MC9S12ZVM-Family Reference Manual

HCS12 Microcontrollers

Rev. 1.3
20 JAN 2014
MC9S12ZVMRMV1

freescale.com

To provide the most up-to-date information, the document revision on the Internet is the most current. A printed copy may be an earlier revision. To verify you have the latest information available, refer to :

freescale.com.

The following revision history table summarizes changes contained in this document. The individual module sections contain revision history tables with more detailed information.

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
12 Dec 2013	1.2	Replaced generic 8-channel TIM section with specific 4-channel TIM section Textual enhancements and corrections throughout Updated electrical parameter section and added parameters for temperataures up to 175°C - Added Table A-5 - Merged Table A-8 and A-9 into Table A-9. Values updated. . - Table A-15. Parameter #2. max changed from 800uA to 1050uA - Table A-15. Inserted new C class parameter ISUPS at 85C. typ. 80uA - Appendices B,D and E. Updated parameter values based on characterization results. - Appendix C. Added parameter values for range above T=150°C - Table F-3. Merged rows 2a and 2b. Merged rows 6a and 6b. - Appendix G. Merged tables G-1 and G-2. - Tables H-1 and H-2 values updated.
20 JAN 2014	1.3	Updated Stop mode description for BDC enabled case Removed false reference to modified clock monitor assert frequency Updated electricals for 175°C Grade0 - Removed temperature range disclaimer from electrical parameter spec.footer - Added sentence above table A-3 - Table D-1. LINPHY parameters 12a and 12b replaced by 12a, 12b and 12c- - Table D-2. LINPHY wake up pulse over whole temperature range - Table E-1. FET gate charge spec. updated

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor 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 Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor 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 Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Chapter 1
Device Overview MC9S12ZVM-Family

Chapter 2
Port Integration Module (S12ZVMPIMV1)

Chapter 3
Memory Mapping Control (S12ZMMCV1)

Chapter 4
Interrupt (S12ZINTV0)

Chapter 5
Background Debug Controller (S12ZBDCV2)

Chapter 6
S12Z Debug (S12ZDBGV2) Module

Chapter 7
ECC Generation Module (SRAM_ECCV1)

Chapter 8
S12 Clock, Reset and Power Management Unit (S12CPMU_UHV_V6)

Chapter 9
Analog-to-Digital Converter (ADC12B_LBA_V1)

Chapter 10
Supply Voltage Sensor - (BATSV3)

Chapter 11
Timer Module (TIM16B4CV3) Block Description

Chapter 12
Freescale's Scalable Controller Area Network (S12MSCANV3)

Chapter 13
Programmable Trigger Unit (PTUV2)

Chapter 14
Pulse Width Modulator with Fault Protection (PMF15B6CV3)

Chapter 15
Serial Communication Interface (S12SCIV6)

Chapter 16
Serial Peripheral Interface (S12SPIV5)

Chapter 17
Gate Drive Unit (GDUV4)

Chapter 18
LIN Physical Layer (S12LINPHYV2)

Chapter 19
128 KB Flash Module (S12ZFTMRZ128K512V2)

Appendix A
MCU Electrical Specifications

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

Appendix C
ADC Electrical Specifications

Appendix D
LINPHY Electrical Specifications

Appendix E
GDU Electrical Specifications

Appendix F
NVM Electrical Parameters

Appendix G
BATS Electrical Specifications

Appendix H
SPI Electrical Specifications

Appendix I
MSCAN Electrical Specifications

Appendix J
Package Information

Appendix K
Ordering Information

Appendix L
Detailed Register Address Map

Chapter 1
Device Overview MC9S12ZVM-Family

Table 1-1. Revision History

Version Number	Revision Date	Sections Affected	Description of Changes
1.4	12.Dec.2012	Section 1.4.8 Section 1.4.9 Section 1.2.1 Figure 1-5	• Added LINPHY dominant timeout to feature list • Changed SCI baud rate select to 16-bit • Added S12ZVML32 to family • Added S12ZVM pinout
1.5	19.Jun.2013	Section 1.2.3 Section 1.4.8 Section 1.6.1	• Documented differences between 0N95G and 1N95G masksets • Updated LINPHY feature list • Updated Part ID table
1.6	03.Jan.2014	Section 1.2.2	• Removed false reference to modified clock monitor assert frequency

1.1 Introduction

The MC9S12ZVM-Family is an automotive 16-bit microcontroller family using the NVM + UHV technology that offers the capability to integrate 40 V analog components. This family reuses many features from the existing S12/S12X portfolio. The particular differentiating features of this family are the enhanced S12Z core, the combination of dual-ADC synchronized with PWM generation and the integration of “high-voltage” analog modules, including the voltage regulator (VREG), Gate Driver Unit (GDU) and a Local Interconnect Network (LIN) physical layer. These features enable a fully integrated single chip solution to drive up to 6 external power MOSFETs for BLDC or PMSM motor drive applications.

The MC9S12ZVM-Family includes error correction code (ECC) on RAM and flash memory, EEPROM for diagnostic or data storage, a fast analog-to-digital converter (ADC) and a frequency modulated phase locked loop (IPLL) that improves the EMC performance. The MC9S12ZVM-Family delivers an optimized solution with the integration of several key system components into a single device, optimizing system architecture and achieving significant space savings. The MC9S12ZVM-Family delivers all the advantages and efficiencies of a 16-bit MCU while retaining the low cost, power consumption, EMC, and code-size efficiency advantages currently enjoyed by users of existing S12(X) families. The MC9S12ZVM-Family is available in two different pinout options, both using the 64-pin LQFP-EP package to accommodate both LIN and CAN based applications. In addition to the I/O ports available in each module, further I/O ports are available with interrupt capability allowing wake-up from stop or wait modes.

The MC9S12ZVM-Family is a general-purpose family of devices suitable for a range of applications, including:

- 3-phase sensorless BLDC motor control for
  - Fuel pump

  - Water pump
  - Oil pump
  - A/C compressor
  - HVAC blower
  - Engine cooling fan
  - Electric vehicle battery cooling fan
- Brush DC motor control that need driving in 2 directions, along with PWM control for
  - Reversible wiper
  - Trunk opener

1.2 Features

This section describes the key features of the MC9S12ZVM-Family.

1.2.1 MC9S12ZVM-Family Member Comparison

Table 1-2 provides a summary of feature set differences within the MC9S12ZVM-Family. All other features are common to all MC9S12ZVM-Family members.

Table 1-2. S12ZVM -Family Differences

Feature	S12ZVML32	S12ZVM32	S12ZVML64	S12ZVMC64	S12ZVML128	S12ZVMC128
Flash (ECC)	32 KB	32 KB	64 KB	64 KB	128 KB	128 KB
EEPROM (ECC)	512 Bytes	–	512 Bytes	512 Bytes	512 Bytes	512 Bytes
RAM (ECC)	2 KB	2 KB	4 KB	4 KB	8 KB	8 KB
LIN Physical layer	1	–	1	–	1	–
CAN VREG	–	–	–	1	–	1
SCI	2	1(1)	2	2	2	2
SPI	1	1	1	1	1	1
ADC channels	4+5	4+5	4+5	4+5	4+5	4+5
PMF channels	6	6	6	6	6	6
TIM channels	4	4	4	4	4	4
MSCAN(2)	–	–	1	1	1	1

1. Options featuring a single SCI include the SCI1 instantiation

2. External CAN physical interface required

1.2.2 Functional Differences Between N06E and 0N95G Masksets

NOTE

N95G also includes bug fixes that are not listed here because they do not constitute

specification changes. Please refer to the Mask Set Errata documents for details.

- Device Level
  - Changed BDC fast clock source from core clock to bus clock
  - Added exposed pad electrical connection to die VSS.
  - Device level current injection immunity improved
  - GDU register address range changed
  - Removed mapping of VRL to PAD7
  - Added ADC reference voltages to IFR
  - Increased over voltage detect thresholds to allow operating range up to 26.5V
- GDU
  - Added status flags for overvoltage on HD pin and low voltage on VLS
  - Blanking time: start internal blanking time generator with HGx/LGx instead of PWM signal
  - Added over current shutdown feature
  - Added low pass filter to desaturation comparators
- SCI V6 replaces V5
  - Enhanced baud rate options
- LINPHY
  - Direct Power Injection (DPI) robustness improvements
  - TX dominant timeout feature
  - Internal pull-up adjusted to stay in 27KOhm to 40KOhm range
- OSC, CPMU:
  - Added full swing Pierce mode
  - Added a configuration bit OMRE (Oscillator Monitor Reset Enable) that will enable the Monitor Reset. By default, clock monitor reset disabled (OMRE=0).
- PTU:
  - Allow swapping the trigger list at every reload event with load_ok active
  - Made the TG0LIST and TG1LIST writable if the associated TG0/TG1 is disabled
  - Allow SW to clear the PTULDOK bit when the PTU is disabled
- FTMRZ:
  - Added wait state configuration option bits for bus accesses
  - Removed interdependency of DFDF and SFDIF bits
  - Changed FTMRZ behavior when forbidden simultaneous P-flash/D-flash operations occur
- DBG:
  - Added register access restrictions when DBG is disarmed but a profiling transmission is still active
  - Added a register bit to indicate that the profiling transmission is still active
- BDC
  - Improved handling of attempted internal accesses during STOP mode

1.2.3 Functional Differences Between 1N95G and 0N95G Masksets

- GDU version changed from V2 to V4
  - Added low side driver shutdown flexibility in overvoltage case
    - Switched off if overvoltage condition and GOCA1=1
    - Switched on if overvoltage condition and GOCA1=0
  - Changed time constant of HD overvoltage monitor to improve noise filtering

1.3 Chip-Level Features

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

- S12Z CPU core
- 128, 64 or 32 KB on-chip flash with ECC
- 512 byte EEPROM with ECC
- 8, 4 or 2 KB 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
- 6-channel, 15-bit pulse width modulator with fault protection (PMF)
- Low side and high side FET pre-drivers for each phase
  - Gate drive pre-regulator
  - LDO (Low Dropout Voltage Regulator) (typically 11V)
  - High side gate supply generated using bootstrap circuit with external diode and capacitor
  - Sustaining charge pump with two external capacitors and diodes
  - High side drain (HD) monitoring on internal ADC channel using HD/5 voltage
- Two parallel analog-to-digital converters (ADC) with 12-bit resolution and up to 9 channels available on external pins
- Programmable Trigger Unit (PTU) for synchronization of PMF and ADC
- One serial peripheral interface (SPI) module
- One serial communication interface (SCI) module with interface to internal LIN physical layer transceiver (with RX connected to a timer channel for frequency calibration purposes, if desired)
- Up to one additional SCI (not connected to LIN physical layer)
- One on-chip LIN physical layer transceiver fully compliant with the LIN 2.2 standard
- 4-channel timer module (TIM) with input capture/output compare
- MSCAN (1 Mbit/s, CAN 2.0 A, B software compatible) module
- On-chip voltage regulator (VREG) for regulation of input supply and all internal voltages
  - Optional VREG ballast control output to supply an external CAN physical layer
- Two current sense circuits for overcurrent detection or torque measurement

- Autonomous periodic interrupt (API)
- 20mA high-current output for use as Hall sensor supply
- Supply voltage sense with low battery warning
- Chip temperature sensor

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, of 16MB linear address space
- Instructions and Addressing modes optimized for C-Programming & Compiler
- Optimized address path so it is capable to run at 50MHz without Flash wait states
    - 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)

- Background debug controller (BDC) with single-wire interface
    - Non-intrusive memory access commands
    - Supports in-circuit programming of on-chip nonvolatile memory

1.4.1.2 Debugger (DBG)

- Enhanced DBG module including:
    - Four comparators (A, B, C and D) each configurable to monitor PC addresses or addresses of data accesses
    - A and C compare full address bus and full 32-bit data bus with data bus mask register
    - B and D compare full address bus only

  - Three modes: simple address/data match, inside address range, or outside address range
  - Tag-type or force-type hardware breakpoint requests
- State sequencer control
- 64 x 64-bit circular trace buffer to capture change-of-flow addresses or address and data of every access
  - Begin, End and Mid alignment of tracing to trigger
- Profiling mode for external visibility of internal program flow

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 MC9S12ZVM-family on the features the following:

- Up to128 KB of program flash memory
  - 32 data bits plus 7 syndrome ECC (error correction code) bits allow single bit fault correction and double fault detection
  - Erase sector size 512 bytes
  - Automated program and erase algorithm
  - User margin level setting for reads
  - Protection scheme to prevent accidental program or erase

1.4.2.3 EEPROM

- Up to 512 bytes 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

- Up to 8 KB of general-purpose RAM with ECC
  - Single bit error correction and double bit error detection

1.4.3 Clocks, Reset & 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 4-20 MHz crystal oscillator/resonator

1.4.3.2 Internal RC Oscillator (IRC)

- Trimmable internal 1MHz reference clock.
  - Trimmed accuracy over -40°C to 150°C junction temperature range: ±1.3%max.

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
  - Trans conductance sized for optimum start-up margin for typical crystals
  - Oscillator pins shared with GPIO functionality

1.4.5 Timer (TIM)

- 4 x 16-bit channels Timer module for input capture or output compare
- 16-bit free-running counter with 8-bit precision prescaler

1.4.6 Pulse width Modulator with Fault protection (PMF)

- 6 x 15-bit channel PWM resolution
- Each pair of channels can be combined to generate a PWM signal (with independent control of edges of PWM signal)
- Dead time insertion available for each complementary pair
- Center-aligned or edge-aligned outputs
- Programmable clock select logic with a wide range of frequencies
- Programmable fault detection

1.4.7 Programmable Trigger Unit (PTU)

- Enables synchronization between PMF and ADC
- 2 trigger input sources and software trigger source
- 2 trigger outputs
- One 16-bit delay register pre-trigger output
- Operation in One-Shot or Continuous modes

1.4.8 LIN physical layer transceiver

- Compliant with LIN Physical Layer 2.2 specification.
- Compliant with the SAE J2602-2 LIN standard.
- Standby mode with glitch-filtered wake-up.
- Slew rate selection optimized for the baud rates: 10.4kBit/s, 20kBit/s and Fast Mode (up to 250kBit/s).
- Switchable 34kW/330kW pull-ups (in shutdown mode, 330kW only)
- Current limitation for LIN Bus pin falling edge.
- Over-current protection.
- LIN TxD-dominant timeout feature monitoring the LPTxD signal.
- Automatic transmitter shutdown in case of an over-current or TxD-dominant timeout.

- Fulfills the OEM "Hardware Requirements for LIN (CAN and FlexRay) Interfaces in Automotive Applications" v1.3.

1.4.9 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
- 16-bit baud rate selection
- Programmable character length
- Programmable polarity for transmitter and receiver
- Active edge receive wakeup
- Break detect and transmit collision detect supporting LIN

1.4.10 Multi-Scalable Controller Area Network (MSCAN)

- Implementation of the CAN protocol — Version 2.0A/B
- 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 either 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter

1.4.11 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

1.4.12 Analog-to-Digital Converter Module (ADC)

- Dual ADC
    - 12-bit resolution
    - Up to 9 external channels & 8 internal channels
    - 2.5us for single 12-bit resolution conversion
    - 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, TempSense, GDU phase, GDU DC-link
- External pins can also be used as digital I/O

1.4.13 Supply Voltage Sensor (BATS)

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

1.4.14 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 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 domain
- Package option for VREG ballast control output to supply external CANPHY

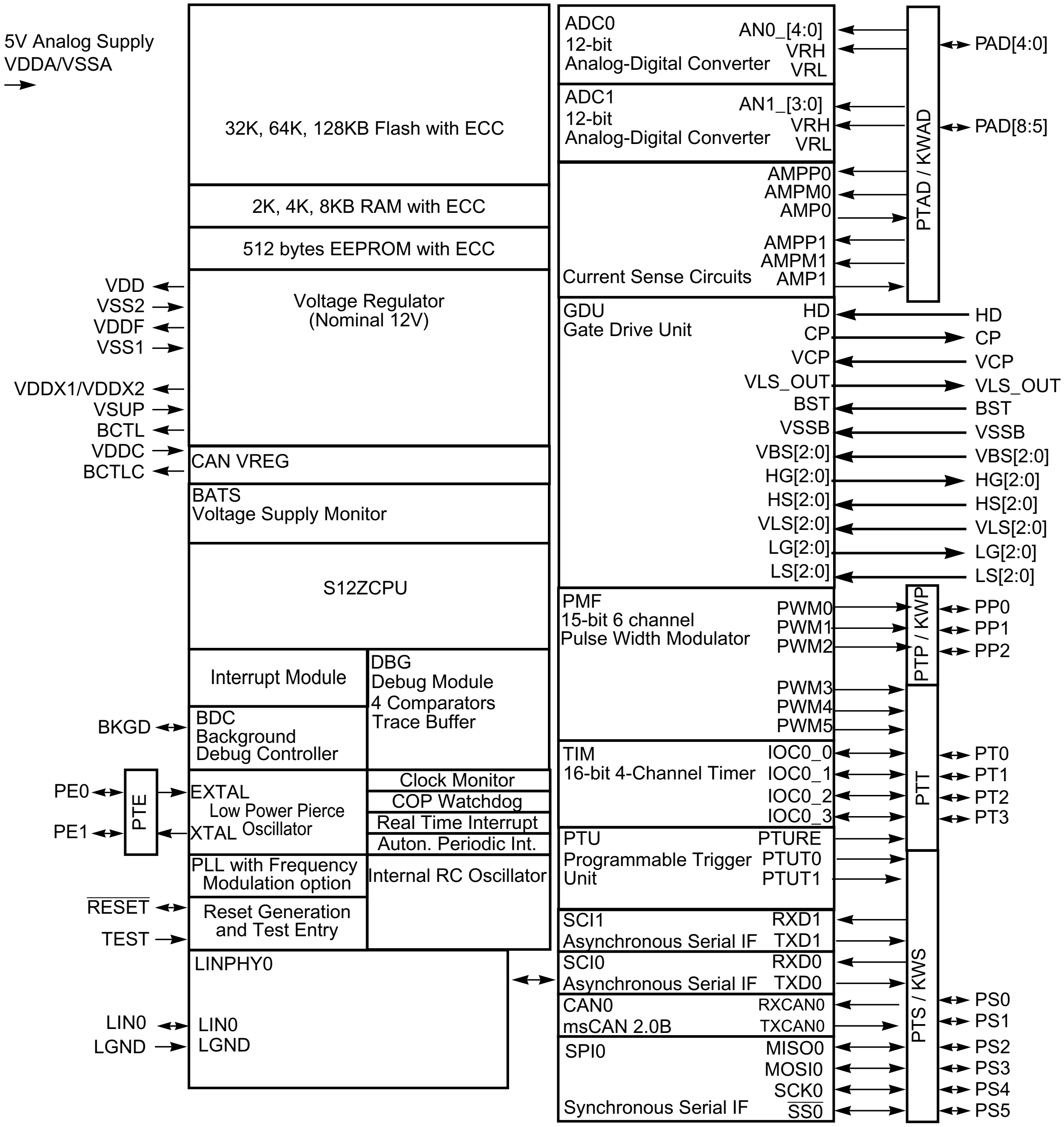
1.4.15 Gate Drive Unit (GDU)

- Low side and high side FET pre-drivers for each phase
- Gate drive pre-regulator LDO (Low Dropout Voltage Regulator)
- High side gate supply done via bootstrap circuit with external diode and capacitor
- Sustaining charge pump with two external capacitors and diodes
- Optional boost convertor configuration with voltage feedback
- FET-Predriver desaturation and error recognition
- Monitoring of FET High Side drain (HD) voltage
- Diagnostic failure management

1.4.16 Current Sense

- 2 channel, integrated op-amp functionality

1.5 Block Diagram



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 1-1. MC9S12ZVM-Family Block Diagram

1.6 Device Memory Map

Table 1-3 shows the device register memory map. All modules that can be instantiated more than once on S12 devices are listed with an index number, even if they are only instantiated once on this device family.

Table 1-3. Module Register Address Ranges

Address	Module	Size (Bytes)
0x0000–0x0003	Part ID Register Section 1.6.1	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
0x0200–0x02FF	PIM	256
0x0300–0x037F	Reserved	128
0x0380–0x039F	FTMRZ	32
0x03A0–0x03BF	Reserved	32
0x03C0–0x03CF	RAM ECC	16
0x03D0–0x04FF	Reserved	304
0x0500–0x053F	PMF	64
0x0540–0x057F	Reserved	64
0x0580–0x059F	PTU	32
0x05A0–0x05BF	Reserved	32
0x05C0–0x05EF	TIM0	48
0x05F0–0x05FF	Reserved	16
0x0600–0x063F	ADC0	64
0x0640–0x067F	ADC1	64
0x0680–0x069F	Reserved	32
(1)0x06A0–0x06BF	GDU	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

Table 1-3. Module Register Address Ranges

Address	Module	Size (Bytes)
0x0718–0x077F	Reserved	104
0x0780–0x0787	SPI0	8
0x0788–0x07FF	Reserved	120
0x0800–0x083F	CAN0	64
0x0840–0x097F	Reserved	320
0x0980–0x0987	LINPHY	8
0x0988–0x0FFF	Reserved	1656

1. Address range = 0x0690-0x069F on Maskset N06E

NOTE

Reserved register space shown above 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.

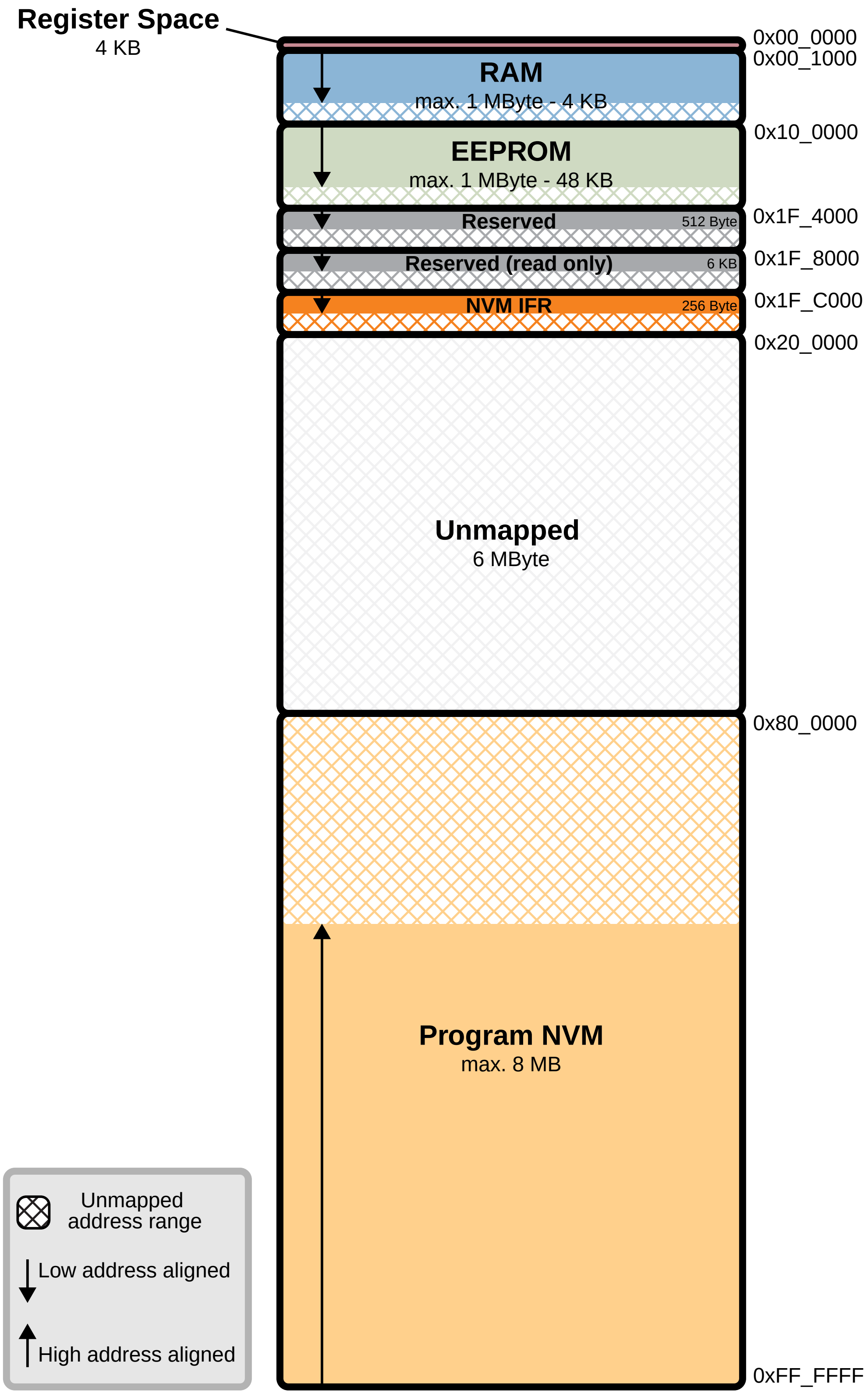


Figure 1-2. MC9S12ZVM-Family Global Memory Map. (See Table 1-2 for individual device details)

1.6.1 Part ID Assignments

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

Table 1-4. Assigned Part ID Numbers

Device	Mask Set Number	Part ID	Bonding Option
MC9S12ZVM128	N06E	0x00170000	LIN
MC9S12ZVM128	N06E	0x00170001	CAN-VREG
MC9S12ZVM128	N56G	0x00171000	LIN
MC9S12ZVM128(1)	N56G	0x00171xxx	CAN-VREG / LIN
MC9S12ZVM128	0N95G	0x00172000	LIN
MC9S12ZVM128	0N95G	0x00172001	CAN-VREG
MC9S12ZVM128	1N95G	0x00172100	LIN
MC9S12ZVM128	1N95G	0x00172101	CAN-VREG
MC9S12ZVM32	TBD	0x00150000	LIN

1. This version for Freescale internal engineering puirposes only

1.7 Signal Description and Device Pinouts

This section describes signals that connect off-chip. It includes pin out diagrams a table of signal properties, and detailed discussion of signals. Internal inter module signal mapping at device level is described in 1.8 Internal Signal Mapping.

1.7.1 Pin Assignment Overview

Table 1-5 provides a summary of which ports are available.

Table 1-5. Port Availability by Package Option

Port	64 LQFP
Port AD	PAD[8:0]
Port E	PE[1:0]
Port P	PP[2:0]
Port S	PS[5:0]
Port T	PT[3:0]
sum of ports	24

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

1.7.2 Detailed External Signal Descriptions

This section describes the properties of signals available at device pins. Signal names associated with modules that can be instantiated more than once on an S12 are indexed, even if the module is only instantiated once on the **MC9S12ZVM-Family.** If a signal already includes a channel number, then the index is inserted before the channel number. Thus ANx_y corresponds to AN instance x, channel number y.

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.

1.7.2.3 MODC — Mode C Signal

The MODC signal is used as an 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.

1.7.2.4 PAD[8:0] / KWAD[8:0] — Port AD, Input Pins of ADC

PAD[8:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWAD[8:0]). These signals can have a pull-up or pull-down device selected and enabled on per signal basis. During and out of reset the pull devices are disabled.

1.7.2.5 PE[1:0] — Port E I/O Signals

PE[1:0] are general-purpose input or output signals. The signals can have a pull-down device, enabled by on a per pin basis. Out of reset the pull-down devices are enabled.

1.7.2.6 PP[2:0] / KWP[2:0] — Port P I/O Signals

PP[2:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWP[2:0]). They can have a pull-up or pull-down device selected and enabled on per signal basis. During and out of reset the pull devices are disabled.

1.7.2.7 PS[5:0] / KWS[5:0] — Port S I/O Signals

PS[5:0] are general-purpose input or output signals. The signals can be configured on per signal basis as interrupt inputs with wake-up capability (KWS[5:0]). They can have a pull-up or pull-down device selected and enabled on per signal basis. During and out of reset the pull-up devices are enabled.

1.7.2.8 PT[3:0] — Port T I/O Signals

PT[3:0] are general-purpose input or output signals. They can have a pull-up or pull-down device selected and enabled on per signal basis. During and out of reset the pull devices are disabled.

1.7.2.9 AN0_[4:0], AN1_[3:0]— ADC Input Signals

These are the analog inputs of the Analog-to-Digital Converters. ADC0 has 5 analog input channels connected to PAD port pins. ADC1 has 4 analog input channels connected to PAD port pins.

1.7.2.10 VRH, VRL — ADC Reference Signals

VRH and VRL are the reference voltage input pins for the analog-to-digital converter.

1.7.2.11 SPI0 Signals

1.7.2.11.1 $\overline{\text{SS0}}$ Signal

This signal is associated with the slave select SS functionality of the serial peripheral interface SPI0.

1.7.2.11.2 SCK0 Signal

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

1.7.2.11.3 MISO0 Signal

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

1.7.2.11.4 MOSI0 Signal

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

1.7.2.12 SCI[1:0] Signals

1.7.2.12.1 RXD[1:0] Signals

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

1.7.2.12.2 TXD[1:0] Signals

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

1.7.2.13 CAN0 Signals

1.7.2.13.1 RXCAN0 Signal

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

1.7.2.13.2 TXCAN0 Signal

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

1.7.2.14 Timer IOC0_[3:0] Signals

The signals IOC0_[3:0] are associated with the input capture or output compare functionality of the timer (TIM0) module.

1.7.2.15 PWM[5:0] Signals

The signals PWM[5:0] are associated with the PMF module digital channel outputs.

1.7.2.16 PTU Signals

1.7.2.16.1 PTUT[1:0] Signals

These signals are the PTU trigger output signals. These signals are routed to pins for debugging purposes.

1.7.2.16.2 PTURE Signal

This signal is the PTU reload enable output signal. This signal is routed to a pin for debugging purposes.

1.7.2.17 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.

1.7.2.18 Oscillator and Clock Signals

1.7.2.18.1 Oscillator Pins — EXTAL and XTAL

EXTAL and XTAL are the crystal driver and external clock pins. On reset all the device clocks are derived from the internal PLLCLK, independent of EXTAL and XTAL. XTAL is the oscillator output.

1.7.2.18.2 ECLK

This signal is associated with the output of the 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.

1.7.2.19 BDC and Debug Signals

1.7.2.19.1 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.

1.7.2.19.2 PDO — Profiling Data Output

This is the profiling data output signal used when the DBG module profiling feature is enabled. This signal is output only and provides a serial, encoded data stream that can be used by external development tools to reconstruct the internal CPU code flow.

1.7.2.19.3 PDOCLK — Profiling Data Output Clock

This is the PDO clock signal used when the DBG module profiling feature is enabled. This signal is output only. During code profiling this is the clock signal that can be used by external development tools to sample the PDO signal.

1.7.2.19.4 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, or trace buffer entry, or to gate trace buffer entries. 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.

1.7.2.20 FAULT5 — External Fault Input

This is the PMF fault input signal, with configurable polarity, that can be used to disable PMF operation when asserted. Asynchronous shutdown of the GDU outputs HG[2:0] and LG[2:0] is not supported. Select QSMPm[1:0] > 0 in PMF.

1.7.2.21 LIN Physical Layer Signals

1.7.2.21.1 LIN0

This pad is connected to the single-wire LIN data bus. This signal is only available on S12ZVML versions.

1.7.2.21.2 LP0TXD

This is the LIN physical layer transmitter input signal.

1.7.2.21.3 LP0RXD

This is the LIN physical layer receiver output signal.

1.7.2.21.4 LP0DR1

This is the LIN LP0DR1 register bit, visible at the designated pin for debug purposes.

1.7.2.22 Gate Drive Unit (GDU) Signals

These are associated with driving the external FETs.

1.7.2.22.1 HD — FET predriver High side Drain Input

This is the drain connection of the external high-side FETs. The voltage present at this input is scaled down by an internal voltage divider, and can be routed to the internal ADC via an analog multiplexer.

This is also used as the LINPHY supply, VLINSUP.

1.7.2.22.2 VBS[2:0] - Bootstrap Capacitor Connections

These signals are the bootstrap capacitor connections for phases HS[2:0]. The capacitor connected between HS[2:0] and these signals provides the gate voltage and current to drive the external FET.

1.7.2.22.3 HG[2:0] - High-Side Gate signals

The pins are the gate drives for the three high-side power FETs. The drivers provide a high current with low impedance to turn on and off the high-side power FETs.

1.7.2.22.4 HS[2:0] - High-Side Source signals

The pins are the source connection for the high-side power FETs and the drain connection for the low-side power FETs. The low voltage end of the bootstrap capacitor is also connected to this pin.

1.7.2.22.5 VLS[2:0] - Voltage Supply for Low -Side Drivers

The pins are the voltage supply pins for the three low-side FET pre-drivers. This pins should be connected to the voltage regulator output pin VLS_OUT.

1.7.2.22.6 LG[2:0] - Low-Side Gate signals

The pins are the gate drives for the low-side power FETs. The drivers provide a high current with low impedance to turn on and off the low-side power FETs.

1.7.2.22.7 LS[2:0] - Low-Side Source signals

The pins are the low-side source connections for the low-side power FETs. The pins are the power ground pins used to return the gate currents from the low-side power FETs.

1.7.2.22.8 CP - Charge Pump Output signal

This pin is the switching node of the charge pump circuit. The supply voltage for charge pump driver is the output of the voltage regulator VLS_OUT. The output voltage of this pin switches typically between 0V and 11V. Must be left unconnected if not used.

1.7.2.22.9 VCP - Charge Pump Input for High-Side Driver Supply

This is the charge pump input for the FET high-side gate drive supply circuit. The pin must be left unconnected if not used.

1.7.2.22.10 BST - Boost signal

This pin provides the basic switching elements required to implement a boost converter for low battery voltage conditions. This requires external diodes, capacitors and a coil. This pin must be left unconnected if not used.

1.7.2.22.11 VSSB - Boost Ground signal

This pin is a separate ground pin for the on chip boost converter switching device.

1.7.2.22.12 VLS_OUT - 11V Voltage Regulator Output

This pin is the output of the integrated voltage regulator. The output voltage is typically V_{VLS}=11V. The input voltage to the voltage regulator is the VSUP pin.

1.7.2.22.13 AMPP[1:0] - Current Sense Amplifier Non-Inverting Input

These are the current sense amplifier non-inverting inputs.

1.7.2.22.14 AMPM[1:0] - Current Sense Amplifier Inverting Input

These are the current sense amplifier inverting inputs.

1.7.2.22.15 AMP[1:0] - Current Sense Amplifier Output

These are the current sense amplifier outputs.

1.7.2.23 CAN Physical Interface Support

The MCU can supply an external CAN physical interface device directly, thus removing the need for an external voltage regulator.

1.7.2.23.1 BCTLC

BCTLC provides the base current of an external bipolar that supplies an external CAN physical interface. This signal is only available on S12ZVMC versions.

1.7.2.23.2 VDDC

VDDC is the CANPHY supply. This is the output voltage of the external bipolar, fed back to the MCU. This signal is only available on S12ZVMC versions.

1.7.2.24 High Current Output — EVDD1

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.

1.7.2.25 BCTL

BCTL is the ballast connection for the on chip voltage regulator. It provides the base current of an external bipolar for the VDDX and VDDA supplies.

1.7.3 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.

1.7.3.1 VDDX1, VDDX2, VSSX1 — Digital I/O Power and Ground Pins

VDDX1, VDDX2 are voltage regulator outputs to supply the digital I/O drivers.

The VSSX1 pin is the ground pin for the digital I/O drivers.

Bypass requirements on VDDX2, VDDX1, VSSX1 depend on how heavily the MCU pins are loaded.

1.7.3.2 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.

1.7.3.3 VDD, VSS2 — Core Power and Ground Pin

The VDD voltage supply of nominally 1.8V is generated by the internal voltage regulator. The return current path is through the VSS2 pin.

1.7.3.4 VDDF, VSS1 — NVM Power and Ground Pin

The VDDF voltage supply of nominally 2.8V is generated by the internal voltage regulator. The return current path is through the VSS1 pin.

1.7.3.5 LGND — LINPHY Ground Pin

LGND is the ground pin for the LIN physical layer LINPHY. This signal is only available on S12ZVM(L) versions, for which it must be connected to board ground, even if the LINPHY is not used.

1.7.3.6 VSUP — Voltage Supply Pin for Voltage Regulator

VSUP is the main supply pin typically coming from the car battery/alternator in the 12V supply voltage range. 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.

1.7.4 Package and Pinouts

The following package options are offered.

- 64LQFP-EP (exposed pad) with internal LIN PHY.
- 64LQFP-EP (exposed pad) without internal LIN PHY but with CAN VREG to support the addition of a low cost external CAN PHY.

The exposed pad must be connected to a grounded contact pad on the PCB.

The exposed pad has an electrical connection within the package to VSSFLAG (VSSX die connection).

The pin out details are shown in the following diagrams. Signals in brackets denote routing options.

NOTE

For the S12ZVM32 derivative the pins 1 and 64 are unused. Pin 64 must be connected to ground and pin1 left unconnected.

The exposed pad on the package bottom must be connected to a grounded contact pad on the PCB.

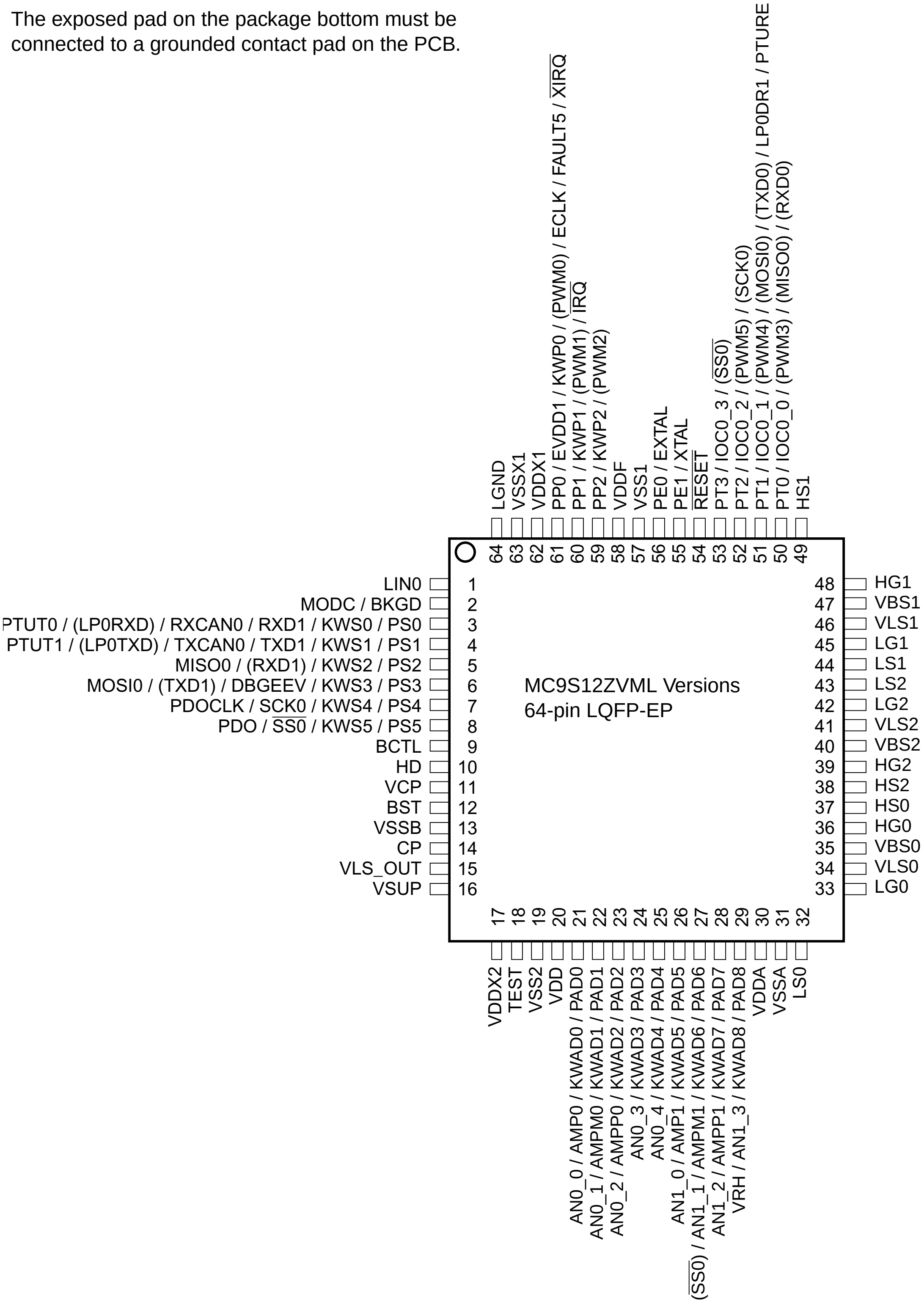


Figure 1-3. MC9S12ZVM-Family 64-pin LQFP pin out - LIN PHY option

The exposed pad on the package bottom must be connected to a grounded contact pad on the PCB.

