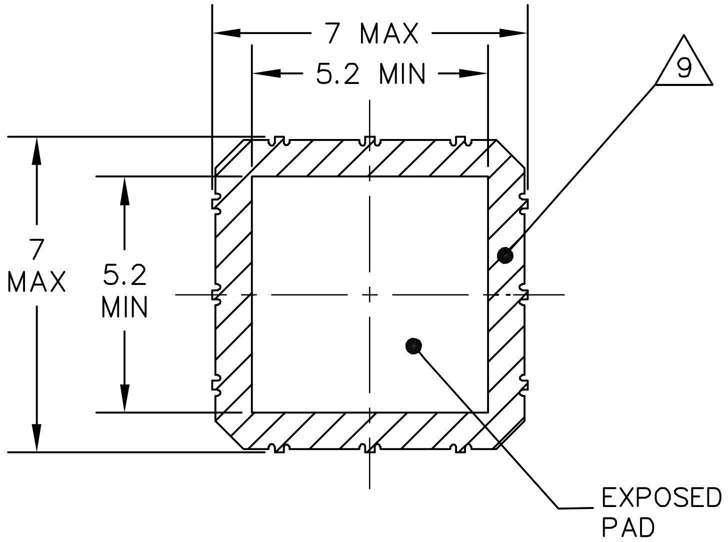


TITLE: LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, 64LD, 6.1 x 6.1 EXPOSED PAD	CASE NUMBER: 1899−03
	STANDARD: JEDEC MS−026 BCD
	SHEET: 1 OF 5

Figure L-2. 64 LQFP Exposed Pad Package

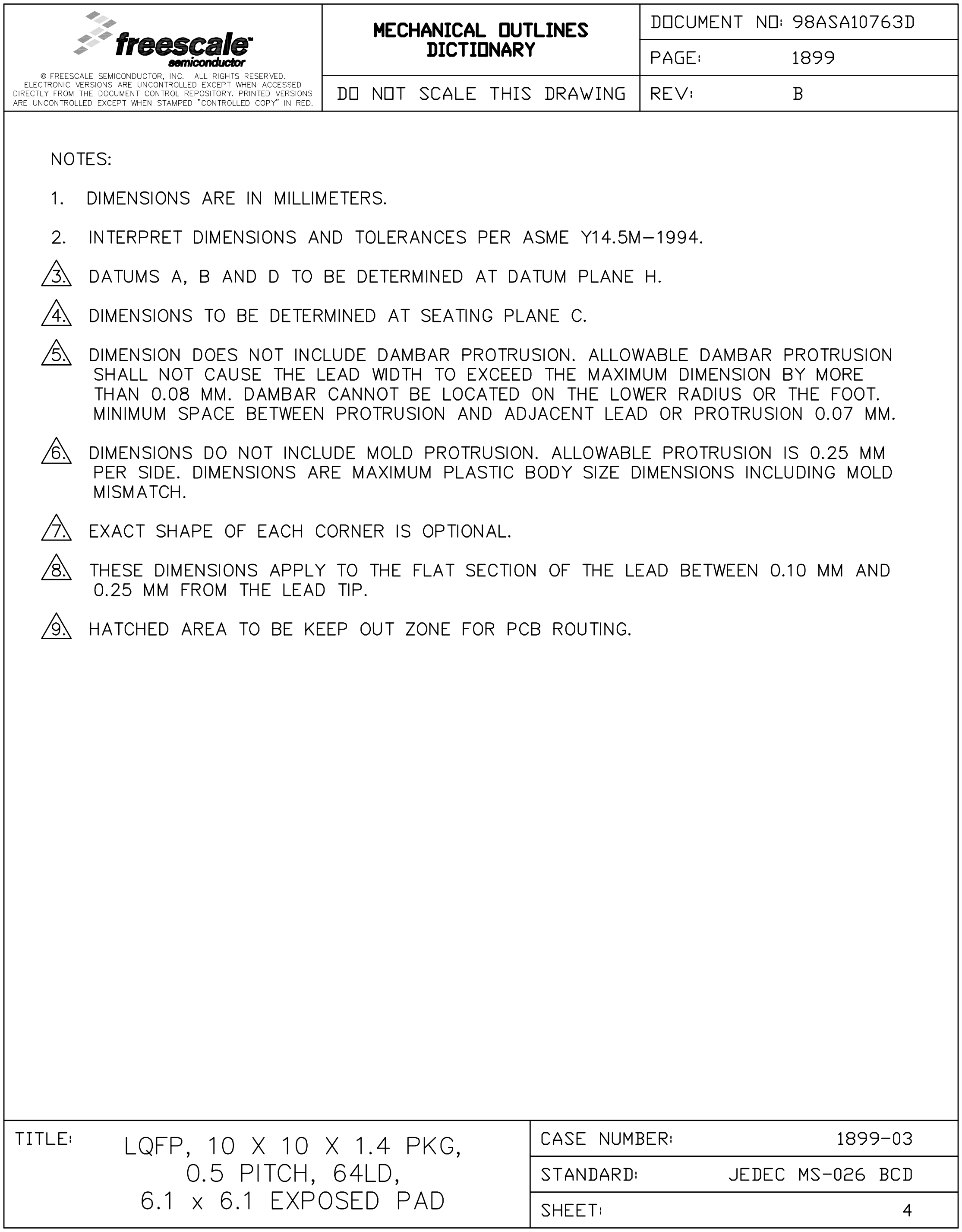


VIEW J–J



EXPOSED PAD
OPTIONAL VIEW

Figure L-3. 64 LQFP Exposed Pad Package



freescale semiconductor

MECHANICAL OUTLINES
DICTIONARY

DO NOT SCALE THIS DRAWING

DOCUMENT NO: 98ASA10763D
PAGE: 1899
REV: B

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M−1994.

3. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.

4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE C.

5. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM DIMENSION BY MORE THAN 0.08 MM. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 MM.

6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 MM PER SIDE. DIMENSIONS ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.

7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

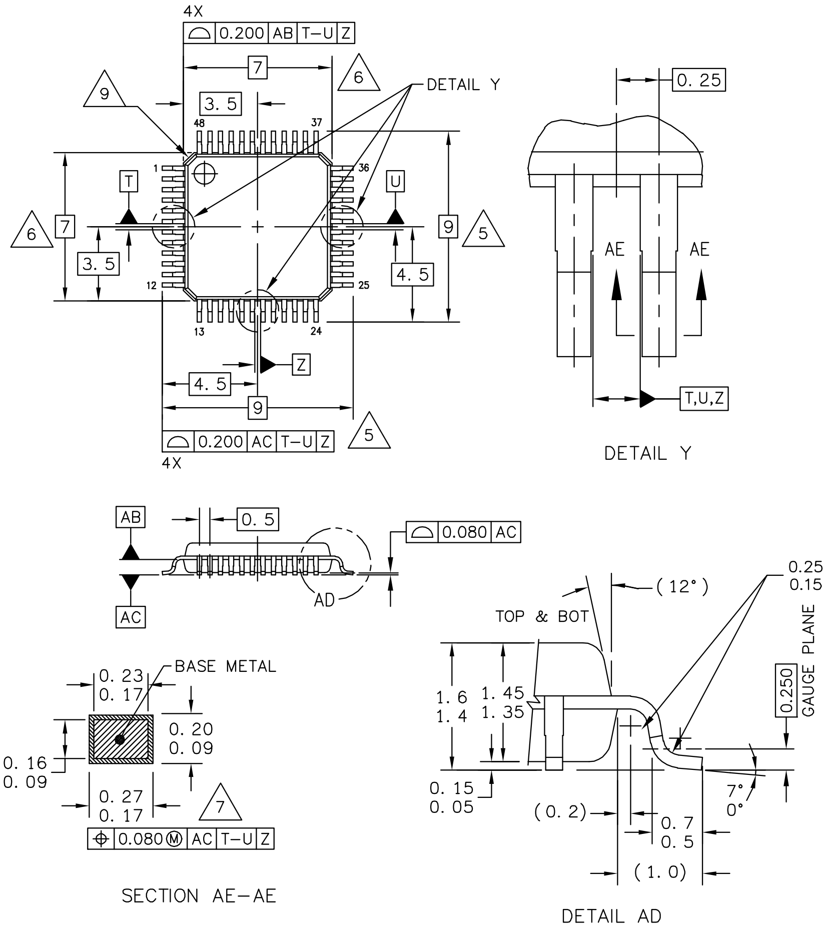
8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.10 MM AND 0.25 MM FROM THE LEAD TIP.