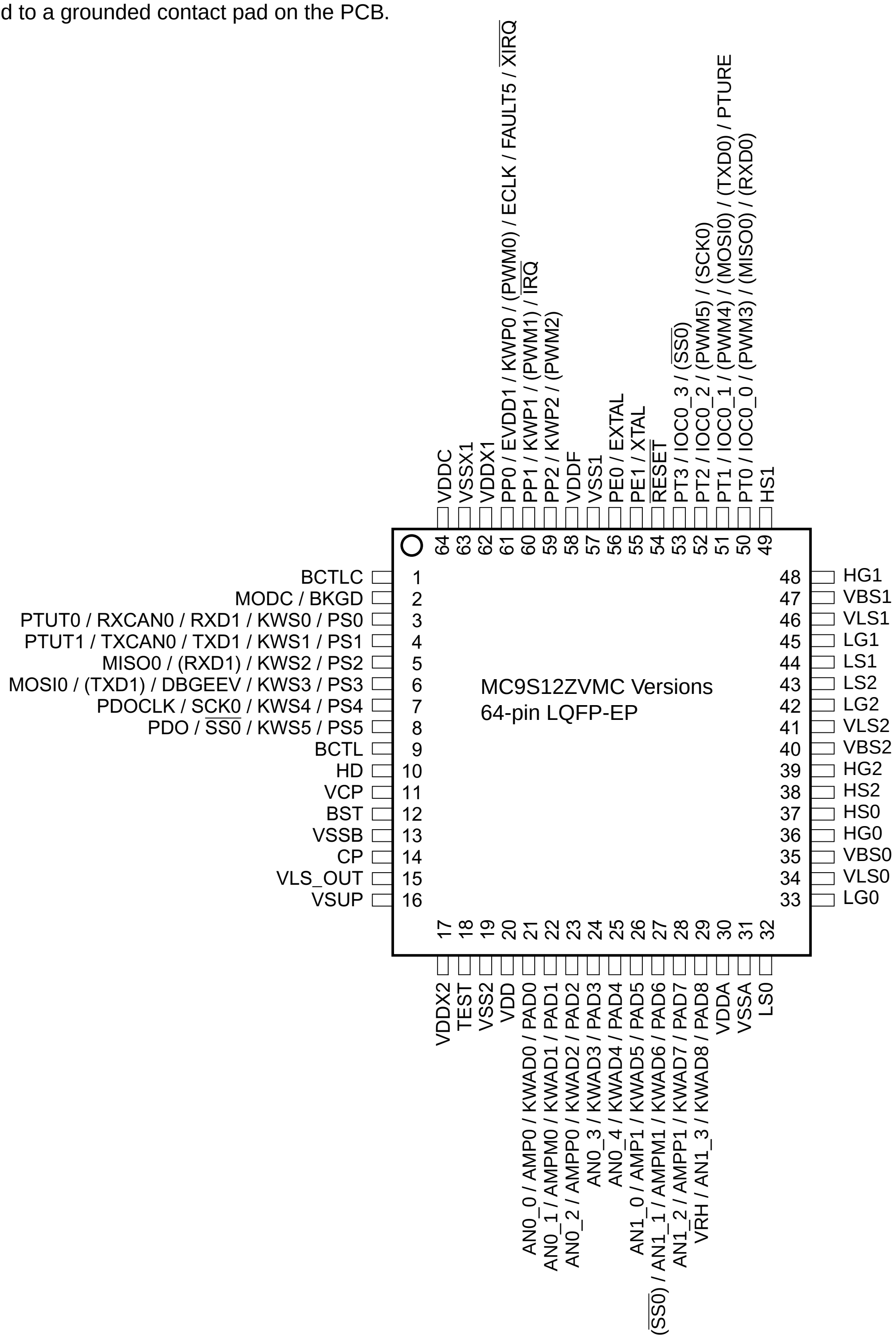


Figure 1-4. MC9S12ZVM-Family 64-pin LQFP pin out - External CAN PHY option

The exposed pad on the package bottom must be connected to a grounded contact pad on the PCB.

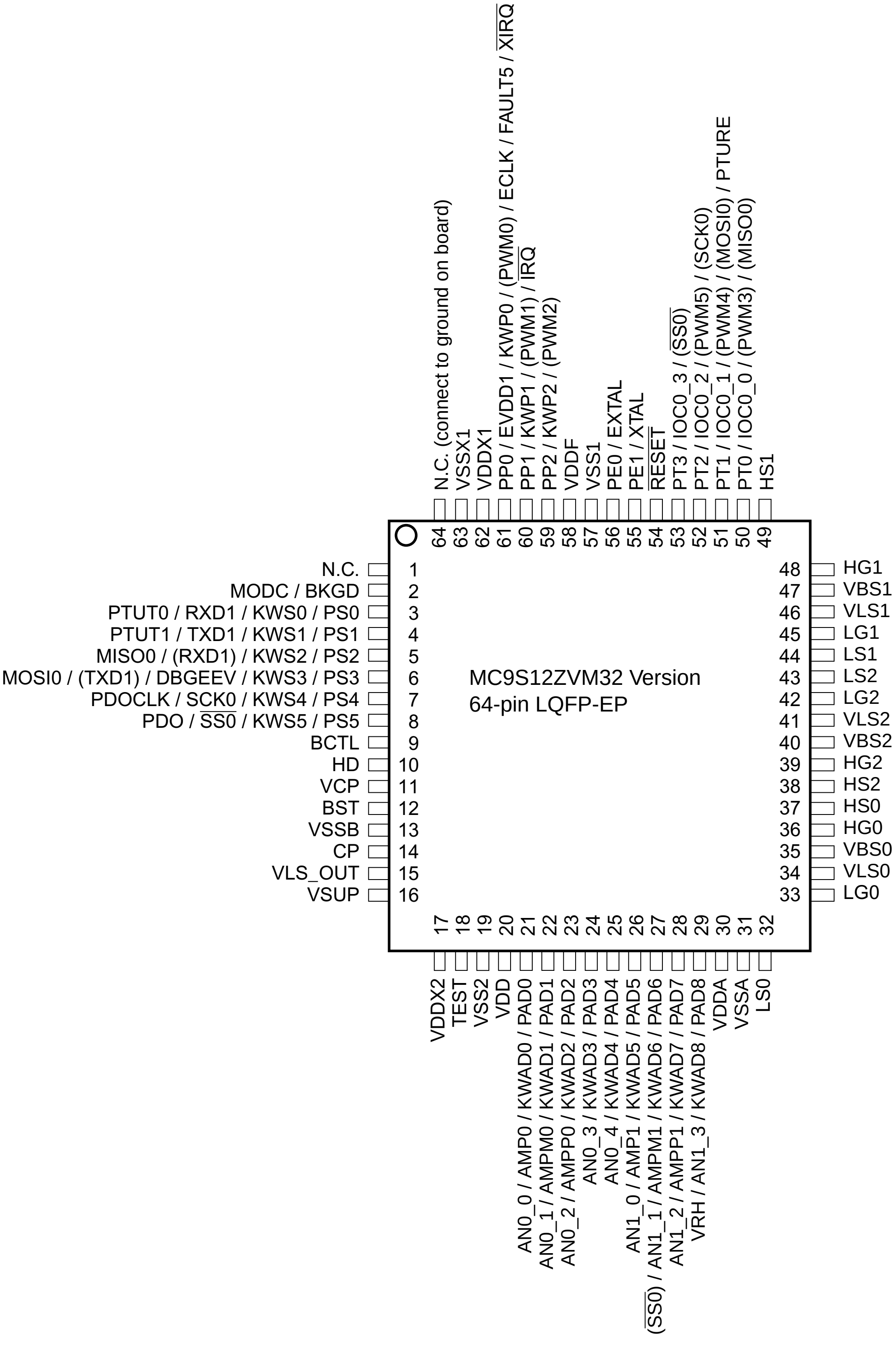


Figure 1-5. MC9S12ZVM32 64-pin LQFP pin out

Table 1-6. Pin Summary (Sheet 1 of 3)

LQFP Option			Function					Power Supply	Internal Pull Resistor	
64 L	64 C	Pin	1st Func.	2nd Func.	3rd Func.	4th Func.	5th Func.		CTRL	Reset State
1	—	LIN	—	—	—	—	—	—	—	—
—	1	BCTLC	—	—	—	—	—	—	—	—
2	2	BKGD	MODC	—	—	—	—	V_{DDX}	—	Up
3	3	PS0	KWS0	RXD1	RXCAN0	LP0RXD	PTUT0	V_{DDX}	PERS/ PPSS	Up
4	4	PS1	KWS1	TXD1	TXCAN0	LP0TXD	PTUT1	V_{DDX}	PERS/ PPSS	Up
5	5	PS2	KWS2	RXD1	MISO0	—	—	V_{DDX}	PERS/ PPSS	Up
6	6	PS3	KWS3	DBGEEV	TXD1	MOSI0	—	V_{DDX}	PERS/ PPSS	Up
7	7	PS4	KWS4	SCK0	PDOCLK	—	—	V_{DDX}	PERS/ PPSS	Up
8	8	PS5	KWS5	$\overline{\text{SS0}}$	PDO	—	—	V_{DDX}	PERS/ PPSS	Up
9	9	BCTL	—	—	—	—	—	—	—	—
10	10	HD	—	—	—	—	—	—	—	—
11	11	VCP	—	—	—	—	—	—	—	—
12	12	BST	—	—	—	—	—	—	—	—
13	13	VSSB	—	—	—	—	—	—	—	—
14	14	CP	—	—	—	—	—	—	—	—
15	15	VLS_OUT	—	—	—	—	—	—	—	—
16	16	VSUP	—	—	—	—	—	V_{SUP}	—	—
17	17	VDDX2	—	—	—	—	—	V_{DDX}	—	—
18	18	TEST	—	—	—	—	—	—	$\overline{\text{RESET}}$	Down
19	19	VSS2	—	—	—	—	—	—	—	—
20	20	VDD	—	—	—	—	—	V_{DD}	—	—
21	21	PAD0	KWAD0	AN0_0	AMP0	—	—	V_{DDA}	PERADL/ PPSADL	Off
22	22	PAD1	KWAD1	AN0_1	AMPM0	—	—	V_{DDA}	PERADL/ PPSADL	Off

Table 1-6. Pin Summary (Sheet 2 of 3)

LQFP Option			Function					Power Supply	Internal Pull Resistor	
64 L	64 C	Pin	1st Func.	2nd Func.	3rd Func.	4th Func.	5th Func.		CTRL	Reset State
23	23	PAD2	KWAD2	AN0_2	AMPP0	—	—	V_{DDA}	PERADL/ PPSADL	Off
24	24	PAD3	KWAD3	AN0_3	—	—	—	V_{DDA}	PERADL/ PPSADL	Off
25	25	PAD4	KWAD4	AN0_4	—	—	—	V_{DDA}	PERADL/ PPSADL	Off
26	26	PAD5	KWAD5	AN1_0	AMP1	—	—	V_{DDA}	PERADL/ PPSADL	Off
27	27	PAD6	KWAD6	AN1_1	AMPM1	$\overline{\text{SS0}}$	—	V_{DDA}	PERADL/ PPSADL	Off
28	28	PAD7	KWAD7	AN1_2	AMPP1	—	—	V_{DDA}	PERADL/ PPSADL	Off
29	29	PAD8	KWAD8	AN1_3	VRH	—	—	V_{DDA}	PERADH/ PPSADH	Off
30	30	VDDA	—	—	—	—	—	V_{DDA}	—	—
31	31	VSSA	—	—	—	—	—	—	—	—
32	32	LS0	—	—	—	—	—	—	—	—
33	33	LG0	—	—	—	—	—	—	—	—
34	34	VLS0	—	—	—	—	—	—	—	—
35	35	VBS0	—	—	—	—	—	—	—	—
36	36	HG0	—	—	—	—	—	—	—	—
37	37	HS0	—	—	—	—	—	—	—	—
38	38	HS2	—	—	—	—	—	—	—	—
39	39	HG2	—	—	—	—	—	—	—	—
40	40	VBS2	—	—	—	—	—	—	—	—
41	41	VLS2	—	—	—	—	—	—	—	—
42	42	LG2	—	—	—	—	—	—	—	—
43	43	LS2	—	—	—	—	—	—	—	—
44	44	LS1	—	—	—	—	—	—	—	—
45	45	LG1	—	—	—	—	—	—	—	—
46	46	VLS1	—	—	—	—	—	—	—	—
47	47	VBS1	—	—	—	—	—	—	—	—
48	48	HG1	—	—	—	—	—	—	—	—

Table 1-6. Pin Summary (Sheet 3 of 3)

LQFP Option			Function					Power Supply	Internal Pull Resistor	
64 L	64 C	Pin	1st Func.	2nd Func.	3rd Func.	4th Func.	5th Func.		CTRL	Reset State
49	49	HS1	—	—	—	—	—	—	—	—
50	50	PT0	IOC0_0	PWM3	MISO0	RXD0	—	V_{DDX}	PERT/ PPST	Off
51	51	PT1	IOC0_1	PWM4	MOSI0	TXD0	LP0DR1/ PTURE	V_{DDX}	PERT/ PPST	Off
52	52	PT2	IOC0_2	PWM5	SCK0	—	—	V_{DDX}	PERT/ PPST	Off
53	53	PT3	IOC0_3	$\overline{\text{SS0}}$	—	—	—	V_{DDX}	PERT/ PPST	Off
54	54	RESET	—	—	—	—	—	V_{DDX}	TEST pin	Up
55	55	PE1	XTAL	—	—	—	—	V_{DDX}	PERE/ PPSE	Down
56	56	PE0	EXTAL	—	—	—	—	V_{DDX}	PERE/ PPSE	Down
57	57	VSS1	—	—	—	—	—	—	—	—
58	58	VDDF	—	—	—	—	—	V_{DDF}	—	—
59	59	PP2	KWP2	PWM2	—	—	—	V_{DDX}	PERP/ PPSP	Off
60	60	PP1	KWP1	PWM1	$\overline{\text{IRQ}}$	—	—	V_{DDX}	PERP/ PPSP	Off
61	61	PP0 / EVDD1	KWP0	PWM0	ECLK	FAULT5	$\overline{\text{XIRQ}}$	V_{DDX}	PERP/ PPSP	Off
62	62	VDDX1	—	—	—	—	—	V_{DDX}	—	—
63	63	VSSX1	—	—	—	—	—	—	—	—
64	—	LGND	—	—	—	—	—	—	—	—
—	64	VDDC	—	—	—	—	—	—	—	—

1.8 Internal Signal Mapping

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

1.8.1 ADC Connectivity

1.8.1.1 ADC Reference Voltages

For both ADC modules, VRH_1 is mapped to VDDA; VRH_0 is mapped to PAD[8]; VRL_0 and VRL_1 are both mapped to VSSA, whereby VRL_1 is the preferred reference for low noise.

1.8.1.2 ADC Internal Channels

The ADC0 and ADC1 internal channel mapping is shown in Table 1-7 and Table 1-8 respectively.

The GDU current sense amplifier outputs are mapped to pins with ADC input functionality. Thus configuring the ADC to convert these pin channels automatically converts the current sense outputs.

The ADC internal temperature sensors must be calibrated by the user. No electrical parameters are specified for these sensors. The VREG temperature sensor electrical parameters are given in the appendices.

Table 1-7. Usage of ADC0 Internal Channels

ADCCMD_1 CH_SEL[5:0]						ADC Channel	**Usage**
0	0	1	0	0	0	Internal_0	ADC0 temperature sensor
0	0	1	0	0	1	Internal_1	VREG temperature sensor or bandgap (V_{BG})(1)
0	0	1	0	1	0	Internal_2	GDU phase multiplexer voltage
0	0	1	0	1	1	Internal_3	GDU DC link voltage monitor
0	0	1	1	0	0	Internal_4	BATS VSUP sense voltage
0	0	1	1	0	1	Internal_5	Reserved
0	0	1	1	1	0	Internal_6	Reserved
0	0	1	1	1	1	Internal_7	Reserved

1. Selectable in CPMU

Table 1-8. Usage of ADC1 Internal Channels

ADCCMD_1 CH_SEL[5:0]						ADC Channel	**Usage**
0	0	1	0	0	0	Internal_0	ADC1 temperature sensor
0	0	1	0	0	1	Internal_1	VREG temperature sensor or bandgap (V_{BG})(1)
0	0	1	0	1	0	Internal_2	GDU phase multiplexer voltage
0	0	1	0	1	1	Internal_3	GDU DC link voltage monitor
0	0	1	1	0	0	Internal_4	Reserved
0	0	1	1	0	1	Internal_5	Reserved
0	0	1	1	1	0	Internal_6	Reserved
0	0	1	1	1	1	Internal_7	Reserved

1. Selectable in CPMU

1.8.2 Motor Control Loop Signals

The motor control loop signals are described in 1.13.3.1 Motor Control Loop Overview

1.8.3 Device Level PMF Connectivity

Table 1-9. Mapping of PMF signals

PMF Connection	**Usage**
Channel0	High-Side Gate and Source Pins HG[0], HS[0]
Channel1	Low-Side Gate and Source Pins LG[0], LS[0]
Channel2	High-Side Gate and Source Pins HG[1], HS[1]
Channel3	Low-Side Gate and Source Pins LG[1], LS[1]
Channel4	High-Side Gate and Source Pins HG[2], HS[2]
Channel5	Low-Side Gate and Source Pins LG[2], LS[2]
FAULT5	External FAULT5 pin
FAULT4	HD Over voltage or GDU over current
FAULT3	VLS under voltage
FAULT2	GDU Desaturation[2] or GDU over current
FAULT1	GDU Desaturation[1] or GDU over current
FAULT0	GDU Desaturation[0] or GDU over current
$\overline{\text{IS2}}$	GDU Phase Status[2]
$\overline{\text{IS1}}$	GDU Phase Status[1]
$\overline{\text{IS0}}$	GDU Phase Status[0]
async_event_edge_sel[1:0]	Tied to b11 (both edges active)

1.8.4 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.

1.8.5 LINPHY Connectivity

The VLINPHY supply is connected to the device HD pin.

1.8.6 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.

The FTMRZ FCLKDIV register is forced to 0x05 by the BDC ERASE_FLASH command. This configures the clock frequency correctly for the initial bus frequency on leaving reset. The bus frequency must not be changed before launching the ERASE_FLASH command.

1.8.7 CPMU Connectivity

The API clock generated in the CPMU is not mapped to a device pin in the MC9S12ZVM-Family.

1.9 Modes of Operation

The MCU can operate in different modes. These are described in 1.9.1 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 1.9.3 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.

1.9.1 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

1.9.1.1 Normal Single-Chip Mode

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

1.9.1.2 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.

1.9.2 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 MC9S12ZVM-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.

The DBG module supports breakpoint, tracing and profiling features. At board level the profiling pins can use the same 6-pin connector typically used for the BDC BKGD pin. The connector pin mapping shown in Figure 1-6 is supported by device evaluation boards and leading development tool vendors.

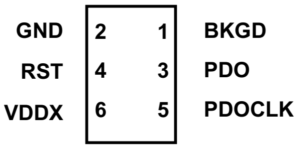


Figure 1-6. Standard Debug Connector Pin Mapping

1.9.3 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. Further to the general system aspects of Stop mode discussed here, the motor control loop specific considerations are described in Section 1.13.3.10.
  - 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. The counters and dividers remain frozen. The autonomous periodic interrupt (API) may remain active but has a very low power consumption. The key pad, SCI and MSCAN transceiver modules 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 and the CPMU continues operation as in run mode. With BDC enabled and BDCCIS bit set, then all clocks remain active to allow BDC access to internal peripherals. If the BDC is enabled and

BDCCIS is clear, then the BDCSI clock remains active, but bus and core clocks are disabled. 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 BATS voltage monitoring remains enabled.

1.10 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.

1.10.1 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

1.10.2 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.

1.10.3 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:

1.10.3.1 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).

1.10.3.2 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.

1.10.4 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

1.10.4.1 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.

1.10.5 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.

1.10.6 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.

1.11 Resets and Interrupts

1.11.1 Resets

Table 1-12. lists all reset sources and the vector locations. Resets are explained in detail in the Chapter 8, “S12 Clock, Reset and Power Management Unit (S12CPMU_UHV_V6)”.

Table 1-12. Reset Sources and Vector Locations

Vector Address	Reset Source	CCR Mask	Local Enable
0xFFFFFC	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
	COP watchdog reset	None	CR[2:0] in CPMUCOP register

1.11.2 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 (Sheet 1 of 4)

Vector Address[1]	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 page2 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 time-out interrupt	I bit	CPMUINT (RTIE)	**See CPMU section**	Yes
Vector base + 0x1CC	TIM0 timer channel 0	I bit	TIM0TIE (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 to Vector base + 0x1B0	Reserved				
Vector base + 0x1AC	TIM0 timer overflow	I bit	TIM0TSCR2(TOI)	No	Yes
Vector base + 0x1A8 to Vector base + 0x1A4	Reserved				
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				

Table 1-13. Interrupt Vector Locations (Sheet 1 of 4)

Vector Address[1]	Interrupt Source	CCR Mask	Local Enable	Wake up from STOP	Wake up from WAIT
Vector base + 0x18C	ADC0 Error	I bit	ADC0EIE (IA_EIE, CMD_EIE, EOL_EIE, TRIG_EIE, RSTAR_EIE, LDOK_EIE) ADC0IE(CONIF_OIE)	No	Yes
Vector base + 0x188	ADC0 conversion sequence abort	I bit	ADC0IE(SEQAD_IE)	No	Yes
Vector base + 0x184	ADC0 conversion complete	I bit	ADC0CONIE[15:0]	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 to Vector base + 0x174	Reserved				
Vector base + 0x170	RAM error	I bit	EECIE (SBEEIE)	No	Yes
Vector base + 0x16C to Vector base + 0x168	Reserved				
Vector base + 0x164	FLASH error	I bit	FERCNFG (SFDIE)	No	Yes
Vector base + 0x160	FLASH command	I bit	FCNFG (CCIE)	No	Yes
Vector base + 0x15C	CAN0 wake-up	I bit	CAN0RIER (WUPIE)	Yes	Yes
Vector base + 0x158	CAN0 errors	I bit	CAN0RIER (CSCIE, OVRIE)	No	Yes
Vector base + 0x154	CAN0 receive	I bit	CAN0RIER (RXFIE)	No	Yes
Vector base + 0x150	CAN0 transmit	I bit	CAN0TIER (TXEIE[2:0])	No	Yes
Vector base + 0x14C to Vector base + 0x148	Reserved				
Vector base + 0x144	LINPHY over-current interrupt	I bit	LPIE (LPDTIE,LPOCIE)	No	Yes
Vector base + 0x140	BATS supply voltage monitor interrupt	I bit	BATIE (BVHIE,BVLIE)	No	Yes
Vector base + 0x13C	GDU Desaturation Error	I bit	GDUIE (GDSEIE)	No	Yes
Vector base + 0x138	GDU Voltage Limit Detected	I bit	GDUIE (GOCIE, GHHDFIE, GLVLSFIE)	No	Yes
Vector base + 0x134 to Vector base + 0x128	Reserved				
Vector base + 0x124	Port S interrupt	I bit	PIES[5:0]	Yes	Yes
Vector base + 0x120	Reserved				

Table 1-13. Interrupt Vector Locations (Sheet 1 of 4)

Vector Address[1]	Interrupt Source	CCR Mask	Local Enable	Wake up from STOP	Wake up from WAIT
Vector base + 0x11C	ADC1 Error	I bit	ADC1EIE (IA_EIE, CMD_EIE, EOL_EIE, TRIG_EIE, RSTAR_EIE, LDOK_EIE) ADC1IE(CONIF_OIE)	No	Yes
Vector base + 0x118	ADC1 conversion sequence abort	I bit	ADC1IE(SEQAD_IE)	No	Yes
Vector base + 0x114	ADC1 conversion complete	I bit	ADC1CONIE[15:0]	No	Yes
Vector base + 0x110	Reserved				
Vector base + 0x10C	Port P interrupt	I bit	PIEP[2:0]	Yes	Yes
Vector base + 0x108	EVDD1 over-current interrupt	I bit	PIEP(OCIE1)	No	Yes
Vector base + 0x104	Low-voltage interrupt (LVI)	I bit	CPMULVCTL (LVIE)	No	Yes
Vector base + 0x100	Autonomous periodical interrupt (API)	I bit	CPMUAPICTRL (APIE)	Yes	Yes
Vector base + 0xFC	High temperature interrupt	I bit	CPMUHTCTL(HTIE)	Yes	Yes
Vector base + 0xF8	Reserved				
Vector base + 0xF4	Port AD interrupt	I bit	PIEADH(PIEADH0) PIEADL(PIEADL[7:0])	Yes	Yes
Vector base + 0xF0	PTU Reload Overrun	I bit	PTUIEH(PTUROIE)	No	Yes
Vector base + 0xEC	PTU Trigger0 Error	I bit	PTUIEL(TG0AEIE, TG0REIE,TG0TEIE)	No	Yes
Vector base + 0xE8	PTU Trigger1 Error	I bit	PTUIEL(TG1AEIE, TG1REIE, TG1TEIE)	No	Yes
Vector base + 0xE4	PTU Trigger0 Done	I bit	PTUIEL(TG0DIE)	No	Yes
Vector base + 0xE0	PTU Trigger1 Done	I bit	PTUIEL(TG1DIE)	No	Yes
Vector base + 0xDC to Vector base + 0xD4	Reserved				
Vector base + 0xD0	PMF Reload A	I bit	PMFENCA(PWMRIEA)	No	Yes
Vector base + 0xCC	PMF Reload B	I bit	PMFENCB(PWMRIEB)	No	Yes
Vector base + 0xC8	PMF Reload C	I bit	PMFENCC(PWMRIEC)	No	Yes
Vector base + 0xC4	PMF Fault	I bit	PMFFIE(FIE[5:0])	No	Yes
Vector base + 0xC0	PMF Reload Overrun	I bit	PMFROIE(PMFROIEA,PMFROIEB,PMFROIEC)	No	Yes
Vector base + 0xBC to Vector base + 0x10	Reserved				

[1] 15 bits vector address based

1.11.3 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.

1.11.3.1 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.

1.11.3.2 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.

1.11.3.3 I/O Pins

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

1.11.3.4 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.

1.12 Module device level dependencies

1.12.1 CPMU COP Configuration

The COP time-out 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 CPMUCOP 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 CPMUCOP Register
1	0
0	1

1.12.2 CPMU High Temperature Trimming

The value loaded from the flash into the CPMUHTTR register is a default value for the device family. There is no device specific trimming carried out during production. The specified V_{HT} value is a typical value that is part dependent and should thus be calibrated.

1.12.3 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 PAD8/VSSA												▨	▨	▨	▨	0x1F_C042 & 0x1F_C043
ADC1 reference conversion using VDDA/VSSA												▨	▨	▨	▨	0x1F_C044 & 0x1F_C045
ADC1 reference conversion using PAD8/VSSA												▨	▨	▨	▨	0x1F_C046 & 0x1F_C047

1.13 Application Information

1.13.1 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 reference voltage is measured during production test using the internal reference voltage V_{BG}, which has a narrow variation over temperature and external voltage supply. V_{BG} is mapped to an internal channel of each ADC module (Table 1-7,Table 1-8). The resulting 12-bit left justified ADC conversion results of V_{BG} are stored to the flash IFR for reference, as listed in Table 1-16.

The measurement conditions of the reference conversion are listed in the device electrical parameters appendix. 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} :

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

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

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

With:

ConvertedADInput:	Result of the analog to digital conversion of the desired pin
ConvertedReference:	Result of internal channel conversion
StoredReference:	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.

1.13.2 SCI Baud Rate Detection

The baud rate for SCI0 and SCI1 is achieved by using a timer channel to measure the data rate on the RXD signal.

1. Establish the link:
   — For SCI0: Set [T0IC3RR1:T0IC3RR0]=0b01 to disconnect IOC0_3 from TIM0 input capture channel 3 and reroute the timer input to the RXD0 signal of SCI0.
   — For SCI1: Set [T0IC3RR1:T0IC3RR0]=0b10 to disconnect IOC0_3 from TIM0 input capture channel 3 and reroute the timer input to the RXD1 signal of SCI1.
2. Determine pulse width of incoming data: Configure TIM0 IC3 to measure time between incoming signal edges.

1.13.3 Motor Control Application Overview

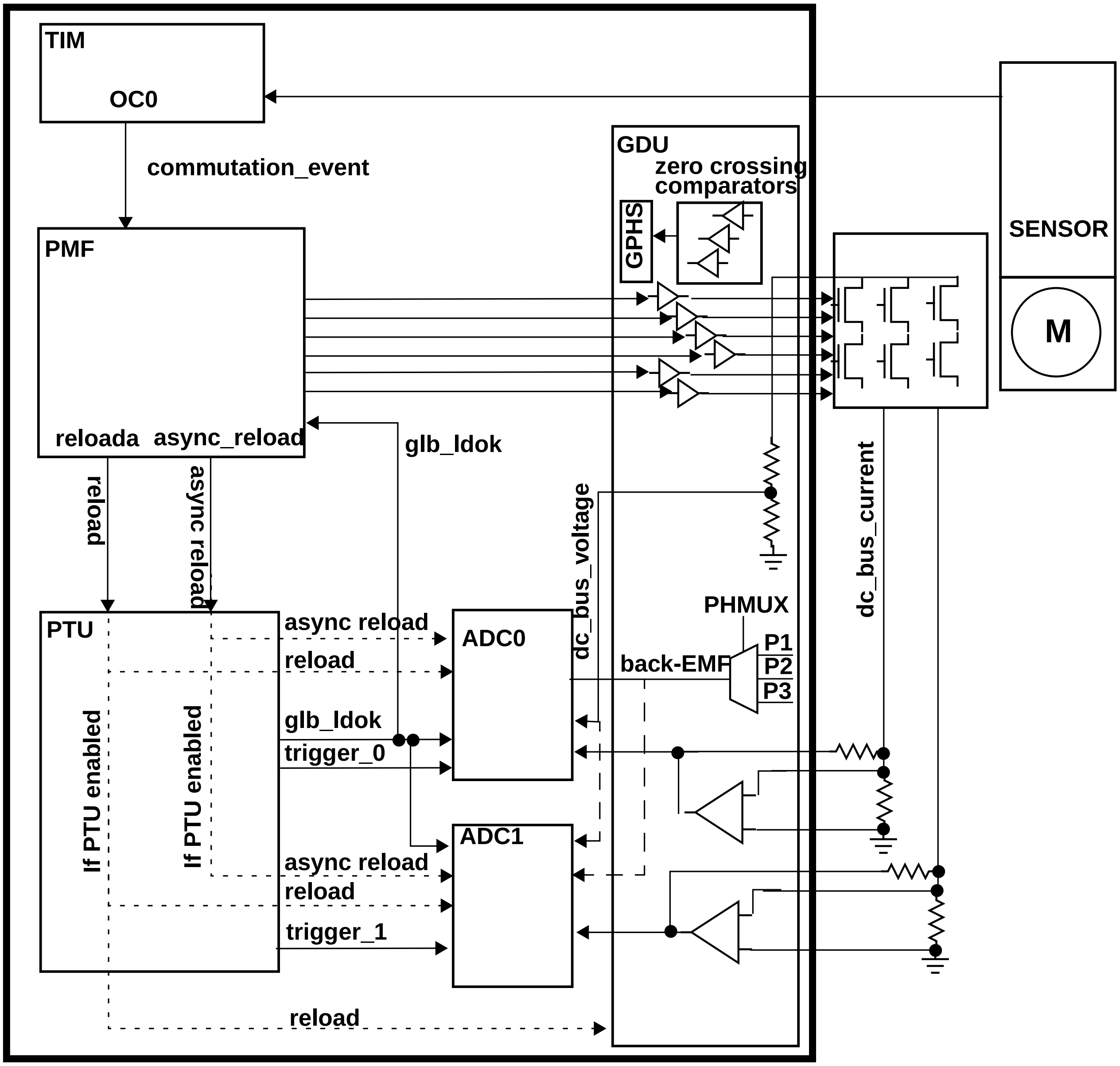
The following sections provide information for using the device in motor control applications. These sections provide a description of motor control loop considerations that are not detailed in the individual module sections, since they concern device level inter module operation specific for motor control. More detailed information is available in application notes**.** The applications described are as follows:

1. BDCM - wiper pumps fans
2. BLDCM - pumps, fans and blowers
   - based on Hall sensors
   - sensorless based on back-EMF zero crossing comparators
   - sensorless based on back-EMF ADC measurements
3. PMSM - high-end wiper, pumps, fans and blowers
   - simple sinewave commutation with position sensor Hall effect, sine-cos
   - FOC with sine-cos position sensor
   - sensorless 3-phase sinewave control

1.13.3.1 Motor Control Loop Overview

The mapping of motor control events at device level as depicted in Figure 1-7 is listed in Table 1-17, whereby the columns list the names used in the module level descriptions

Figure 1-7. Internal Control Loop Configuration



The control loop consists of the PMF, GDU, ADC and PTU modules. The control loop operates using either static, dynamic or asynchronous timing. In the following text the event names given in **bold type** correspond to those shown in Figure 1-7. The PTU and ADC operate using lists stored in memory. These lists define trigger points for the PTU, commands for the ADC and results from the ADC. If the PTU is enabled the reload and async_reload events are immediately passed through to the ADC and GDU modules.

Table 1-17. Control Loop Events

Device Level Event	TIM	PMF	PTU	ADC0	ADC1
commutation_event	OC0(1)	commutation_event	—	—	—
reload	—	reloada(2)	reload	Restart	Restart
async_reload	—	async_reload	async_reload	Seq_abort	Seq_abort
trigger_0	—	—	trigger_0	Trigger	—
trigger_1	—	—	trigger_1	—	Trigger
glb_ldok	—	glb_ldok	glb_ldok	LoadOK	LoadOK

1. TIM channel OC0 must be configured to toggle on both edges.

2. PMF events reloadb and reloadc are not connected at device level

Each control loop cycle is started by a PMF **reload** event. The PMF reload event restarts the PTU time base. If the PTU is enabled, the reload is immediately passed through to the ADC and GDU modules.

The PMF generates the **reload** event at the required PWM reload frequency. The PMF reload event causes the PTU time base to restart, to acquire the first trigger times from the list and the ADCs to start loading the ADC conversion command from the Command Sequence List (CSL).

NOTE

> In the PTU there is 7 bus cycle maximum time window after the **reload** event assertion to access the first trigger times from the list. In this window the trigger can not be generated. In the ADC there is 10 bus cycle maximum time window after the **reload** event assertion to access the first ADC command from the list. In this window the ADC conversion can not be started. If the measurement is control loop related these delays are negligible due to much larger delays in the PWM-GDU-feedback loop.

When the trigger time is encountered the corresponding PTU trigger generates the **trigger_x** event for the associated ADC. For simultaneous sampling the PTU generates simultaneous **trigger_x** events for both ADCs. At the **trigger_x** event the ADC starts the first conversion of the next conversion sequence in the CSL (the first ADC command is already downloaded).

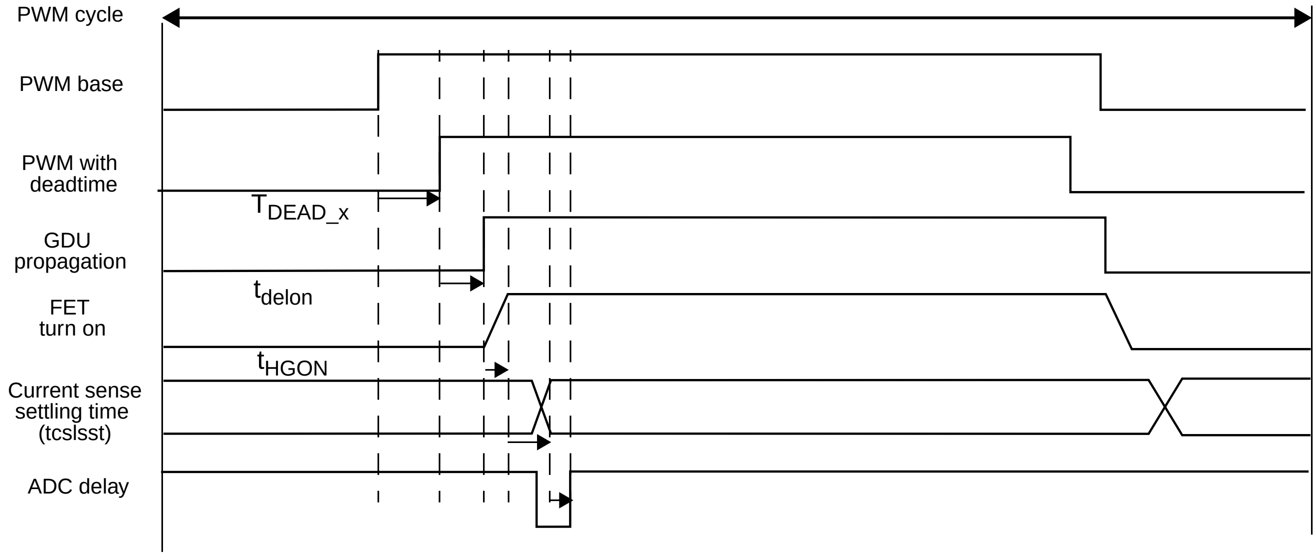
A commutation event is used by the PMF to generate an async_reload event. The async_reload is used by the PTU to update lists and re-initialize the trigger lists. If the PTU is enabled the async_reload is immediately passed through to the ADC.

1.13.3.2 Control Loop Timing Considerations

Delays within the separate control loop elements require consideration to ensure correct synchronization.

Regarding the raw PWM signal as the starting point and stepping through the control loop stages, the factors shown in Figure 1-8 contribute to delays within the control loop, starting with the deadtime insertion, going through the external FETs and back into the internal ADC measurements of external voltages and currents.

Figure 1-8. Control Loop Delay Overview



The PWM deadtime (T_{DEAD_X}) is an integral number of bus clock cycles, configured by the PMF deadtime registers.

The GDU propagation delays (t_{delon}, t_{deloff}) are specified in the electrical parameter Table E-1.

The FET turn on times (t_{HGON}) are load dependent but are specified for particular loads in the electrical parameter Table E-1.

The current sense amplifier delay is highly dependent on external components.

The ADC delay until a result is available is specified as the conversion period N_{CONV} in Table C-1.

1.13.3.3 Static Timing Operation

The timing frame is static if it is the same in every control cycle (defined by reload frequency) and is relative to start of the control cycle. The only settings modified from one control cycle to the next one are the PWM duty cycle registers.

The main control cycle synchronization event is the PMF **reload** event. The PMF **reload** event can be generated every *n* PWM periods.
This mode can optionally be extended by a timer channel trigger to PMF to change the PWM channel operation (e.g. used for BLDCM commutation). In this case, the PMF configuration can propagate the trigger through the control loop or can prevent propagation so the static timing of the control cycle and inter-block coherency are not affected by the trigger.

At the end of the conversion sequence the first ADC command from the new sequence is loaded and the ADC*x* waits for the next **trigger_x**. The PTU continues to generate the **trigger_x** events for each trigger time from the list until a new **reload** or **async_reload** occurs.

Before the upcoming **reload** event the CPU:

- reads the ADC results from the buffered Conversion Result List
- clears the conversion complete flag
- services the **reload** by setting new duty cycle values
- sets the PTULDOK bit (corresponding to **glb_ldok**) to signal the duty cycle coherence

The CPU actions are typically performed in an ISR triggered by the conversion complete flag.

1.13.3.4 Static Timing Fault Handling

The following Faults and/or errors can occur:

- Desaturation error, Overvoltage, Undervoltage, Temperature sensor, External fault

The application run-time error is handled by the GDU without CPU interaction. Firstly the FETs are disabled and the PMF signals switched to an inactive state. To re-enable the operation first the GDU fault and then PWM fault must be cleared, to automatically re-enable the FET driving at the next PWM boundary.

- PTU reload overrun error

This is an application run-time error caused by the CPU not setting PTULDOK on time. Servicing this type of error is application dependent and may range from a further reload attempt to a total shut down.

- PTU trigger generator reload error, PTU trigger generator error

Since all timing is static, this error should only occur during application debugging. This type of error occurring in a static timing configuration indicates possible data corruption. This can be serviced by a control loop shutdown.

- PTU memory access error, Memory access double bit ECC error

This type of error occurring in an application indicates data corruption. This can be serviced by a control loop shutdown.

- ADC sequence overrun, ADC command overrun, ADC command error

Since all timing is static, this error should only occur during application debugging. This type of error occurring in an application indicates possible data corruption. This can be serviced by a control loop shutdown.

1.13.3.5 Dynamic Timing Operation

The timing frame is dynamic if the following are modified on a cycle by cycle basis:

- PMF - duty cycle value registers (PMF_VAL*x*), modulo registers
- PTU - Trigger Event List (PTU_TEL*x*)
- ADC - Command Sequence List (ADC*x*_CSL)

The main philosophy is that all cycle-by-cycle settings for cycle *n* need to be done within cycle *n-1*. The main control cycle synchronization event is the PMF **reload** event, which can be generated every *n* PWM periods.

This mode can optionally be extended by a timer channel trigger PMF to change PWM channel operation (e.g. used for BLDCM commutation).

The event flow is the same as for static timing.

Before the upcoming **reload** event the CPU:

- reads the ADC results from the buffered Conversion Result List
- clears the conversion complete flag
- services the **reload** by setting new duty cycle values and a new PMF modulo value
- updates the non-active PTU_TEL*x*
- updates the non-active ADC*x*_CSL
- sets the PTULDOK bit (corresponding to **glb_ldok**) to signal the duty cycle coherence

The CPU actions are typically performed in an ISR triggered by the conversion complete flag.

1.13.3.6 Dynamic Timing Fault Handling

The following Faults and/or errors can occur:

- Desaturation error, Overvoltage, Undervoltage, Temperature sensor, External fault

The application run-time error is handled by the GDU without CPU interaction. Firstly the FETs are disabled and the PMF signals switched to an inactive state. To re-enable the operation first the GDU fault and then PWM fault must be cleared, to automatically re-enable the FET driving at the next PWM boundary.

- PTU reload overrun error

This is an application run-time error caused by the CPU not setting PTULDOK on time. Servicing this type of error is application dependent and may range from a further reload attempt to a total shut down.

- PTU trigger generator reload error, PTU trigger generator error

This indicates an application run-time error caused by a settings mismatch. Servicing this type of error is application dependent. In some cases, the ADC values for the current control cycle can be ignored.

- PTU memory access error, Memory access double bit ECC error

This type of error occurring in an application indicates possible data corruption. This can be serviced by a control loop shutdown.

- ADC sequence overrun, ADC command overrun, ADC command error

This indicates an application run-time error caused by a settings mismatch. Servicing this type of error is application dependent. In some cases, the ADC values for the current control cycle can be ignored.

1.13.3.7 Asynchronous Timing

This case is an extension of the dynamic timing case by an asynchronous event generated by the Timer. Note the asynchronous term is referenced to the control cycle.

The timing frame is the same as in dynamic timing case plus it can be asynchronously restarted at any time within the control cycle.

At the asynchronous **commutation_event**

- **the PMF actions are:**

1. counter re-start, re-initialization
2. PWM configuration re-initialization according to the selected PWM settings (center/edge-aligned pattern, normal/inverted type etc.)
3. re-initialization of the dead time generators (in case the commutation takes place at a time when one of the dead times is being generated)
4. re-initialization of the PWM outputs according to pre-set PWM channel output settings in double buffered registers (mask, swap, output control)
5. re-initialization of the automatic fault clearing
6. generates **async_reload** event for the PTU
7. optionally updates the PWM duty cycle values based on LDOK state

- **the PTU actions are:**

1. abortion of the **trigger_x** event generation
2. re-initialization and re-start the PTU counter
3. update of the current list index TGxList based on the **glb_ldok** state
4. fetch first trigger time from updated TGxList
5. passes the **async_reload** event immediately to the ADC (if the PTU is enabled)
6. generates the **reload** event for the ADC

- **the ADC actions are:**

1. the conversion in progress is completed
2. the ADC conversion sequence is aborted and the SEQA flag is set to indicate that the final conversion occurred during the abortion process (potentially coinciding with a commutation and is thus less precise than under normal conditions)
3. update of the current lists index ADxLists
4. re-start of the conversion sequencing upon successful abortion - fetches the first ADC command from the ADC*x*_CSL, re-sets the result pointer to the top of the list

   Note: in case the lists index ADxLists is not updated at the sequence abortion the new restarted A/D conversions will overwrite the previously converted results.