9. HATCHED AREA TO BE KEEP OUT ZONE FOR PCB ROUTING.

TITLE: LQFP, 10 X 10 X 1.4 PKG,
0.5 PITCH, 64LD,
6.1 x 6.1 EXPOSED PAD

CASE NUMBER: 1899-03
STANDARD: JEDEC MS-026 BCD
SHEET: 4

Figure L-4. 48 LQFP Package



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE	
TITLE: LQFP, 48 LEAD, 0.50 PITCH (7.0 X 7.0 X 1.4)	DOCUMENT NO: 98ASH00962A		REV: G
	CASE NUMBER: 932-03		14 APR 2005
	STANDARD: JEDEC MS-026-BBC		

Figure L-5. 48 LQFP Package

NOTES:

1. DIMENSIONS AND TOLERANCING PER ASME Y14.5M–1994.

2. CONTROLLING DIMENSION: MILLIMETER.

3. DATUM PLANE AB IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.

4. DATUMS T, U, AND Z TO BE DETERMINED AT DATUM PLANE AB.

5. DIMENSIONS TO BE DETERMINED AT SEATING PLANE AC.

6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.250 PER SIDE. DIMENSIONS DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE AB.

7. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.350.

8. MINIMUM SOLDER PLATE THICKNESS SHALL BE 0.0076.

9. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE	
TITLE: LQFP, 48 LEAD, 0.50 PITCH (7.0 X 7.0 X 1.4)	DOCUMENT NO: 98ASH00962A		REV: G
	CASE NUMBER: 932–03		14 APR 2005
	STANDARD: JEDEC MS–026–BBC		

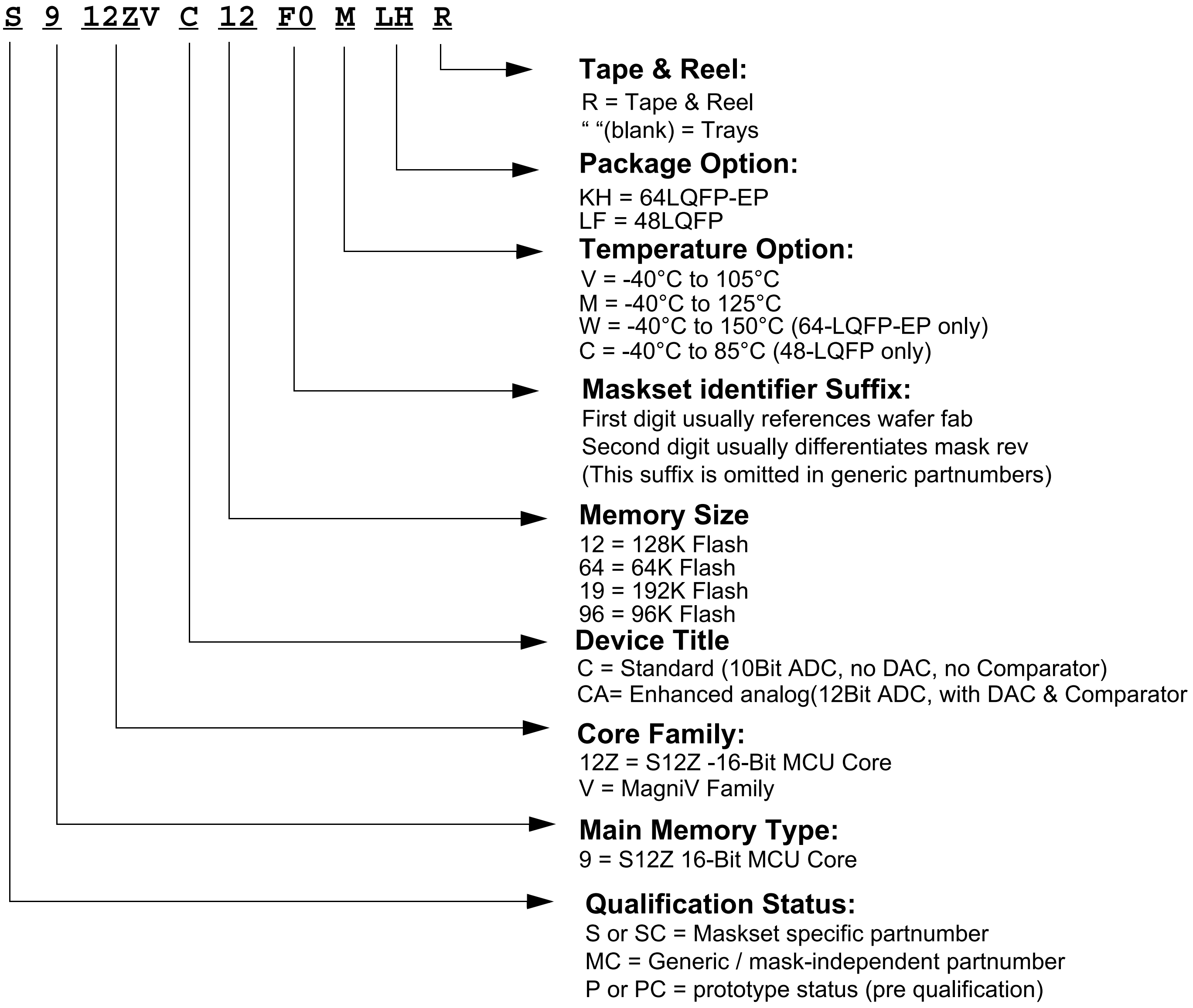
Figure L-6.

Appendix M
Ordering Information

Customers can choose either the mask-specific partnumber or the generic, mask-independent partnumber. Ordering a mask-specific partnumber enables the customer to specify which particular maskset they receive whereas ordering the generic partnumber means that the currently preferred maskset (which may change over time) is shipped. In either case, the marking on the device always shows the generic, mask-independent partnumber and the mask set number. The below figure illustrates the structure of a typical mask-specific ordering number.

NOTES

Not every combination is offered. Table 1-4 lists available derivatives.The mask identifier suffix and the Tape & Reel suffix are always both omitted from the partnumber which is actually marked on the device.