- **the GDU actions are:**

1. standard operation

1.13.3.8 Control Loop Startup Guidelines

The sequence for control loop start up is to firstly configure the signal measurement (inputs/feedback). Once the measurement is properly configured (correct value is measured at defined time) the output actuation (control action) is configured. The following modules are involved in signal measurements.

- TIM (to identify asynchronous commutation) [BLDC applications only]

- PMF (to generate main synchronous events for PTU and ADC)
- PTU (to generate delay relative to synchronous events generated by PMF)
- ADC (to acquire analog signals under synchronous control)
- GDU (zero crossing comparators, Back-EMF muxing) [application dependent]

The TIM OC0 channel identifies the commutation event and restarts the PMF counter. In order to establish this link TIM and PMF need to be configured and started. Then to sample accurately within one PMF cycle the PTU needs to be used, so the next step is to configure the PTU to establish PMF to PTU link. The PTU sends triggers to the ADC to perform a measurement of control signals. So the next step is to configure the ADC. In some cases the GDU involvement is required and therefore configured.

The control action involves the PMF (to generate the duty cycle for GDU) and the GDU (to propagate the signal to the MOSFETs). Since the PMF has already been configured for the measurements, only the GDU need be configured to complete startup. Sometimes the GDU can be configured earlier but the GDU output is always enabled last.

The recommended startup sequence is summarized as follows:

- Configure TIM and PMF to establish the link between TIM OC0 commutation event and PMF
- Configure PTU to establish the PMF to PTU link and ensure correct sampling within PMF cycle
- Configure the ADC
- Configure the GDU

1.13.3.9 Control Loop Shutdown Guidelines

1. Remove energy stored in the system after the power stage
   kinetic energy - stop all rotating/moving mass
   magnetic energy - gracefully drive currents to zero
2. Put GDU and PMF outputs to safe state

1.13.3.10 Control Loop Stop Mode Considerations

In Stop mode the PWM, PTU, ADC can not run because the bus clock is not running. Thus the GDU must transition to a disabled state. Before entering Stop mode the application must perform the following steps:

1. Remove energy stored in the system after the power stage
   kinetic energy - stop all rotating/moving mass
   magnetic energy - gracefully drive currents to zero
2. Put GDU and PMF outputs to safe state
3. Verify GDU and PMF safe states
4. Verify fault flags and service if necessary
5. Execute the STOP instruction

The return from stop is expected in reverse order:

1. On returning from Stop mode the clocks are automatically enabled coherently
2. Initialize and check device proper functionality (charge pump etc.)

3. Check functionality of the external system
4. Initializes control loop operation, however with PMF and GDU outputs still in safe state
5. Read the ADC values to check the system
6. Start driving energy into the system
   based on the measurements from the previous step, the PWM duty cycle values are calculated
7. PMF and GDU outputs are enabled (actively driven)

The device does not support putting the FETs in an active driving state during STOP as the GDU charge pump clock is not running. This means the device cannot be put in stop mode if the FETS need to be in an active driving state to protect the system from external energy supply (e.g. externally driven motor-generator).

NOTE

> It is imperative, that whatever the modules perform on entering/exiting Stop mode, the pre-set complementary mode of operation and dead time insertion must be guaranteed all the times.

1.13.3.11 Application Signal Visibility

In typical motor control applications, TIM OC0 is used internally to indicate commutation events. To switch off OC0 visibility at port pin PT0:

- Disable output compare signal on pin PT0 in TIM: OCPD[OCPD0]=0b1.

1.13.3.12 Debug Signal Visibility

Depending on required visibility of internal signals on port pins enable the following registers:

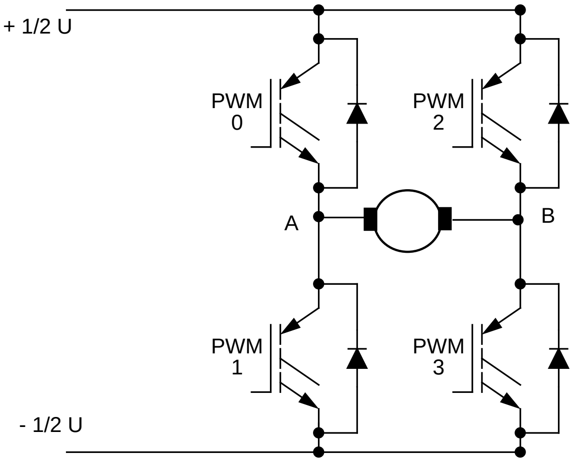
- Set [PWMPRR]=0b1 in PIM if monitoring of internal PWM waveforms is needed. PWM5-3 are driven out on pins PT[2:0] and PWM2-0 on pins PP[2:0].
- Enable output compare channel OC0 to output commutation event on pin PT0 in TIM: OCPD[OCPD0]=0b0.
- Set PTUDEBUG[PTUREPE]=0b1 in PTU to output the reload event.
- Set PTUDEBUG[PTUTxPE]=0b1 with x=0,1 in PTU to output the trigger events.

1.13.4 BDCM Complementary Mode Operation

This section describes BDCM control using center aligned complementary mode with deadtime insertion.

The DC Brushed motor power stage topology is a classical full bridge as shown in Figure 1-9. The DC Brushed motor is driven by the DC voltage source. A rotational field is created by means of commutator and brushes on the motor. These drives are still very popular because sophisticated calculations and algorithms such as commutation, waveform generation, or space vector modulation are not required.

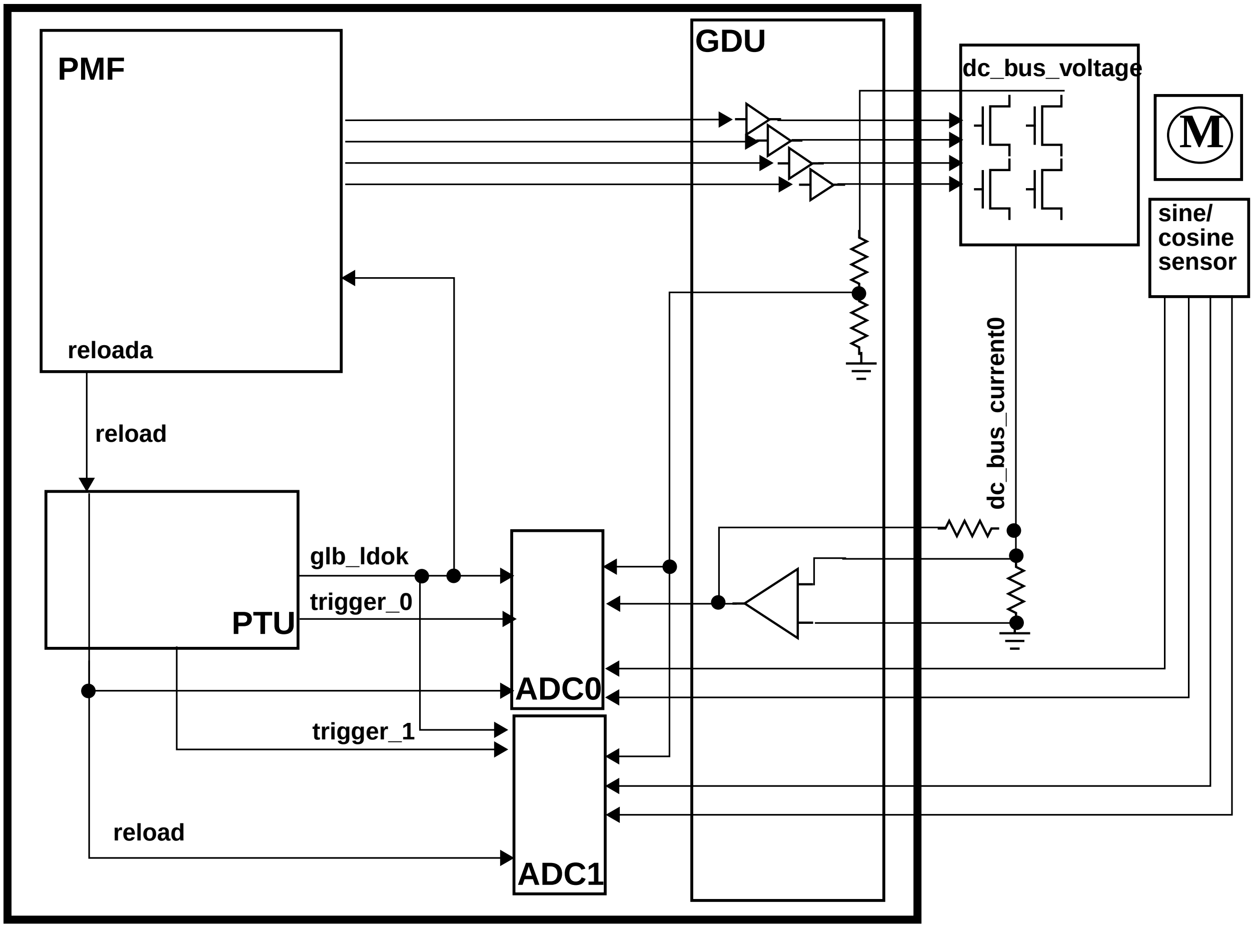
Figure 1-9. DC Brushed Motor External Configuration



Usually the control consists of an outer, speed control loop with inner current (torque) control loop. The inner loop controls DC voltage applied onto the motor winding. The control loop is calculated regularly within a given period. In most cases, this period matches the PWM reload period.

Driving the DC motor from a DC voltage source, the motor can work in all four quadrants. The complementary mode of operation with deadtime insertion is needed for smooth reversal of the motor current (motor torque), hence smooth full four quadrant control. Usually the center-aligned PWM is chosen to lower electromagnetic emissions.

Figure 1-10. BDCM Control Loop Configuration

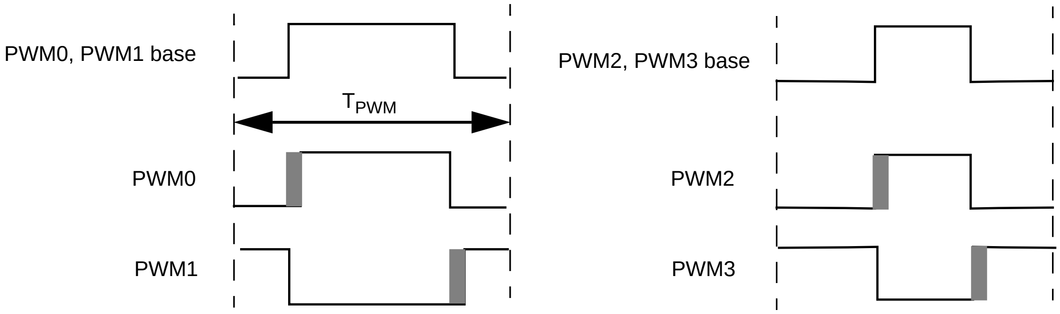


The PWM frequency selection is always a compromise between audible noise, electromagnetic emissions, current ripples and power switching losses.

The BDCM control loop goal is to provide a controlled DC voltage to the motor winding, whereby it is controlled cycle-by-cycle using a speed, current or torque feedback loop.

The center aligned PWM waveforms generated by the PMF module are applied to the bridge as shown in Figure 1-11 whereby the base waveform for PWM0 and PWM1 is depicted at the top and the complementary PWM0 and PWM1 waveforms are shown with deadtime insertion depicted by the gray phases before the switching edges.

Figure 1-11. BDCM Complementary Mode Waveform



Assuming first quadrant operation, forward accelerating operation, the applied voltage at node A must exceed the applied voltage at node B (Figure 1-9). Thus the PWM0 duty cycle must exceed the PWM2 duty cycle.

The PWM duty cycle of PWM0 defines the voltage at the first power stage branch.

The PWM duty cycle of PWM2 defines the voltage at the second power stage branch.

Modulating the PWM duty cycle every period using the function F_{PWM} then the duty cycle is expressed as:

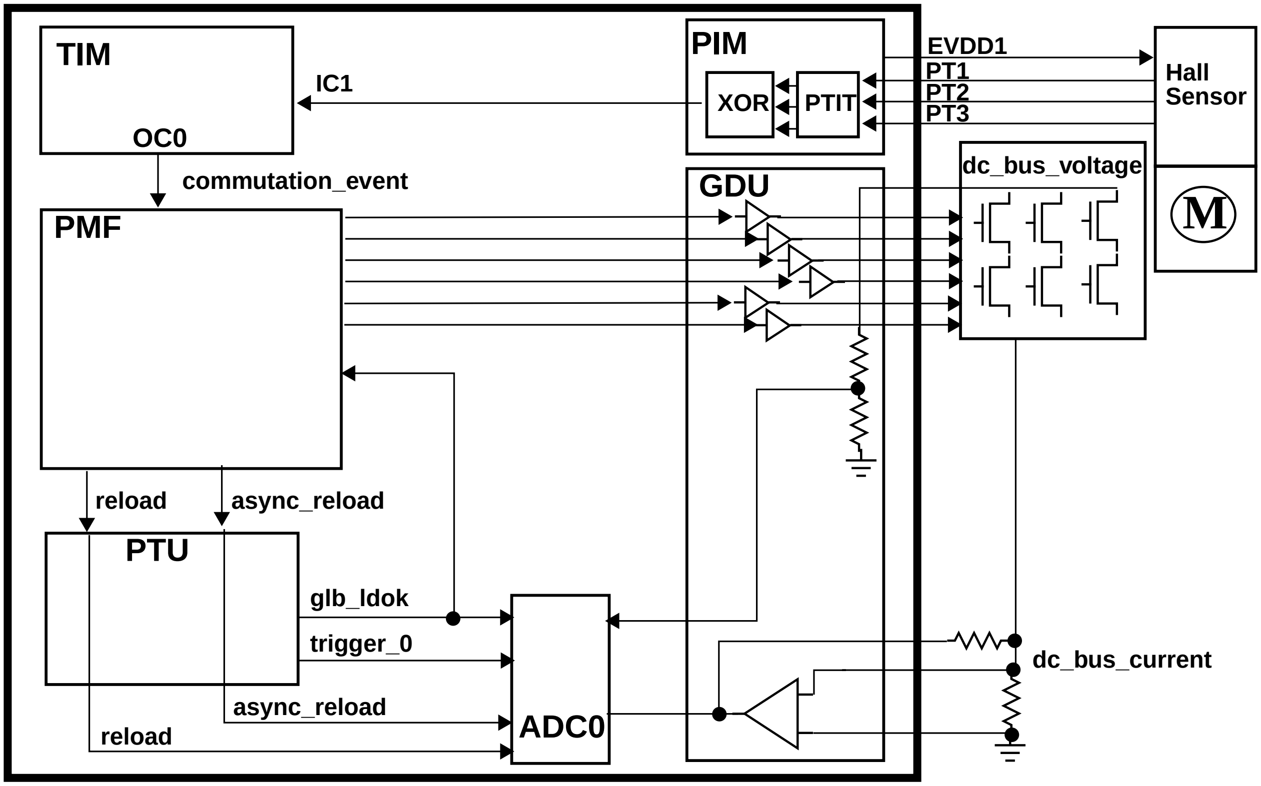
PWM0 duty-cycle = $0.5 + (0.5 * F_{PWM})$; For $-1 <= F_{PWM} <= 1$;

PWM2 duty-cycle = $0.5 - (0.5 * F_{PWM})$

1.13.5 BLDC Six-Step Commutation

1.13.5.1 Hall Sensor Triggered Commutation

Figure 1-12. BLDC Configuration With Hall Sensors



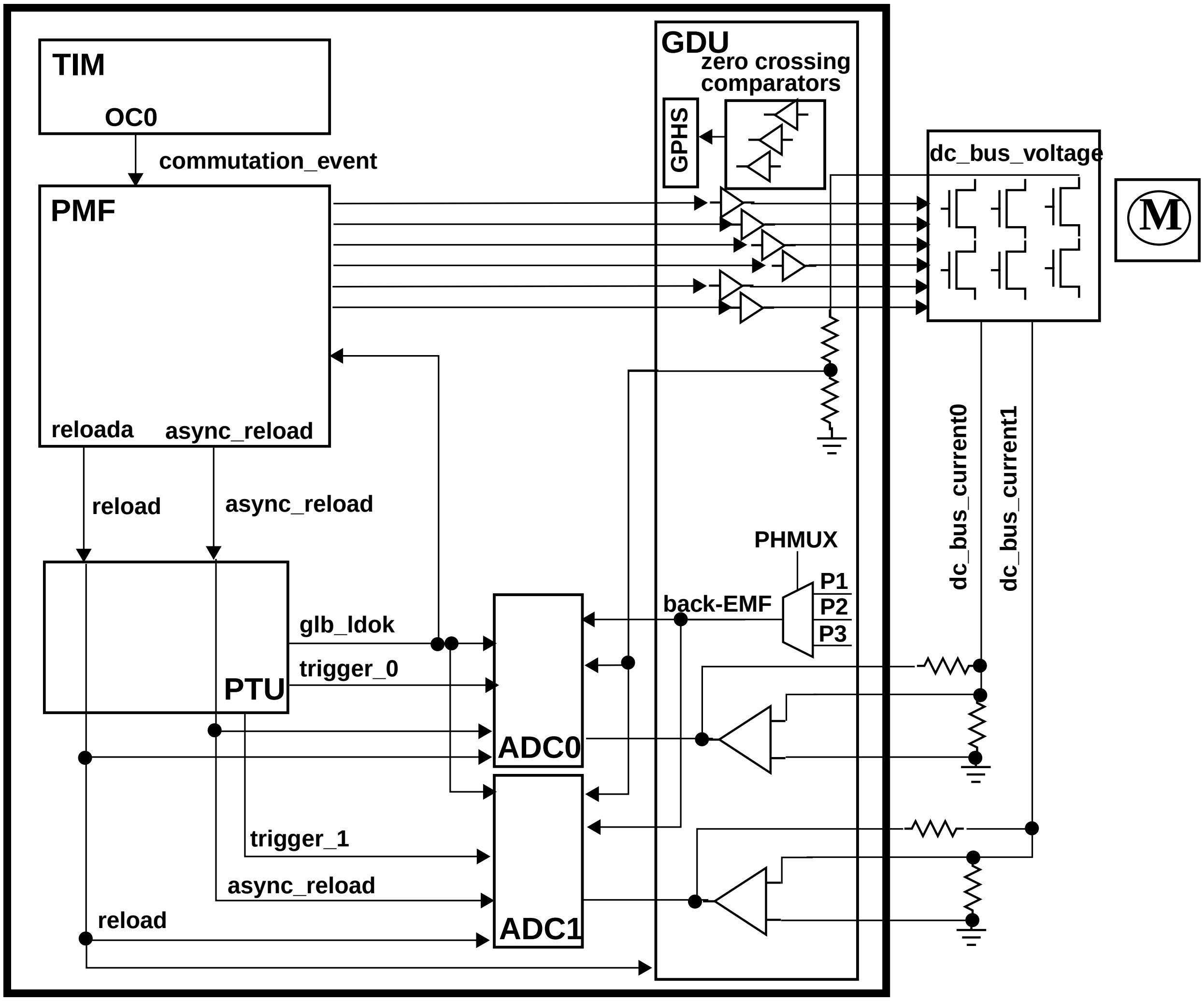
This BLDC application uses Hall sensor signals to create commutation triggers. The integrated sense amplifier and an ADC module are used to measure DC bus current, for torque calculation. The DC bus voltage measurement is used in the control algorithm to counter-modulate the PWM such that the variation of the DC-bus voltage does not affect the motor current closed loop. The configuration is as follows:

1. Connect the three Hall sensor signals from the motor to input pins PT3-1.
2. Set [T0IC1RR=1] in the register MODRR2 to establish the link from Hall sensor input pins to TIM input capture channel 1.
3. Setup TIM IC1 for speed measurement of XORed Hall sensor signals. Enable interrupt on both edges.
4. Enable TIM OC0 and select toggle action on output compare event: TCTL2[OM0:OL0]=01.
5. Configure PMF for edge-aligned PWM mode with or without restart at commutation: PMFENCx[RSTRT]. If using the restart option, then select generator A as reload signal source and keep the following configurations at their default setting: multi timebase generators (PMFCFG0[MTG]=b0), reload frequency (PMFFQCx[LDFQx]=b0), prescaler (PMFFQCx[PRSCx]=b00).
6. Enable PMF commutation event input: PMFCFG1[ENCE]=1.

7. Read port register PTIT[3:1] to determine starting sector.
8. Startup motor by applying PWM to the related motor phase.
9. In IC1 interrupt ISR calculate the delay to next commutation and store value to output compare register. Update registers with next values of mask and swap.
10. On next output compare event the buffered mask and swap information is transferred to the active PWM registers to execute the commutation.

1.13.5.2 Sensorless Commutation

Figure 1-13. Sensorless BLDC Configuration



To calculate the commutation time in a sensorless motor system the back-EMF zero crossing event of the currently non-fed phase within an electrical rotation cycle must be determined. For fast motor rotation, the ADC is used to measure the back-EMF voltage and the DC bus voltage to determine the zero crossing time. For slow motor rotation the GPHS register can be polled. In either case the zero crossing event is handled by the CPU monitoring flags or responding to interrupts. The TIM then generates the commutation_event under CPU control, based on the zero crossing time.

1. Enable TIM OC0 and select toggle action on output compare event: TCTL2[OM0:OL0]=0b01.
2. Enable PMF commutation event input: PMFCFG1[ENCE]=0b1.
3. Enable internal ADC channel for measuring the phase voltages from the muxed GDU outputs.
4. Align rotor to stator field. Initialize phase MUX using register GDUPHMUX.
5. Startup motor by applying PWM to an arbitrary motor phase.
6. Take samples of the phase voltages periodically based on PWM cycle to detect zero crossing.
7. Calculate the delay to next commutation and store value to output compare register. Update registers with next values of mask and swap.
8. On next output compare event the buffered mask and swap information are transferred to the active PWM register to execute the commutation.

1.13.6 PMSM Control

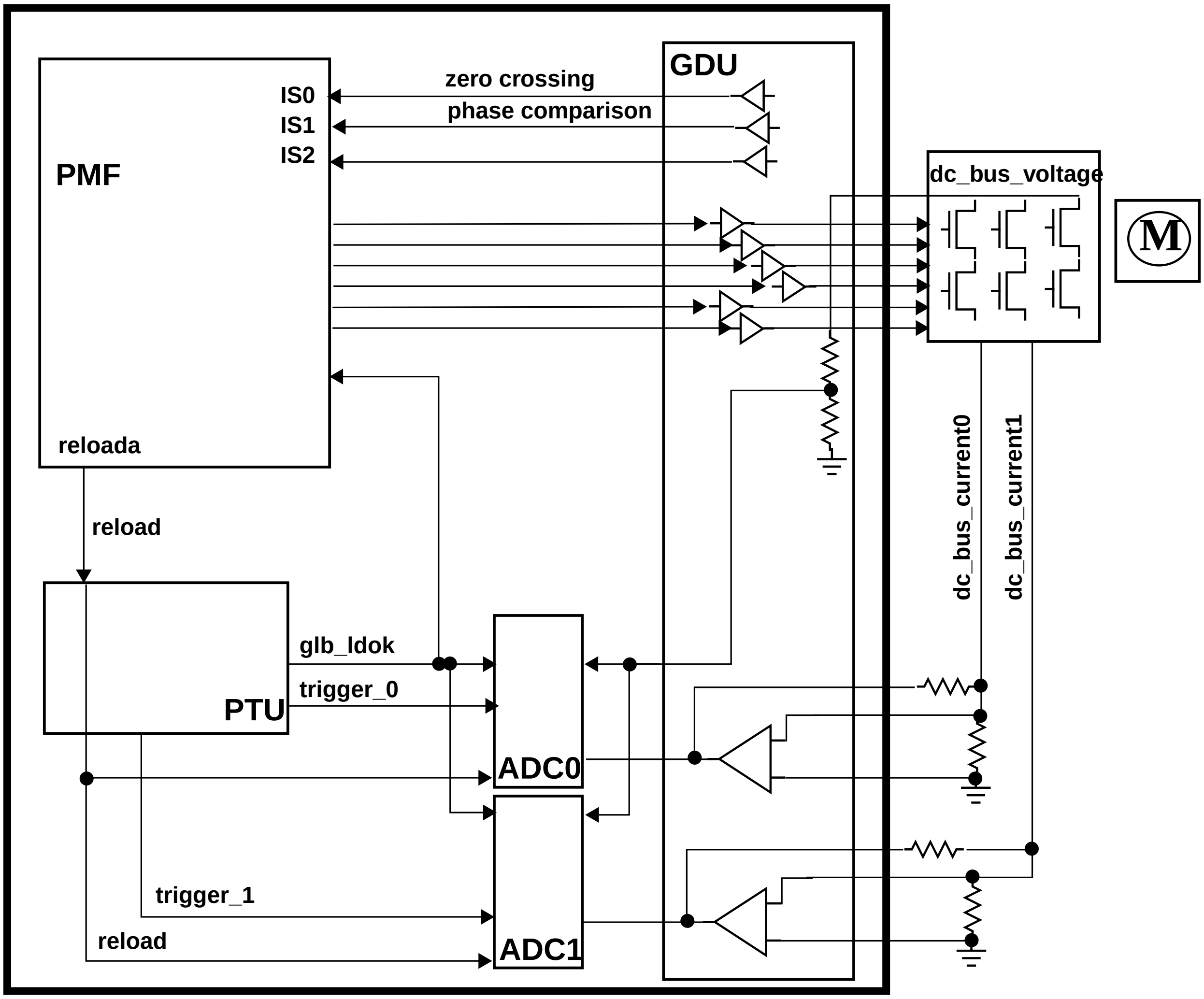
PMSM control drives all 3 phases simultaneously with sinusoidal waveforms. Both sensorless and Sine-Cosine position sensor control loop operation are supported.

1.13.6.1 PMSM Sensorless Operation

In this configuration the PMSM stator winding currents are driven sinusoidally and the back EMF waveform is also sinusoidal. Thus all 3 phases are active simultaneously. The rotor position and speed are determined by the current and calculated voltages respectively. The back EMF voltage is calculated based on the currents.

1. Configure PMF for complementary mode operation.
2. Configure PMF for center aligned or phase shifted operation.
3. Select correct PMF deadtime insertion based on external FET switches.
4. Enable GDU current sense opamps for measuring the phase currents from 2 external shunts.
5. Map the output pin of each current sense opamp to the ADC input.
6. Optionally use GDU phase comparators for zero crossing detection to correct deadtime distortion.
7. Fetch targeted motor speed parameter from external source (e.g. SCI)
8. Configure PMF period and duty cycle.
9. Startup motor by applying FOC startup algorithm.
10. Take samples of the phase currents periodically based on PWM cycle to determine motor speed.
11. Calculate FOC algorithm to determine back EMF and motor position.

Figure 1-14. Sensorless PMSM Control Loop Configuration



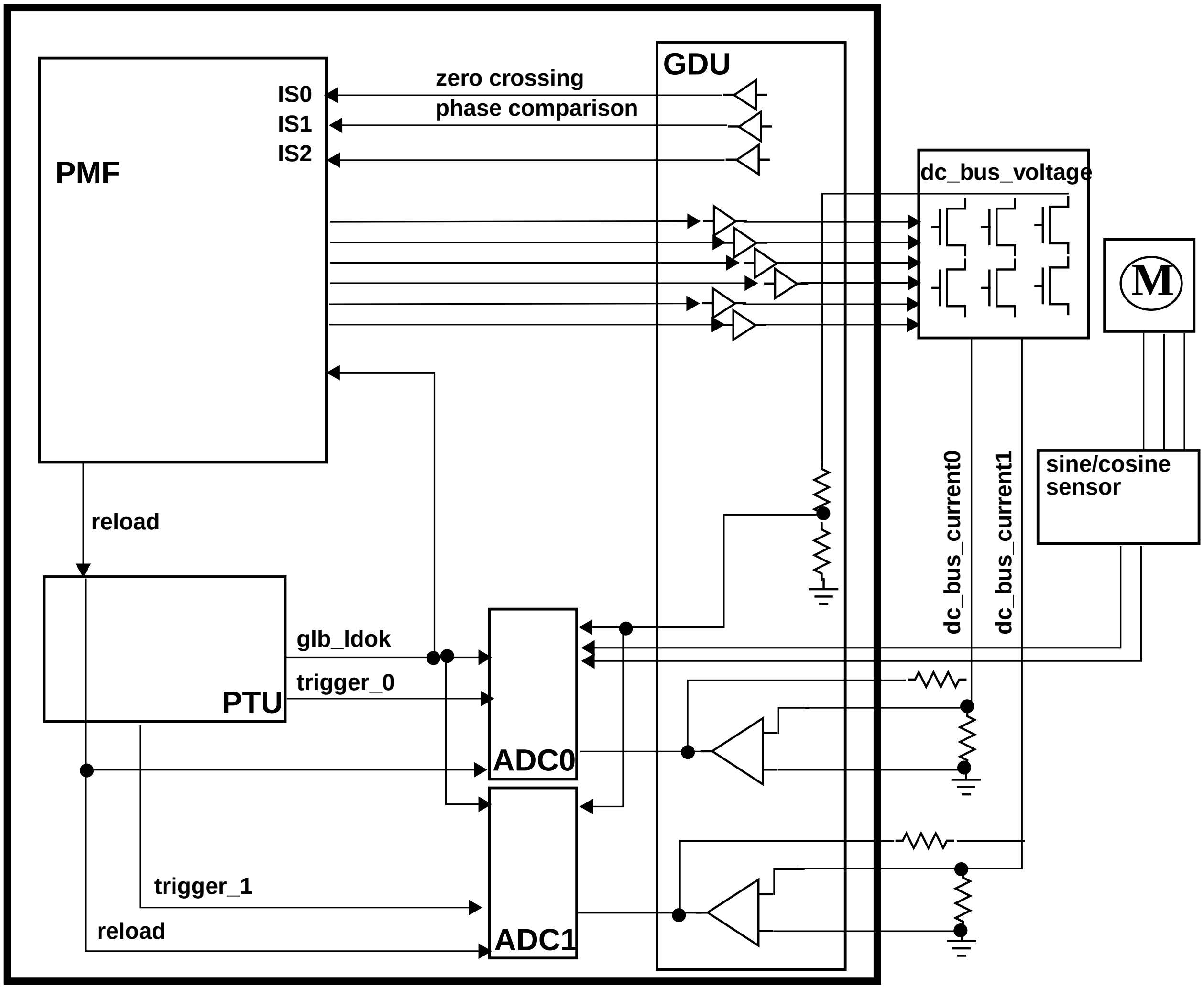
1.13.6.2 PMSM Operation With Sine-Cosine Position Sensor

In this configuration the PMSM stator winding currents are driven sinusoidally and the back EMF waveform is also sinusoidal. Thus all 3 phases are active simultaneously. The back EMF voltage is calculated based on the currents. The rotor position and speed are determined by a sine/cosine sensor, which generates sinusoidal sine/cosine signals, indicating the angle of the rotor in relation to sensor windings. The sensor is supplied by the EVDD1 pin.

1. Configure PMF for complementary mode operation.
2. Configure PMF for center aligned or phase shifted operation.
3. Select correct PMF deadtime insertion based on external FET switches.
4. Enable GDU current sense opamps for measuring the phase currents from external shunts.
5. Map the output pin of each current sense opamp to the ADC input.

6. Map the sine/cosine input signals to ADC input channels.
7. Configure the EVDD1 pin as output.
8. Optionally use GDU phase comparators for zero crossing detection to correct dead time distortion.
9. Fetch targeted motor speed parameter from external source (e.g. SCI)
10. Configure PMF period and duty cycle.
11. Start motor by applying startup algorithm.
12. Sample the sine/cosine voltages periodically based on PWM cycle to determine motor position.
13. Use FOC algorithm to determine back EMF and motor speed.

Figure 1-15. PMSM Sine/Cosine Control Loop Configuration



1.13.6.3 Dead time Distortion Correction

PMSM motor control applications driven by sinusoidal voltages by default require zero crossing information of phase currents to determine the point in time to change sign of deadtime compensation value to be added to duty cycles.

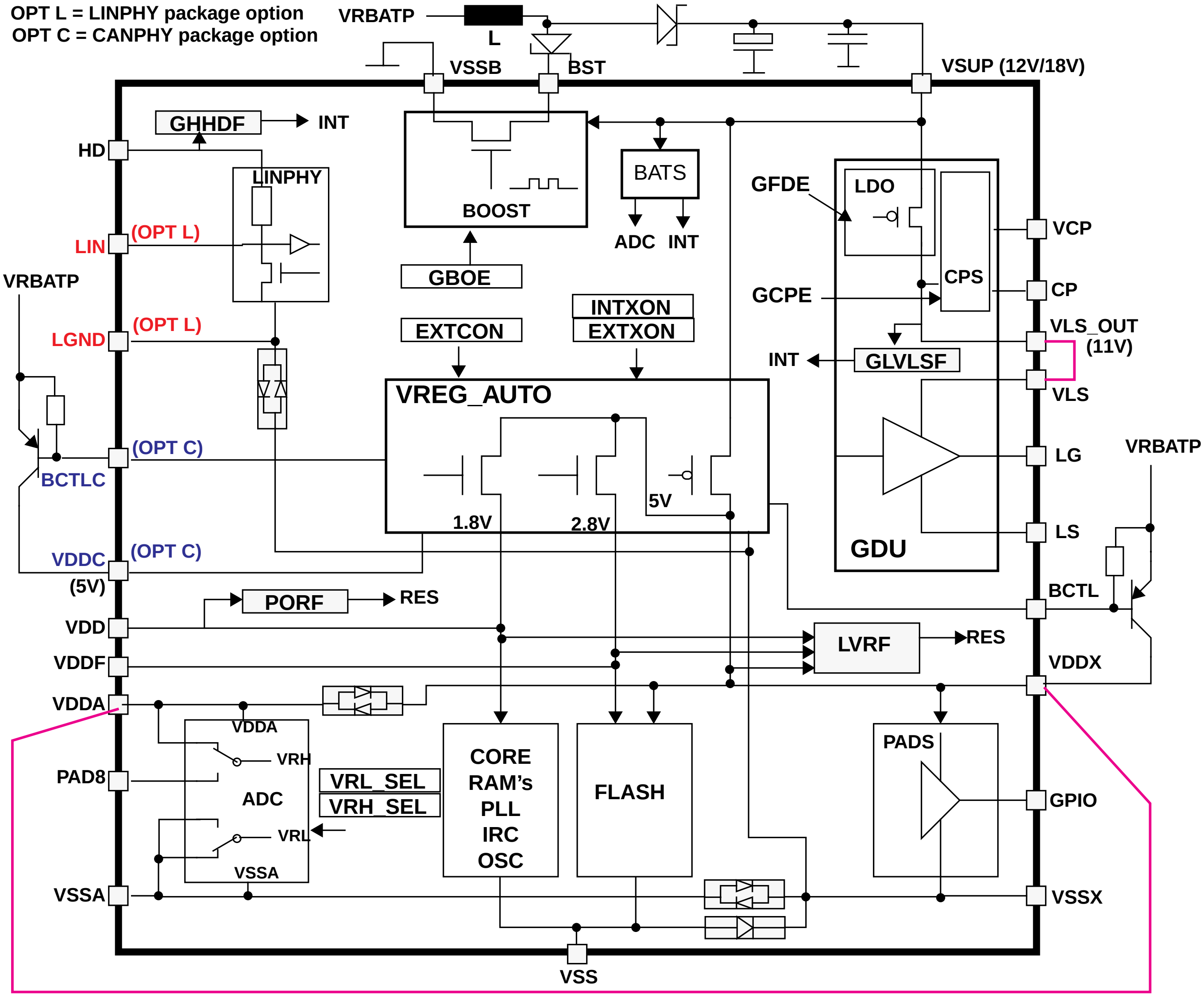
The GDU phase comparator signals are connected internally to the PMF ISx inputs. This allows the dead time distortion correction to be applied directly based on the phase status.

1. Align rotor to stator field.
2. Await phase comparator status change.
3. Switch to alternate duty cycle register to compensate distortion.

1.13.7 Power Domain Considerations

The MC9S12ZVM-Family power domains are illustrated in Figure 1-16. More detailed information is included in the individual module descriptions.

Figure 1-16. Power Domain Overview



The system supply voltage VRBATP is a reverse battery protected input voltage. It must be protected against reverse battery connections and must not be connected directly to the battery voltage (VBAT).

The device supply voltage VSUP provides the input voltage for the internal regulator, VREG_AUTO, which generates the voltages VDDX, VDD and VDDF. The VDDX domain supplies the device I/O pins, VDDA supplies the ADC and internal bias current generators. The VDDA and VDDX pins must be connected at board level, they are not connected directly internally. ESD protection diodes exist between VDDX and VDDA, therefore forcing a common operating range. The VDD domain supplies the internal device logic. The VDDF domain supplies sections of the internal Flash NVM circuitry.

The device supports the use of an external PNP to supplement the VDDX supply, for reducing on chip power dissipation. In this configuration, most of the current flowing from VRBATP to VDDX, flows through the external PNP. This configuration, using the BCTL pin, can be enabled by register bits EXTXON and INTXON.

A supply for an external CANPHY is offered via external device pins BCTLC and VDDC, whereby BCTLC provides the base current of an external PNP and VDDC is the CANPHY supply (output voltage of the external PNP). This is only available in the CANPHY package option. This configuration can be enabled by the register bit EXTCON.

The LINPHY pull-up resistor is internally connected to the HD voltage. This is chosen as opposed to VSUP to ensure that the LINPHY is not disturbed by the internal VSUP boost circuit. The external connections for the HD pin must ensure a reverse battery protection.

The ADC register bit VRH_SEL maps the ADC reference VRH to VDDA or to the device pad PAD8.

1.13.7.1 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.

The GDU high side drain voltage, pin HD, is monitored within the GDU and mapped to an interrupt. A connection to the ADC is provided for accurate measurement of a scaled HD level.

1.13.7.2 FET-Predriver (GDU) Supplies

A dedicated low drop regulator is used to generate the VLS_OUT voltage from VSUP. The VLS_OUT voltage is used to supply the low side drivers and can be directly connected to the VLS inputs of each low side driver. For FET-predriver operation at lower VSUP levels, a boost circuit can be enabled by the GBOE register bit. The boost circuit requires Shottky diodes, a coil and capacitors, as shown in Figure 1-16. More detailed information is included in the GDU module description.

1.13.7.2.1 Bootstrap Precharge

The FET-predriver high side driver must provide a sufficient gate-source voltage and sufficient charge for the gate capacitance of the external FETs. A bootstrap circuit is used to provide sufficient charge, whereby the capacitor C_{BS} is first charged to VLS_OUT via an external diode, when the low side driver is active Figure 1-17. When the high side driver switches on, the charge on this capacitor, supplies the FET-predriver via the VBSx pin. The C_{BS} capacitor can only be charged if the low side driver is active, so after a long period of inactivity of the low side driver, the C_{BS} capacitor becomes discharged. In this case, the low side driver must be switched on to charge C_{BS} before commencing high side driving. The time it takes to discharge the bootstrap capacitor C_{BS} can be calculated from the size of the bootstrap capacitor C_{BS} and the leakage current on VBSx pin.

The bootstrap capacitors must be precharged before turning on the high-side drivers for the first time. This can be done by using the PMF software output control mechanism:

```
PMFOUTC = 0x3F;                        // SW control on all outputs
PMFOUTB = 0x2A;                        // All high-sides off, all low-sides on
```

The PWM signals should be configured to start with turning on the low-side before the high-side drivers in order to assure precharged bootstraps. Therefore invert the PWM signals:

```
PMFCINV = 0x3F;                        // Invert all channels to precharge bootstraps
```

1.13.7.2.2 High Side Charge Pump For 100% Duty Cycle

A charge pump voltage is used to supply the high side FET-predriver with enough current to maintain the gate source voltage. To generate this voltage an external charge pump is driven by the pin CP, switching between 0V and 11V. The pumped voltage is then applied to the pin VCP.

At 100% duty cycle operation the low-side turn on time is zero during a masked commutation cycle before the high-side gates are attempted to be turned on. This can cause bootstrap charge to decay.

In order to speed-up the high-side gate voltage level directly after commutation, the software should drive the first PWM cycle with a duty cycle meeting an on-time of at least $t_{minpulse}$ for the low-side drivers and then switch back to 100% again.

The recommended procedure is to use the manual correction method (PMFCCTL[ISENS]) as described for the BLDC applications:

Set odd PMF values to alternative duty cycle. At commutation event when one of the three high-side drivers it turned on (every 120˚) set the PMFCCTL[IPOLx] bits and clear them in at next PWM reload event.