S 9 12ZV C 12 F0 M LH R

Tape & Reel:
R = Tape & Reel
“ “(blank) = Trays

Package Option:
KH = 64LQFP-EP
LF = 48LQFP

Temperature Option:
V = -40°C to 105°C
M = -40°C to 125°C
W = -40°C to 150°C (64-LQFP-EP only)
C = -40°C to 85°C (48-LQFP only)

Maskset identifier Suffix:
First digit usually references wafer fab
Second digit usually differentiates mask rev
(This suffix is omitted in generic partnumbers)

Memory Size
12 = 128K Flash
64 = 64K Flash
19 = 192K Flash
96 = 96K Flash

Device Title
C = Standard (10Bit ADC, no DAC, no Comparator)
CA= Enhanced analog(12Bit ADC, with DAC & Comparator

Core Family:
12Z = S12Z -16-Bit MCU Core
V = MagniV Family

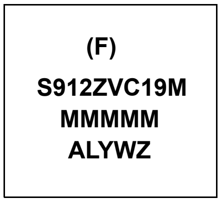
Main Memory Type:
9 = S12Z 16-Bit MCU Core

Qualification Status:
S or SC = Maskset specific partnumber
MC = Generic / mask-independent partnumber
P or PC = prototype status (pre qualification)

Example Top Side Marking

Characters “YW” in the topside trace code line (ALYWZ) = JW for work week 23, JX for week 24, JY for week 25...

Figure M-1. Example Top Side Marking



Appendix N
Detailed Register Address Map

Table N-1. Revision History Table

Revision Number	Revision Date	Sections Affected	Description Of Changes
0.01	22-Aug-2013		Initial version
0.02	25-Oct-2013		Added instance suffix to SENTTX register names

The following tables show the detailed register map of the MC9S12ZVC-Family.

NOTE

Smaller derivatives within the MC9S12ZVC-Family feature a subset of the listed modules.

N.1 0x0000–0x0003 Part ID

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0000	PARTID0	R	0	0	0	0	0	0	0	0
		W								
0x0001	PARTID1	R	0	0	0	1	0	1	1	1
		W								
0x0002	PARTID2	R	0	0	0	0	0	0	0	0
		W								
0x0003	PARTID3	R	Revision Dependent							
		W								

N.2 0x0010–0x001F S12ZINT

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0010	IVBR	R	IVB_ADDR[15:8]							
		W								
0x0011		R	IVB_ADDR[7:1]							0
		W								
0x0016	INT_XGPRIO	R	0	0	0	0	0	XILVL[2:0]		
		W								
0x0017	INT_CFADDR	R	0	INT_CFADDR[6:3]				0	0	0
		W								
0x0018	INT_CFDATA0	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x0019	INT_CFDATA1	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								

N.2 0x0010–0x001F S12ZINT

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001A	INT_CFDATA2	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001B	INT_CFDATA3	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001C	INT_CFDATA4	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001D	INT_CFDATA5	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001E	INT_CFDATA6	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001F	INT_CFDATA7	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								

N.3 0x0070-0x008F S12ZMMC

Address	**Name**		**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
0x0070	MODE	R	MODC	0	0	0	0	0	0	0
		W								
0x0071-0x007F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0080	MMCECH	R	ITR[3:0]				TGT[3:0]			
		W								
0x0081	MMCECL	R	ACC[3:0]				ERR[3:0]			
		W								
0x0082	MMCCCRH	R	CPUU	0	0	0	0	0	0	0
		W								
0x0083	MMCCCRL	R	0	CPUX	0	CPUI	0	0	0	0
		W								
0x0084	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0085	MMCPCH	R	CPUPC[23:16]							
		W								
0x0086	MMCPCM	R	CPUPC[15:8]							
		W								
0x0087	MMCPCL	R	CPUPC[7:0]							
		W								
0x0088-0x00FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.4 0x0100-0x017F S12ZDBG

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0100	DBGC1	R	ARM	0	reserved	BDMBP	BRKCPU	reserved	EEVE1	0
		W		TRIG						
0x0101	DBGC2	R	0	0	0	0	0	0	ABCM	
		W								
0x0102	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.4 0x0100-0x017F S12ZDBG (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0103	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0104	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0105	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0106	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0107	DBGSCR1	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0108	DBGSCR2	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0109	DBGSCR3	R	C3SC1	C3SC0	0	0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x010A	DBGEFR	R	0	TRIGF	0	EEVF	ME3	0	ME1	ME0
		W								
0x010B	DBGSR	R	0	0	0	0	0	SSF2	SSF1	SSF0
		W								
0x010C- 0x010F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0110	DBGACTL	R	0	NDB	INST	0	RW	RWE	reserved	COMPE
		W								
0x0111- 0x0114	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0115	DBGAAH	R	DBGAA[23:16]							
		W								
0x0116	DBGAAM	R	DBGAA[15:8]							
		W								
0x0117	DBGAAL	R	DBGAA[7:0]							
		W								
0x0118	DBGAD0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x0119	DBGAD1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x011A	DBGAD2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x011B	DBGAD3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

N.4 0x0100-0x017F S12ZDBG (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x011C	DBGADM0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x011D	DBGADM1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x011E	DBGADM2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x011F	DBGADM3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0120	DBGBCTL	R	0	0	INST	0	RW	RWE	reserved	COMPE
		W								
0x0121- 0x0124	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0125	DBGBAH	R	DBGBA[23:16]							
		W								
0x0126	DBGBAM	R	DBGBA[15:8]							
		W								
0x0127	DBGBAL	R	DBGBA[7:0]							
		W								
0x0128- 0x012F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0130- 0x013F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0140	DBGDCTL	R	0	0	INST	0	RW	RWE	reserved	COMPE
		W								
0x0141- 0x0144	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0145	DBGDAH	R	DBGDA[23:16]							
		W								
0x0146	DBGDAM	R	DBGDA[15:8]							
		W								
0x0147	DBGDAL	R	DBGDA[7:0]							
		W								
0x0148- 0x017F	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.5 0x0200-0x037F S12ZVCPIM