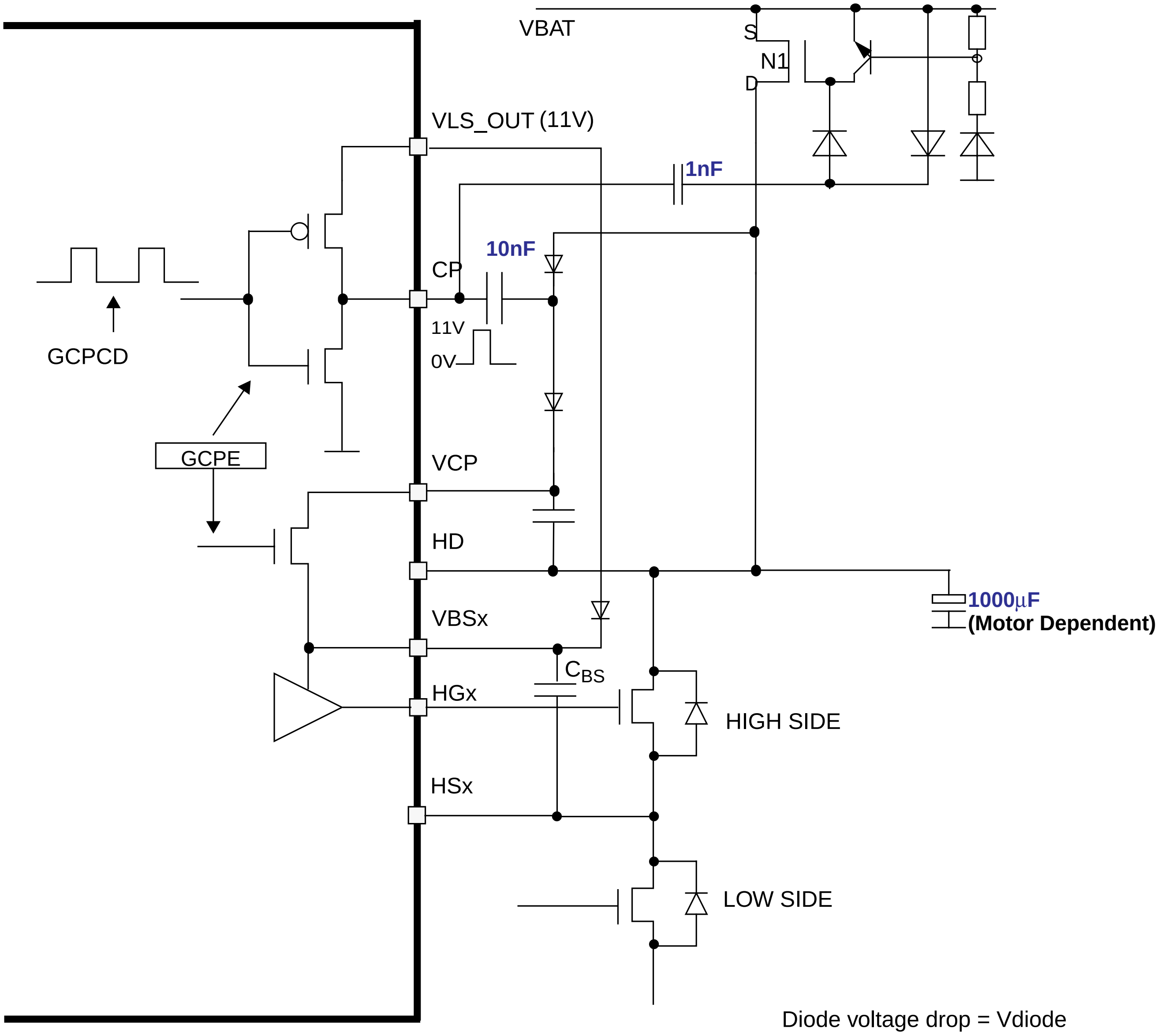
Given unipolar switching mode:

```
// TIM OC0 ISR:
if ((PMFOUTC == 0x1c) || (PMFOUTC == 0x07) || (PMFOUTC == 0x31)) // all high-side turn-on sectors
         PMFCCTL = 0x17; // select odd PMF values

// PMF reload ISR:
         PMFCCTL = 0x10; // select even PMF values
```

The GDU high side drain voltage, pin HD, is supplied from VBAT through a reverse battery protection circuit. In a typical application the charge pump is used to switch on an external NMOS, N1, with source connected to VBAT, by generating a voltage of VBAT+VLS-(2xVdiode). In a reverse battery scenario, the external bipolar turns on, ensuring that the HD pin is isolated from VBAT by the external NMOS, N1.

Figure 1-17. High Side Supply and Charge Pump Concept



Chapter 2
Port Integration Module (S12ZVMPIMV1)

Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V01.00	11 Nov 2012		• Initial release
V01.01	20 Feb 2013	Table 2-1	• Removed VRL • Corrected typo

2.1 Introduction

2.1.1 Overview

The S12ZVM-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
- 4-pin port T associated with either combination of 4 TIM channels, 3 PWM channels of PMF, 1 SPI and 1 SCI
- 6-pin port S with pin interrupts and key-wakeup function; associated with 1 MSCAN, 1 SCI and 1 SPI modules
- 3-pin port P with pin interrupts and key-wakeup function; associated with
    - $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$ interrupt inputs
    - 3 PWM channels of PMF
    - ECLK output
- 9-pin port AD associated with 9 ADC channels shared among two ADC and two GDU AMP modules - inputs can be used as an external interrupt and key-wakeup source

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 T, S, P and AD when used as general-purpose I/O
- Control registers to enable pull devices and select pullups/pulldowns on ports E, T, S, P and AD
- Control register to enable open-drain (wired-or) mode on port S
- Control register to enable digital input buffers on port AD
- Interrupt flag register for pin interrupts and key-wakeup (KWU) on port S, P and AD
- 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:
  - SPI0 to alternative pins
  - Various SCI0-LINPHY0 routing options supporting standalone use and conformance testing
  - Optional RXD0 to TIM0 link
  - Optional RXD1 to TIM0 link
  - PWM channels to GDU and/or pins
  - 3 pin input mux to one TIM0 IC channel

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
- Interrupt input with glitch filtering

2.2 External Signal Description

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

Table 2-1 shows all pins with the pins and functions that are controlled by the PIM. Routing options are denoted in parenthesis.

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 2-1. Pin Functions and Priorities

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Routing Register Bit	Pin Function after Reset
-	BKGD	MODC[2]	I	MODC input during $\overline{\text{RESET}}$	—	BKGD
		BKGD	I/O	S12ZBDC communication	—	
E	PE1	XTAL	-	CPMU OSC signal	—	GPIO
		PTE[1]	I/O	General-purpose	—	
	PE0	EXTAL	-	CPMU OSC signal	—	
		PTE[0]	I/O	General-purpose	—	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Routing Register Bit	Pin Function after Reset
AD	PAD8	VRH	I	ADC0&1 voltage reference high	—	GPIO
		AN1_3	I	ADC1 analog input	—	
		PTADH[0]/ KWADH[0]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD7	AMPP1	I	GDU AMP1 non-inverting input (+)	—	
		AN1_2	I	ADC1 analog input	—	
		PTADL[7]/ KWADL[7]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD6	AMPM1	I	GDU AMP1 inverting input (-)		
		($\overline{\text{SS0}}$)	I/O	SPI0 slave select	SPI0RR SPI0SSRR	
		AN1_1	I	ADC1 analog input	—	
		PTADL[6]/ KWADL[6]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD5	AMP1	O	GDU AMP1 output	—	
		AN1_0	I	ADC1 analog input	—	
		PTADL[5]/ KWADL[5]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD4-3	AN0_4:AN0_3	I	ADC0 analog input	—	
		PTADL[4:3]/ KWADL[4:3]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD2	AMPP0	I	GDU AMP0 non-inverting input (+)	—	
		AN0_2	I	ADC0 analog input	—	
		PTADL[2]/ KWADL[2]/	I/O	General-purpose; with interrupt and wakeup	—	
	PAD1	AMPM0	I	GDU AMP0 inverting input (-)	—	
		AN0_1	I	ADC0 analog input	—	
		PTADL[1]/ KWADL[1]	I/O	General-purpose; with interrupt and wakeup	—	
	PAD0	AMP0	O	GDU AMP0 output	—	
		AN0_0	I	ADC0 analog input	—	
		PTADL[0]/ KWADL[0]	I/O	General-purpose; with interrupt and wakeup	—	

Port	Pin Name	Pin Function & Priority(1)	I/O	Description	Routing Register Bit	Pin Function after Reset
T	PT3	($\overline{\text{SS0}}$)	I/O	SPI0 slave select	SPI0RR SPI0SSRR	GPIO
		(IOC0_3)(3)	I/O	TIM0 channel 3	T0IC3RR1-0	
		PTT[3]	I/O	General-purpose	—	
	PT2	(SCK0)	I/O	SPI0 serial clock	SPI0RR	
		(PWM5)	O	PWM channel 5	PWM54RR PWMPRR	
		IOC0_2	I/O	TIM0 channel 2	—	
		PTT[2]	I/O	General-purpose	—	
	PT1	PTURE	O	PTU reload event	—	
		(TXD0)/ (LPDC0)	O	SCI0 transmit/ LPTXD0 direct control by LP0DR[LP0DR1]	S0L0RR2-0	
		(MOSI0)	I/O	SPI0 master out/slave in	SPI0RR	
		(PWM4)	O	PWM channel 4	PWM54RR PWMPRR	
		(IOC0_1)[3]	I/O	TIM0 channel 1	T0IC1RR	
		PTT[1]	I/O	General-purpose	—	
	PT0	(RXD0)	I	SCI0 receive	S0L0RR2-0	
		(MISO0)	I/O	SPI0 master in/slave out	SPI0RR	
		(PWM3)	O	PWM channel 3	PWM32RR PWMPRR	
		IOC0_0	I/O	TIM0 channel 0	—	
		PTT[0]	I/O	General-purpose	—	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Routing Register Bit	Pin Function after Reset
S	PS5	PDO	O	DBG profiling data output	—	GPIO
		$\overline{\text{SS0}}$	I/O	SPI0 slave select	SPI0RR SPI0SSRR	
		PTS[5]/ KWS[5]	I/O	General-purpose; with interrupt and wakeup	—	
	PS4	PDOCLK	O	DBG profiling clock	—	
		SCK0	I/O	SPI0 serial clock	SPI0RR	
		PTS[4]/ KWS[4]	I/O	General-purpose; with interrupt and wakeup	—	
	PS3	MOSI0	I/O	SPI0 master out/slave in	SPI0RR	
		(TXD1)	O	SCI1 transmit	SCI1RR	
		DBGEEV	I	DBG external event	—	
		PTS[3]/ KWS[3]	I/O	General-purpose; with interrupt and wakeup	—	
	PS2	MISO0	I/O	SPI0 master in/slave out	SPI0RR	
		(RXD1)	I	SCI1 receive	SCI1RR	
		PTS[2]/ KWS[2]	I/O	General-purpose; with interrupt and wakeup	—	
	PS1	PTUT1	O	PTU trigger 1	—	
		(LPTXD0)	I	LINPHY0 transmit input	S0L0RR2-0	
		TXCAN0	O	MSCAN0 transmit	—	
		TXD1	O	SCI1 transmit	SCI1RR	
		PTS[1]/ KWS[1]	I/O	General-purpose; with interrupt and wakeup	—	
	PS0	PTUT0	O	PTU trigger 0	—	
		(LPRXD0)	O	LINPHY0 receive output	S0L0RR2-0	
		RXCAN0	I	MSCAN0 receive	—	
		RXD1	I	SCI1 receive	SCI1RR	
		PTS[0]/ KWS[0]	I/O	General-purpose; with interrupt and wakeup	—	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Routing Register Bit	Pin Function after Reset
P	PP2	(PWM2)	O	PWM channel 2	PWM32RR PWMPRR	GPIO
		PTP[2]/ KWP[2]	I/O	General-purpose; with interrupt and wakeup	—	
	PP1	$\overline{\text{IRQ}}$	I	Maskable level- or falling edge-sensitive interrupt	—	
		(PWM1)	O	PWM channel 1	PWM10RR PWMPRR	
		PTP[1]/ KWP[1]	I/O	General-purpose; with interrupt and wakeup	—	
	PP0	$\overline{\text{XIRQ}}$	I	Non-maskable level-sensitive interrupt[4]	—	
		FAULT5	I	PMF fault	—	
		ECLK	O	Free-running clock	—	
		(PWM0)	O	PWM channel 0 with over-current interrupt; high-current capable (20 mA)	PWM10RR PWMPRR	
		PTP[0]/ KWP[0]/ EVDD1	I/O	General-purpose; with interrupt and wakeup Switchable external power supply output with over-current interrupt; high-current capable (20 mA)	—	

1. Signals in parentheses denote alternative module routing pins.

2. Function active when $\overline{\text{RESET}}$ asserted.

3. Routable input capture function.

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 with the X bit set (refer to S12ZCPU reference manual) is not available.

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	0	0	SPI0SSRR	SPI0RR	SCI1RR	S0L0RR2-0		
		W								
0x0201	MODRR1	R	0	0	0	0	PWMPRR	PWM54RR	PWM32RR	PWM10RR
		W								
0x0202	MODRR2	R	0	0	0	0	T0IC3RR1-0		T0IC1RR	0
		W								
0x0203– 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	PIMMISC	R	0	0	0	0	0	0	OCPE1	0
		W								
0x020B– 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	0	0	0	0	0	0	0	PTADH0
		W								
0x0281	PTADL	R	PTADL7	PTADL6	PTADL5	PTADL4	PTADL3	PTADL2	PTADL1	PTADL0
		W								
0x0282	PTIADH	R	0	0	0	0	0	0	0	PTIADH0
		W								
0x0283	PTIADL	R	PTIADL7	PTIADL6	PTIADL5	PTIADL4	PTIADL3	PTIADL2	PTIADL1	PTIADL0
		W								
0x0284	DDRADH	R	0	0	0	0	0	0	0	DDRADH0
		W								
0x0285	DDRADL	R	DDRADL7	DDRADL6	DDRADL5	DDRADL4	DDRADL3	DDRADL2	DDRADL1	DDRADL0
		W								
0x0286	PERADH	R	0	0	0	0	0	0	0	PERADH0
		W								
0x0287	PERADL	R	PERADL7	PERADL6	PERADL5	PERADL4	PERADL3	PERADL2	PERADL1	PERADL0
		W								
0x0288	PPSADH	R	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0	PIEADH0
		W								
0x028D	PIEADL	R	PIEADL7	PIEADL6	PIEADL5	PIEADL4	PIEADL3	PIEADL2	PIEADL1	PIEADL0
		W								
0x028E	PIFADH	R	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0	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	0	0	0	0	PTT3	PTT2	PTT1	PTT0
		W								
0x02C1	PTIT	R	0	0	0	0	PTIT3	PTIT2	PTIT1	PTIT0
		W								
0x02C2	DDRT	R	0	0	0	0	DDRT3	DDRT2	DDRT1	DDRT0
		W								
0x02C3	PERT	R	0	0	0	0	PERT3	PERT2	PERT1	PERT0
		W								
0x02C4	PPST	R	0	0	0	0	PPST3	PPST2	PPST1	PPST0
		W								
0x02C5– 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	0	0	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		W								
0x02D1	PTIS	R	0	0	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		W								
0x02D2	DDRS	R	0	0	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		W								
0x02D3	PERS	R	0	0	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		W								
0x02D4	PPSS	R	0	0	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		W								
0x02D5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02D6	PIES	R	0	0	PIES5	PIES4	PIES3	PIES2	PIES1	PIES0
		W								
0x02D7	PIFS	R	0	0	PIFS5	PIFS4	PIFS3	PIFS2	PIFS1	PIFS0
		W								
0x02D8–0x02DE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02DF	WOMS	R	0	0	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		W								
0x02E0–0x02EF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F0	PTP	R	0	0	0	0	0	PTP2	PTP1	PTP0
		W								
0x02F1	PTIP	R	0	0	0	0	0	PTIP2	PTIP1	PTIP0
		W								
0x02F2	DDRP	R	0	0	0	0	0	DDRP2	DDRP1	DDRP0
		W								
0x02F3	PERP	R	0	0	0	0	0	PERP2	PERP1	PERP0
		W								
0x02F4	PPSP	R	0	0	0	0	0	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	OCIE1	0	0	0	0	PIEP2	PIEP1	PIEP0
		W								
0x02F7	PIFP	R	OCIF1	0	0	0	0	PIFP2	PIFP1	PIFP0
		W								
0x02F8–0x02FC	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02FD	RDRP	R	0	0	0	0	0	0	0	RDRP0
		W								
0x02FE–0x02FF	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/write(1)

	7	6	5	4	3	2	1	0
R	0	0	SPI0SSRR	SPI0RR	SCI1RR	S0L0RR2-0		
W								
	—	—	SPI0 $\overline{\text{SS0}}$	SPI0	SCI1	SCI0-LINPHY0 (see Figure 2-2)		
Reset	0	0	0	0	0	0	0	0

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

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

Table 2-2. MODRR0 Routing Register Field Descriptions

Field	Description
5 SPI0SSR	**Module Routing Register** — SPI0 SS0 routing 1 $\overline{\text{SS0}}$ on PAD6 0 $\overline{\text{SS0}}$ based on SPI0RR
4 SPI0RR	**Module Routing Register** — SPI0 routing 1 MISO0 on PT0; MOSI0 on PT1; SCK0 on PT2; $\overline{\text{SS0}}$ on PT3 0 MISO0 on PS2; MOSI0 on PS3; SCK0 on PS4; $\overline{\text{SS0}}$ on PS5
3 SCI1RR	**Module Routing Register** — SCI1 routing 1 TXD1 on PS3; RXD1 on PS2 0 TXD1 on PS1; RXD1 on PS0
2-0 S0L0RR2-0	**Module Routing Register** — SCI0-LINPHY0 routing Selection of SCI0-LINPHY0 interface routing options to support probing and conformance testing. Refer to Figure 2-2 for an illustration and Table 2-3 for preferred settings. SCI0 must be enabled for TXD0 routing to take effect on pins. LINPHY0 must be enabled for LPRXD0 and LPDC0 routings to take effect on pins.

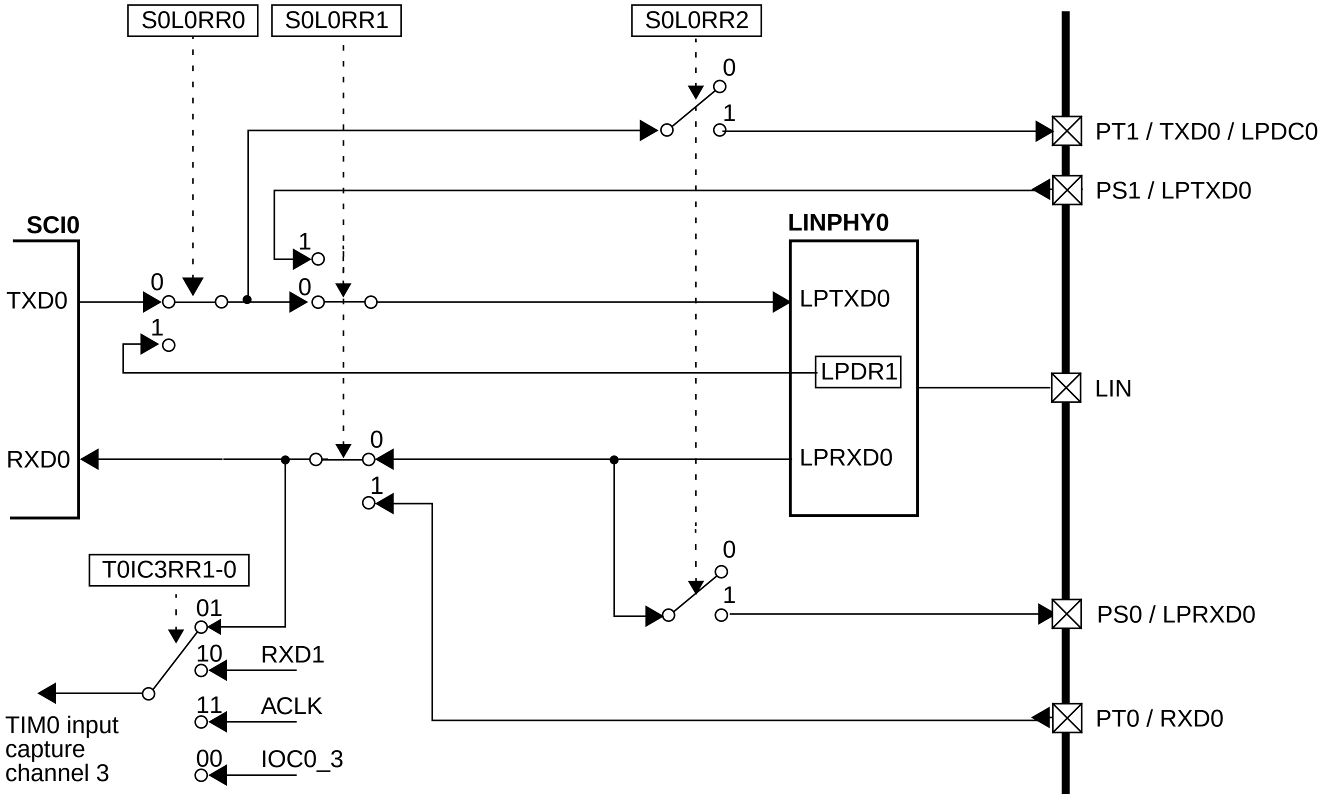
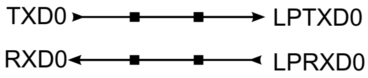
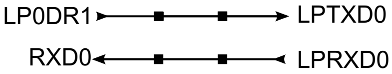
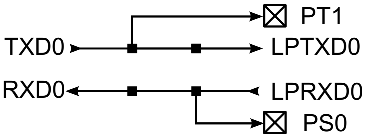
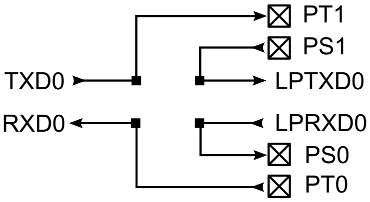


Figure 2-2. SCI0-to-LINPHY0 Routing Options Illustration

Table 2-3. Preferred Interface Configurations

S0L0RR[2:0]	Signal Routing	Description
000		Default setting: SCI0 connects to LINPHY0, interface internal only
001		Direct control setting: LP0DR[LPDR1] register bit controls LPTXD0, interface internal only
100		Probe setting: SCI0 connects to LINPHY0, interface accessible on 2 external pins
110		Conformance test setting: Interface opened and all 4 signals routed externally

NOTE

For standalone usage of SCI0 on external pins set [S0L0RR2:S0L0RR0]=0b110 and disable the LINPHY0 (LPCR[LPE]=0). This releases PS0 and PS1 to other associated functions and maintains TXD0 and RXD0 signals on PT1 and PT0, respectively, if no other function with higher priority takes precedence.

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	0	0	0	0	PWMPRR	PWM54RR	PWM32RR	PWM10RR
W								
	—	—	—	—	PWM probe	PWM4 PWM5 GDU/pins	PWM2 PWM3 GDU/pins	PWM0 PWM1 GDU/pins
Reset	0	0	0	0	0	0	0	0

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

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

Table 2-4. MODRR1 Routing Register Field Descriptions

Field	Description
3 PWMPRR	**Module Routing Register** — PWM probe All six internal PWM outputs can be probed on related external pins. 1 All PWM channels connected to related PWM[5:0] pins 0 All PWM channels disconnected from related PWM[5:0] pins
2 PWM54RR	**Module Routing Register** — PWM4 and PWM5 routing The PWM channel pair can be configured for internal use with the GDU or with its related external pins only. If set the signal routing to the pins is established and the related GDU inputs are forced low. 1 PWM4 to PT1; PWM5 to PT2 0 PWM4 to GDU; PWM5 to GDU
1 PWM32RR	**Module Routing Register** — PWM2 and PWM3 routing The PWM channel pair can be configured for internal use with the GDU or with its related external pins only. If set the signal routing to the pins is established and the related GDU inputs are forced low. 1 PWM2 to PP2; PWM3 to PT0 0 PWM2 to GDU; PWM3 to GDU
0 PWM10RR	**Module Routing Register** — PWM0 and PWM1 routing The PWM channel pair can be configured for internal use with the GDU or with its related external pins only. If set the signal routing to the pins is established and the related GDU inputs are forced low. 1 PWM0 to PP0; PWM1 to PP1 0 PWM0 to GDU; PWM1 to GDU

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	0	0	0	0	T0IC3RR1-0		T0IC1RR	0
W								
					TIM0 IC3		TIM0 IC1	
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 2-5. MODRR2 Routing Register Field Descriptions

Field	Description
3-2 T0IC3RR1-0	**Module Routing Register** — TIM0 IC3 routing One out of four different sources can be selected as input to timer channel 3. 11 TIM0 input capture channel 3 is connected to ACLK 10 TIM0 input capture channel 3 is connected to RXD1 01 TIM0 input capture channel 3 is connected to RXD0 00 TIM0 input capture channel 3 is connected to PT3
1 T0IC1RR	**Module Routing Register** — TIM0 IC1 routing Timer input capture channel 1 can be used to determine the asynchronous commutation event in BLDC motor applications with Hall sensors. An integrated XOR gate supports direct connection of the three sensor inputs to the device. 1 TIM0 input capture channel 1 is connected to logically XORed input signals of pins PT3-1 0 TIM0 input capture channel 1 is connected to PT1

2.3.2.4 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 2-5. ECLK Control Register (ECLKCTL)

1. Read: Anytime
   Write: Anytime

Table 2-6. 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

2.3.2.5 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 2-6. IRQ Control Register (IRQCR)

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

Table 2-7. IRQCR Register Field Descriptions

Field	Description
7 IRQE	**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	**IRQ enable** — 1 $\overline{\text{IRQ}}$ pin is connected to interrupt logic 0 $\overline{\text{IRQ}}$ pin is disconnected from interrupt logic

2.3.2.6 PIM Miscellaneous Register (PIMMISC)

Address 0x020A

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	OCPE1	0
W	▨	▨	▨	▨	▨	▨		▨
Reset	0	0	0	0	0	0	0	0

Figure 2-7. PIM Miscellaneous Register (PIMMISC)

[1] Read: Anytime
Write:Anytime

Table 2-8. PIM Miscellaneous Register Field Descriptions

Field	Description
1 OCPE1	Over-Current Protection Enable — Activate over-current detector on PP0 Refer to Section 2.5.2, "Over-Current Protection on EVDD1" 1 PP0 over-current detector enabled 0 PP0 over-current detector disabled

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

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.

2.3.2.8 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 2-22.

2.3.3.1 Port Data Register

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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-9. Port Data Register Field Descriptions

Field	Description
7-0 PTx7-0	**Port** — **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 2-1), 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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-10. 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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-11. 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

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:	0	0	0	0	0	0	1	1
Ports S:	0	0	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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-12. 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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-13. 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. 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

Read: Anytime

Address 0x028C PIEADH
0x028D PIEADL
0x02D6 PIES

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-14. 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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-15. 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 2.4.5, “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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-16. 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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-17. 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

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 2-22 to determine the implemented bits in the respective register. Unimplemented bits read zero.

Table 2-18. 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 Polarity Select Register (PPSP)

Address 0x02F4 PPSP

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PPSP2	PPS1P	PPSP0
W	▨	▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	0

Figure 2-21. Port P Polarity Select Register

1. Read: Anytime
   Write: Anytime

Table 2-19. Port P Polarity Select Register Field Descriptions

Field	Description
2-1 PPSP	**See Section 2.3.3.5, “Polarity Select Register”**
0 PPSP	**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. This bit also selects the polarity of the active interrupt edge. This bit selects if a high or a low level on FAULT5 generates a fault event in PMF. 1 Pulldown device selected; rising edge selected; active-high level selected on FAULT5 input 0 Pullup device selected; falling edge selected; active-low level selected on FAULT5 input

2.3.4.2 Port P Interrupt Enable Register (PIEP)

Read: Anytime

Address 0x02F6 PIEP

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	OCIE1	0	0	0	0	PIEP2	PIEP1	PIEP0
W		▨	▨	▨	▨			
Reset	0	0	0	0	0	0	0	0

Figure 2-22. Port P Interrupt Enable Register

1. Read: Anytime
   Write: Anytime

Table 2-20. Port P Interrupt Enable Register Field Descriptions

Field	Description
7 OCIE1	**Over-Current Interrupt Enable register** — 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)
2-0 PIEP2-0	**See Section 2.3.3.6, “Port Interrupt Enable Register”**

2.3.4.3 Port P Interrupt Flag Register (PIFP)

Address 0x02F7 PIFP

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	OCIF1	0	0	0	0	PIFP2	PIFP1	PIFP0
W								
Reset	0	0	0	0	0	0	0	0

Figure 2-23. Port P Interrupt Flag Register

1. Read: Anytime
   Write: Anytime, write 1 to clear

Table 2-21. Port P Interrupt Flag Register Field Descriptions

Field	Description
7 OCIF1	**Over-Current Interrupt Flag register** — This flag asserts if an over-current condition is detected on PP0 (Section 2.4.6, “Over-Current Interrupt”). 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
2-0 PIFP2-0	**See Section 2.3.3.7, “Port Interrupt Flag Register”**

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 2-22 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 2-22. 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	0	0	0	0	0	0	0	0	-	-
ADL	7-0	7-0	7-0	7-0	7-0	7-0	7-0	7-0	-	-
T	3-0	3-0	3-0	3-0	3-0	-	-	-	-	-
S	5-0	5-0	5-0	5-0	5-0	5-0	5-0	-	-	5-0
P	2-0	2-0	2-0	2-0	2-0	2-0	2-0	-	0	-

Table 2-23 shows the effect of enabled peripheral features on I/O state and enabled pull devices.

Table 2-23. 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
TIM0 output compare	IOC0_x	Forced output	Forced off
TIM0 input capture	IOC0_x	None(2)	None(3)
SPI0	MISO0, MOSI0, SCK0, $\overline{\text{SS0}}$	Controlled input/output	Forced off if output
SCIx transmitter	TXDx	Forced output	Forced off
SCIx receiver	RXDx	Forced input	None[3]
MSCAN0	TXCAN0	Forced output	Forced off
	RXCAN0	Forced input	Pulldown forced off
S12ZDBG	PDO, PDOCLK	Forced output	Forced off
	DBGEEV	None[2]	None[3]
PTU	PTURE, PTUT1-0	Forced output	Forced off
PWM channel	PWMx	Forced output	Forced off
PMF fault input	FAULT5	Forced input	None[3]

Table 2-23. Effect of Enabled Features

Enabled Feature(1)	Related Signal(s)	Effect on I/O state	Effect on enabled pull device
ADCx	ANx_y	None[2] (4)	None[3]
	VRH, VRL		
AMPx	AMPx, AMPPx, AMPMx	None[2] [4]	None[3]
IRQ	$\overline{\text{IRQ}}$	Forced input	None[3]
XIRQ	$\overline{\text{XIRQ}}$	Forced input	None[3]
LINPHY0	LPTXD0	Forced input	None[3]
	LPRXD0	Forced output	Forced off

[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 (DIENADx) must be set to logic level “1”.

2.4.3 Pin I/O Control

Figure 2-24 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 2-23). 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 2-1 will take precedence on the pin.

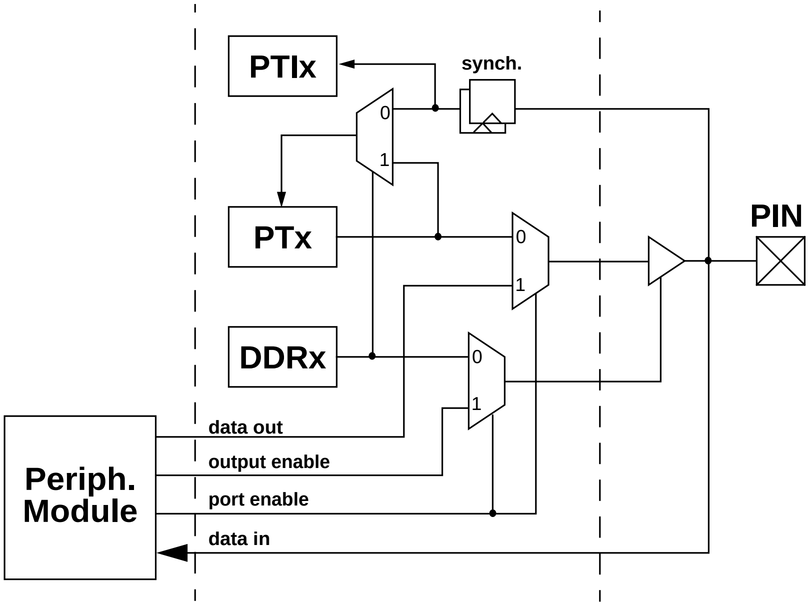


Figure 2-24. Illustration of I/O pin functionality

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 2-24. PIM Interrupt Sources

Module Interrupt Sources	Local Enable
$\overline{\text{XIRQ}}$	None
$\overline{\text{IRQ}}$	IRQCR[IRQEN]
Port S pin interrupt	PIES[PIES5-PIES0]
Port P pin interrupt	PIEP[PIEP2-PIEP0]
Port AD pin interrupt	PIEADH[PIEADH0] PIEADL[PIEADL7-PIEADL0]
PP0 over-current interrupt	PIEP[OCIE1]

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.

2.4.5 Pin interrupts and Key-Wakeup (KWU)

Ports S, P and AD 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 2-25). 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).

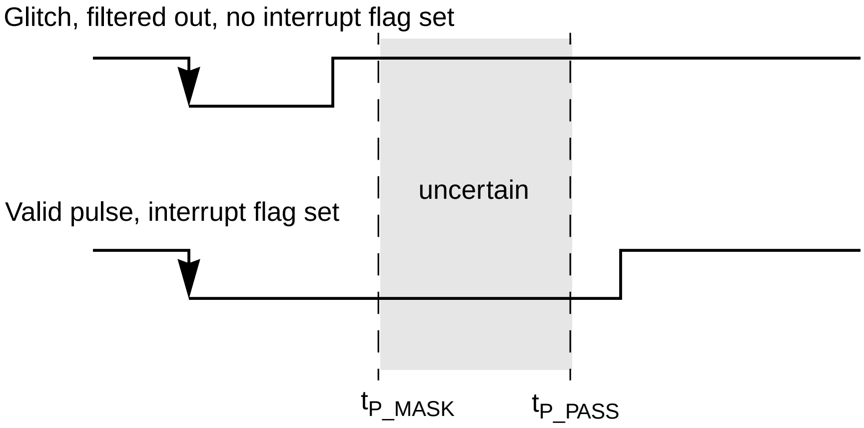


Figure 2-25. Interrupt Glitch Filter (here: active low level selected)

2.4.6 Over-Current Interrupt

In case of an over-current condition on PP0 (see Section 2.5.2, "Over-Current Protection on EVDD1") the over-current interrupt flag PIFP[OCIF1] asserts. This flag generates an interrupt if the enable bit PIEP[OCIE1] is set.

An asserted flag immediately forces the output pin low to protect the device. The flag must be cleared to re-enable the driver.

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 Over-Current Protection on EVDD1

Pin PP0 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 is supplied by the digital pad 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 with a current direction 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 OCPE1 bit in register PIMMISC.

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 2.4.6, "Over-Current Interrupt".

Chapter 3
Memory Mapping Control (S12ZMMCV1)

Table 3-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.03	27 Jul 2012		Corrected Table 3-9
V01.04	27 Jul 2012		Added feature tags
V01.05	6 Aug 2012		Fixed wording
V01.06	12 Feb 2013	Figure 3-8 3.3.2.2/3-122	• Changed “KByte:to “KB” • Corrected the description of the MMCECH/L register •

3.1 Introduction

The S12ZMMC module controls the access to all internal memories and peripherals for the S12ZCPU, and the S12ZBDC module. It also provides access to the RAM for ADCs and the PTU module. The S12ZMMC determines the address mapping of the on-chip resources, regulates access priorities and enforces memory protection. Figure 3-1 shows a block diagram of the S12ZMMC module.

3.1.1 Glossary

Table 3-2. Glossary Of Terms

Term	Definition
MCU	Microcontroller Unit
CPU	S12Z Central Processing Unit
BDC	S12Z Background Debug Controller
ADC	Analog-to-Digital Converter
PTU	Programmable Trigger Unit
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

3.1.2 Overview

The S12ZMMC provides access to on-chip memories and peripherals for the S12ZCPU, the S12ZBDC, the PTU, 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.

3.1.3 Features

- S12ZMMC mode operation control
- Memory mapping for S12ZCPU and S12ZBDC, PTU and ADCs
  - Maps peripherals and memories into a 16 MByte address space for the S12ZCPU, the S12ZBDC, the PTU, and the ADCs
  - 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

3.1.4 Modes of Operation

3.1.4.1 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.

3.1.4.2 Power modes

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

3.1.5 Block Diagram

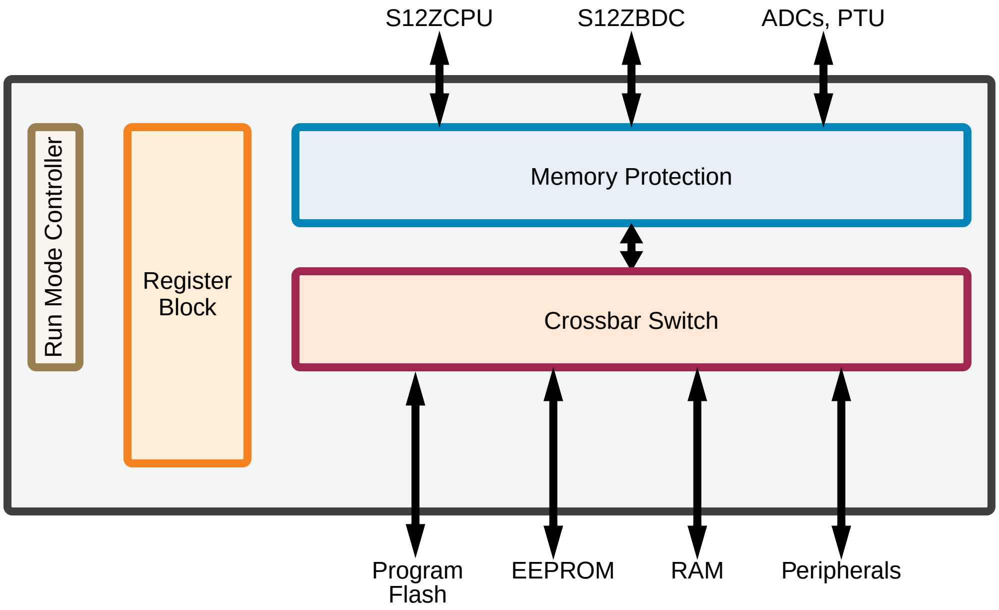


Figure 3-1. S12ZMMC Block Diagram

3.2 External Signal Description

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

Table 3-3. 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.

3.3 Memory Map and Register Definition

3.3.1 Memory Map

A summary of the registers associated with the MMC block is shown in Figure 3-2. 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 3-2. S12ZMMC Register Summary

3.3.2 Register Descriptions

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

3.3.2.1 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 3-3).

▨ = Unimplemented or Reserved

Figure 3-3. Mode Register (MODE)

Read: Anytime.

Write: Only if a transition is allowed (see Figure 3-4).

The MODE register determines the operating mode of the MCU.

CAUTION

Table 3-4. 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 3-4 illustrates the only valid mode transition from special single-chip mode to normal single chip mode.

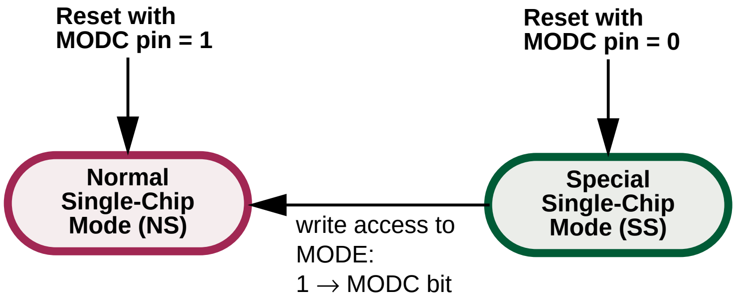


Figure 3-4. Mode Transition Diagram

3.3.2.2 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 3-5. Error Code Register (MMCEC)

Read: Anytime

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

Table 3-5. 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:ADC0 4:ADC1 5:PTU 6-15: 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 access 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.

3.3.2.3 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 3-6. Captured S12ZCPU Condition Code Register (MMCCCRH, MMCCCRL)

Read: Anytime

Write: Never

Table 3-6. 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.

3.3.2.4 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 3-7. Captured S12ZCPU Program Counter (MMCPCH, MMCPCM, MMCPCL)

Read: Anytime

Write: Never

Table 3-7. 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.

3.4 Functional Description

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

3.4.1 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 3-8. The global address space is used by the S12ZCPU, ADCs, PTU, and the S12ZBDC module.

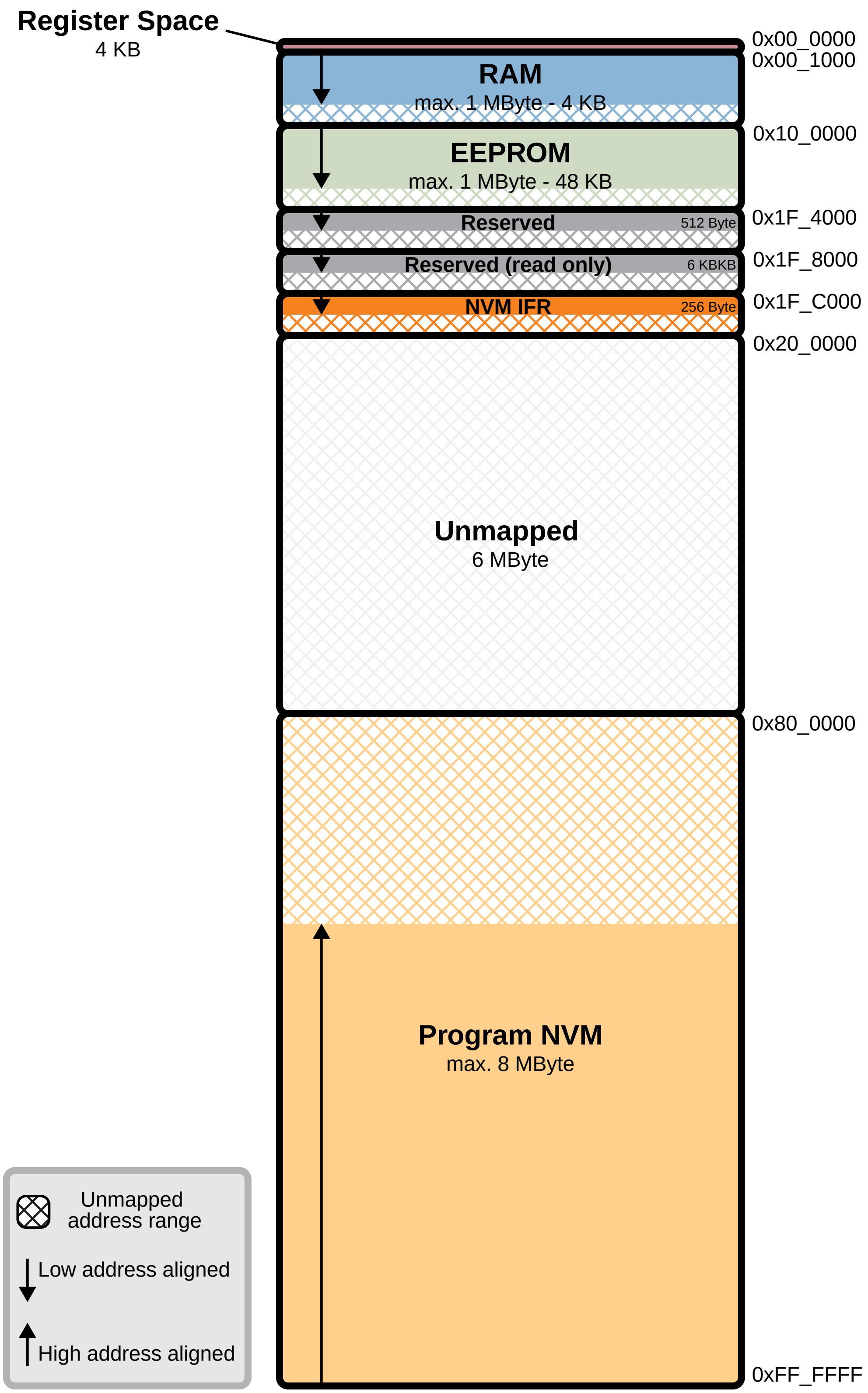


Figure 3-8. Global Memory Map

3.4.2 Illegal Accesses

The S12ZMMC module monitors all memory traffic for illegal accesses. See Table 3-9 for a complete list of all illegal accesses.