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0200	MODRR0	R	IIC0RR1-0		SCI1RR	SCI0RR	SPI0RR	M0C0RR2-0		
		W								
0x0201	MODRR1	R	T1IC3RR	T1IC2RR	0	0	0	TRIG0NEG	TRIG0RR1-0	
		W								
0x0202	MODRR2	R	P0C7RR	0	0	0	P0C3RR	0	0	0
		W								
0x0203	MODRR3	R	0	0	0	T0IC3RR1	T0IC3RR0	T0IC2RR	T0IC1RR	0
		W								
0x0204– 0x0207	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0208	ECLKCTL	R	NECLK	0	0	0	0	0	0	0
		W								
0x0209	IRQCR	R	IRQE	IRQEN	0	0	0	0	0	0
		W								
0x020A– 0x020D	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x020E	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x020F	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0210– 0x025F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0260	PTE	R	0	0	0	0	0	0	PTE1	PTE0
		W								
0x0261	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0262	PTIE	R	0	0	0	0	0	0	PTIE1	PTIE0
		W								
0x0263	Reserved	R	0	0	0	0	0	0	0	0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0264	DDRE	R	0	0	0	0	0	0	DDRE1	DDRE0
		W								
0x0265	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0266	PERE	R	0	0	0	0	0	0	PERE1	PERE0
		W								
0x0267	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0268	PPSE	R	0	0	0	0	0	0	PPSE1	PPSE0
		W								
0x0269– 0x027F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0280	PTADH	R	PTADH7	PTADH6	PTADH5	PTADH4	PTADH3	PTADH2	PTADH1	PTADH0
		W								
0x0281	PTADL	R	PTADL7	PTADL6	PTADL5	PTADL4	PTADL3	PTADL2	PTADL1	PTADL0
		W								
0x0282	PTIADH	R	PTIADH7	PTIADH6	PTIADH5	PTIADH4	PTIADH3	PTIADH2	PTIADH1	PTIADH0
		W								
0x0283	PTIADL	R	PTIADL7	PTIADL6	PTIADL5	PTIADL4	PTIADL3	PTIADL2	PTIADL1	PTIADL0
		W								
0x0284	DDRADH	R	DDRADH7	DDRADH6	DDRADH5	DDRADH4	DDRADH3	DDRADH2	DDRADH1	DDRADH0
		W								
0x0285	DDRADL	R	DDRADL7	DDRADL6	DDRADL5	DDRADL4	DDRADL3	DDRADL2	DDRADL1	DDRADL0
		W								
0x0286	PERADH	R	PERADH7	PERADH6	PERADH5	PERADH4	PERADH3	PERADH2	PERADH1	PERADH0
		W								
0x0287	PERADL	R	PERADL7	PERADL6	PERADL5	PERADL4	PERADL3	PERADL2	PERADL1	PERADL0
		W								
0x0288	PPSADH	R	PPSADH7	PPSADH6	PPSADH5	PPSADH4	PPSADH3	PPSADH2	PPSADH1	PPSADH0
		W								
0x0289	PPSADL	R	PPSADL7	PPSADL6	PPSADL5	PPSADL4	PPSADL3	PPSADL2	PPSADL1	PPSADL0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x028A–0x028B	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x028C	PIEADH	R W	PIEADH7	PIEADH6	PIEADH5	PIEADH4	PIEADH3	PIEADH2	PIEADH1	PIEADH0
0x028D	PIEADL	R W	PIEADL7	PIEADL6	PIEADL5	PIEADL4	PIEADL3	PIEADL2	PIEADL1	PIEADL0
0x028E	PIFADH	R W	PIFADH7	PIFADH6	PIFADH5	PIFADH4	PIFADH3	PIFADH2	PIFADH1	PIFADH0
0x028F	PIFADL	R W	PIFADL7	PIFADL6	PIFADL5	PIFADL4	PIFADL3	PIFADL2	PIFADL1	PIFADL0
0x0290–0x0297	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0298	DIENADH	R W	DIENADH7	DIENADH6	DIENADH5	DIENADH4	DIENADH3	DIENADH2	DIENADH1	DIENADH0
0x0299	DIENADL	R W	DIENADL7	DIENADL6	DIENADL5	DIENADL4	DIENADL3	DIENADL2	DIENADL1	DIENADL0
0x029A–0x02BF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02C0	PTT	R W	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
0x02C1	PTIT	R	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
		W								
0x02C2	DDRT	R W	DDRT7	DDRT6	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
0x02C3	PERT	R W	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
0x02C4	PPST	R W	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
0x02C5–0x02CE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02CF	Reserved	R	0	0	0	0	0	0	0	0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x02D0	PTS	R	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		W								
0x02D1	PTIS	R	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		W								
0x02D2	DDRS	R	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		W								
0x02D3	PERS	R	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		W								
0x02D4	PPSS	R	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		W								
0x02D5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02D6	PIES	R	PIES7	PIES6	PIES5	PIES4	PIES3	PIES2	PIES1	PIES0
		W								
0x02D7	PIFS	R	PIFS7	PIFS6	PIFS5	PIFS4	PIFS3	PIFS2	PIFS1	PIFS0
		W								
0x02D8– 0x02DE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02DF	WOMS	R	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		W								
0x02E0– 0x02EF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F0	PTP	R	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
		W								
0x02F1	PTIP	R	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
		W								
0x02F2	DDRP	R	DDRP7	DDRP6	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
		W								
0x02F3	PERP	R	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
		W								
0x02F4	PPSP	R	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSP0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x02F5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F6	PIEP	R	PIEP7	PIEP6	PIEP5	PIEP4	PIEP3	PIEP2	PIEP1	PIEP0
		W								
0x02F7	PIFP	R	PIFP7	PIFP6	PIFP5	PIFP4	PIFP3	PIFP2	PIFP1	PIFP0
		W								
0x02F8	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F9	OCPEP	R	0	OCPEP6	OCPEP5	OCPEP4	0	OCPEP2	0	OCPEP0
		W								
0x02FA	OCIEP	R	0	OCIEP6	OCIEP5	OCIEP4	0	OCIEP2	0	OCIEP0
		W								
0x02FB	OCIFP	R	0	OCIFP6	OCIFP5	OCIFP4	0	OCIFP2	0	OCIFP0
		W								
0x02FC	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02FD	RDRP	R	0	RDRP6	RDRP5	RDRP4	0	RDRP2	0	RDRP0
		W								
0x02FE–0x02FF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0300–0x030F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0310	PTJ	R	0	0	0	0	0	0	PTJ1	PTJ0
		W								
0x0311	PTIJ	R	0	0	0	0	0	0	PTIJ1	PTIJ0
		W								
0x0312	DDRJ	R	0	0	0	0	0	0	DDRJ1	DDRJ0
		W								
0x0313	PERJ	R	0	0	0	0	0	0	PERJ1	PERJ0
		W								
0x0314	PPSJ	R	0	0	0	0	0	0	PPSJ1	PPSJ0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0315–0x031E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x031F	WOMJ	R	0	0	0	0	0	0	WOMJ1	WOMJ0
		W								
0x0320–0x032F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0330	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0331	PTIL	R	0	0	0	0	0	0	PTIL1	PTIL0
		W								
0x0332	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0333	PTPSL	R	0	0	0	0	0	0	PTPSL1	PTPSL0
		W								
0x0334	PPSL	R	0	0	0	0	0	0	PPSL1	PPSL0
		W								
0x0335	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0336	PIEL	R	0	0	0	0	0	0	PIEL1	PIEL0
		W								
0x0337	PIFL	R	0	0	0	0	0	0	PIFL1	PIFL0
		W								
0x0338–0x0339	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x033A	PTABYPL	R	0	0	0	0	0	0	PTABYPL1	PTABYPL0
		W								
0x033B	PTADIRL	R	0	0	0	0	0	0	PTADIRL1	PTADIRL0
		W								
0x033C	DIENL	R	0	0	0	0	0	0	DIENL1	DIENL0
		W								
0x033D	PTAENL	R	0	0	0	0	0	0	PTAENL1	PTAENL0
		W								

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x033E	PIRL	R	0	0	0	0	0	0	PIRL1	PIRL0
		W								
0x033F	PTTEL	R	0	0	0	0	0	0	PTTEL1	PTTEL0
		W								
0x0340– 0x037F	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.6 0x0380-0x039F FTMRZ192K2K

Address	Name		7	6	5	4	3	2	1	0
0x0380	FCLKDIV	R	FDIVLD	FDIVLCK	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
		W								
0x0381	FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
		W								
0x0382	FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
		W								
0x0383	FPSTAT	R	FPOVRD	0	0	0	0	0	0	WSTAT ACK
		W								
0x0384	FCNFG	R	CCIE	0	ERSAREQ	IGNSF	WSTAT[1:0]		FDFD	FSFD
		W								
0x0385	FERCNFG	R	0	0	0	0	0	0	DFDIE	SFDIE
		W								
0x0386	FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
		W								
0x0387	FERSTAT	R	0	0	0	0	0	0	DFDIF	SFDIF
		W								
0x0388	FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
		W								
0x0389	DFPROT	R	DPOPEN	0	0	0	DPS3	DPS2	DPS1	DPS0
		W								
0x038A	FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
		W								
0x038B	FRSV1	R	0	0	0	0	0	0	0	0
		W								

N.6 0x0380-0x039F FTMRZ192K2K (continued)

Address	Name		7	6	5	4	3	2	1	0
0x038C	FCCOB0HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x038D	FCCOB0LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x038E	FCCOB1HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x038F	FCCOB1LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0390	FCCOB2HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0391	FCCOB2LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0392	FCCOB3HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0393	FCCOB3LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0394	FCCOB4HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0395	FCCOB4LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
0x0396	FCCOB5HI	R W	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
0x0397	FCCOB5LO	R W	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0

N.7 0x03C0-0x03CF SRAM_ECC_32D7P

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x03C0	ECCSTAT	R	0	0	0	0	0	0	0	RDY
		W								
0x03C1	ECCIE	R	0	0	0	0	0	0	0	SBEEIE
		W								
0x03C2	ECCIF	R	0	0	0	0	0	0	0	SBEEIF
		W								
0x03C3 - 0x03C6	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.7 0x03C0-0x03CF SRAM_ECC_32D7P

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x03C7	ECCDPTRH	R	DPTR[23:16]							
		W								
0x03C8	ECCDPTRM	R	DPTR[15:8]							
		W								
0x03C9	ECCDPTRL	R	DPTR[7:1]							0
		W								
0x03CA - 0x03CB	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x03CC	ECCDDH	R	DDATA[15:8]							
		W								
0x03CD	ECCDDL	R	DDATA[7:0]							
		W								
0x03CE	ECCDE	R	0	0	DECC[5:0]					
		W								
0x03CF	ECCDCMD	R	ECCDRR	0	0	0	0	0	ECCDW	ECCDR
		W								

N.8 0x0400-0x042F TIM1

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0400 TIOS	R					IOS3	IOS2	IOS1	IOS0
	W								
0x0401 CFORC	R	0	0	0	0	0	0	0	0
	W					FOC3	FOC2	FOC1	FOC0
0x0402 Reserved	R								
0x0403 Reserved	R								
0x0404 TCNTH	R	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
	W								
0x0405 TCNTL	R	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
	W								
0x0406 TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
	W								
0x0407 TTOV	R					TOV3	TOV2	TOV1	TOV0
	W								
0x0408 TCTL1	R								
	W								
0x0409 TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
	W								
0x040A TCTL3	R								
	W								
0x040B TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
	W								
0x040C TIE	R					C3I	C2I	C1I	C0I
	W								
0x040D TSCR2	R	TOI	0	0	0		PR2	PR1	PR0
	W								
0x040E TFLG1	R					C3F	C2F	C1F	C0F
	W								
0x040F TFLG2	R	TOF	0	0	0	0	0	0	0
	W								
0x0410–0x041F TCxH–TCxL[1]	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0420–0x042B Reserved	R								
	W								
0x042C OCPD	R					OCPD3	OCPD2	OCPD1	OCPD0
	W								
0x042D Reserved	R								
0x042E PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
	W								

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x042F Reserved	R								
	W								

[1] The register is available only if corresponding channel exists.

N.9 0x0480-0x04AF PWM0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0480	PWME	R	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
		W								
0x0481	PWMPOL	R	PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0
		W								
0x0482	PWMCLK	R	PCLK7	PCLKL6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0
		W								
0x0483	PWMPRCLK	R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
		W								
0x0484	PWMCAE	R	CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0
		W								
0x0485	PWMCTL	R	CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0
		W								
0x0486	PWMCLKAB	R	PCLKAB7	PCLKAB6	PCLKAB5	PCLKAB4	PCLKAB3	PCLKAB2	PCLKAB1	PCLKAB0
		W								
0x0487	RESERVED	R	0	0	0	0	0	0	0	0
		W								
0x0488	PWMSCLA	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0489	PWMSCLB	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x048A - 0x048B	RESERVED	R	0	0	0	0	0	0	0	0
		W								
0x048C	PWMCNT0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x048D	PWMCNT1	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x048E	PWMCNT2	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0

N.9 0x0480-0x04AF PWM0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x048F	PWMCNT3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0490	PWMCNT4	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0491	PWMCNT5	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0492	PWMCNT6	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0493	PWMCNT7	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0494	PWMPER0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0495	PWMPER1	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0496	PWMPER2	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0497	PWMPER3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0498	PWMPER4	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0499	PWMPER5	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049A	PWMPER6	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049B	PWMPER7	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049C	PWMDTY0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049D	PWMDTY1	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049E	PWMDTY2	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x049F	PWMDTY32	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

N.9 0x0480-0x04AF PWM0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x04A0	PWMDTY42	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x04A1	PWMDTY52	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x04A2	PWMDTY62	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x04A3	PWMDTY72	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x04A4 - 0x04AF	RESERVED	R	0	0	0	0	0	0	0	0
		W								

N.10 0x0500-0x052F PWM1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0500	PWME	R	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
		W								
0x0501	PWMPOL	R	PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0
		W								
0x0502	PWMCLK	R	PCLK7	PCLKL6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0
		W								
0x0503	PWMPRCLK	R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
		W								
0x0504	PWMCAE	R	CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0
		W								
0x0505	PWMCTL	R	CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0
		W								
0x0506	PWMCLKAB	R	PCLKAB7	PCLKAB6	PCLKAB5	PCLKAB4	PCLKAB3	PCLKAB2	PCLKAB1	PCLKAB0
		W								
0x0507	RESERVED	R	0	0	0	0	0	0	0	0
		W								
0x0508	PWMSCLA	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0509	PWMSCLB	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x050A - 0x050B	RESERVED	R	0	0	0	0	0	0	0	0
		W								

N.10 0x0500-0x052F PWM1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x050C	PWMCNT0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x050D	PWMCNT1	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x050E	PWMCNT2	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x050F	PWMCNT3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0510	PWMCNT4	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0511	PWMCNT5	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0512	PWMCNT6	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0513	PWMCNT7	R	Bit 7	6	5	4	3	2	1	Bit 0
		W	0	0	0	0	0	0	0	0
0x0514	PWMPER0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0515	PWMPER1	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0516	PWMPER2	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0517	PWMPER3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0518	PWMPER4	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0519	PWMPER5	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x051A	PWMPER6	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x051B	PWMPER7	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x051C	PWMDTY0	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

N.10 0x0500-0x052F PWM1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x051D	PWMDTY1	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x051E	PWMDTY2	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x051F	PWMDTY32	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0520	PWMDTY42	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0521	PWMDTY52	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0522	PWMDTY62	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0523	PWMDTY72	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0524 - 0x052F	RESERVED	R	0	0	0	0	0	0	0	0
		W								

N.11 0x05C0-0x05EF TIM0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05C0	TIM0TIOS	R	0	0	0	0	IOS3	IOS2	IOS1	IOS0
		W								
0x05C1	TIM0CFORC	R	0	0	0	0	0	0	0	0
		W					FOC3	FOC2	FOC1	FOC0
0x05C2	Reserved	R W								
0x05C3	Reserved	R W								
0x05C4	TIM0TCNTH	R W	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
0x05C5	TIM0TCNTL	R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
0x05C6	TIM0TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
		W								