Table 3-9. Illegal memory accesses

		S12ZCPU	S12ZBDC	ADCs and PTU
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 an ADC or PTU module trigger error interrupts. See ADC and PTU 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.

3.4.3 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, ADC or PTU 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 4
Interrupt (S12ZINTV0)

Table 4-1. 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

4.1 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.

4.1.1 Glossary

The following terms and abbreviations are used in the document.

Table 4-2. 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

4.1.2 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).
- One X-bit maskable interrupt vector request associated with $\overline{\text{XIRQ}}$ (at address vector base[1] + 0x0001D8).

[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 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.

4.1.3 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 4.5.3, “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 4.5.3, “Wake Up from Stop or Wait Mode” for details.

4.1.4 Block Diagram

Figure 4-1 shows a block diagram of the INT module.

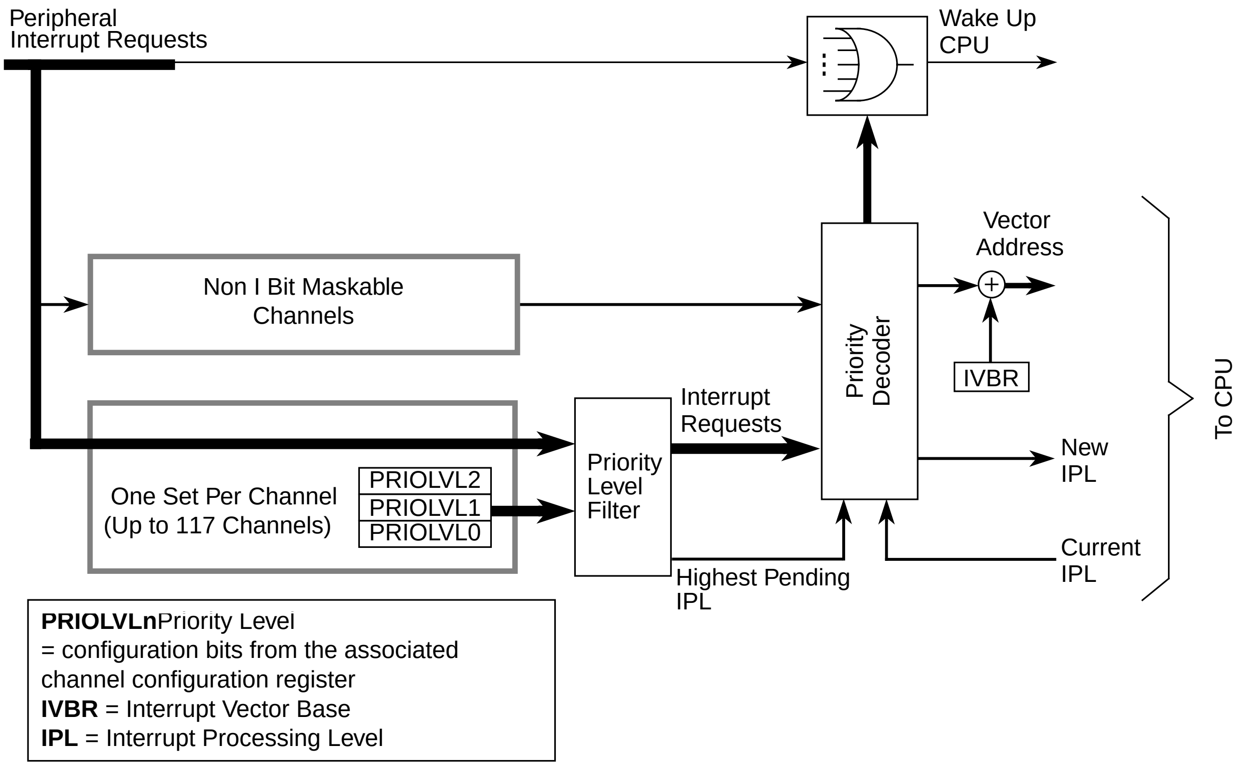


Figure 4-1. INT Block Diagram

4.2 External Signal Description

The INT module has no external signals.

4.3 Memory Map and Register Definition

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

4.3.1 Module Memory Map

Table 4-3 gives an overview over all INT module registers.

Table 4-3. 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 4-3. 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

4.3.2 Register Descriptions

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

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000010	IVBR	R	IVB_ADDR[15:8]							
		W								
0x000011		R	IVB_ADDR[7:1]							0
		W								▨
0x000017	INT_CFADDR	R	0	INT_CFADDR[6:3]				0	0	0
		W	▨					▨	▨	▨
0x000018	INT_CFDATA0	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x000019	INT_CFDATA1	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x00001A	INT_CFDATA2	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x00001B	INT_CFDATA3	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			
0x00001C	INT_CFDATA4	R	0	0	0	0	0	PRIOLVL[2:0]		
		W	▨	▨	▨	▨	▨			

▨ = Unimplemented or Reserved

Figure 4-2. 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 4-2. INT Register Summary

4.3.2.1 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 4-3. Interrupt Vector Base Register (IVBR)

Read: Anytime

Write: Anytime

Table 4-4. 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).

4.3.2.2 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 4-4. Interrupt Configuration Address Register (INT_CFADDR)

Read: Anytime

Write: Anytime

Table 4-5. 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.

4.3.2.3 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 4-5. 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 4-6. 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 4-7. 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 4-8. 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 4-9. 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 4-10. 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 4-11. 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 4-12. 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 4-6. 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 4-7 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 4-7. 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

Table 4-7. Interrupt Priority Levels

Priority	PRIOLVL2	PRIOLVL1	PRIOLVL0	Meaning
	1	0	1	Priority level 5
	1	1	0	Priority level 6
high	1	1	1	Priority level 7

4.4 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.

4.4.1 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.

4.4.2 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.

4.4.2.1 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.

4.4.3 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)).

4.4.4 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

4.4.5 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 4-8. 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 4-8. 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

4.4.6 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 4-13 illustrates the vector table entry format.

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

Figure 4-13. Interrupt Vector Table Entry

4.5 Initialization/Application Information

4.5.1 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).

4.5.2 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 4-14 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

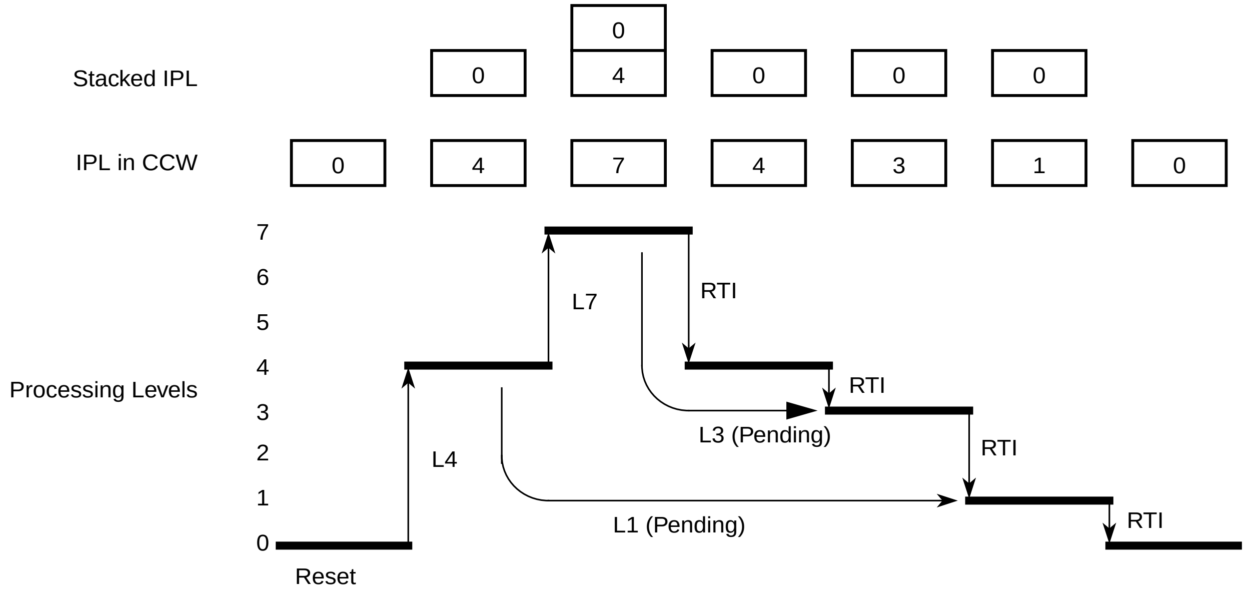


Figure 4-14. Interrupt Processing Example

4.5.3 Wake Up from Stop or Wait Mode

4.5.3.1 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 5
Background Debug Controller (S12ZBDCV2)

Table 5-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V2.04	03.Dec.2012	Section 5.1.3.3	Included BACKGROUND/ Stop mode dependency
V2.05	22.Jan.2013	Section 5.3.2.2	Improved NORESP description and added STEP1/ Wait mode dependency
V2.06	22.Mar.2013	Section 5.3.2.2	Improved NORESP description of STEP1/ Wait mode dependency
V2.07	11.Apr.2013	Section 5.1.3.3.1	Improved STOP and BACKGROUND interdepency description
V2.08	31.May.2013	Section 5.4.4.4 Section 5.4.7.1	Removed misleading WAIT and BACKGROUND interdepency description Added subsection dedicated to Long-ACK
V2.09	29.Aug.2013	Section 5.4.4.12	Noted that READ_DBGTB is only available for devices featuring a trace buffer.
V2.10	21.Oct.2013	Section 5.1.3.3.2	Improved description of NORESP dependence on WAIT and BACKROUND

5.1 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.

5.1.1 Glossary

Table 5-2. 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

5.1.2 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

5.1.3 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.

5.1.3.1 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.

5.1.3.2 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.

5.1.3.3 Low-Power Modes

5.1.3.3.1 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 5-3

Table 5-3. 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 5-5) 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.

5.1.3.3.2 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.

5.1.4 Block Diagram

A block diagram of the BDC is shown in Figure 5-1.

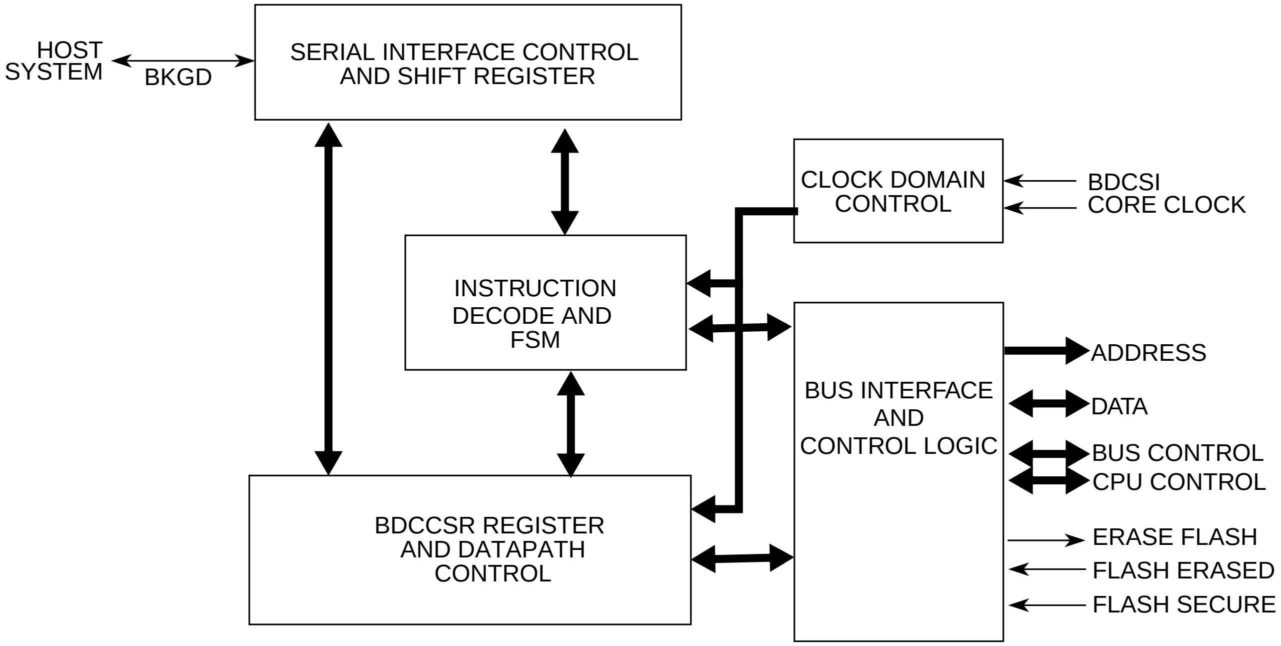


Figure 5-1. BDC Block Diagram

5.2 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 5.4.6” for more details.

5.3 Memory Map and Register Definition

5.3.1 Module Memory Map

Table 5-4 shows the BDC memory map.

Table 5-4. BDC Memory Map

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

5.3.2 Register Descriptions

The BDC registers are shown in Figure 5-2. 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 5-2. BDC Register Summary

5.3.2.1 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 5-3. 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 5-5. 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 5-7. 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 5-3). 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.

5.3.2.2 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 5-4. 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 5-6. 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 5-6. 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 5.4.5.1). 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 5-6. 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.

5.4 Functional Description

5.4.1 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.

5.4.2 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.

5.4.3 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 5-5. 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.

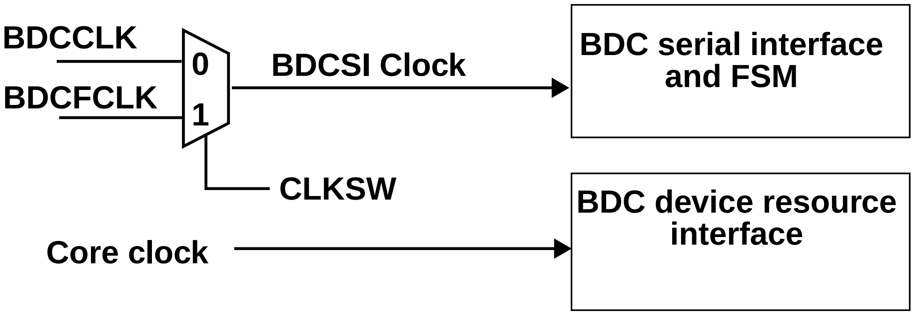


Figure 5-5. Clock Switch

5.4.4 BDC Commands

BDC commands can be classified into three types as shown in Table 5-7.

Table 5-7. 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 5-8 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 5-8. 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 5-8. 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 5-8. 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.

5.4.4.1 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).

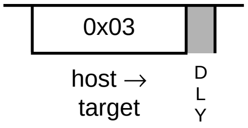
The host measures the low time of this 128-cycle SYNC response pulse and determines the correct speed for subsequent BDC communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the serial protocol can easily tolerate this speed 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 5.4.8.

5.4.4.2 ACK_DISABLE

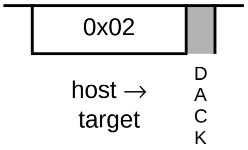
Disable host/target handshake protocol **Always Available**



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.

5.4.4.3 ACK_ENABLE

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

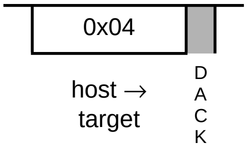


Enables the hardware handshake protocol in the serial communication. The hardware handshake is implemented by an acknowledge (ACK) pulse issued by the target MCU in response to a host command. The ACK_ENABLE command is interpreted and executed in the BDC logic without the need to interface with the CPU. 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 5-8 indicates which commands support the ACK hardware handshake protocol.

For additional information about the hardware handshake protocol, refer to Section 5.4.7," and Section 5.4.8."

5.4.4.4 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.

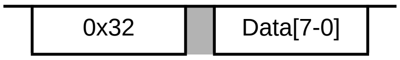
A BACKGROUND command issued during stop or wait modes cannot immediately force active BDM because the WAI instruction does not end until an interrupt occurs. For the detailed mode dependency description refer to Section 5.1.3.3.

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.

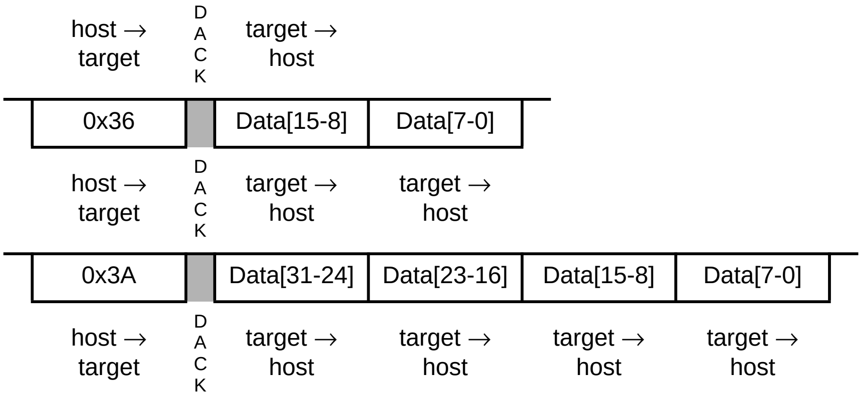
5.4.4.5 DUMP_MEM.sz, DUMP_MEM.sz_WS

DUMP_MEM.sz

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

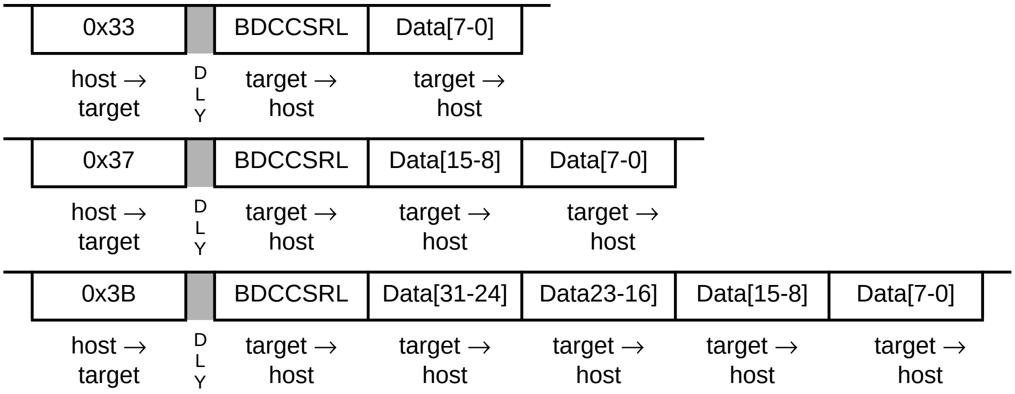


DUMP_MEM.sz



DUMP_MEM.sz_WS

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



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 5.4.5.2".

NOTE

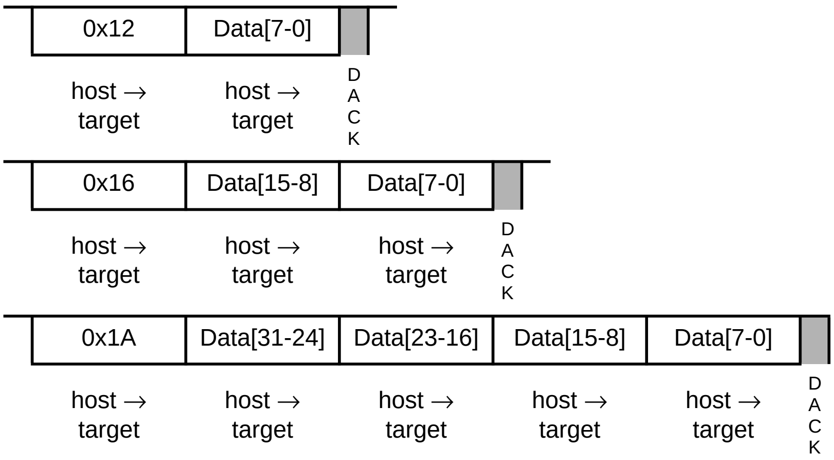
> DUMP_MEM{_WS} is a valid command only when preceded by SYNC, NOP, READ_MEM{_WS}, or another DUMP_MEM{_WS} command. Otherwise, an illegal command response is returned, setting the ILLCMD bit. NOP can be used for inter-command padding without corrupting the address pointer.

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.

5.4.4.6 FILL_MEM.sz, FILL_MEM.sz_WS

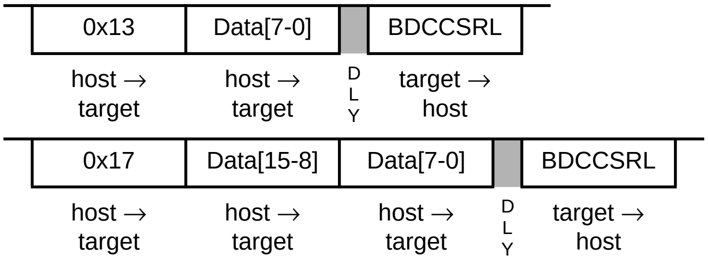
FILL_MEM.sz

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

0x1B	Data[31-24]	Data[23-16]	Data[15-8]	Data[7-0]		BDCCSRL
host → target	host → target	host → target	host → target	host → target	DLY	target → host

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 5.4.5.2"

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.

5.4.4.7 GO

Go **Non-intrusive**

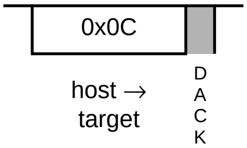
0x08	
host → target	DACK

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.

5.4.4.8 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.

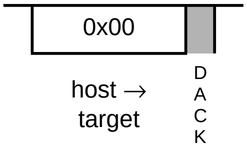
If a GO_UNTIL is not acknowledged then a SYNC command must be issued to end the pending GO_UNTIL.

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.

5.4.4.9 NOP

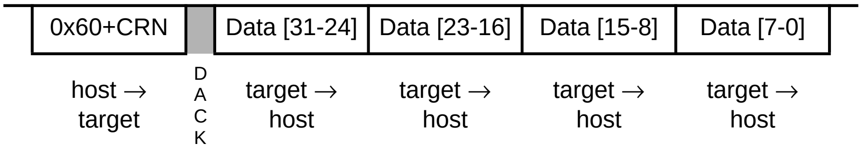
No operation **Active Background**



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

5.4.4.10 READ_Rn

Read CPU register **Active Background**



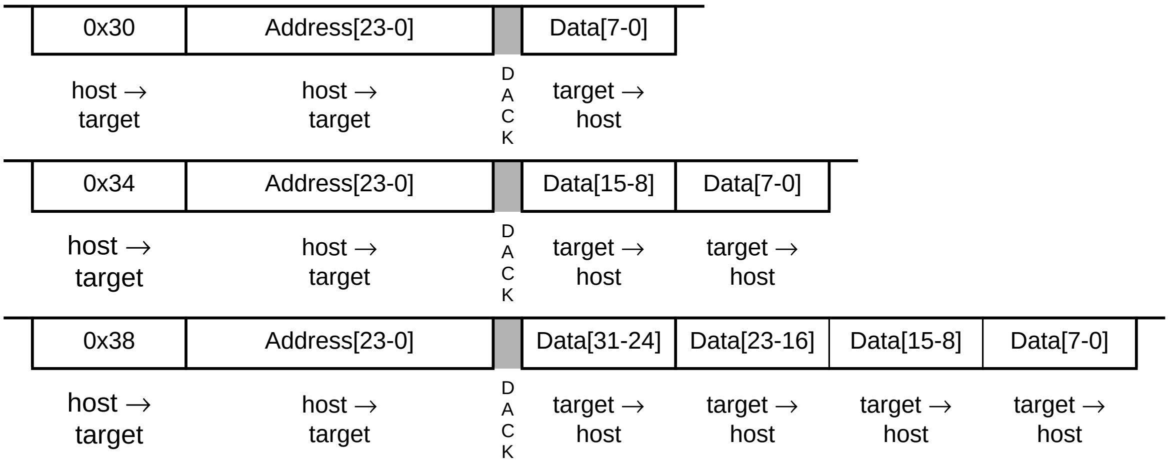
This command reads the selected CPU registers and returns the 32-bit result. Accesses to CPU registers are always 32-bits wide, regardless of implemented register width. Bytes that are not implemented return zero. The register is addressed through the CPU register number (CRN). See Section 5.4.5.1 for 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.

5.4.4.11 READ_MEM.sz, READ_MEM.sz_WS

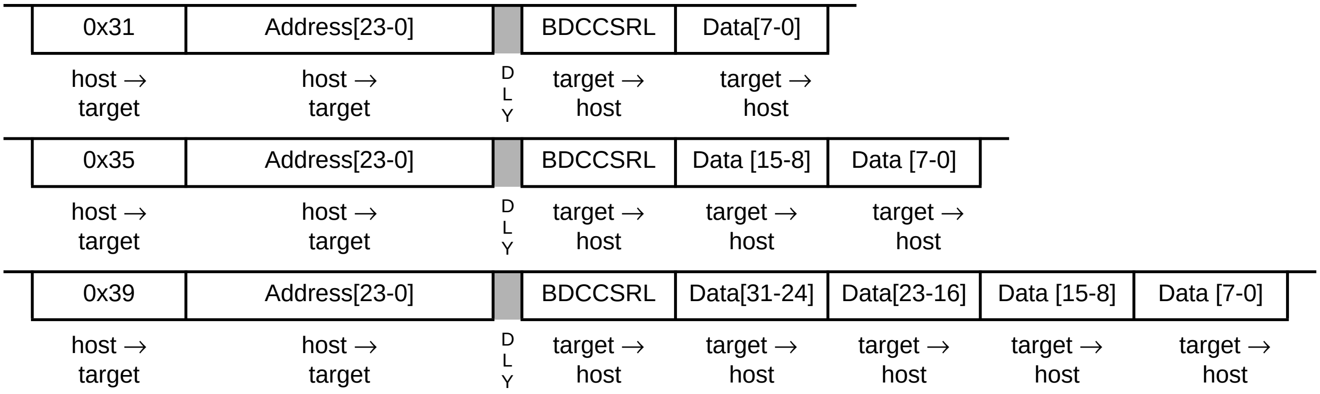
READ_MEM.sz

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



READ_MEM.sz_WS

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



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 5.4.5.2". 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.

5.4.4.12 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.

5.4.4.13 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 5.4.5.2”. 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.

5.4.4.14 READ_BDCCSR

Read BDCCSR Status Register **Always Available**

0x2D		BDCCSR [15:8]	BDCCSR [7-0]
host → target	DLY	target → host	target → host

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

5.4.4.15 SYNC_PC

Sample current PC **Non-intrusive**

0x01		PC data[23–16]	PC data[15–8]	PC data[7–0]
host → target	DACK	target → host	target → host	target → host

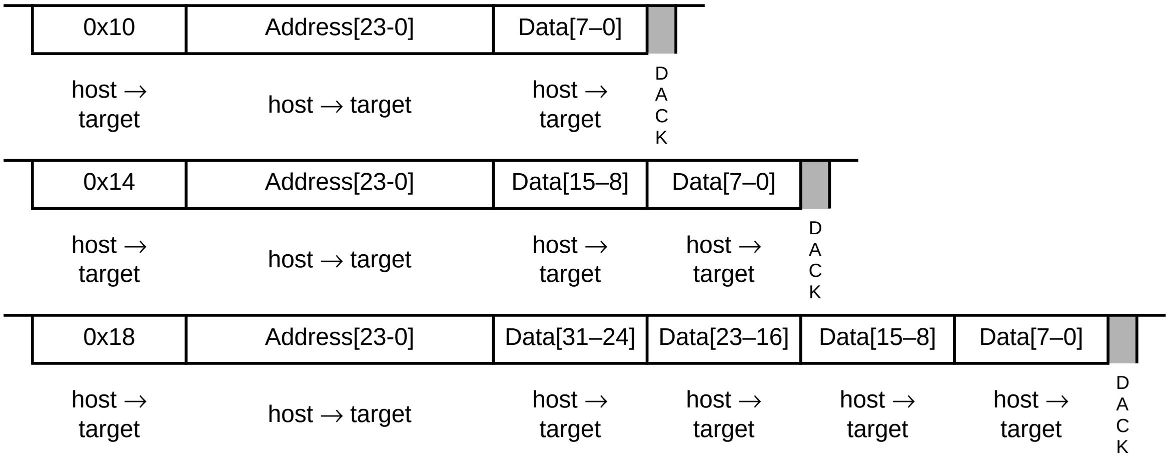
This command returns the 24-bit CPU PC value to the host. Unsuccessful SYNC_PC accesses return 0xEE for each byte. If enabled, an ACK pulse is driven before the data bytes are transmitted. The value of 0xEE is returned if a timeout occurs, whereby NORESP is set. This can occur if the CPU is executing 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.

5.4.4.16 WRITE_MEM.sz, WRITE_MEM.sz_WS

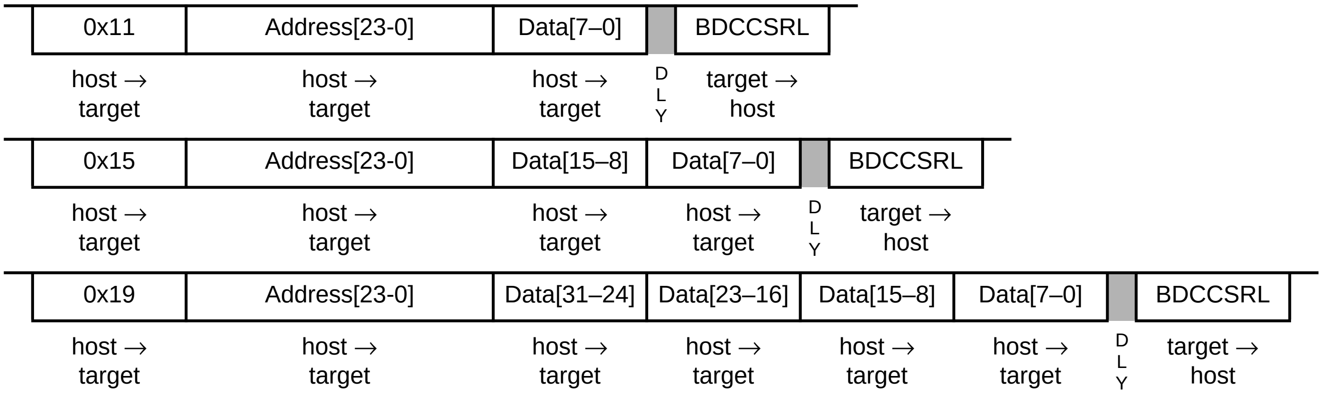
WRITE_MEM.sz

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



WRITE_MEM.sz_WS

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

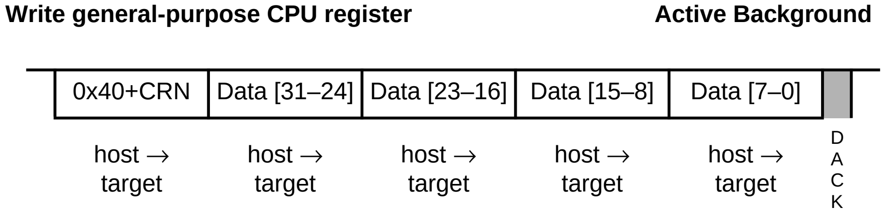


Write data to the specified memory address. The address is transmitted as three 8-bit packets (msb to lsb) immediately after the command.

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 5.4.5.2”.

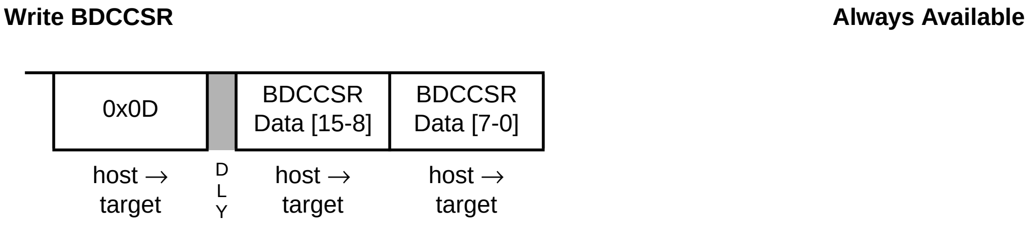
5.4.4.17 WRITE_Rn



If the device is in active BDM, this command writes the 32-bit operand to the selected CPU general-purpose register. See Section 5.4.5.1 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.

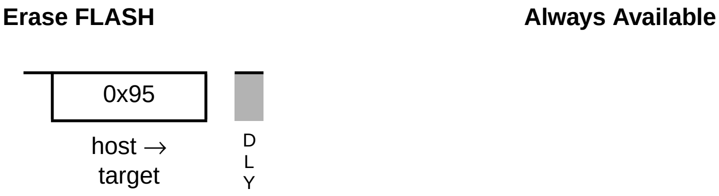
If the device is not in active BDM, this command is rejected as an illegal operation, the ILLCMD bit is set and no operation is performed.

5.4.4.18 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.

5.4.4.19 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.

5.4.4.20 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 5.1.3.3.2.

5.4.5 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

5.4.5.1 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 5-9. 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 5-9. 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

5.4.5.2 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 5-10. 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 5-10. 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			

5.4.5.2.1 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 5-11, 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 5-11. 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	▨	▨

5.4.5.2.2 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 5-12 shows some examples of this.

Table 5-12. 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

5.4.6 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 5-6 and that of target-to-host in Figure 5-7 and Figure 5-8. 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 5-6. 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 5-6 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.

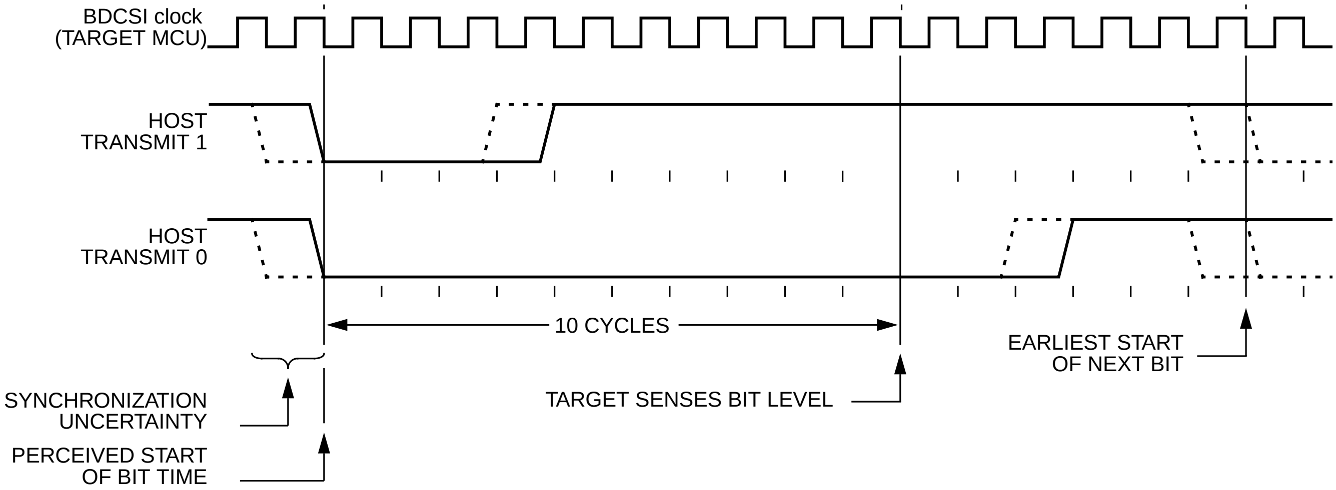


Figure 5-6. BDC Host-to-Target Serial Bit Timing

Figure 5-7 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.

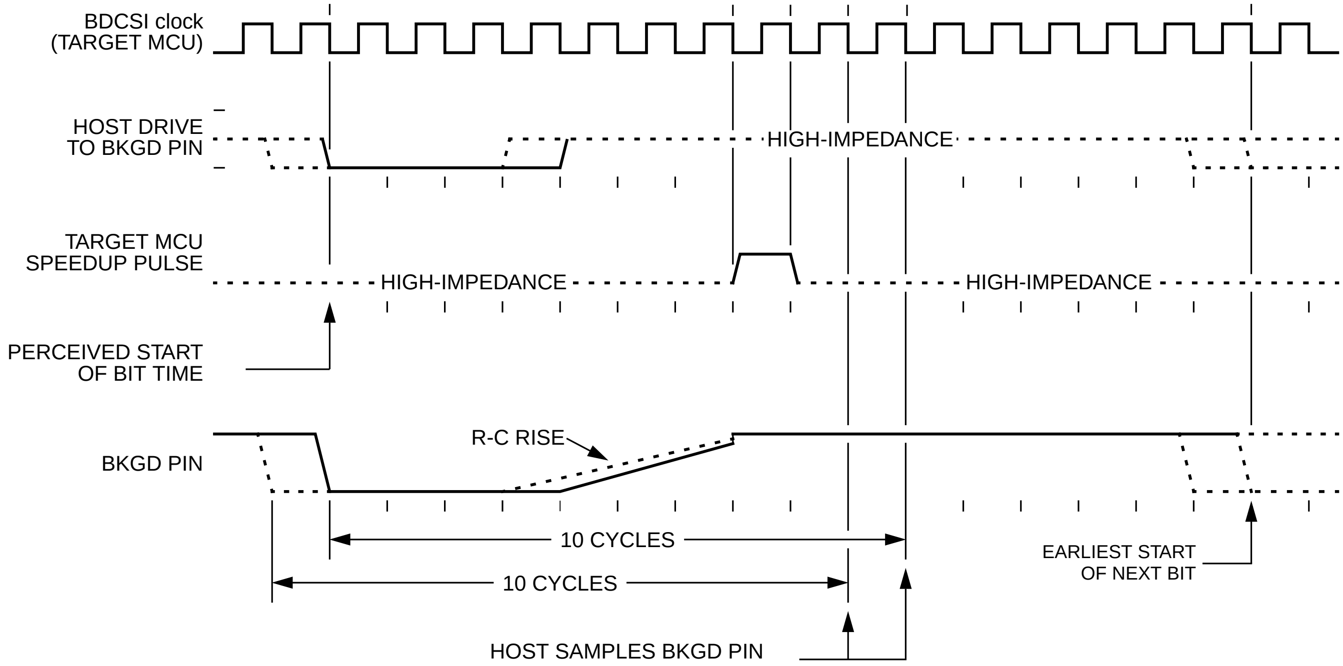


Figure 5-7. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 5-8 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.

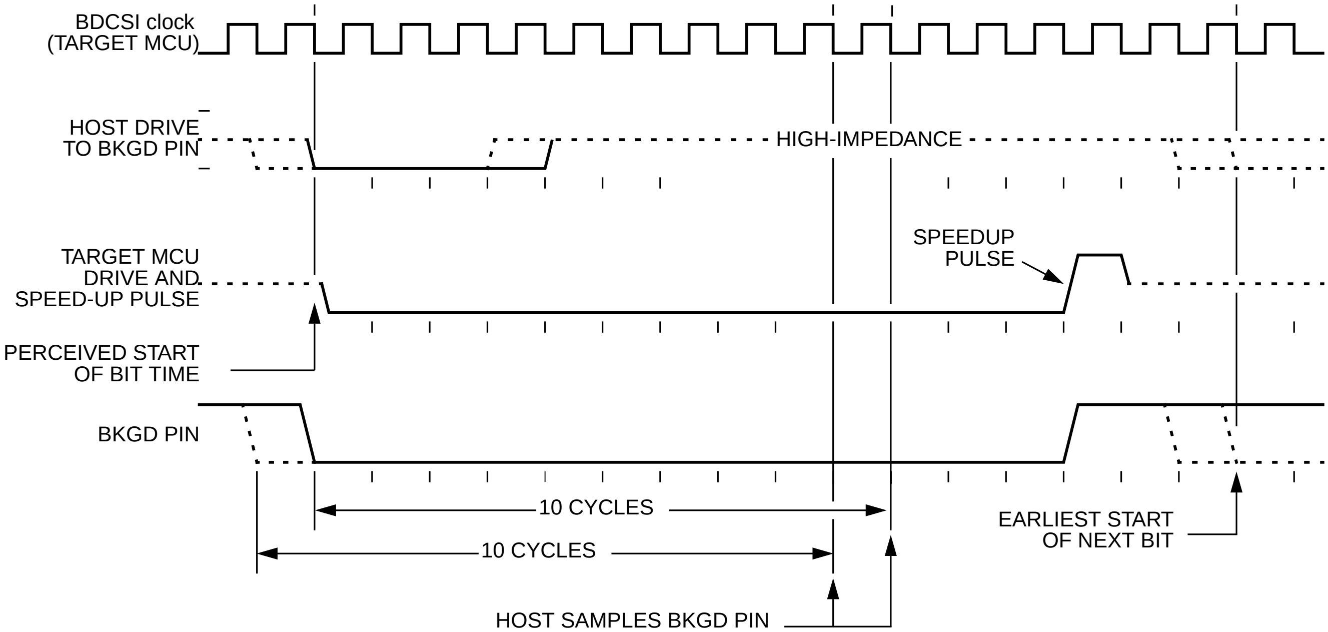


Figure 5-8. BDC Target-to-Host Serial Bit Timing (Logic 0)

5.4.7 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 5-9). 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.