N.11 0x05C0-0x05EF TIM0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05C7	TIM0TTOV	R	0	0	0	0	TOV3	TOV2	TOV1	TOV0
		W								
0x05C8	TIM0TCTL1	R	0	0	0	0	0	0	0	0
		W								
0x05C9	TIM0TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		W								
0x05CA	TIM0TCTL3	R	0	0	0	0	0	0	0	0
		W								
0x05CB	TIM0TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		W								
0x05CC	TIM0TIE	R	0	0	0	0	C3I	C2I	C1I	C0I
		W								
0x05CD	TIM0TSCR2	R	TOI	0	0	0		PR2	PR1	PR0
		W								
0x05CE	TIM0TFLG1	R	0	0	0	0	C3F	C2F	C1F	C0F
		W								
0x05CF	TIM0TFLG2	R	TOF	0	0	0	0	0	0	0
		W								
0x05D0	TIM0TC0H	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x05D1	TIM0TC0L	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x05D2	TIM0TC1H	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x05D3	TIM0TC1L	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x05D4	TIM0TC2H	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x05D5	TIM0TC2L	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x05D6	TIM0TC3H	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x05D7	TIM0TC3L	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x05D8– 0x05DF	Reserved	R								
		W								

N.11 0x05C0-0x05EF TIM0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05E0	Reserved	R								
		W								
0x05E1	Reserved	R								
		W								
0x05E2	Reserved	R								
		W								
0x05E3	Reserved	R								
		W								
0x05E4–0x05EB	Reserved	R								
		W								
0x05EC	TIM0OCPD	R	0	0	0	0	OCPD3	OCPD2	OCPD1	OCPD0
		W								
0x05ED	Reserved	R								
		W								
0x05EE	TIM0PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
		W								
0x05EF	Reserved	R								
		W								

N.12 0x0600-0x063F ADC0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0600	ADC0CTL_0	R	ADC_EN	ADC_SR	FRZ_MOD	SWAI	ACC_CFG[1:0]		STR_SEQA	MOD_CFG
		W								
0x0601	ADC0CTL_1	R	CSL_BMOD	RVL_BMOD	SMOD_ACC	AUT_RSTA	0	0	0	0
		W								
0x0602	ADC0STS	R	CSL_SEL	RVL_SEL	DBECC_ERR	Reserved	READY	0	0	0
		W								
0x0603	ADC0TIM	R	0	PRS[6:0]						
		W								
0x0604	ADC0FMT	R	DJM	0	0	0	0	SRES[2:0]		
		W								
0x0605	ADC0FLWCTL	R	SEQA	TRIG	RSTA	LDOK	0	0	0	0
		W								
0x0606	ADC0EIE	R	IA_EIE	CMD_EIE	EOL_EIE	Reserved	TRIG_EIE	RSTAR_EIE	LDOK_EIE	0
		W								
0x0607	ADC0IE	R	SEQAD_IE	CONIF_OIE	Reserved	0	0	0	0	0
		W								

N.12 0x0600-0x063F ADC0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0608	ADC0EiF	R	IA_EIF	CMD_EIF	EOL_EIF	Reserved	TRIG_EIF	RSTAR_EIF	LDOK_EIF	0
		W								
0x0609	ADC0IF	R	SEQAD_IF	CONIF_OIF	Reserved	0	0	0	0	0
		W								
0x060A	ADC0CONIE_0	R	CON_IE[15:8]							
		W								
0x060B	ADC0CONIE_1	R	CON_IE[7:1]							EOL_IE
		W								
0x060C	ADC0CONIF_0	R	CON_IF[15:8]							
		W								
0x060D	ADC0CONIF_1	R	CON_IF[7:1]							EOL_IF
		W								
0x060E	ADC0IMDRI_0	R	CSL_IMD	RVL_IMD	0	0	0	0	0	0
0x060F	ADC0IMDRI_1	R	0	RIDX_IMD						
		W								
0x0610	ADC0EOLRI	R	CSL_EOL	RVL_EOL	0	0	0	0	0	0
		W								
0x0611	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0612	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0613	Reserved	R	Reserved						0	0
		W								
0x0614	ADC0CMD_0	R	CMD_SEL		0	0	INTFLG_SEL[3:0]			
		W								
0x0615	ADC0CMD_1	R	VRH_SEL	VRL_SEL	CH_SEL[5:0]					
		W								
0x0616	ADC0CMD_2	R	SMP[4:0]					0	0	Reserved
		W								
0x0617	ADC0CMD_3	R	Reserved	Reserved	Reserved					
		W								
0x0618	Reserved	R	Reserved							
		W								
0x0619	Reserved	R	Reserved							
		W								
0x061A	Reserved	R	Reserved							
		W								
0x061B	Reserved	R	Reserved							
		W								
0x061C	ADC0CIDX	R	0	0	CMD_IDX[5:0]					
		W								
0x061D	ADC0CBP_0	R	CMD_PTR[23:16]							
		W								
0x061E	ADC0CBP_1	R	CMD_PTR[15:8]							
		W								
0x061F	ADC0CBP_2	R	CMD_PTR[7:2]						0	0
		W								

N.12 0x0600-0x063F ADC0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0620	ADC0RIDX	R	0	0	RES_IDX[5:0]					
		W	▨	▨	▨					
0x0621	ADC0RBP_0	R	0	0	0	0	RES_PTR[19:16]			
		W	▨	▨	▨	▨				
0x0622	ADC0RBP_1	R	RES_PTR[15:8]							
		W								
0x0623	ADC0RBP_2	R	RES_PTR[7:2]						0	0
		W							▨	▨
0x0624	ADC0CROFF0	R	0	CMDRES_OFF0[6:0]						
		W	▨	▨						
0x0625	ADC0CROFF1	R	0	CMDRES_OFF1[6:0]						
		W	▨							
0x0626	Reserved	R	0	0	0	0	Reserved			
		W	▨	▨	▨	▨				
0x0627	Reserved	R	Reserved							
		W								
0x0628	Reserved	R	Reserved						0	0
		W							▨	▨
0x0629	Reserved	R	Reserved	0	Reserved					
		W	▨	▨	▨					
0x062A- 0x063F	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

N.13 0x0680-0x0687 DAC8B5V

Address Offset Register Name		**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
0x0680 DACCTL	R	FVR	DRIVE	0	0	0	DACM[2:0]		
	W			▨	▨	▨			
0x0681 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0682 DACVOL	R	VOLTAGE[7:0]							
	W								
0x0683 - 0x0686 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0687 DACDEBUG	R W	0	BUF_EN	DAC_EN	S3	S2n	S2p	S1n	S1p

▨ = Unimplemented

N.14 0x0690-0x0697 ACMP0

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0690	ACMPC0	R	ACE	ACOPE	ACOPS	ACDLY	ACHYS1-0		ACMOD1-0	
		W								
0x0691	ACMPC1	R	0	0	ACPSEL1-0		0	0	ACNSEL1-0	
		W								
0x0692	ACMPC2	R	0	0	0	0	0	0	0	ACIE
		W								
0x0693	ACMPS	R	ACO	0	0	0	0	0	0	ACIF
		W								
0x0694– 0x0697	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.15 0x0698-0x069F ACMP1

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0698	ACMPC0	R	ACE	ACOPE	ACOPS	ACDLY	ACHYS1-0		ACMOD1-0	
		W								
0x0699	ACMPC1	R	0	0	ACPSEL1-0		0	0	ACNSEL1-0	
		W								
0x069A	ACMPC2	R	0	0	0	0	0	0	0	ACIE
		W								
0x069B	ACMPS	R	ACO	0	0	0	0	0	0	ACIF
		W								
0x069C– 0x069F	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.16 0x06C0-0x06DF CPMU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06C0	CPMU RESERVED00	R	0	0	0	0	0	0	0	0
		W								
0x06C1	CPMU RESERVED01	R	0	0	0	0	0	0	0	0
		W								
0x06C2	CPMU RESERVED02	R	0	0	0	0	0	0	0	0
		W								
0x06C3	CPMURFLG	R	0	PORF	LVRF	0	COPRF	0	OMRF	PMRF
		W								
0x06C4	CPMU SYNR	R	VCOFRQ[1:0]		SYNDIV[5:0]					
		W								
0x06C5	CPMU REFDIV	R	REFFRQ[1:0]		0	0	REFDIV[3:0]			
		W								
0x06C6	CPMU POSTDIV	R	0	0	0	POSTDIV[4:0]				
		W								
0x06C7	CPMUIFLG	R	RTIF	0	0	LOCKIF	LOCK	0	OSCIF	UPOSC
		W								
0x06C8	CPMUINT	R	RTIE	0	0	LOCKIE	0	0	OSCIE	0
		W								
0x06C9	CPMUCLKS	R	PLLSEL	PSTP	CSAD	COP OSCSEL1	PRE	PCE	RTI OSCSEL	COP OSCSEL0
		W								
0x06CA	CPMUPLL	R	0	0	FM1	FM0	0	0	0	0
		W								
0x06CB	CPMURTI	R	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
		W								
0x06CC	CPMUCOP	R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
		W			WRTMASK					
0x06CD	RESERVED CPMUTEST0	R	0	0	0	0	0	0	0	0
		W								
0x06CE	RESERVED CPMUTEST1	R	0	0	0	0	0	0	0	0
		W								
0x06CF	CPMU ARMCOP	R	0	0	0	0	0	0	0	0
		W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x06D0	CPMU HTCTL	R	0	0	VSEL	0	HTE	HTDS	HTIE	HTIF
		W								
0x06D1	CPMU LVCTL	R	0	0	0	0	0	LVDS	LVIE	LVIF
		W								
0x06D2	CPMU APICTL	R	APICLK	0	0	APIES	APIEA	APIFE	APIE	APIF
		W								

N.16 0x06C0-0x06DF CPMU (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06D3	CPMUACLKTR	R	ACLKTR5	ACLKTR4	ACLKTR3	ACLKTR2	ACLKTR1	ACLKTR0	0	0
		W								
0x06D4	CPMUAPIRH	R	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
		W								
0x06D5	CPMUAPIRL	R	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
		W								
0x06D6	RESERVED CPMUTEST3	R	0	0	0	0	0	0	0	0
		W								
0x06D7	CPMUHTTR	R	HTOE	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
		W								
0x06D8	CPMU IRCTRIMH	R	TCTRIM[4:0]					0	IRCTRIM[9:8]	
		W								
0x06D9	CPMU IRCTRIML	R	IRCTRIM[7:0]							
		W								
0x06DA	CPMUOSC	R	OSCE	Reserved	Reserved	Reserved				
		W								
0x06DB	CPMUPROT	R	0	0	0	0	0	0	0	PROT
		W								
0x06DC	RESERVED CPMUTEST2	R	0	0	0	0	0	0	0	0
		W								
0x06DD	CPMU VREGCTL	R	0	0	0	0	0	EXTCON	EXTXON	INTXON
		W								
0x06DE	CPMU RESERVED1E	R	0	0	0	0	0	0	0	0
		W								
0x06DF	CPMU RESERVED1F	R	0	0	0	0	0	0	0	0
		W								

N.17 0x06F0-0x06F7 BATS

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06F0	BATE	R	0	BVHS	BVLS[1:0]		BSUAE	BSUSE	0	0
		W								
0x06F1	BATSR	R	0	0	0	0	0	0	BVHC	BVLC
		W								
0x06F2	BATIE	R	0	0	0	0	0	0	BVHIE	BVLIE
		W								
0x06F3	BATIF	R	0	0	0	0	0	0	BVHIF	BVLIF
		W								
0x06F4 - 0x06F5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x06F6 - 0x06F7	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								

N.18 0x0700-0x0707 SCI0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0700	SCI0BDH[1]	R	SBR15	SBR14	SBR13	SBR12	SBR11	SBR10	SBR9	SBR8
		W								
0x0701	SCI0BDL[1]	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		W								
0x0702	SCI0CR1[1]	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		W								
0x0700	SCI0ASR1[2]	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
		W								
0x0701	SCI0ACR1[2]	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
		W								
0x0702	SCI0ACR2[2]	R	IREN	TNP1	TNP0	0	0	BERRM1	BERRM0	BKDFE
		W								
0x0703	SCI0CR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		W								
0x0704	SCI0SR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		W								
0x0705	SCI0SR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
		W								

N.18 0x0700-0x0707 SCI0 (continued)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0706	SCI0DRH	R	R8	T8	0	0	0	0	0	0
		W								
0x0707	SCI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

1 These registers are accessible if the AMAP bit in the SCISR2 register is set to zero.

2 These registers are accessible if the AMAP bit in the SCISR2 register is set to one.

N.19 0x0710-0x0717 SCI1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0710	SCI1BDH[1]	R	SBR15	SBR14	SBR13	SBR12	SBR11	SBR10	SBR9	SBR8
		W								
0x0711	SCI1BDL[1]	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		W								
0x0712	SCI1CR1[1]	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		W								
0x0710	SCI1ASR1[2]	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
		W								
0x0711	SCI1ACR1[2]	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
		W								
0x0712	SCI1ACR2[2]	R	IREN	TNP1	TNP0	0	0	BERRM1	BERRM0	BKDFE
		W								
0x0713	SCI1CR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		W								
0x0714	SCI1SR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		W								
0x0715	SCI1SR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
		W								
0x0716	SCI1DRH	R	R8	T8	0	0	0	0	0	0
		W								
0x0717	SCI1DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

1 These registers are accessible if the AMAP bit in the SCISR2 register is set to zero.

2 These registers are accessible if the AMAP bit in the SCISR2 register is set to one.

N.20 0x0780-0x0787 SPI0

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0780	SPI0CR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		W								
0x0781	SPI0CR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		W								
0x0782	SPI0BR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		W								
0x0783	SPI0SR	R	SPIF	0	SPTEF	MODF	0	0	0	0
		W								
0x0784	SPI0DRH	R	R15	R14	R13	R12	R11	R10	R9	R8
		W	T15	T14	T13	T12	T11	T10	T9	T8
0x0785	SPI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0
0x0786	Reserved	R								
		W								
0x0787	Reserved	R								
		W								

N.21 0x0790-0x0797 SPI1

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0790	SPI0CR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		W								
0x0791	SPI0CR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		W								
0x0792	SPI0BR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		W								
0x0793	SPI0SR	R	SPIF	0	SPTEF	MODF	0	0	0	0
		W								
0x0794	SPI0DRH	R	R15	R14	R13	R12	R11	R10	R9	R8
		W	T15	T14	T13	T12	T11	T10	T9	T8
0x0795	SPI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