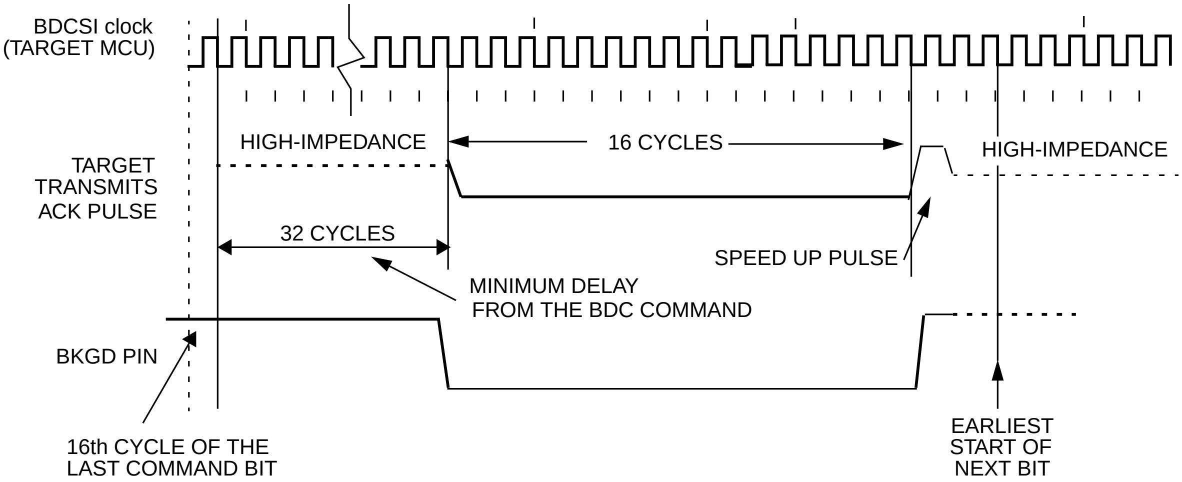


Figure 5-9. Target Acknowledge Pulse (ACK)

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 5-9 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 5-10 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.

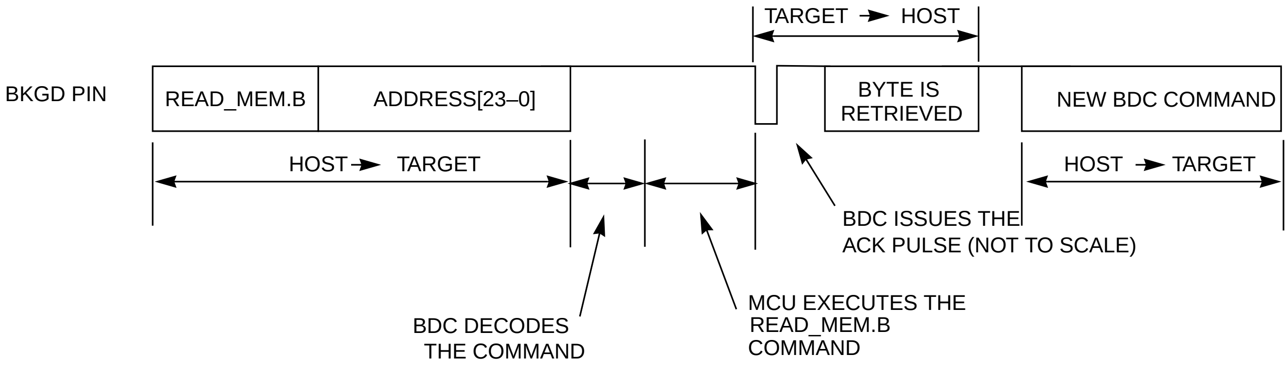


Figure 5-10. Handshake Protocol at Command Level

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.

5.4.7.1 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.

5.4.8 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 5.4.4.1”, 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 5.4.4.1”.

Figure 5-11 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.

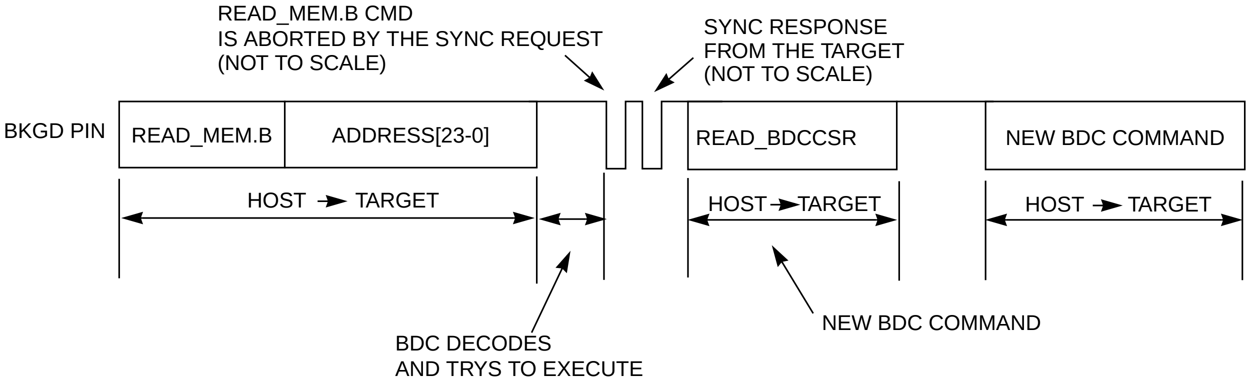


Figure 5-11. ACK Abort Procedure at the Command Level (Not To Scale)

Figure 5-12 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.

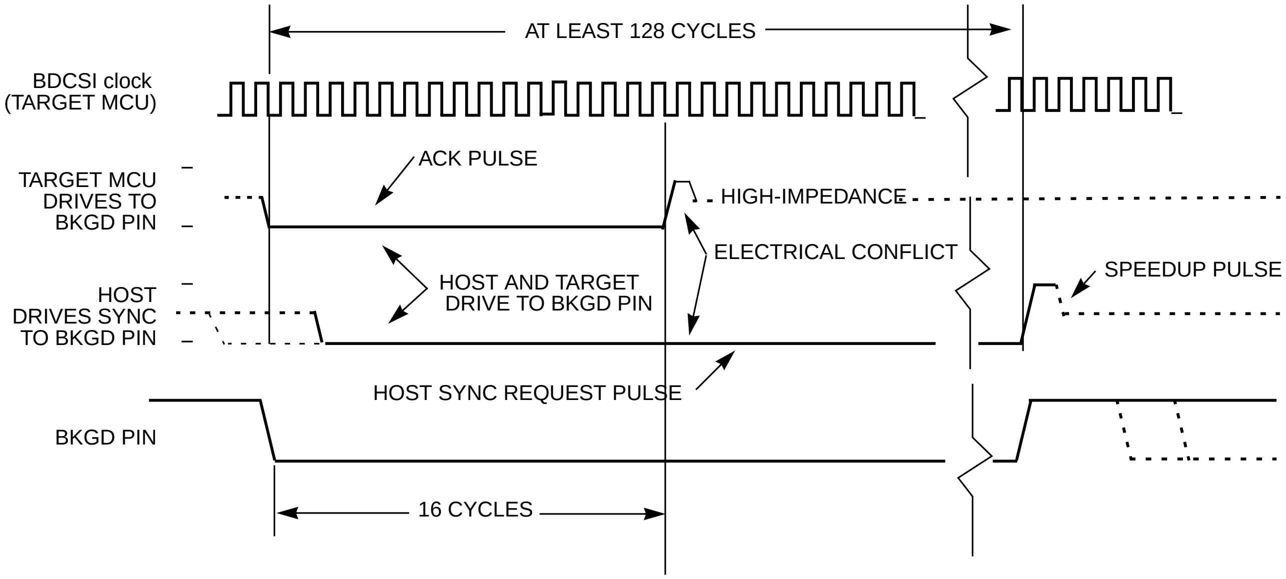


Figure 5-12. ACK Pulse and SYNC Request Conflict

5.4.9 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 5.5.1/5-180). 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.

5.4.10 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.

5.4.11 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.

5.5 Application Information

5.5.1 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 6
S12Z Debug (S12ZDBGV2) Module

Table 6-1. Revision History Table

Revision Number	Revision Date	Sections Affected	Description Of Changes
2.04	19.APR.2012	Section 6.4.5.2.1	Documented DBGTB read dependency on PROFILE bit
2.05	23.MAY.2012	General	Formatting changes to support DBGV3 from single source
2.06	10.SEP.2012	Section 6.4.5.3	Added NOTE about PC trace buffer entries for Comp D timestamps
2.07	18.OCT.2012	General	Formatting corrections
2.08	16.NOV.2012	Section 6.5.1	Modified step over breakpoint information
2.09	19.DEC.2012	General	Formatting corrections
2.10	28.JUN.2013	General Section 6.3.2.21 Section 6.3.2.1 Section 6.3.2.5	Emphasized need to set TSOURCE for tracing or profiling Corrected DBGCDM write access dependency Corrrected ARM versus PTACT dependency Modified DBGTBH read access dependencies
2.11	15.JUL.2013	Section 6.3.2	Added explicit names to state control register bit fields

6.1 Introduction

The DBG module provides on-chip breakpoints and trace buffer 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

Table 6-2. Glossary Of Terms

Term	Definition
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
Data Line	64-bit data entity
CPU	S12Z CPU module
Trigger	A trace buffer input that triggers tracing start, end or mid point

6.1.2 Overview

The comparators monitor the bus activity of the CPU. A single comparator match or a series of matches can trigger bus tracing and/or generate breakpoints. A state sequencer determines if the correct series of matches occurs. Similarly an external event can trigger bus tracing and/or generate breakpoints.

The trace buffer is visible through a 2-byte window in the register address map and can be read out using standard 16-bit word reads.

6.1.3 Features

- Four comparators (A, B, C, and D)
  - Comparators A and C compare the full address bus and full 32-bit data bus
  - Comparators A and C feature 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)
- Trace control
  - Tracing session triggered by state sequencer
  - Begin, End, and Mid alignment of tracing to trigger
- Four trace modes
  - Normal: change of flow (COF) PC information is stored (see Section 6.4.5.2.1) for change of flow definition.
  - Loop1: same as Normal but inhibits consecutive duplicate source address entries
  - Detail: address and data for all read/write access cycles are stored
  - Pure PC: All program counter addresses are stored.
- 2 Pin (data and clock) profiling interface
  - Output of code flow information

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

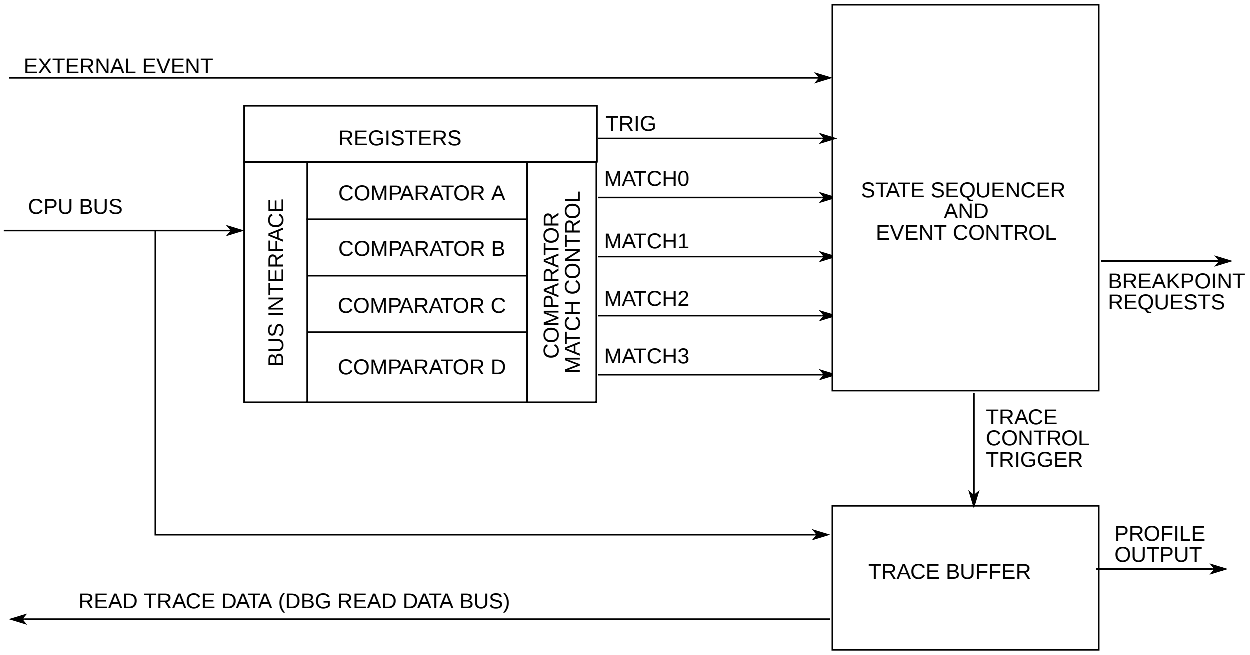


Figure 6-1. Debug Module Block Diagram

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, or trace buffer entry, or to gate trace buffer entries. With the external event function enabled, a falling edge at the external event pin constitutes an event. Rising edges have no effect. If configured for gating trace buffer entries, then a low level at the pin allows entries, but a high level suppresses entries. 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.

6.2.2 Profiling Output

The DBG module features a profiling data output signal PDO. The mapping of this signal to a device pin is specified in the device specific documentation. The device pin is enabled for profiling by setting the PDOE bit. The profiling function can be enabled by the PROFILE bit in the DBGTCRL control register. This signal is output only and provides a serial, encoded data stream that can be used by external development tools to reconstruct the internal CPU code flow, as specified in Section 6.4.6. During code profiling the device PDOCLK output is used as a clock signal.

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	EEVE	
		W		TRIG						
0x0101	DBGC2	R	0	0	0	0	CDCM		ABCM	
		W								

Figure 6-2. Quick Reference to DBG Registers

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0102	DBGTCRH	R	reserved	TSOURCE	TRANGE		TRCMOD		TALIGN	
		W								
0x0103	DBGTCRL	R	0	0	0	0	DSTAMP	PDOE	PROFILE	STAMP
		W								
0x0104	DBGTB	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0105	DBGTB	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0106	DBGCNT	R	0	CNT						
		W								
0x0107	DBGSCR1	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0108	DBGSCR2	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0109	DBGSCR3	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x010A	DBGEFR	R	PTBOVF	TRIGF	0	EEVF	ME3	ME2	ME1	ME0
		W								
0x010B	DBGSR	R	TBF	0	0	PTACT	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								

Figure 6-2. Quick Reference to DBG Registers

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x011B	DBGAD3	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x011C	DBGADM0	R W	Bit 31	30	29	28	27	26	25	Bit 24
0x011D	DBGADM1	R W	Bit 23	22	21	20	19	18	17	Bit 16
0x011E	DBGADM2	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x011F	DBGADM3	R W	Bit 7	6	5	4	3	2	1	Bit 0
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 W	DBGBA[23:16]							
0x0126	DBGBAM	R W	DBGBA[15:8]							
0x0127	DBGBAL	R W	DBGBA[7:0]							
0x0128- 0x012F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0130	DBGCCTL	R	0	NDB	INST	0	RW	RWE	reserved	COMPE
		W								
0x0131- 0x0134	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0135	DBGCAH	R W	DBGCA[23:16]							
0x0136	DBGCAM	R W	DBGCA[15:8]							
0x0137	DBGCAL	R W	DBGCA[7:0]							
0x0138	DBGCD0	R W	Bit 31	30	29	28	27	26	25	Bit 24
0x0139	DBGCD1	R W	Bit 23	22	21	20	19	18	17	Bit 16
0x013A	DBGCD2	R W	Bit 15	14	13	12	11	10	9	Bit 8

Figure 6-2. Quick Reference to DBG Registers

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x013B	DBGCD3	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x013C	DBGCDM0	R W	Bit 31	30	29	28	27	26	25	Bit 24
0x013D	DBGCDM1	R W	Bit 23	22	21	20	19	18	17	Bit 16
0x013E	DBGCDM2	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x013F	DBGCDM3	R W	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 TRIG	reserved	BDMBP	BRKCPU	reserved	EEVE	
Reset	0	0	0	0	0	0	0	0

Figure 6-3. Debug Control Register (DBGC1)

Read: Anytime

Write: Bit 7 Anytime with the exception that it cannot be set if PTACT is set. 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 and PTACT is clear.

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. If tracing is enabled, the breakpoint is generated on completion of the tracing session. If tracing is not enabled, the breakpoint is generated immediately. Please refer to Section 6.4.7 for further details. 0 Breakpoints disabled 1 Breakpoints enabled
1–0 EEVE	**External Event Enable** — The EEVE bits configure the external event function. Table 6-4 explains the bit encoding.

Table 6-4. EEVE Bit Encoding

EEVE	Description
00	External event function disabled
01	External event forces a trace buffer entry if tracing is enabled
10	External event is mapped to the state sequencer, replacing comparator channel 3
11	External event pin gates trace buffer entries

6.3.2.2 Debug Control Register2 (DBGC2)

Address: 0x0101

	7	6	5	4	3	2	1	0
R	0	0	0	0	CDCM		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 and PTACT is clear.

This register configures the comparators for range matching.

Table 6-5. DBGC2 Field Descriptions

Field	Description
3–2 CDCM[1:0]	**C and D Comparator Match Control** — These bits determine the C and D comparator match mapping as described in Table 6-6.
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 6-7.

Table 6-6. CDCM Encoding

CDCM	Description
00	Match2 mapped to comparator C match....... Match3 mapped to comparator D match.
01	Match2 mapped to comparator C/D inside range....... Match3 disabled.
10	Match2 mapped to comparator C/D outside range....... Match3 disabled.
11	Reserved(1)

1. Currently defaults to Match2 mapped to inside range: Match3 disabled.

Table 6-7. 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 Trace Control Register High (DBGTCRH)

Address: 0x0102

	7	6	5	4	3	2	1	0
R W	reserved	TSOURCE	TRANGE		TRCMOD		TALIGN	
Reset	0	0	0	0	0	0	0	0

Figure 6-5. Debug Trace Control Register (DBGTCRH)

Read: Anytime.

Write: Anytime the module is disarmed and PTACT is clear.

WARNING

DBGTCR[7] is reserved. Setting this bit maps the tracing to an unimplemented bus, thus preventing proper operation.

This register configures the trace buffer for tracing and profiling.

Table 6-8. DBGTCRH Field Descriptions

Field	Description
6 TSOURCE	**Trace Control Bits** — The TSOURCE enables the tracing session. 0 No CPU tracing/profiling selected 1 CPU tracing/profiling selected
5–4 TRANGE	**Trace Range Bits** — The TRANGE bits allow filtering of trace information from a selected address range when tracing from the CPU in Detail mode. These bits have no effect in other tracing modes. To use a comparator for range filtering, the corresponding COMPE bit must remain cleared. If the COMPE bit is set then the comparator is used to generate events and the TRANGE bits have no effect. See Table 6-9 for range boundary definition.
3–2 TRCMOD	**Trace Mode Bits** — See Section 6.4.5.2 for detailed Trace Mode descriptions. In Normal Mode, change of flow information is stored. In Loop1 Mode, change of flow information is stored but redundant entries into trace memory are inhibited. In Detail Mode, address and data for all memory and register accesses is stored. See Table 6-10.
1–0 TALIGN	**Trigger Align Bits** — These bits control whether the trigger is aligned to the beginning, end or the middle of a tracing or profiling session. See Table 6-11.

Table 6-9. TRANGE Trace Range Encoding

TRANGE	Tracing Range
00	Trace from all addresses (No filter)
01	Trace only in address range from $00000 to Comparator D
10	Trace only in address range from Comparator C to $FFFFFF
11	Trace only in range from Comparator C to Comparator D

Table 6-10. TRCMOD Trace Mode Bit Encoding

TRCMOD	Description
00	Normal

Table 6-10. TRCMOD Trace Mode Bit Encoding

TRCMOD	Description
01	Loop1
10	Detail
11	Pure PC

Table 6-11. TALIGN Trace Alignment Encoding

TALIGN	Description
00	Trigger ends data trace
01	Trigger starts data trace
10	32 lines of data trace follow trigger
11(1)	Reserved

1. Tracing/Profiling disabled.

6.3.2.4 Debug Trace Control Register Low (DBGTCRL)

Address: 0x0103

	7	6	5	4	3	2	1	0
R	0	0	0	0	DSTAMP	PDOE	PROFILE	STAMP
W	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-6. Debug Trace Control Register Low (DBGTCRL)

Read: Anytime.

Write: Anytime the module is disarmed and PTACT is clear.

This register configures the profiling and timestamp features

Table 6-12. DBGTCRL Field Descriptions

Field	Description
3 DSTAMP	**Comparator D Timestamp Enable** — This bit, when set, enables Comparator D matches to generate timestamps in Detail, Normal and Loop1 trace modes. 0 Comparator D match does not generate timestamp 1 Comparator D match generates timestamp if timestamp function is enabled
2 PDOE	**Profile Data Out Enable** — This bit, when set, configures the device profiling pins for profiling. 0 Device pins not configured for profiling 1 Device pins configured for profiling
1 PROFILE	**Profile Enable** — This bit, when set, enables the profile function, whereby a subsequent arming of the DBG activates profiling. When PROFILE is set, the TRCMOD bits are ignored. 0 Profile function disabled 1 Profile function enabled

Table 6-12. DBGTCRL Field Descriptions (continued)

Field	Description
0 STAMP	**Timestamp Enable** — This bit, when set, enables the timestamp function. The timestamp function adds a timestamp to each trace buffer entry in Detail, Normal and Loop1 trace modes. 0 Timestamp function disabled 1 Timestamp function enabled

6.3.2.5 Debug Trace Buffer Register (DBGTB)

Address: 0x0104, 0x0105

	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
POR	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Other Resets	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Figure 6-7. Debug Trace Buffer Register (DBGTB)

Read: Only when unlocked AND not armed and the TSOURCE bit is set, otherwise an error code (0xEE) is returned. Only aligned word read operations are supported. Misaligned word reads or byte reads return the error code 0xEE for each byte. The PROFILE bit must be clear to read profiling data,

Write: Aligned word writes when the DBG is disarmed and both PTACT and PROFILE are clear unlock the trace buffer for reading but do not affect trace buffer contents.

Table 6-13. DBGTB Field Descriptions

Field	Description
15–0 Bit[15:0]	**Trace Buffer Data Bits** — The Trace Buffer Register is a window through which the lines of the trace buffer may be read 16 bits at a time. Each valid read of DBGTB increments an internal trace buffer pointer which points to the next address to be read. When the ARM bit is written to 1 the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by writing to DBGTB with an aligned word write when the module is disarmed. The DBGTB register can be read only as an aligned word. Byte reads or misaligned access of these registers returns 0xEE and does not increment the trace buffer pointer. Similarly word reads while the debugger is armed or trace buffer is locked return 0xEEEE. The POR state is undefined Other resets do not affect the trace buffer contents.

6.3.2.6 Debug Count Register (DBGCNT)

Address: 0x0106

	7	6	5	4	3	2	1	0
R	0	CNT						
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	—	—	—	—	—	—	—
POR	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-8. Debug Count Register (DBGCNT)

Read: Anytime.

Write: Never.

Table 6-14. DBGCNT Field Descriptions

Field	Description
6–0 CNT[6:0]	**Count Value** — The CNT bits [6:0] indicate the number of valid data lines stored in the trace buffer. Table 6-15 shows the correlation between the CNT bits and the number of valid data lines in the trace buffer. When the CNT rolls over to zero, the TBF bit in DBGSR is set. Thereafter incrementing of CNT continues if configured for end-alignment or mid-alignment. The DBGCNT register is cleared when ARM in DBGC1 is written to a one. The DBGCNT register is cleared by power-on-reset initialization but is not cleared by other system resets. If a reset occurs during a debug session, the DBGCNT register still indicates after the reset, the number of valid trace buffer entries stored before the reset occurred. The DBGCNT register is not decremented when reading from the trace buffer.

Table 6-15. CNT Decoding Table

TBF (DBGSR)	CNT[6:0]	Description
0	0000000	No data valid
0	0000001	32 bits of one line valid
0	0000010 0000100 0000110 .. 1111100	1 line valid 2 lines valid 3 lines valid .. 62 lines valid
0	1111110	63 lines valid
1	0000000	64 lines valid; if using Begin trigger alignment, ARM bit is cleared and the tracing session ends.
1	0000010 .. 1111110	64 lines valid, oldest data has been overwritten by most recent data

6.3.2.7 Debug State Control Register 1 (DBGSCR1)

Address: 0x0107

	7	6	5	4	3	2	1	0
R W	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
Reset	0	0	0	0	0	0	0	0

Figure 6-9. Debug State Control Register 1 (DBGSCR1)

Read: Anytime.

Write: If DBG is not armed and PTACT is clear.

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 6.3.2.12”. Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 6-16. 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.
5–4 C2SC[1:0]	Channel 2 State Control. These bits select the targeted next state whilst in State1 following a match2.
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 6-17. 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.

6.3.2.8 Debug State Control Register 2 (DBGSCR2)

Address: 0x0108

	7	6	5	4	3	2	1	0
R W	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
Reset	0	0	0	0	0	0	0	0

Figure 6-10. Debug State Control Register 2 (DBGSCR2)

Read: Anytime.

Write: If DBG is not armed and PTACT is clear.

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 6.3.2.12”. Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 6-18. 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.

Table 6-18. DBGSCR2 Field Descriptions (continued)

Field	Description
5–4 C2SC[1:0]	Channel 2 State Control. These bits select the targeted next state whilst in State2 following a match2.
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 6-19. 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.

6.3.2.9 Debug State Control Register 3 (DBGSCR3)

Address: 0x0109

	7	6	5	4	3	2	1	0
R W	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
Reset	0	0	0	0	0	0	0	0

Figure 6-11. Debug State Control Register 3 (DBGSCR3)

Read: Anytime.

Write: If DBG is not armed and PTACT is clear.

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 6.3.2.12”. Comparators must be enabled by setting the comparator enable bit in the associated DBGxCTL control register.

Table 6-20. 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.
5–4 C2SC[1:0]	Channel 2 State Control. These bits select the targeted next state whilst in State3 following a match2.

Table 6-20. DBGSCR3 Field Descriptions (continued)

Field	Description
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 6-21. 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.

6.3.2.10 Debug Event Flag Register (DBGEFR)

Address: 0x010A

	7	6	5	4	3	2	1	0
R	PTBOVF	TRIGF	0	EEVF	ME3	ME2	ME1	ME0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 6-12. 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 6-22. DBGEFR Field Descriptions

Field	Description
7 PTBOVF	**Profiling Trace Buffer Overflow Flag** — Indicates the occurrence of a trace buffer overflow event during a profiling session. 0 No trace buffer overflow event 1 Trace buffer overflow event
6 TRIGF	**TRIG Flag** — Indicates the occurrence of a TRIG event during the debug session. 0 No TRIG event 1 TRIG event

Table 6-22. DBGEFR Field Descriptions

Field	Description
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**.**

6.3.2.11 Debug Status Register (DBGSR)

Address: 0x010B

	7	6	5	4	3	2	1	0
R	TBF	0	0	PTACT	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 6-13. Debug Status Register (DBGSR)

Read: Anytime.

Write: Never.

Table 6-23. DBGSR Field Descriptions

Field	Description
7 TBF	**Trace Buffer Full** — The TBF bit indicates that the trace buffer has been filled with data since it was last armed. If this bit is set, then all trace buffer lines contain valid data, regardless of the value of DBGCNT bits CNT[6:0]. The TBF bit is cleared when ARM in DBGC1 is written to a one. The TBF is cleared by the power on reset initialization. Other system generated resets have no affect on this bit
4 PTACT	**Profiling Transmission Active** — The PTACT bit, when set, indicates that the profiling transmission is still active. When clear, PTACT then profiling transmission is not active. The PTACT bit is set when profiling begins with the first PTS format entry to the trace buffer. The PTACT bit is cleared when the profiling transmission ends.
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 6-24**.**

Table 6-24. SSF[2:0] — State Sequence Flag Bit Encoding

SSF[2:0]	Current State
000	State0 (disarmed)
001	State1
010	State2
011	State3

Table 6-24. SSF[2:0] — State Sequence Flag Bit Encoding

SSF[2:0]	Current State
100	Final State
101,110,111	Reserved

6.3.2.12 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 6-14. Debug Comparator A Control Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-25. 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 6-26 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 6-26. 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 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 6-15. Debug Comparator A Address Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-27. 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

6.3.2.14 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 6-16. Debug Comparator A Data Register (DBGAD)

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

This register can be accessed with a byte resolution, whereby DBGAD0, DBGAD1, DBGAD2, DBGAD3 map to DBGAD[31:0] respectively.

Table 6-28. 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

6.3.2.15 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 6-17. Debug Comparator A Data Mask Register (DBGADM)

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

This register can be accessed with a byte resolution, whereby DBGADM0, DBGADM1, DBGADM2, DBGADM3 map to DBGADM[31:0] respectively.

Table 6-29. 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

6.3.2.16 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 6-18. Debug Comparator B Control Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-30. 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 6-31 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 6-31. 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

6.3.2.17 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 6-19. Debug Comparator B Address Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-32. 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

6.3.2.18 Debug Comparator C Control Register (DBGCCTL)

Address: 0x0130

	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 6-20. Debug Comparator C Control Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-33. DBGCCTL 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 not used 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

Table 6-34 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 6-34. 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

Table 6-34. Read or Write Comparison Logic Table

RWE Bit	RW Bit	RW Signal	Comment
1	1	1	Read match

6.3.2.19 Debug Comparator C Address Register (DBGCAH, DBGCAM, DBGCAL)

Address: 0x0135, DBGCAH

	23	22	21	20	19	18	17	16
R W	DBGCA[23:16]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0136, DBGCAM

	15	14	13	12	11	10	9	8
R W	DBGCA[15:8]							
Reset	0	0	0	0	0	0	0	0

Address: 0x0137, DBGCAL

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

Figure 6-21. Debug Comparator C Address Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-35. DBGCAH, DBGCAM, DBGCAL Field Descriptions

Field	Description
23–16 DBGCA [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 DBGCA [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

6.3.2.20 Debug Comparator C Data Register (DBGCD)

Address: 0x0138, 0x0139, 0x013A, 0x013B

	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 6-22. Debug Comparator C Data Register (DBGCD)

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

This register can be accessed with a byte resolution, whereby DBGCD0, DBGCD1, DBGCD2, DBGCD3 map to DBGCD[31:0] respectively.

XGATE data accesses have a maximum width of 16-bits and are mapped to DBGCD[15:0].

Table 6-36. DBGCD Field Descriptions

Field	Description
31–16 Bits[31:16] (DBGCD0, DBGCD1)	**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] (DBGCD2, DBGCD3)	**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

6.3.2.21 Debug Comparator C Data Mask Register (DBGCDM)

Address: 0x013C, 0x013D, 0x013E, 0x013F

	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 6-23. Debug Comparator C Data Mask Register (DBGCDM)

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

This register can be accessed with a byte resolution, whereby DBGCDM0, DBGCDM1, DBGCDM2, DBGCDM3 map to DBGCDM[31:0] respectively.

XGATE data accesses have a maximum width of 16-bits and are mapped to DBGCDM[15:0].

Table 6-37. DBGCDM Field Descriptions

Field	Description
31–16 Bits[31:16] (DBGCDM0, DBGCDM1)	**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] (DBGCDM2, DBGCDM3)	**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

6.3.2.22 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 6-24. Debug Comparator D Control Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-38. 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 6-39 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 6-39. 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

6.3.2.23 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 6-25. Debug Comparator D Address Register

Read: Anytime.

Write: If DBG not armed and PTACT is clear.

Table 6-40. 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

Table 6-40. DBGDAH, DBGDAM, DBGDAL Field Descriptions

Field	Description
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 supports storing of data in the trace buffer and can be used to generate breakpoints to the CPU. The DBG module is made up of comparators, control logic, the trace buffer, 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 6-26).

The state sequencer can transition freely between the states 1, 2 and 3. On transition to Final State bus tracing can be triggered. On completion of tracing the state sequencer enters State0. If tracing is disabled or End aligned tracing is enabled then the state sequencer transitions immediately from Final State to State0. The transition to State0 generates breakpoints if breakpoints are enabled.

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.

The trace buffer is visible through a 2-byte window in the register address map and can be read out using standard 16-bit word reads.

6.4.2 Comparator Modes

The DBG contains four comparators, A, B, C, 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, comparators A and C 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”).

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.

Comparators C and D can also be used to select an address range to trace from, when tracing CPU accesses in Detail mode. This is determined by the TRANGE bits in the DBGTCRH register. The TRANGE encoding is shown in Table 6-9. If the TRANGE bits select a range definition using comparator D and the COMPE bit is clear, then comparator D is configured for trace range definition. By setting the COMPE bit the comparator is configured for address bus comparisons, the TRANGE bits are ignored and the tracing range function is disabled. Similarly if the TRANGE bits select a range definition using comparator C and the COMPE bit is clear, then comparator C is configured for trace range definition.

Match[0, 1, 2, 3] map directly to Comparators [A, B, C, D] respectively, except in range modes (see Section 6.3.2.2”). Comparator priority rules are described in the event priority section (Section 6.4.3.5”).

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 6-41. 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 6-41. 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

Table 6-41. Comparator Address Bus Matches

Access	Address	ADDR[n]	ADDR[n+1]	ADDR[n+2]	ADDR[n+3]
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

Comparators A and C feature 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 6-42, whereby the Address column refers to the lowest 2 bits of the lowest accessed address. This corresponds to the most significant data byte.

Table 6-42. 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 6-43 the Access Address column refers to the address bits[1:0] of the lowest accessed address (most significant data byte).

Table 6-43. 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			

			Memory Address[2:0]						
Case	**Access Address**	**Access Size**	**000**	**001**	**010**	**011**	**100**	**101**	**110**
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 6-43 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 6-44. 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. Similarly when using the CD comparator pair for a range comparison, the data bus can be used for qualification by using the comparator C data and data mask registers. The DBGACTL/DBGCCTL RW and RWE bits can be used to qualify the range comparison on either a read or a write access. The corresponding DBGBCTL/DBGDCTL bits are ignored. The DBGACTL/DBGCCTL COMPE/INST bits are used for range comparisons. The DBGBCTL/DBGDCTL COMPE/INST bits are ignored in range modes.

6.4.2.4.1 Inside Range (CompAC_Addr ≤ address ≤ CompBD_Addr)

In the Inside Range comparator mode, either comparator pair A and B or comparator pair C and D 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 < CompAC_Addr or address > CompBD_Addr)

In the Outside Range comparator mode, either comparator pair A and B or comparator pair C and D 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, or force a trace buffer entry, or gate trace buffer entries.

If configured to change the state sequencer state, then the external match is mapped to DBGSCRx bits C3SC[1:0]. In this configuration, internal comparator channel3 is de-coupled from the state sequencer but can still be used for timestamps. 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 tracing session and/or breakpoint by writing the TRIG bit in DBGC1 to a logic “1”. This forces the state sequencer into the Final State. If configured for End aligned tracing or for no tracing, the transition to Final State is followed immediately by a transition to State0. If configured for Begin- or Mid Aligned tracing, the state sequencer remains in Final State until tracing is complete, then it transitions to State0.

Breakpoints, if enabled, are issued on the transition to State0.

6.4.3.4 Profiling Trace Buffer Overflow Event

During code profiling a trace buffer overflow forces the state sequencer into the disarmed State0 and, if breakpoints are enabled, issues a breakpoint request to the CPU.

6.4.3.5 Event Priorities

If simultaneous events occur, the priority is resolved according to Table 6-45. 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 (3,2,1,0) 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 6-45. Event Priorities

Priority	Source	Action
Highest	TB Overflow	Immediate force to state 0, generate breakpoint and terminate tracing

Table 6-45. Event Priorities

	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
	Match2	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

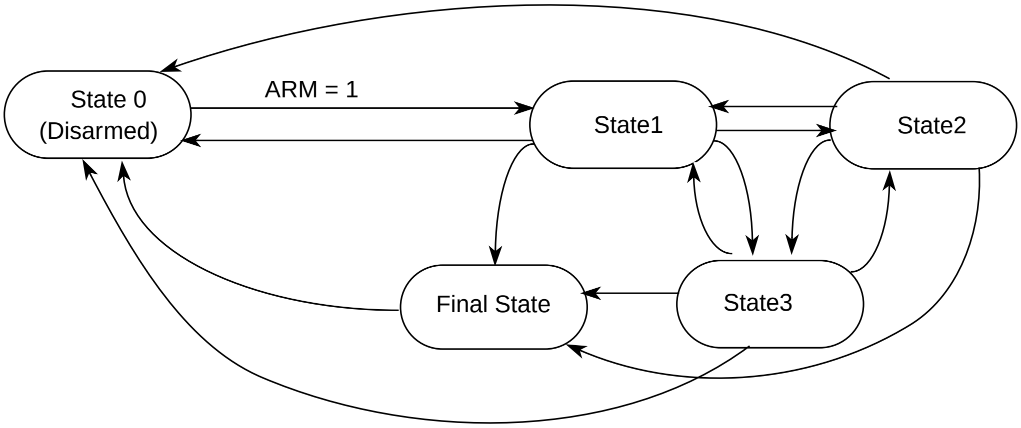


Figure 6-26. State Sequencer Diagram

The state sequencer allows a defined sequence of events to provide a breakpoint and/or a trigger point for tracing of data in the trace buffer. 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). 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

On entering Final State a trigger may be issued to the trace buffer according to the trigger position control as defined by the TALIGN field (see Section 6.3.2.3").

If tracing is enabled and either Begin or Mid aligned triggering is selected, the state sequencer remains in Final State until completion of the trace. On completion of the trace the state sequencer returns to State0 and the debug module is disarmed; if breakpoints are enabled, a breakpoint request is generated.

If tracing is disabled or End aligned triggering is selected, then 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 Trace Buffer Operation

The trace buffer is a 64 lines deep by 64-bits wide RAM array. If the TSOURCE bit is set the DBG module can store trace information in the RAM array in a circular buffer format. Data is stored in mode dependent formats, as described in the following sections. After each trace buffer entry, the counter register DBGCNT is incremented. Trace buffer rollover is possible when configured for End- or Mid-Aligned tracing, such that older entries are replaced by newer entries. Tracing of CPU activity is disabled when the BDC is active.

The RAM array can be accessed through the register DBGTB using 16-bit wide word accesses. After each read, the internal RAM pointer is incremented so that the next read will receive fresh information. Reading the trace buffer whilst the DBG is armed returns invalid data and the trace buffer pointer is not incremented.

In Detail mode the address range for CPU access tracing can be limited to a range specified by the TRANGE bits in DBGTCRH. This function uses comparators C and D to define an address range inside which accesses should be traced. Thus traced accesses can be restricted, for example, to particular register or RAM range accesses.

The external event pin can be configured to force trace buffer entries in Normal or Loop1 trace modes. All tracing modes support trace buffer gating. In Pure PC and Detail modes external events do not force trace buffer entries.

If the external event pin is configured to gate trace buffer entries then any trace mode is valid.

6.4.5.1 Trace Trigger Alignment

Using the TALIGN bits (see Section 6.3.2.3”) it is possible to align the trigger with the end, the middle, or the beginning of a tracing session.

If End or Mid-Alignment is selected, tracing begins when the ARM bit in DBGC1 is set and State1 is entered. The transition to Final State if End-Alignment is selected, ends the tracing session. The transition to Final State if Mid-Alignment is selected signals that another 32 lines are traced before ending the tracing session. Tracing with Begin-Alignment starts at the trigger and ends when the trace buffer is full.

Table 6-46. Tracing Alignment

TALIGN	Tracing Begin	Tracing End
00	On arming	At trigger
01	At trigger	When trace buffer is full
10	On arming	When 32 trace buffer lines have been filled after trigger
11	Reserved	

6.4.5.1.1 Storing with Begin-Alignment

Storing with Begin-Alignment, data is not stored in the trace buffer until the Final State is entered. Once the trigger condition is met the DBG module remains armed until 64 lines are stored in the trace buffer.

Using Begin-Alignment together with opcode address comparisons, if the instruction is about to be executed then the trace is started. If the trigger is at the address of a COF instruction, whilst tracing COF addresses, then that COF address is stored to the trace buffer. If breakpoints are enabled, the breakpoint is generated upon entry into State0 on completion of the tracing session; thus the breakpoint does not occur at the instruction boundary.

6.4.5.1.2 Storing with Mid-Alignment

Storing with Mid-Alignment, data is stored in the trace buffer as soon as the DBG module is armed. When the trigger condition is met, another 32 lines are traced before ending the tracing session, irrespective of the number of lines stored before the trigger occurred, then the DBG module is disarmed and no more data is stored. Using Mid-Alignment with opcode address triggers, if the instruction is about to be executed then the trace is continued for another 32 lines. If breakpoints are enabled, the breakpoint is generated upon entry into State0 on completion of the tracing session; thus the breakpoint does not occur at the instruction boundary. When configured for Compressed Pure-PC tracing, the MAT info bit is set to indicate the last PC entry before a trigger event.

6.4.5.1.3 Storing with End-Alignment

Storing with End-Alignment, data is stored in the trace buffer until the Final State is entered. Following this trigger, the DBG module immediately transitions to State0. If the trigger is at the address of a COF instruction the trigger event is not stored in the trace buffer.

6.4.5.2 Trace Modes

The DBG module can operate in four trace modes. The mode is selected using the TRCMOD bits in the DBGTCRH register. Normal, Loop1 and Detail modes can be configured to store a timestamp with each entry, by setting the STAMP bit. The modes are described in the following subsections.

In addition to the listed trace modes it is also possible to use code profiling to fill the trace buffer with a highly compressed COF format. This can be subsequently read out in the same fashion as the listed trace modes (see Section 6.4.6).

6.4.5.2.1 Normal Mode

In Normal Mode, change of flow (COF) program counter (PC) addresses are stored.

CPU COF addresses are defined as follows:

- Source address of taken conditional branches (bit-conditional, and loop primitives)
- Destination address of indexed JMP and JSR instruction.s
- Destination address of RTI and RTS instructions.
- Vector address of interrupts

BRA, BSR, BGND as well as non-indexed JMP and JSR instructions are not classified as change of flow and are not stored in the trace buffer.

COF addresses stored include the full address bus of CPU and an information byte, which contains bits to indicate whether the stored address was a source, destination or vector address.

NOTE

When a CPU indexed jump instruction is executed, the destination address is stored to the trace buffer on instruction completion, indicating the COF has taken place. If an interrupt occurs simultaneously then the next instruction carried out is actually from the interrupt service routine. The instruction at the destination address of the original program flow gets executed after the interrupt service routine.

In the following example an IRQ interrupt occurs during execution of the indexed JMP at address MARK1. The NOP at the destination (SUB_1) is not executed until after the IRQ service routine but the destination address is entered into the trace buffer to indicate that the indexed JMP COF has taken place.

```
         LD       X,#SUB_1
MARK1:   JMP      (0,X)                    ; IRQ interrupt occurs during execution of this
MARK2:   NOP                               ;

SUB_1:   NOP                               ; JMP Destination address TRACE BUFFER ENTRY 1
                                           ; RTI Destination address TRACE BUFFER ENTRY 3
         NOP                               ;
ADDR1:   DBNE     D0,PART5                 ; Source address TRACE BUFFER ENTRY 4

IRQ_ISR: LD       D1,#$F0                  ; IRQ Vector $FFF2 = TRACE BUFFER ENTRY 2
         ST       D1,VAR_C1
         RTI                               ;
```

The execution flow taking into account the IRQ is as follows

```
         LD       X,#SUB_1
MARK1:   JMP      (0,X)                    ;
IRQ_ISR: LD       D1,#$F0                  ;
         ST       D1,VAR_C1
         RTI                               ;
SUB_1:   NOP
         NOP                               ;
ADDR1:   DBNE     D0,PART5                 ;
```

The Normal Mode trace buffer format is shown in the following tables. Whilst tracing in Normal or Loop1 modes each array line contains 2 data entries, thus in this case the DBGCNT[0] is incremented after each separate entry. Information byte bits indicate if an entry is a source, destination or vector address.

The external event input can force trace buffer entries independent of COF occurrences, in which case the EEVI bit is set and the PC value of the last instruction is stored to the trace buffer. If the external event coincides with a COF buffer entry a single entry is made with the EEVI bit set.

Normal mode profiling with timestamp is possible when tracing from a single source by setting the STAMP bit in DBGTCRL. This results in a different format (see Table 6-48).

Table 6-47. Normal and Loop1 Mode Trace Buffer Format without Timestamp

Mode	8-Byte Wide Trace Buffer Line							
	7	6	5	4	3	2	1	0

Table 6-47. Normal and Loop1 Mode Trace Buffer Format without Timestamp

CPU	CINF1	CPCH1	CPCM1	CPCL1	CINF0	CPCH0	CPCM0	CPCL0
	CINF3	CPCH3	CPCM3	CPCL3	CINF2	CPCH2	CPCM2	CPCL2

Table 6-48. Normal and Loop1 Mode Trace Buffer Format with Timestamp

Mode	8-Byte Wide Trace Buffer Line							
	7	6	5	4	3	2	1	0
CPU	Timestamp	Timestamp	Reserved	Reserved	CINF0	CPCH0	CPCM0	CPCL0
	Timestamp	Timestamp	Reserved	Reserved	CINF1	CPCH1	CPCM1	CPCL1

CINF contains information relating to the CPU.

CPU Information Byte CINF For Normal And Loop1 Modes

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CET		0	0	CTI	EEVI	0	TOVF

Figure 6-27. CPU Information Byte CINF

Table 6-49. CINF Bit Descriptions

Field	Description
7–6 CET	**CPU Entry Type Field** — Indicates the type of stored address of the trace buffer entry as described in Table 6-50
3 CTI	**Comparator Timestamp Indicator** — This bit indicates if the trace buffer entry corresponds to a comparator timestamp. 0 Trace buffer entry initiated by trace mode specification conditions or timestamp counter overflow 1 Trace buffer entry initiated by comparator D match
2 EEVI	**External Event Indicator** — This bit indicates if the trace buffer entry corresponds to an external event. 0 Trace buffer entry not initiated by an external event 1 Trace buffer entry initiated by an external event
0 TOVF	**Timestamp Overflow Indicator** — Indicates if the trace buffer entry corresponds to a timestamp overflow 0 Trace buffer entry not initiated by a timestamp overflow 1 Trace buffer entry initiated by a timestamp overflow

Table 6-50. CET Encoding

CET	Entry Type Description
00	Non COF opcode address (entry forced by an external event)
01	Vector destination address

Table 6-50. CET Encoding

CET	Entry Type Description
10	Source address of COF opcode
11	Destination address of COF opcode

6.4.5.2.2 Loop1 Mode

Loop1 Mode, similarly to Normal Mode also stores only COF address information to the trace buffer, it however allows the filtering out of redundant information.

The intent of Loop1 Mode is to prevent the trace buffer from being filled entirely with duplicate information from a looping construct such as delays using the DBNE instruction. The DBG monitors trace buffer entries and prevents consecutive duplicate address entries resulting from repeated branches.

Loop1 Mode only inhibits consecutive duplicate source address entries that would typically be stored in most tight looping constructs. It does not inhibit repeated entries of destination addresses or vector addresses, since repeated entries of these could indicate a bug in application code that the DBG module is designed to help find.

The trace buffer format for Loop1 Mode is the same as that of Normal Mode.

6.4.5.2.3 Detail Mode

When tracing CPU activity in Detail Mode, address and data of data and vector accesses are traced. The information byte indicates the size of access and the type of access (read or write).

ADRH, ADRM, ADRL denote address high, middle and low byte respectively. The numerical suffix indicates which tracing step. DBGCNT increments by 2 for each line completed.

If timestamps are enabled then each CPU entry can span 2 trace buffer lines, whereby the second line includes the timestamp. If a valid PC occurs in the same cycle as the timestamp, it is also stored to the trace buffer and the PC bit is set. The second line featuring the timestamp is only stored if no further data access occurs in the following cycle. This is shown in Table 6-52, where data accesses 2 and 3 occur in consecutive cycles, suppressing the entry2 timestamp. If 2 lines are used for an entry, then DBGCNT increments by 4. A timestamp line is indicated by bit1 in the TSINF byte. The timestamp counter is only reset each time a timestamp line entry is made. It is not reset when the data and address trace buffer line entry is made.

Table 6-51. Detail Mode Trace Buffer Format without Timestamp

Mode	8-Byte Wide Trace Buffer Line							
	7	6	5	4	3	2	1	0
CPU Detail	CDATA31	CDATA21	CDATA11	CDATA01	CINF1	CADRH1	CADRM1	CADRL1
	CDATA32	CDATA22	CDATA12	CDATA02	CINF2	CADRH2	CADRM2	CADRL2

Table 6-52. Detail Mode Trace Buffer Format with Timestamp

Mode	8-Byte Wide Trace Buffer Line							
	7	6	5	4	3	2	1	0
CPU Detail	CDATA31	CDATA21	CDATA11	CDATA01	CINF1	CADRH1	CADRM1	CADRL1
	Timestamp	Timestamp	Reserved	Reserved	TSINF1	CPCH1	CPCM1	CPCL1
	CDATA32	CDATA22	CDATA12	CDATA02	CINF2	CADRH2	CADRM2	CADRL2
	CDATA33	CDATA23	CDATA13	CDATA03	CINF3	CADRH3	CADRM3	CADRL3
	Timestamp	Timestamp	Reserved	Reserved	TSINF3	CPCH3	CPCM3	CPCL3

Detail Mode data entries store the bytes aligned to the address of the MSB accessed (Byte1 Table 6-53). Thus accesses split across 32-bit boundaries are wrapped around.

Table 6-53. Detail Mode Data Byte Alignment

Access Address	Access Size	CDATA31	CDATA21	CDATA11	CDATA01
00	32-bit	Byte1	Byte2	Byte3	Byte4
01	32-bit	Byte4	Byte1	Byte2	Byte3
10	32-bit	Byte3	Byte4	Byte1	Byte2
11	32-bit	Byte2	Byte3	Byte4	Byte1
00	24-bit	Byte1	Byte2	Byte3	▨
01	24-bit	▨	Byte1	Byte2	Byte3
10	24-bit	Byte3	▨	Byte1	Byte2
11	24-bit	Byte2	Byte3	▨	Byte1
00	16-bit	Byte1	Byte2	▨	▨
01	16-bit	▨	Byte1	Byte2	▨
10	16-bit	▨	▨	Byte1	Byte2
11	16-bit	Byte2	▨	▨	Byte1
00	8-bit	Byte1	▨	▨	▨
01	8-bit	▨	Byte1	▨	▨
10	8-bit	▨	▨	Byte1	▨
11	8-bit	▨	▨	▨	Byte1
		▨	Denotes byte that is not accessed.		

Information Bytes

BYTE	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CINF	CSZ		CRW	0	0	0	0	0
TSINF	0	0	0	0	CTI	PC	1	TOVF

Figure 6-28. Information Bytes CINF and XINF

When tracing in Detail Mode, CINF provides information about the type of CPU access being made.

TSINF provides information about a timestamp. Bit1 indicates if the byte is a TSINF byte.

Table 6-54. CINF Field Descriptions

Field	Description
7–6 CSZ	**Access Type Indicator** — This field indicates the CPU access size. 00 8-bit Access 01 16-bit Access 10 24-bit Access 11 32-bit Access
5 CRW	**Read/Write Indicator** — Indicates if the corresponding stored address corresponds to a read or write access. 0 Write Access 1 Read Access

Table 6-55. TSINF Field Descriptions

Field	Description
3 CTI	**Comparator Timestamp Indicator** — This bit indicates if the trace buffer entry corresponds to a comparator timestamp. 0 Trace buffer entry initiated by trace mode specification conditions or timestamp counter overflow 1 Trace buffer entry initiated by comparator D match
2 PC	**Program Counter Valid Indicator** — Indicates if the PC entry is valid on the timestamp line. 0 Trace buffer entry does not include PC value 1 Trace buffer entry includes PC value
0 TOVF	**Timestamp Overflow Indicator** — Indicates if the trace buffer entry corresponds to a timestamp overflow 0 Trace buffer entry not initiated by a timestamp overflow 1 Trace buffer entry initiated by a timestamp overflow

6.4.5.2.4 Pure PC Mode

In Pure PC Mode, the PC addresses of all opcodes loaded into the execution stage, including illegal opcodes, are stored.

Tracing from a single source, compression is implemented to increase the effective trace depth. A compressed entry consists of the lowest PC byte only. A full entry consists of all PC bytes. If the PC remains in the same 256 byte range, then a compressed entry is made, otherwise a full entry is made. The full entry is always the last entry of a record.

Each trace buffer line consists of 7 payload bytes, PLB0-6, containing full or compressed CPU PC addresses and 1 information byte to indicate the type of entry (compressed or base address) for each payload byte.

Each trace buffer line is filled from right to left. The final entry on each line is always a base address, used as a reference for the previous entries on the same line. Whilst tracing, a base address is typically stored in bytes[6:4], the other payload bytes may be compressed or complete addresses as indicated by the info byte bits.

Table 6-56. Pure PC Mode Trace Buffer Format Single Source

Mode	8-Byte Wide Trace Buffer Line							
	7	6	5	4	3	2	1	0
CPU	CXINF	BASE	BASE	BASE	PLB3	PLB2	PLB1	PLB0

If the info bit for byte3 indicates a full CPU PC address, whereby bytes[5:3] are used, then the info bit mapped to byte[4] is redundant and the byte[6] is unused because a line overflow has occurred. Similarly a base address stored in bytes[4:2] causes line overflow, so bytes[6:5] are unused.

CXINF[6:4] indicate how many bytes in a line contain valid data, since tracing may terminate before a complete line has been filled.

CXINF Information Byte Source Tracing

	7	6	5	4	3	2	1	0
CXINF	MAT	PLEC			NB3	NB2	NB1	NB0

Figure 6-29. Pure PC Mode CXINF

Table 6-57. CXINF Field Descriptions

Field	Description
MAT	**Mid Aligned Trigger**— This bit indicates a mid aligned trigger position. When a mid aligned trigger occurs, the next trace buffer entry is a base address and the counter is incremented to a new line, independent of the number of bytes used on the current line. The MAT bit is set on the current line, to indicate the position of the trigger. When configured for begin or end aligned trigger, this bit has no meaning. NOTE: In the case when ARM and TRIG are simultaneously set together in the same cycle that a new PC value is registered, then this PC is stored to the same trace buffer line and MAT set. 0 Line filled without mid aligned trigger occurrence 1 Line last entry is the last PC entry before a mid aligned trigger
PLEC[2:0]	**Payload Entry Count**— This field indicates the number of valid bytes in the trace buffer line Binary encoding is used to indicate up to 7 valid bytes.
NBx	**Payload Compression Indicator**— This field indicates if the corresponding payload byte is the lowest byte of a base PC entry 0 Corresponding payload byte is a not the lowest byte of a base PC entry 1 Corresponding payload byte is the lowest byte of a base PC entry

Pure PC mode tracing does not support timestamps or external event entries.

6.4.5.3 Timestamp

When set, the STAMP bit in DBGTCRL configures the DBG to add a timestamp to trace buffer entries in Normal, Loop1 and Detail trace buffer modes. The timestamp is generated from a 16-bit counter and is stored to the trace buffer line each time a trace buffer entry is made.

The number of core clock cycles since the last entry equals the timestamp + 1. The core clock runs at twice the frequency of the bus clock. The timestamp of the first trace buffer entry is 0x0000. With timestamps enabled trace buffer entries are initiated in the following ways:

- according to the trace mode specification, for example COF PC addresses in Normal mode
- on a timestamp counter overflow
  If the timestamp counter reaches 0xFFFF then a trace buffer entry is made, with timestamp= 0xFFFF and the timestamp overflow bit TOVF is set.
- on a match of comparator D
  If STAMP and DSTAMP are set then comparator D is used for forcing trace buffer entries with timestamps. The state control register settings determine if comparator D is also used to trigger the state sequencer. Thus if the state control register configuration does not use comparator D, then it is used solely for the timestamp function. If comparator D initiates a timestamp then the CTI bit is set in the INFO byte. This can be used in Normal/Loop1 mode to indicate when a particular data access occurs relative to the PC flow. For example when the timing of an access may be unclear due to the use of indexes.

NOTE

If comparator D is configured to match a PC address then associated timestamps trigger a trace buffer entry during execution of the previous instruction. Thus the PC stored to the trace buffer is that of the previous instruction.The comparator must contain the PC address of the instruction's first opcode byte

Timestamps are disabled in Pure PC mode.

6.4.5.4 Reading Data from Trace Buffer

The data stored in the trace buffer can be read using either the background debug controller (BDC) module or the CPU provided the DBG module is not armed and is configured for tracing by TSOURCE. When the ARM bit is set the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by an aligned word write to DBGTB when the module is disarmed. The trace buffer can only be read through the DBGTB register using aligned word reads. Reading the trace buffer while the DBG module is armed, or trace buffer locked returns 0xEE and no shifting of the RAM pointer occurs. Any byte or misaligned reads return 0xEE and do not cause the trace buffer pointer to increment to the next trace buffer address.

Reading the trace buffer is prevented by internal hardware whilst profiling is active because the RAM pointer is used to indicate the next row to be transmitted. Thus attempted reads of DBGTB do not return valid data when the PROFILE bit is set. To initialize the pointer and read profiling data, the PROFILE bit must be cleared and remain cleared.

The trace buffer data is read out first-in first-out. By reading CNT in DBGCNT the number of valid 64-bit lines can be determined. DBGCNT does not decrement as data is read.

Whilst reading, an internal pointer is used to determine the next line to be read. After a tracing session, the pointer points to the oldest data entry, thus if no overflow has occurred, the pointer points to line0. The pointer is initialized by each aligned write to DBGTB to point to the oldest data again. This enables an interrupted trace buffer read sequence to be easily restarted from the oldest data entry. After reading all trace buffer lines, the next read wraps around and returns the contents of line0.

The least significant word of each 64-bit wide array line is read out first. All bytes, including those containing invalid information are read out.

6.4.5.5 Trace Buffer Reset State

The trace buffer contents are not initialized by a system reset. Thus should a system reset occur, the trace session information from immediately before the reset occurred can be read out. The DBGCNT bits are not cleared by a system reset. Thus should a reset occur, the number of valid lines in the trace buffer is indicated by DBGCNT. The internal pointer is cleared by a system reset. It can be initialized by an aligned word write to DBGTB following a reset during debugging, so that it points to the oldest valid data again. Debugging occurrences of system resets is best handled using mid or end trigger alignment since the reset may occur before the trace trigger, which in the begin trigger alignment case means no information would be stored in the trace buffer.

6.4.6 Code Profiling

6.4.6.1 Code Profiling Overview

Code profiling supplies encoded COF information on the PDO pin and the reference clock on the PDOCLK pin. If the TSOURCE bit is set then code profiling is enabled by setting the PROFILE bit. The associated device pin is configured for code profiling by setting the PDOE bit. Once enabled, code profiling is activated by arming the DBG. During profiling, if PDOE is set, the PDO operates as an output pin at a half the internal bus frequency, driving both high and low.

Independent of PDOE status, profiling data is stored to the trace buffer and can be read out in the usual manner when the debug session ends and the PROFILE bit has been cleared.

The external debugger uses both edges of the clock output to strobe the data on PDO. The first PDOCLK edge is used to sample the first data bit on PDO.

Figure 6-30. Profiling Output Interface

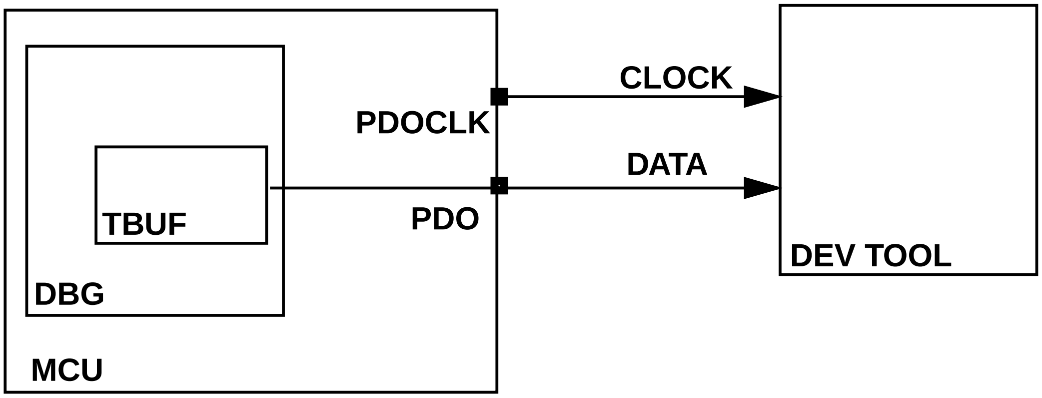
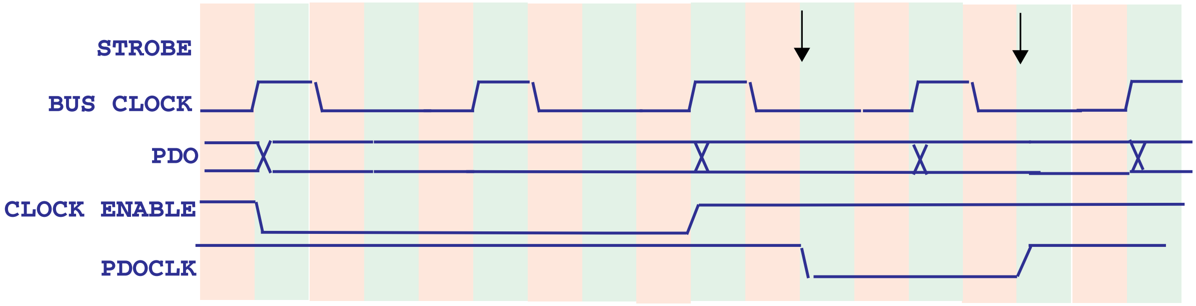


Figure 6-31 shows the profiling clock, PDOCLK, whose edges are offset from the bus clock, to ease setup and hold time requirements relative to PDO, which is synchronous to the bus clock.

Figure 6-31. PDO Profiling Clock Control



The trace buffer is used as a temporary storage medium to store COF information before it is transmitted. COF information can be transmitted whilst new information is written to the trace buffer. The trace buffer data is transmitted at PDO least significant bit first. After the first trace buffer entry is made, transmission begins in the first clock period in which no further data is written to the trace buffer.

If a trace buffer line transmission completes before the next trace buffer line is ready, then the clock output is held at a constant level until the line is ready for transfer.

6.4.6.2 Profiling Configuration, Alignment and Mode Dependencies

The PROFILE bit must be set and the DBG armed to enable profiling. Furthermore the PDOE bit must be set to configure the PDO and PDOCLK pins for profiling.

If TALIGN is configured for End-Aligned tracing then profiling begins as soon as the module is armed.

If TALIGN is configured for Begin-aligned tracing, then profiling begins when the state sequencer enters Final State and continues until a software disarm or trace buffer overflow occurs; thus profiling does not terminate after 64 line entries have been made.

Mid-Align tracing is not supported whilst profiling; if the TALIGN bits are configured for Mid-Align tracing when PROFILE is set, then the alignment defaults to end alignment.

Profiling entries continue until either a trace buffer overflow occurs or the DBG is disarmed by a state machine transition to State0. The profiling output transmission continues, even after disarming, until all trace buffer entries have been transmitted. The PTACT bit indicates if a profiling transmission is still active. The PTBOVF indicates if a trace buffer overflow has occurred.

The profiling timestamp feature is used only for the PTVB and PTW formats, thus differing from timestamps offered in other modes.

Profiling does not support trace buffer gating. The external pin gating feature is ignored during profiling.

When the DBG module is disarmed but profiling transmission is ongoing, register write accesses are suppressed.

When the DBG module is disarmed but profiling transmission is still ongoing, reading from the DBGTB returns the code 0xEE.

6.4.6.3 Code Profiling Internal Data Storage Format

When profiling starts, the first trace buffer entry is made to provide the start address. This uses a 4 byte format (PTS), including the INFO byte and a 3-byte PC start address. In order to avoid trace buffer overflow a fully compressed format is used for direct (conditional branch) COF information.

Table 6-58. Profiling Trace buffer line format

Format	**8-Byte Wide Trace Buffer Line**							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PTS					PC Start Address			INFO
PTIB	Indirect	Indirect	Indirect	Direct	Direct	Direct	Direct	INFO
PTHF			0	Direct	Direct	Direct	Direct	INFO
PTVB	Timestamp	Timestamp	Vector	Direct	Direct	Direct	Direct	INFO
PTW	Timestamp	Timestamp	0	Direct	Direct	Direct	Direct	INFO

The INFO byte indicates the line format used. Up to 4 bytes of each line are dedicated to branch COFs. Further bytes are used for storing indirect COF information (indexed jumps and interrupt vectors). Indexed jumps force a full line entry with the PTIB format and require 3-bytes for the full 24-bit destination address. Interrupts force a full line entry with the PTVB format, whereby vectors are stored as a single byte and a 16-bit timestamp value is stored simultaneously to indicate the number of bus cycles relative to the previous COF. At each trace buffer entry the 16-bit timestamp counter is cleared. The device vectors use address[8:0] whereby address[1:0] are constant zero for vectors. Thus the value stored to the PTVB vector byte is equivalent to (Vector Address[8:1]).

After the PTS entry, the pointer increments and the DBG begins to fill the next line with direct COF information. This continues until the direct COF field is full or an indirect COF occurs, then the INFO byte and, if needed, indirect COF information are entered on that line and the pointer increments to the next line.

If a timestamp overflow occurs, indicating a 65536 bus clock cycles without COF, then an entry is made with the TSOVF bit set, INFO[6] (Table 6-59) and profiling continues.

If a trace buffer overflow occurs, a final entry is made with the TBOVF bit set, profiling is terminated and the DBG is disarmed. Trace buffer overflow occurs when the trace buffer contains 64 lines pending transmission.

Whenever the DBG is disarmed during profiling, a final entry is made with the TERM bit set to indicate the final entry.

When a final entry is made then by default the PTW line format is used, except if a COF occurs in the same cycle in which case the corresponding PTIB/PTVB/PTHF format is used. Since the development tool receives the INFO byte first, it can determine in advance the format of data it is about to receive. The transmission of the INFO byte starts when a line is complete. Whole bytes are always transmitted. The grey shaded bytes of Table 6-58 are not transmitted.

Figure 6-32. INFO byte encoding

7	6	5	4	3	2	1	0
0	TSOVF	TBOVF	TERM	Line Format			

Table 6-59. Profiling Format Encoding

INFO[3:0]	Line Format	Source	Description
0000	PTS	CPU	Initial CPU entry
0001	PTIB	CPU	Indexed jump with up to 31 direct COFs
0010	PTHF	CPU	31 direct COFs without indirect COF
0011	PTVB	CPU	Vector with up to 31 direct COFs
0111	PTW	CPU	Error (Error codes in INFO[7:4])
Others	Reserved	CPU	Reserved
INFO[7:4]	**Bit Name**		**Description**
INFO[7]	Reserved	CPU	Reserved
INFO[6]	TSOVF	CPU	Timestamp Overflow
INFO[5]	TBOVF	CPU	Trace Buffer Overflow
INFO[4]	TERM	CPU	Profiling terminated by disarming
Vector[7:0]	Vector[7:0]	CPU	Device Interrupt Vector Address [8:1]

6.4.6.4 Direct COF Compression

Each branch COF is stored to the trace buffer as a single bit (0=branch not taken, 1=branch taken) until an indirect COF (indexed jump, return, or interrupt) occurs. The branch COF entries are stored in the byte fields labelled “Direct” in Table 6-58. These entries start at byte1[0] and continue through to byte4[7], or until an indirect COF occurs, whichever occurs sooner. The entries use a format whereby the left most asserted bit is always the stop bit, which indicates that the bit to its right is the first direct COF and byte1[0] is the last COF that occurred before the indirect COF. This is shown in Table 6-60, whereby the Bytes 4 to 1 of the trace buffer are shown for 3 different cases. The stop bit field for each line is shaded.

In line0, the left most asserted bit is Byte4[7]. This indicates that all remaining 31 bits in the 4-byte field contain valid direct COF information, whereby each 1 represents branch taken and each 0 represents branch not taken. The stop bit of line1 indicates that all 30 bits to it’s right are valid, after the 30th direct COF entry, an indirect COF occurred, that is stored in bytes 7 to 5. In this case the bit to the left of the stop bit is redundant. Line2 indicates that an indirect COF occurred after 8 direct COF entries. The indirect COF address is stored in bytes 7 to 5. All bits to the left of the stop bit are redundant.

Line	Byte4								Byte3								Byte2								Byte1							
Line0	1	0	0	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	1	0	0	0	0	1	1	0	0	0	0	1	1	0
Line1	0	1	1	0	0	1	0	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0
Line2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1

Table 6-60. Profiling Direct COF Format

6.4.7 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.
- Through a profiling trace buffer overflow event.

Breakpoints are not generated by software writes to DBGC1 that clear the ARM bit.

6.4.7.1 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.

If a tracing session is selected by TSOURCE, the transition to State0 occurs when the tracing session has completed, thus if Begin or Mid aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace. If End aligned tracing or no tracing session is selected, the transition to State0 and associated breakpoints are immediate.

6.4.7.2 Breakpoints Generated Via The TRIG Bit

When TRIG is written to “1”, the Final State is entered. If a tracing session is selected by TSOURCE, State0 is entered and breakpoints are requested only when the tracing session has completed, thus if Begin or Mid aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace. If no tracing session is selected, the state sequencer enters State0 immediately and breakpoints are requested. TRIG breakpoints are possible even if the DBG module is disarmed.

6.4.7.3 DBG Breakpoint Priorities

If a TRIG occurs after Begin or Mid aligned tracing has already been triggered by a comparator instigated transition to Final State, then TRIG no longer has an effect. When the associated tracing session is complete, the breakpoint occurs. Similarly if a TRIG is followed by a subsequent comparator match, it has no effect, since tracing has already started.

6.4.7.3.1 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 6-61.

Table 6-61. 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 Debugging Through Reset

To debug through reset, the debugger can recognize a reset occurrence and pull the device BKGD pin low. This forces the device to leave reset in special single chip (SSC) mode, because the BKGD pin is used as the MODC signal in the reset phase. When the device leaves reset in SSC mode, CPU execution is halted and the device is in active BDM. Thus the debugger can configure the DBG for tracing and breakpoints before returning to application code execution. In this way it is possible to analyze the sequence of events emerging from reset. The recommended handling of the internal reset scenario is as follows:

- When a reset occurs the debugger pulls BKGD low until the reset ends, forcing SSC mode entry.
- Then the debugger reads the reset flags to determine the cause of reset.
- If required, the debugger can read the trace buffer to see what happened just before reset. Since the trace buffer and DBGCNT register are not affected by resets other than POR.
- The debugger configures and arms the DBG to start tracing on returning to application code.
- The debugger then sets the PC according to the reset flags.
- Then the debugger returns to user code with GO or STEP1.

6.5.3 Breakpoints from other S12Z sources

The DBG is neither affected by CPU BGND instructions, nor by BDC BACKGROUND commands.

6.5.4 Code Profiling

The code profiling data output pin PDO is mapped to a device pin that can also be used as GPIO in an application. If profiling is required and all pins are required in the application, it is recommended to use the device pin for a simple output function in the application, without feedback to the chip. In this way the application can still be profiled, since the pin has no effect on code flow.

The PDO provides a simple bit stream that must be strobed at both edges of the profiling clock when profiling. The external development tool activates profiling by setting the DBG ARM bit, with PROFILE and PDOE already set. Thereafter the first bit of the profiling bit stream is valid at the first rising edge of the profiling clock. No start bit is provided. The external development tool must detect this first rising edge after arming the DBG. To detect the end of profiling, the DBG ARM bit can be monitored using the BDC.

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 W	DPTR[23:16]							
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 W	DPTR[15:8]							
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 W	DDATA[15:8]							
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 W	DDATA[7:0]							
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.

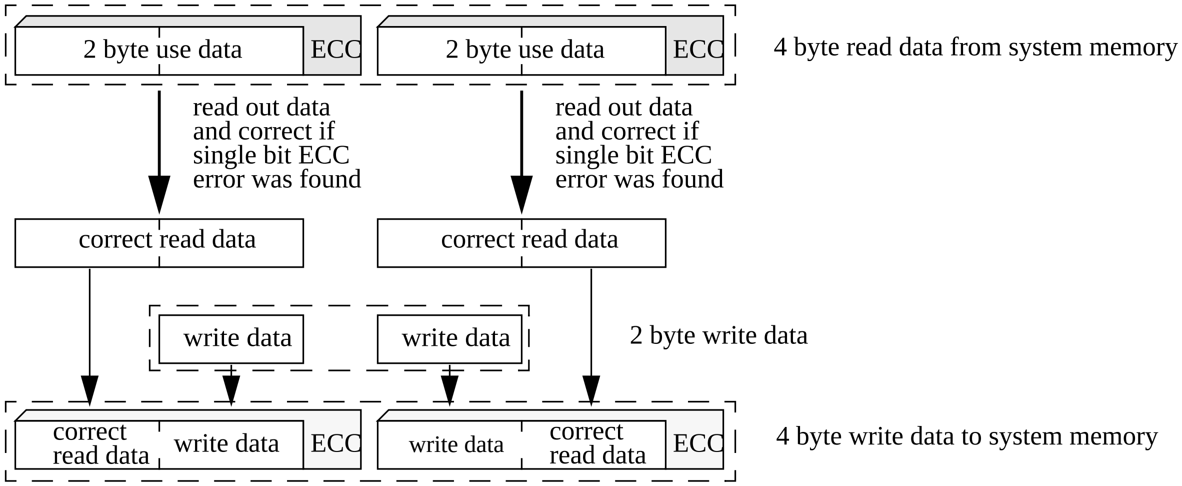


Figure 7-9. 2 byte non-aligned write access

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_V6)

Revision History

Rev. No. (Item No)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V06.02	20 Dec. 2012		• Format and font corrections • Table 8-31. CPMUOSC2 Field Descriptions: removed Bit6 and Bit4-0 description as these bits no longer exist.
V06.03	18 June 2013		• EXTCON register Bit: correct reset value to 1 • PMRF register Bit: corrected description • Memory map: corrected address typo CPMUAPIRH register
V06.04	21 Aug. 2013		• corrected bit numbering for CSAD Bit • f_{PLLRST} changed to f_{VCORST} • corrected typo in heading of CPMUOSC2 Field Description • changed frequency upper limit of external Pierce Oscillator (XOSCLCP) from 16MHz to 20MHz
V06.05	3 Jan. 2014		• corrected description of CSAD Bit

8.1 Introduction

This specification describes the function of the Clock, Reset and Power Management Unit (S12CPMU_UHV_V6).

- 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_V6 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 based 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 time-out
  - 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_V6.

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_V6 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 (please refer to CSAD bit description for details) 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 (please refer to CSAD bit description for details) 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_V6 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_V6 COP Control Register (CPMUCOP)

8.1.3 S12CPMU_UHV_V6 Block Diagram

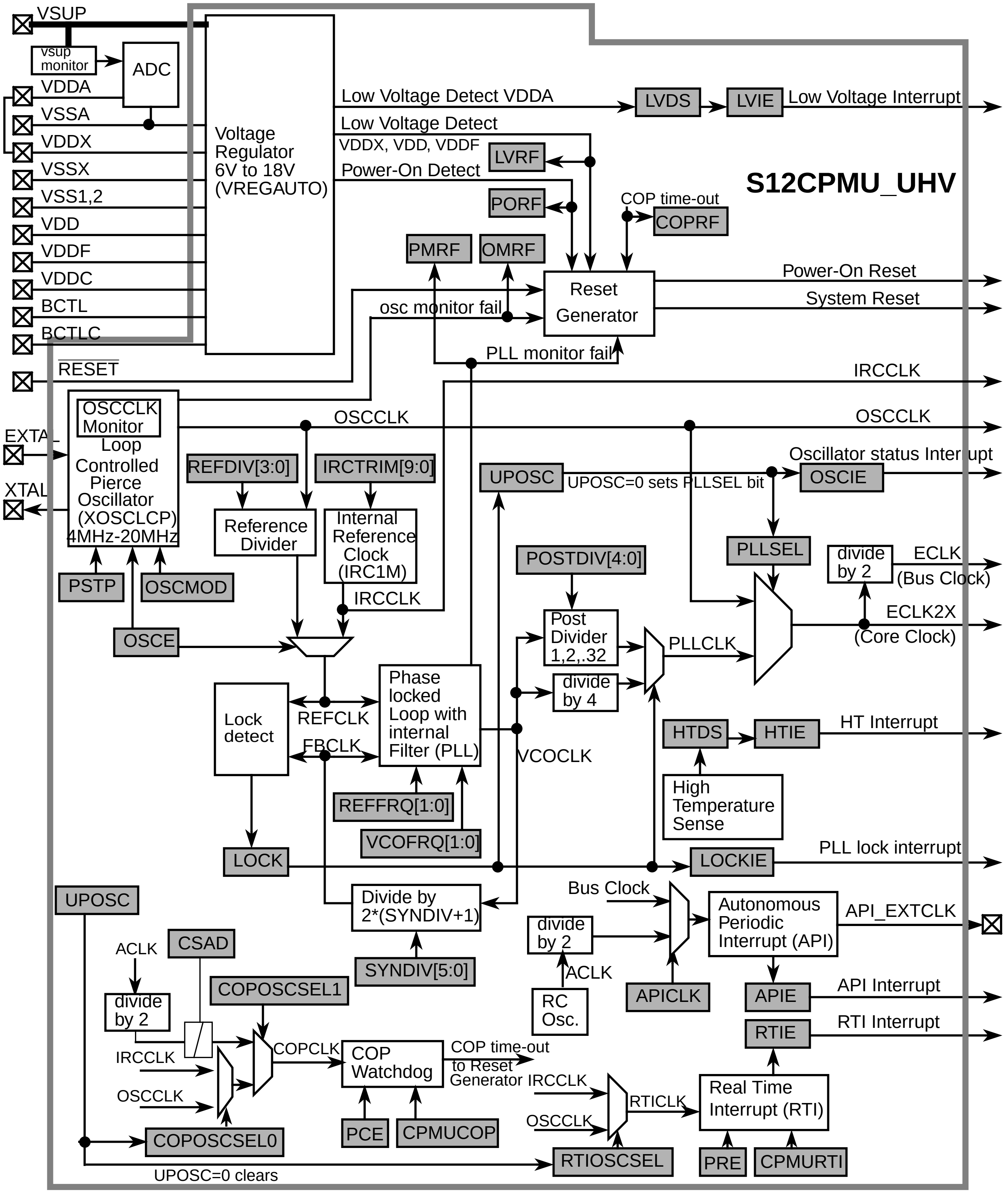


Figure 8-1. Block diagram of S12CPMU_UHV_V6

Figure 8-2 shows a block diagram of the XOSCLCP.

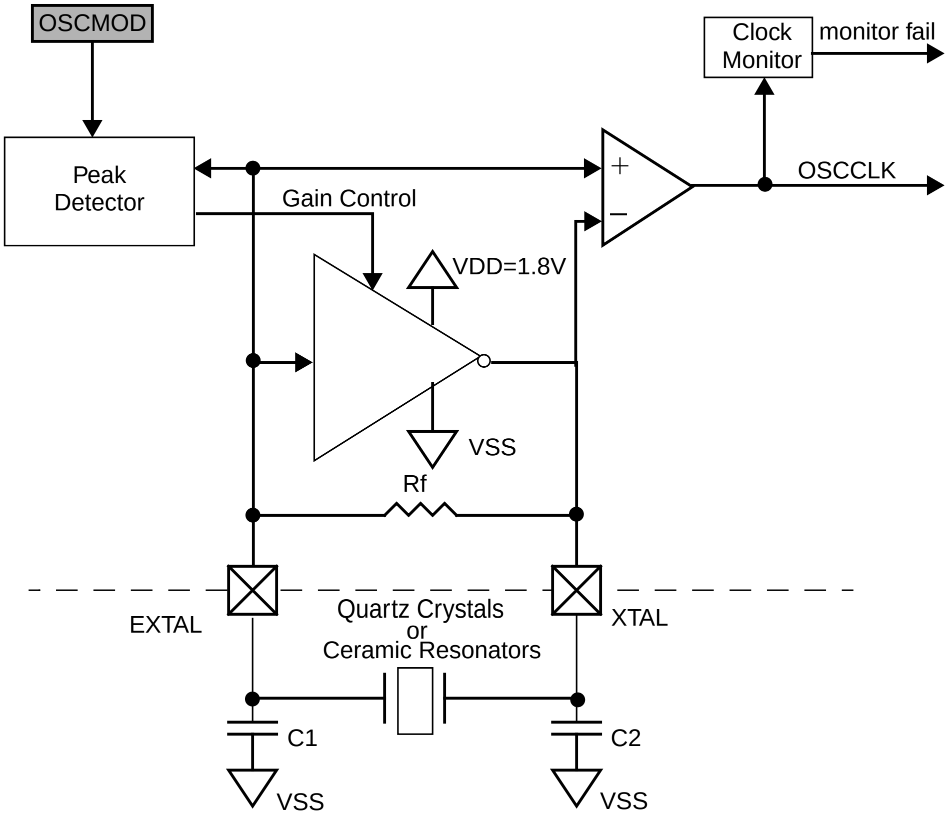


Figure 8-2. XOSCLCP Block Diagram

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

> Freescale 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.

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.

VDDX has to be connected externally to VDDA.

8.2.6 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 5.6KΩ resistor between emitter and base of the BJT is required.

8.2.7 VSS1,2 — Core Ground Pins

VSS1,2 are the core logic supply return pins. They must be grounded.

8.2.8 VDD— Core Logic Supply Pin

VDD is the supply domain for the core logic.

An off-chip decoupling capacitor (220 nF(X7R ceramic)) between VDD and VSS is required and can improve the quality of this supply.

This supply domain is monitored by the Low Voltage Reset circuit and The Power On Reset circuit.

8.2.9 VDDF— NVM Logic Supply Pin

VDDF is the supply domain for the NVM logic.

An off-chip decoupling capacitor (220 nF(X7R ceramic)) between VDDF and VSS is required and can improve the quality of this supply.

This supply domain is monitored by the Low Voltage Reset circuit.

8.2.10 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.

8.2.11 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_V6.

8.3.1 Module Memory Map

The S12CPMU_UHV_V6 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_V6 registers and their individual bits.

Address order is as listed in Figure 8-3

8.3.2.1 S12CPMU_UHV_V6 Reset Flags Register (CPMURFLG)

This register provides S12CPMU_UHV_V6 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_V6 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. 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_V6 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_V6 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_V6 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} <= 80MHz	01
Reserved	10
80MHz < f_{VCO}<= 100MHz	11

8.3.2.3 S12CPMU_UHV_V6 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_V6 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_V6 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_V6 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_V6 Interrupt Flags Register (CPMUIFLG)

This register provides S12CPMU_UHV_V6 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_V6 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_V6 Interrupt Enable Register (CPMUINT)

This register enables S12CPMU_UHV_V6 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_V6 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_V6 Clock Select Register (CPMUCLKS)

This register controls S12CPMU_UHV_V6 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_V6 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** — If this bit is set the ACLK for the COP in Stop Mode is disabled. Hence the COP is static while in Stop Mode and continues to operate after exit from Stop Mode. **For CSAD = 1 and COP is running on ACLK (COPOSCSEL1 = 1) the following applies:** Due to clock domain crossing synchronization there is a latency time of 2 ACLK cycles to enter Stop Mode. **After exit from STOP mode (when interrupt service routine is entered) the software has to wait for 2 ACLK cycles before it is allowed to enter Stop mode again (STOP instruction). It is absolutely forbidden to enter Stop Mode before this time of 2 ACLK cycles has elapsed.** 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_V6 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_V6 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_V6 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_V6 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_V6 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_V6 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_V6’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_V6’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_V6 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_V6 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)

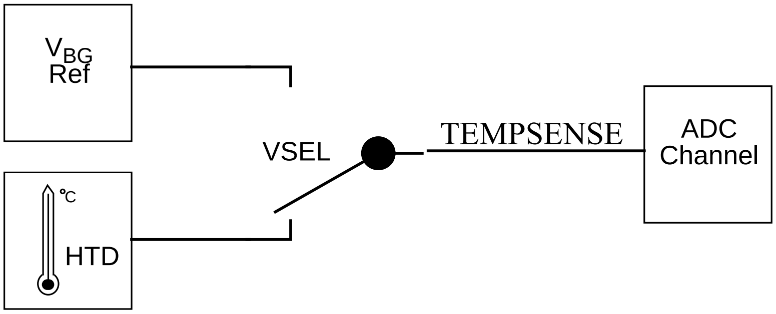
Read: Anytime

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)

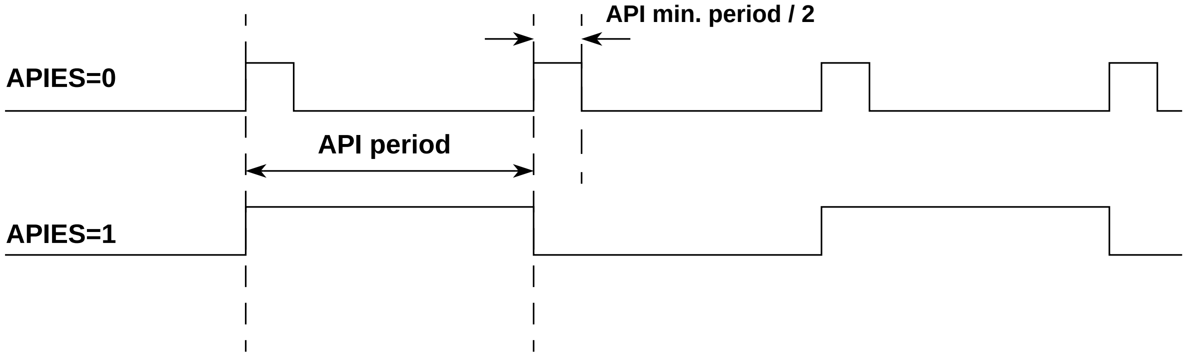
Read: Anytime

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_V6'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_V6 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_V6 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_V6 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_V6 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.