N.21 0x0790-0x0797 SPI1

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0796	Reserved	R								
		W								
0x0797	Reserved	R								
		W								

N.22 0x07C0-0x07C7 IIC0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x07C0	IBAD	R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
		W								
0x07C1	IBFD	R	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
		W								
0x07C2	IBCR	R	IBEN	IBIE	MS/$\overline{\text{SL}}$	Tx/$\overline{\text{Rx}}$	TXAK	0	0	IBSWAI
		W						RSTA		
0x07C3	IBSR	R	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
		W								
0x07C4	IBDR	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
0x07C5	IBCR2	R	GCEN	ADTYPE	0	0	0	ADR10	ADR9	ADR8
		W								
0x07C6 - 0x07C7	Reserved	R	0	0	0	0	0	0	0	0
		W								

N.23 0x0800-0x083F CAN0

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0800 CANCTL0	R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
	W		▨		▨				
0x0001 CANCTL1	R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
	W							▨	▨
0x0802 CANBTR0	R	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
	W								
0x0803 CANBTR1	R	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
	W								
0x0804 CANRFLG	R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
	W			▨	▨	▨	▨		
0x0805 CANRIER	R	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
	W								
0x0806 CANTFLG	R	0	0	0	0	0	TXE2	TXE1	TXE0
	W	▨	▨	▨	▨	▨			
0x0807 CANTIER	R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
	W	▨	▨	▨	▨	▨			
0x0808 CANTARQ	R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
	W	▨	▨	▨	▨	▨			
0x0809 CANTAAK	R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x080A CANTBSEL	R	0	0	0	0	0	TX2	TX1	TX0
	W	▨	▨	▨	▨	▨			
0x080B CANIDAC	R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
	W	▨	▨			▨	▨	▨	▨
0x080C Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000D CANMISC	R	0	0	0	0	0	0	0	BOHOLD
	W	▨	▨	▨	▨	▨	▨	▨	
0x080E CANRXERR	R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x080F CANTXERR	R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0810–0x0813 CANIDAR0–3	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x0814–0x0817 CANIDMRx	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0818–0x081B CANIDAR4–7	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x081C–0x081F CANIDMR4–7	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0820–0x082F CANRXFG	R	See Section 17.3.3, “Programmer’s Model of Message Storage”							
	W								
0x0830–0x083F CANTXFG	R	See Section 17.3.3, “Programmer’s Model of Message Storage”							
	W								

▨ = Unimplemented or Reserved

N.24 0x0990-0x0997 CANPHY

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0990	CPDR	R	CPDR7	0	0	0	0	0	CPDR1	CPDR0
		W	▨	▨	▨	▨	▨	▨		▨
0x0991	CPCR	R	CPE	SPE	WUPE1-0		0	SLR2-0		
		W					▨			
0x0992	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0993	CPSR	R	CPCHVH	CPCHVL	CPCLVH	CPCLVL	CPDT	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x0994	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0995	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0996	CPIE	R	0	0	0	CPVFIE	CPDTIE	0	0	CPOCIE
		W	▨	▨	▨			▨	▨	
0x0997	CPIF	R	CHVHIF	CHVLIF	CLVHIF	CLVLIF	CPDTIF	0	CHOCIF	CLOCIF
		W						▨		

▨ = Unimplemented or Reserved

N.25 0x09A0-0x09AF SENTTX

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x09A0	STTICKRATE	R	0	0	PRE[13:8]					
		W	▨	▨						
		R	PRE[7:0]							
		W								
0x09A2	STPPULSE	R	PPEN	PPFIXED	0	0	0	PPCOUNT[10:8]		
		W			▨	▨	▨			
		R	PPCOUNT[7:0]							
		W								
0x09A4	STCONFIG	R	TXINIT	TXEN	0	0	0	DNIBBLECOUNT[2:0]		
		W			▨	▨	▨			
		R	0	0	0	OPTEDGE	SINGLE	CRCSCN	CRCLEG	CRCBYP
		W	▨	▨	▨					
0x09A6	STINTEN	R	0	0	0	PPREIE	TUIE	CSIE	TCIE	TBEIE
		W	▨	▨	▨					

▨ = Unimplemented or Reserved

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x09A7	STINTFLG	R	0	0	0	PPRE	TU	CS	TC	TBE
		W	▨	▨	▨					
0x09A8	STTXBUF	R W	STATCONF[3:0]				CRC[3:0]			
		R W	DATA0[3:0]				DATA1[3:0]			
0x09AA		R W	DATA2[3:0]				DATA3[3:0]			
		R W	DATA4[3:0]				DATA5[3:0]			
0x09AC – 0x09AF	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

How to Reach Us:

Home Page:

nxp.com

Web Support

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. ARM, AMBA, ARM Powered, Artisan, Cortex, Jazelle, Keil, SecurCore, Thumb, TrustZone, and μVision are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. ARM7, ARM9, ARM11, big.LITTLE, CoreLink, CoreSight, DesignStart, Mali, mbed, NEON, POP, Sensinode, Socrates, ULINK and Versatile are trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.
© 2019 NXP B.V.

Document Number: MC9S12ZVCRMV2
Rev. 2.1
March 19, 2019

OCEAN CHIPS
Океан Электроники
Поставка электронных компонентов

Компания «Океан Электроники» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Поставка оригинальных импортных электронных компонентов напрямую с производств Америки, Европы и Азии, а так же с крупнейших складов мира;
- Широкая линейка поставок активных и пассивных импортных электронных компонентов (более 30 млн. наименований);
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Помощь Конструкторского Отдела и консультации квалифицированных инженеров;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Поставка электронных компонентов под контролем ВП;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- При необходимости вся продукция военного и аэрокосмического назначения проходит испытания и сертификацию в лаборатории (по согласованию с заказчиком);
- Поставка специализированных компонентов военного и аэрокосмического уровня качества (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Actel, Aeroflex, Peregrine, VPT, Syfer, Eurofarad, Texas Instruments, MS Kennedy, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits, General Dynamics и др.);

Компания «Океан Электроники» является официальным дистрибьютором и эксклюзивным представителем в России одного из крупнейших производителей разъемов военного и аэрокосмического назначения **«JONHON»**, а так же официальным дистрибьютором и эксклюзивным представителем в России производителя высокотехнологичных и надежных решений для передачи СВЧ сигналов **«FORSTAR»**.

AVIC JONHON

«JONHON» (основан в 1970 г.)

Разъемы специального, военного и аэрокосмического назначения:

(Применяются в военной, авиационной, аэрокосмической, морской, железнодорожной, горно- и нефтедобывающей отраслях промышленности)

«FORSTAR» (основан в 1998 г.)

ВЧ соединители, коаксиальные кабели, кабельные сборки и микроволновые компоненты:

(Применяются в телекоммуникациях гражданского и специального назначения, в средствах связи, РЛС, а так же военной, авиационной и аэрокосмической отраслях промышленности).

FORSTAR®

Телефон: 8 (812) 309-75-97 (многоканальный)
Факс: 8 (812) 320-03-32
Электронная почта: ocean@oceanchips.ru
Web: http://oceanchips.ru/
Адрес: 198099, г. Санкт-Петербург, ул. Калинина, д. 2, корп. 4, лит. А