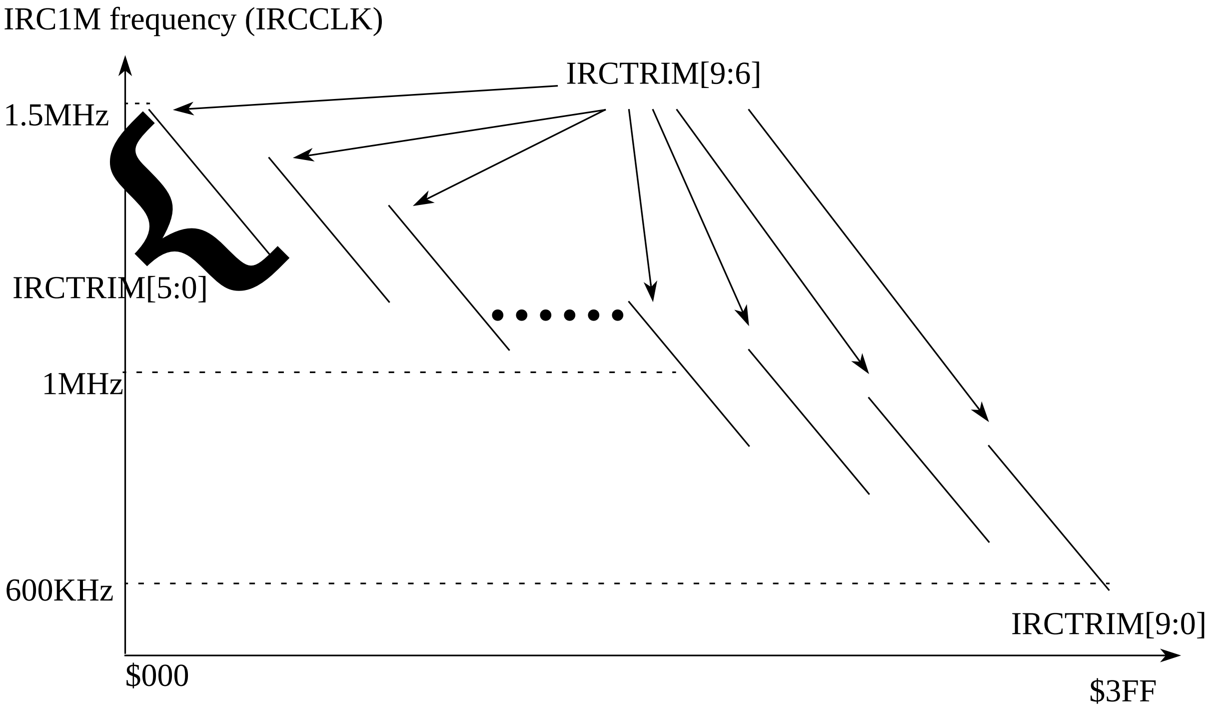


Figure 8-29. IRC1M Frequency Trimming Diagram

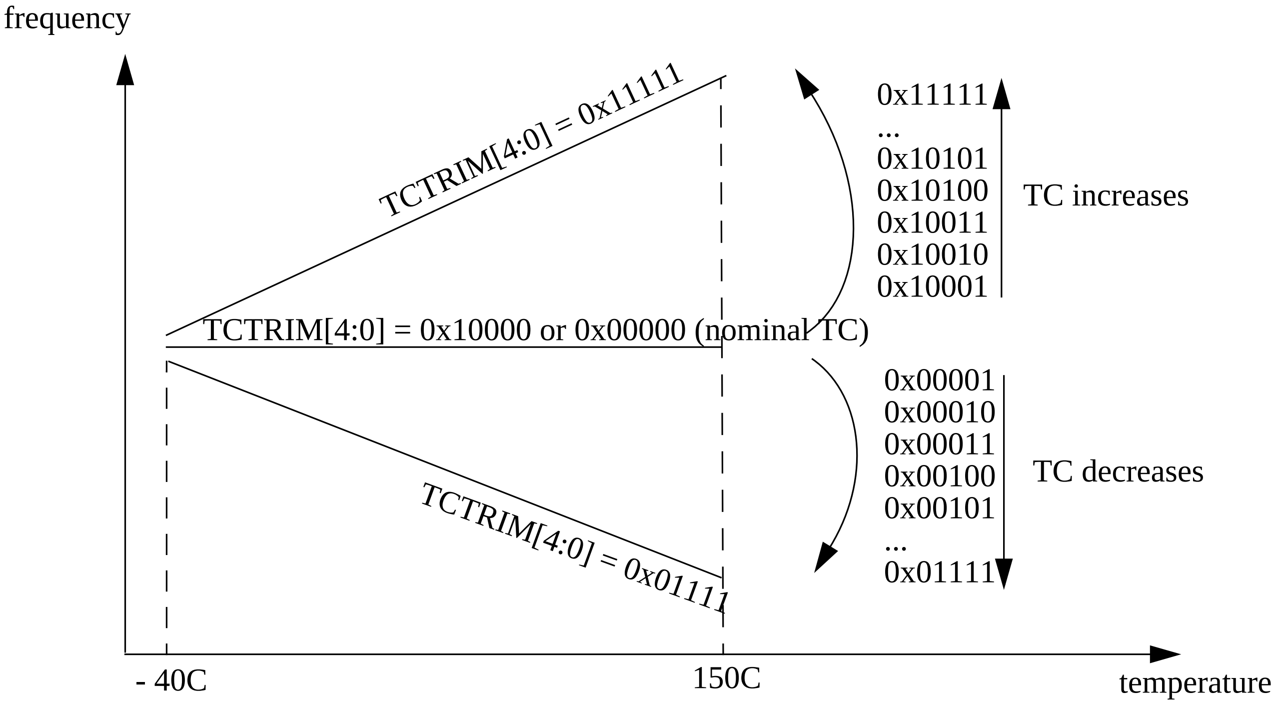


Figure 8-30. Influence of TCTRIM[4:0] on the Temperature Coefficient

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_V6 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_V6 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_V6 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_V6 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_V6'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_V6 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_V6 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
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
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)

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.

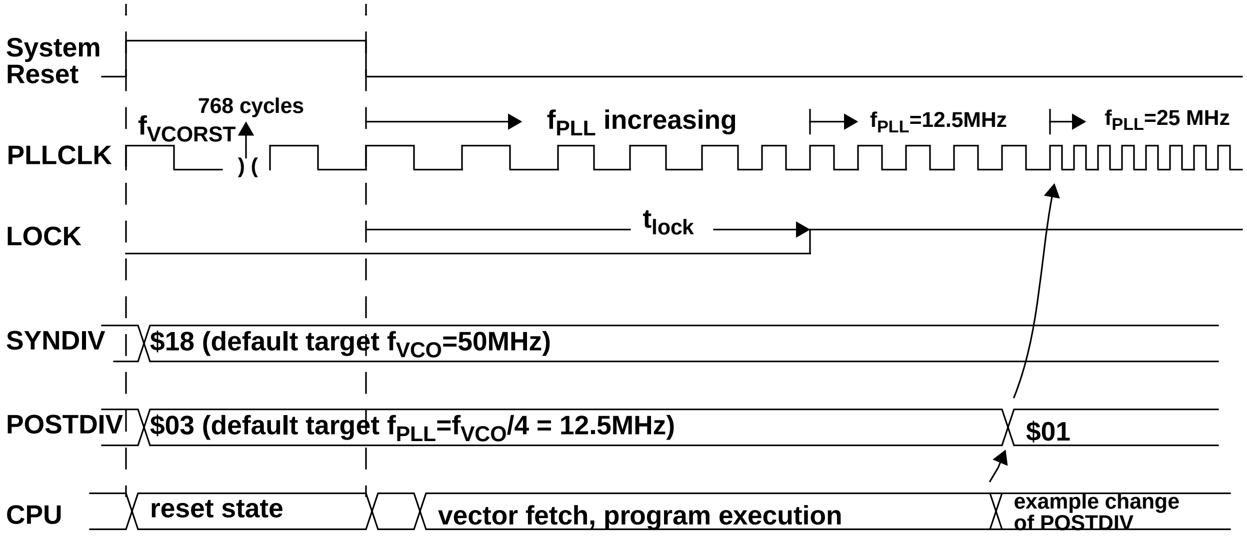
If PLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and for instance check the LOCK bit. If interrupt requests are disabled, software can poll the LOCK bit continuously (during PLL start-up) or at periodic intervals. In either case, only when the LOCK bit is set, 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.

8.4.2 Startup from Reset

An example for startup of the clock system from Reset is given in Figure 8-36.

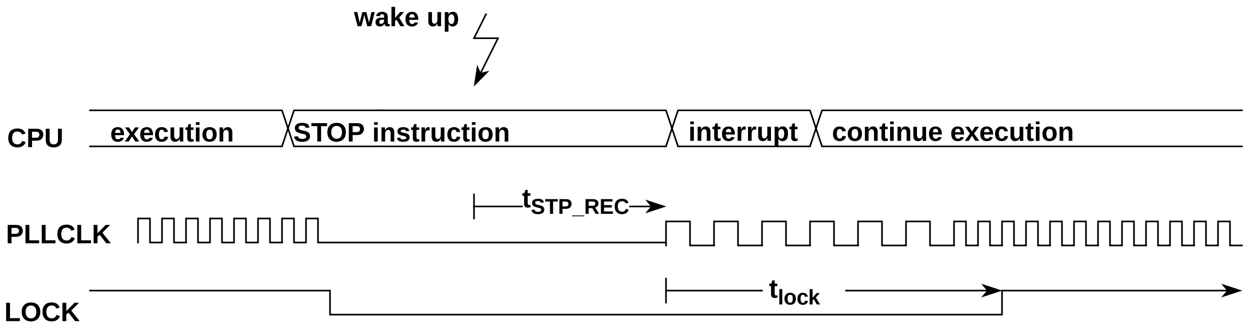
Figure 8-36. Startup of clock system after Reset



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



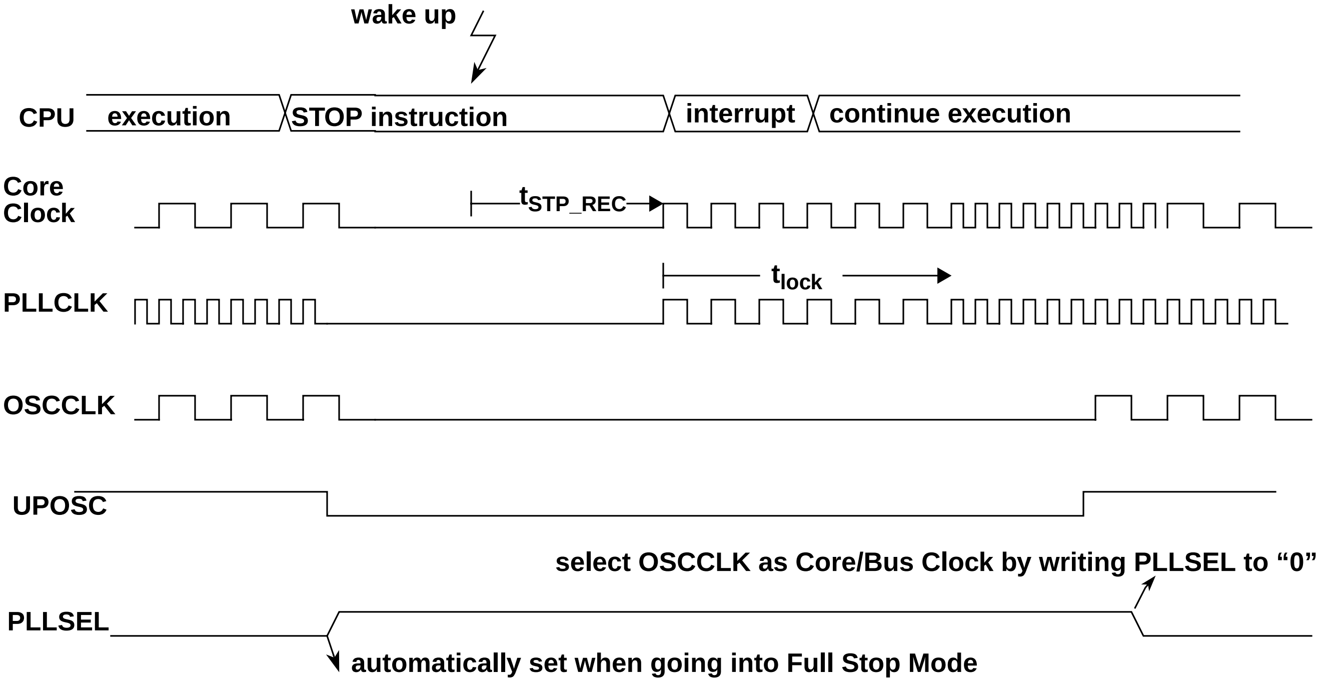
Depending on the COP configuration there might be an additional significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time occurs if COP clock source is ACLK and the CSAD bit is set (please refer to CSAD bit description for details).

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



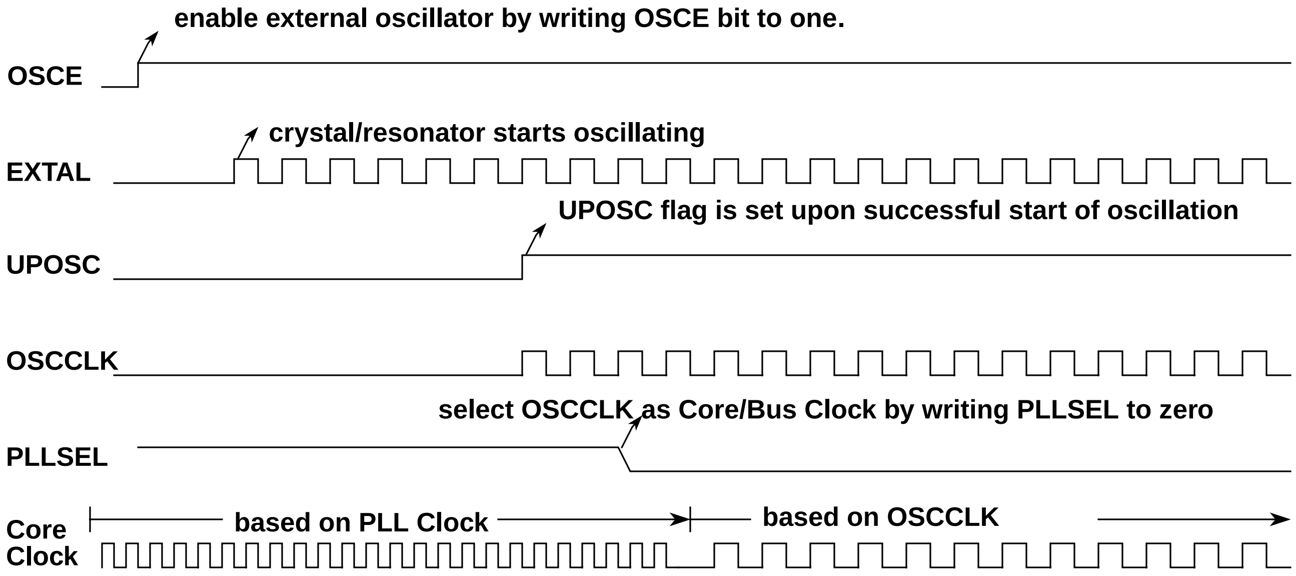
Depending on the COP configuration there might be a significant latency time until COP is active again after exit from Stop Mode due to clock domain crossing synchronization. This latency time occurs if COP clock source is ACLK and the CSAD bit is set (please refer to CSAD bit description for details).

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 Bus clock source 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

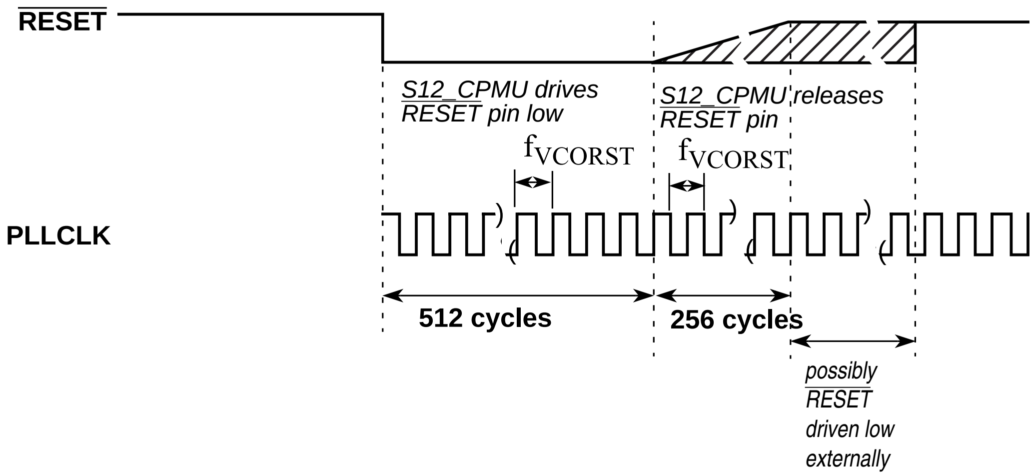
8.5.2 Description of Reset Operation

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_V6 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_V6 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.

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 occurs if COP clock source is ACLK and the CSAD bit is set (please refer to CSAD bit description for details).

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_V6 are listed in Table 8-35. Refer to MCU specification for related vector addresses and priorities.

Table 8-35. S12CPMU_UHV_V6 Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
RTI time-out interrupt	I bit	CPMUINT (RTIE)
PLL lock interrupt	I bit	CPMUINT (LOCKIE)

Table 8-35. S12CPMU_UHV_V6 Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
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_V6 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].

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.

---

[1] For details please refer to “8.4.6 System Clock Configurations”

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
V1.38	30. Apr 2013	9.4.2.13/9-335	Provided more detailed information regarding captured information in bits RIDX_IMD[5:0] for different scenarios of Sequence Abort Event execution.
V1.39	02. Jul 2013	9.4.2.6/9-324	Update of: Timing considerations for Restart Mode
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 µs). 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).

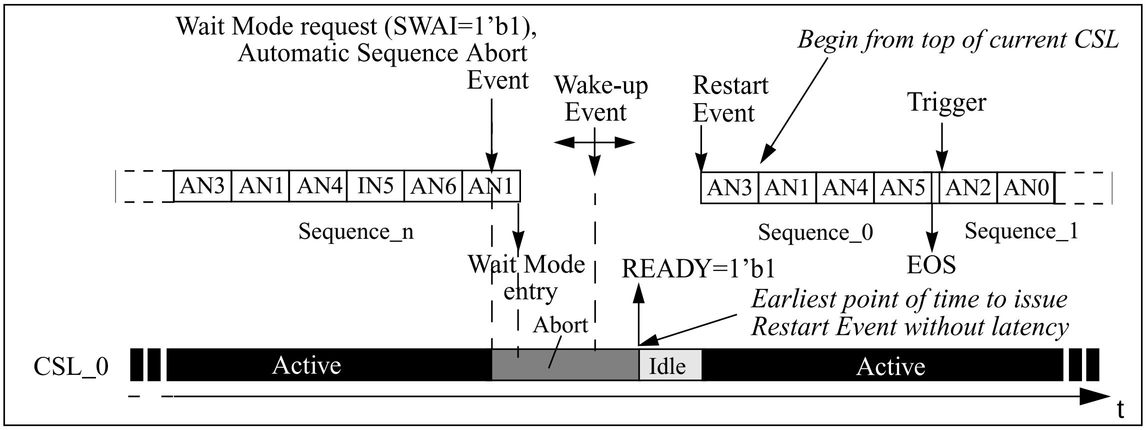


Figure 9-1. Conversion Flow Control Diagram - Wait Mode (SWAI=1’b1, AUT_RSTA=1’b0)

- **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

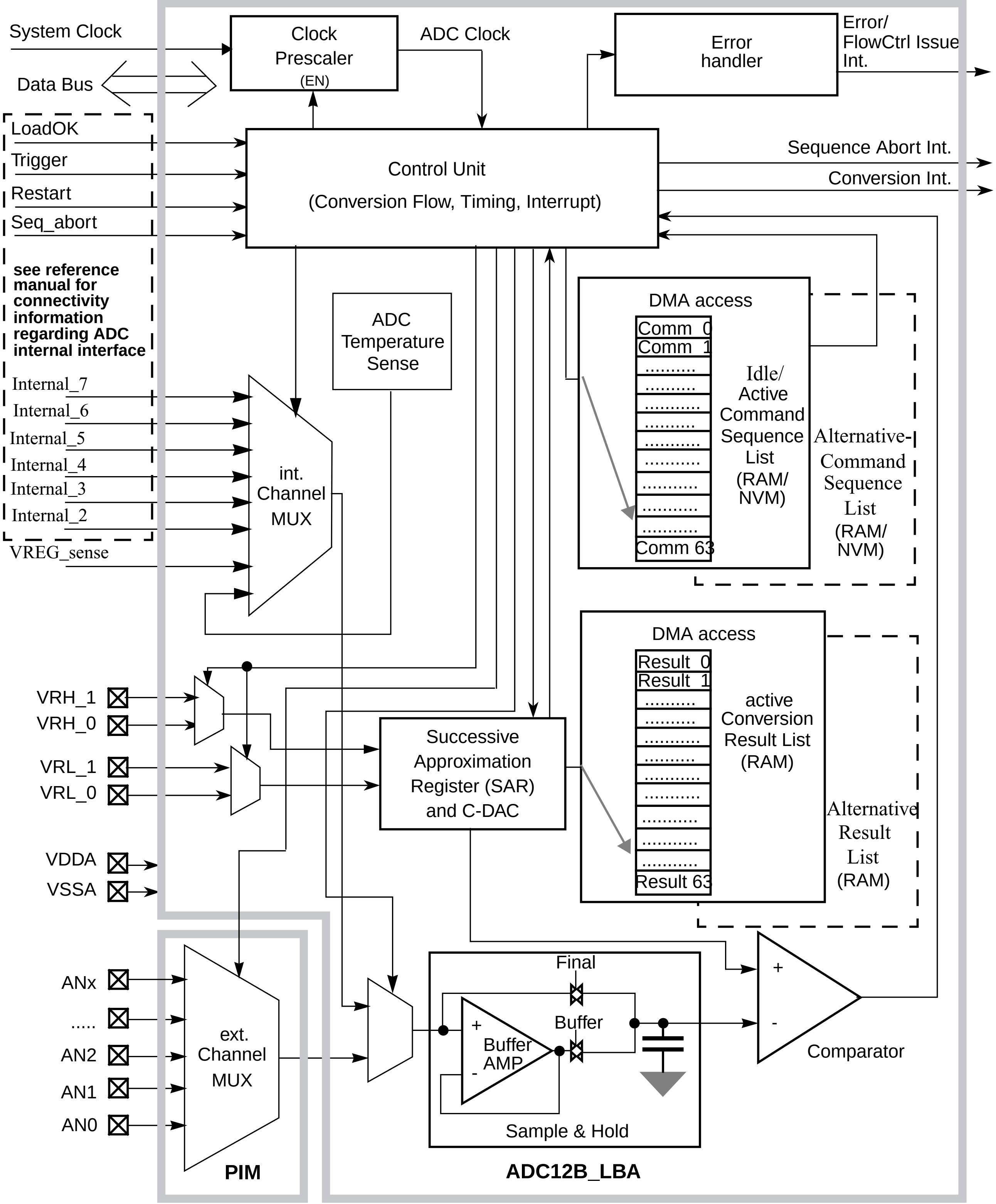


Figure 9-2. ADC12B_LBA Block Diagram

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.

Module Base + 0x000C

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

▨ = Unimplemented or Reserved

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	Internal_0 (ADC temperature sense)
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.

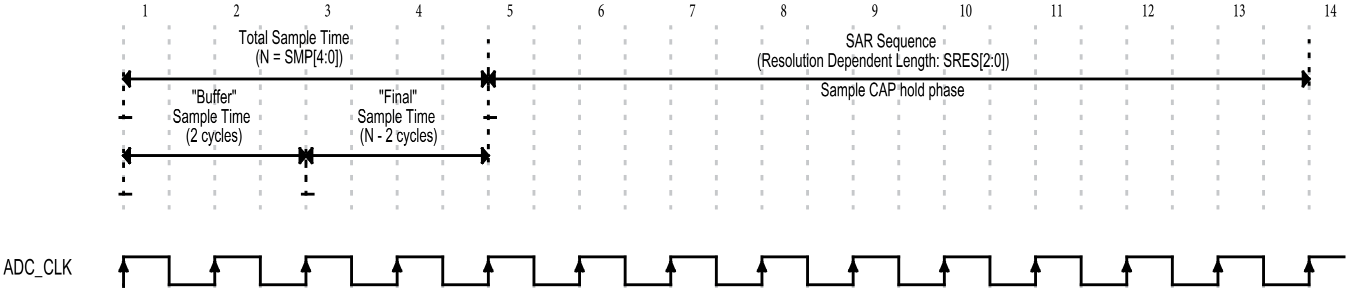


Figure 9-28. Sampling and Conversion Timing Example (8-bit Resolution, 4 Cycle Sampling)

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.

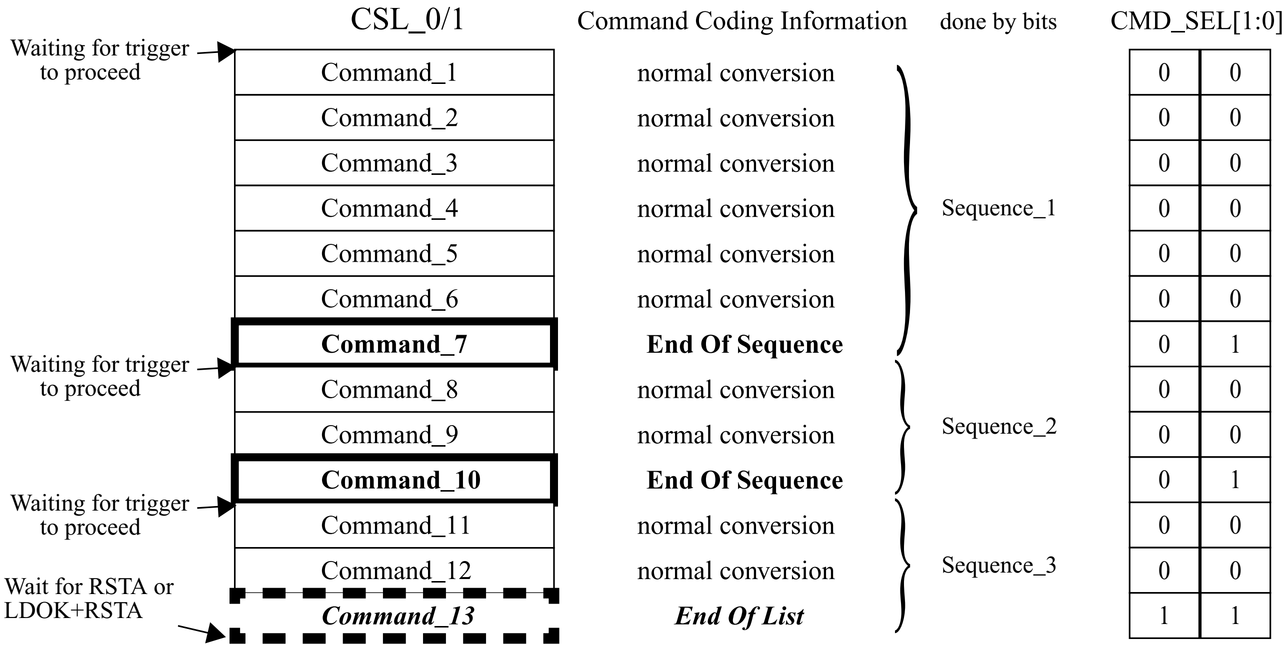
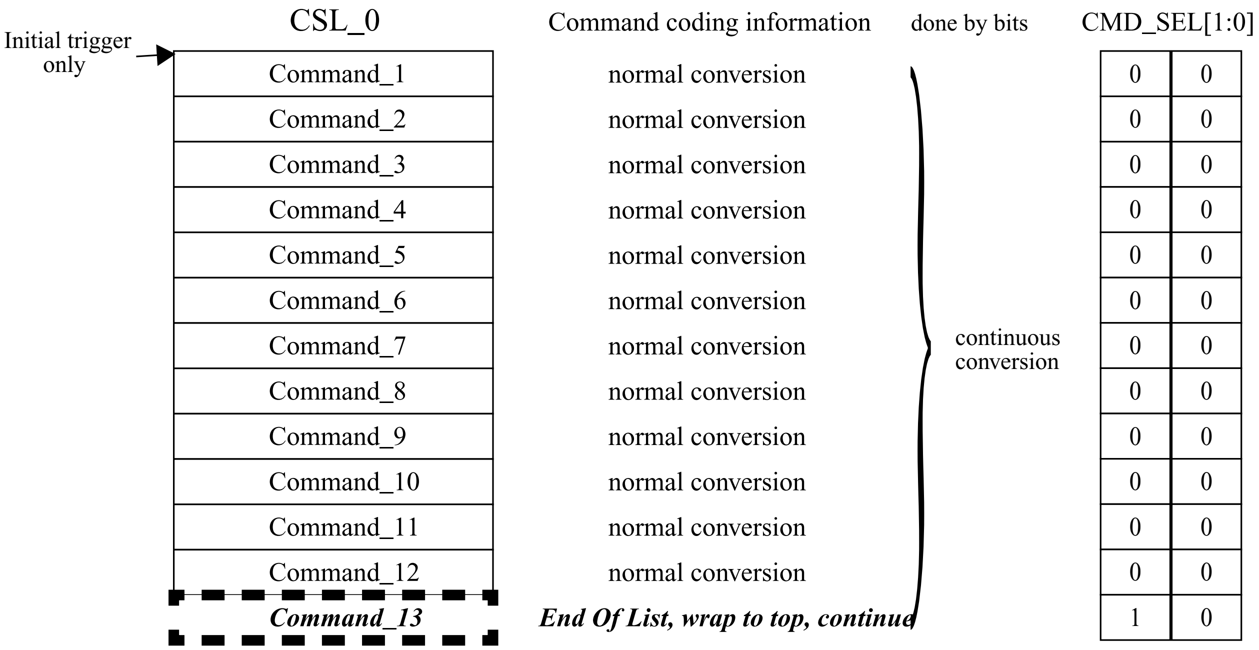


Figure 9-29. Example CSL with sequences and an “End Of List” command type identifier

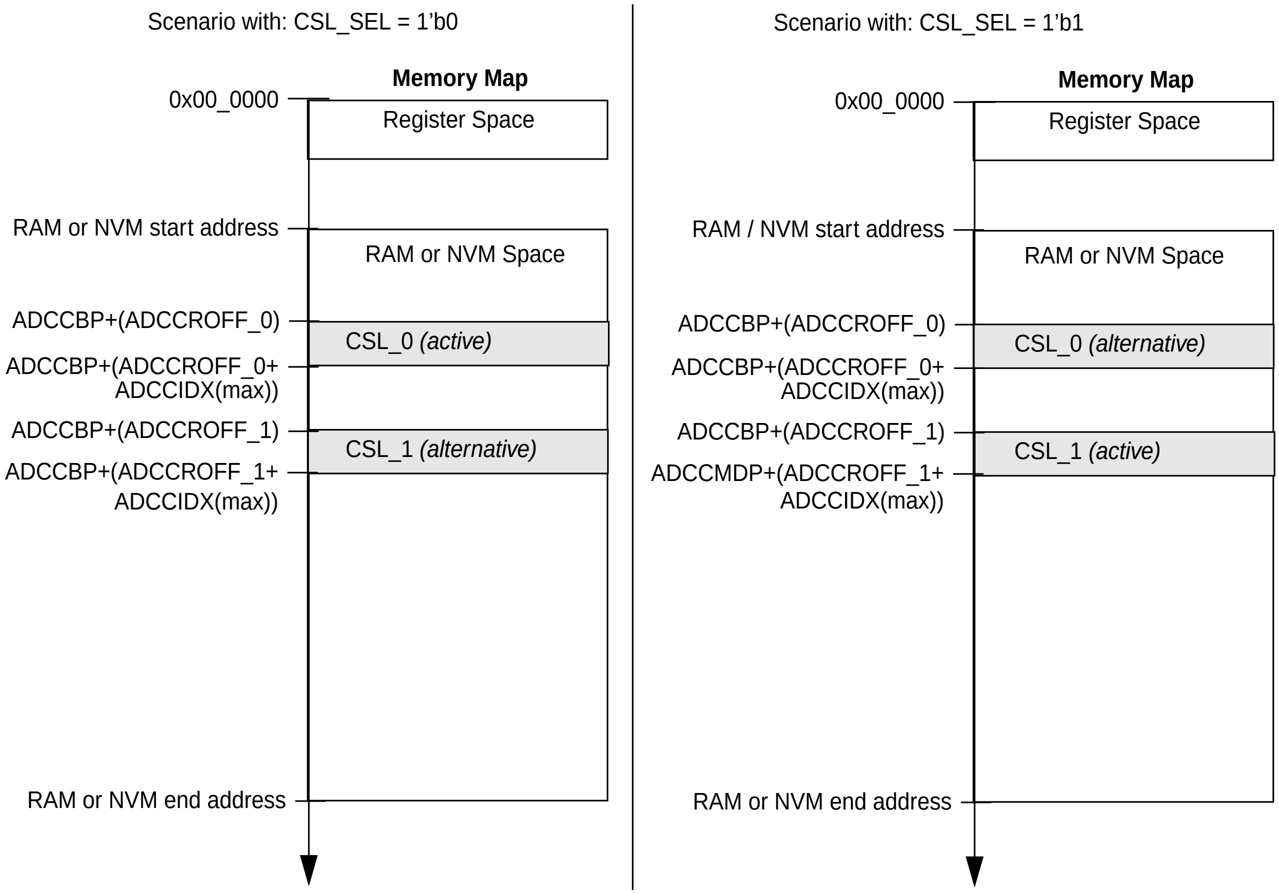


CSL_0	Command coding information	done by bits	CMD_SEL[1:0]	
Command_1	normal conversion	continuous conversion	0	0
Command_2	normal conversion		0	0
Command_3	normal conversion		0	0
Command_4	normal conversion		0	0
Command_5	normal conversion		0	0
Command_6	normal conversion		0	0
Command_7	normal conversion		0	0
Command_8	normal conversion		0	0
Command_9	normal conversion		0	0
Command_10	normal conversion		0	0
Command_11	normal conversion		0	0
Command_12	normal conversion		0	0
Command_13	***End Of List, wrap to top, continue***		1	0

Figure 9-30. Example CSL for continues conversion

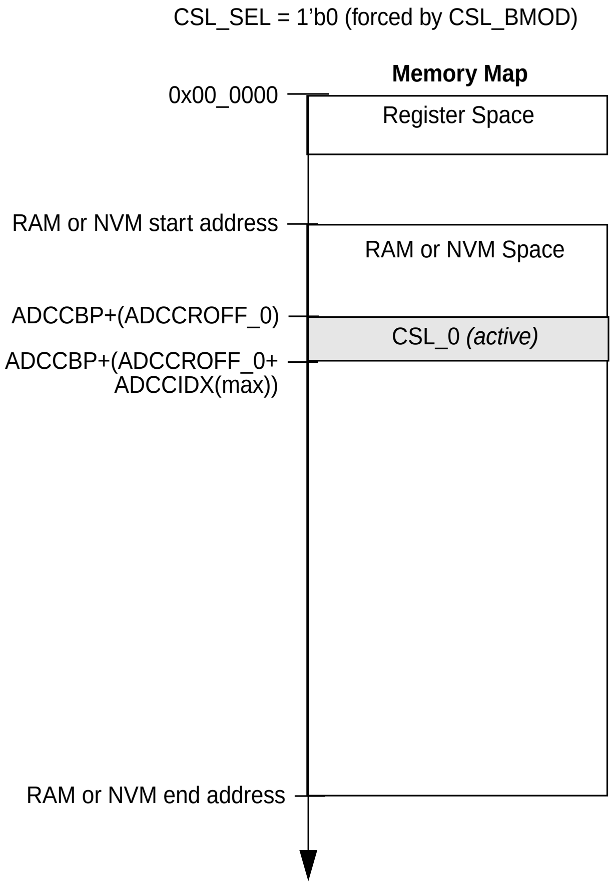
9.5.3.2.2 Introduction of the two Command Sequence Lists (CSLs)

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

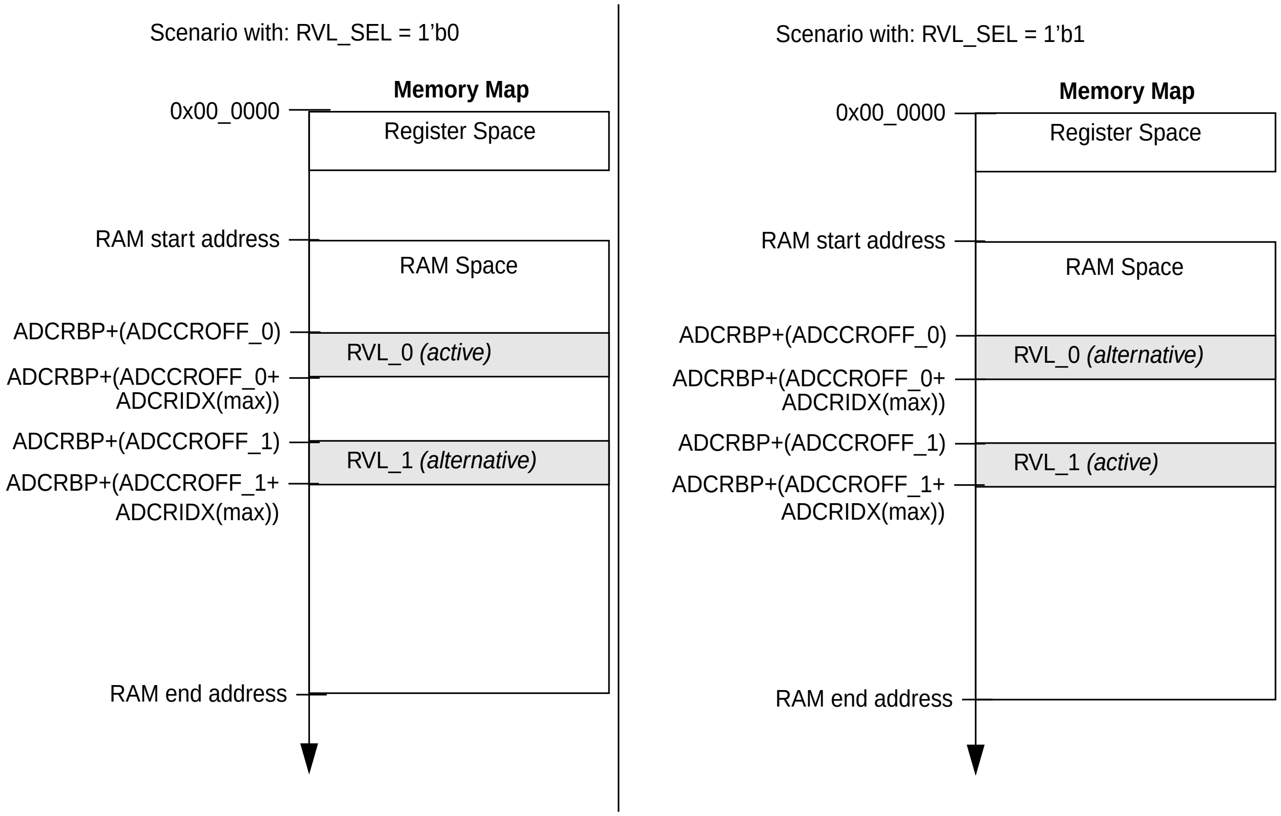
While the ADC is enabled, one CSL is active (indicated by bit CSL_SEL) and the corresponding list should not be modified anymore. At the same time the alternative CSL can be modified to prepare the ADC for new conversion sequences in CSL double buffered mode. When the ADC is enabled, the command address 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:

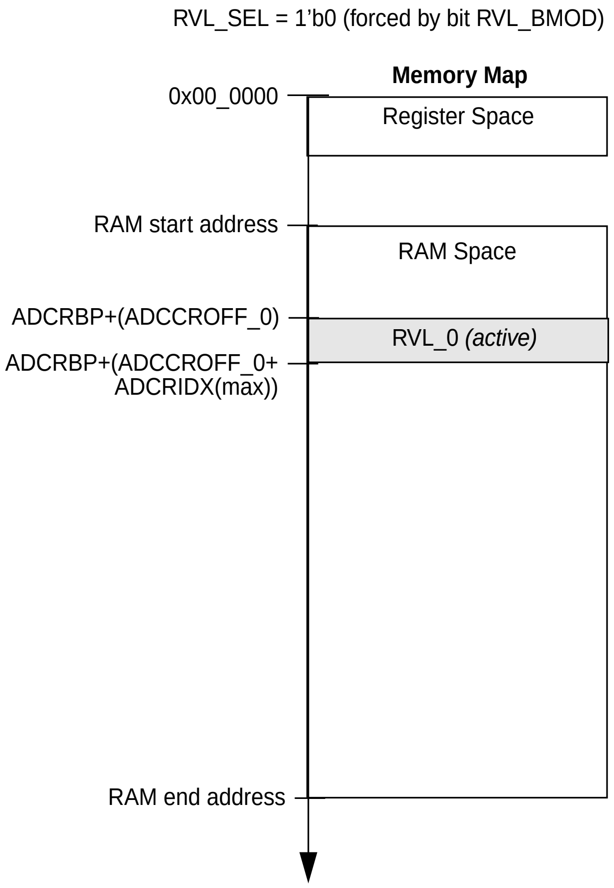
- Each time an “End Of List” command type got executed followed by the first conversion from top of the next CSL and related (first) result is about to be stored
- A CSL got aborted (bit SEQA=1’b1) and ADC enters idle state (becomes ready for new flow control events)

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



Note: Address register names in () are not absolute addresses instead they are a sample offset or sample index

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

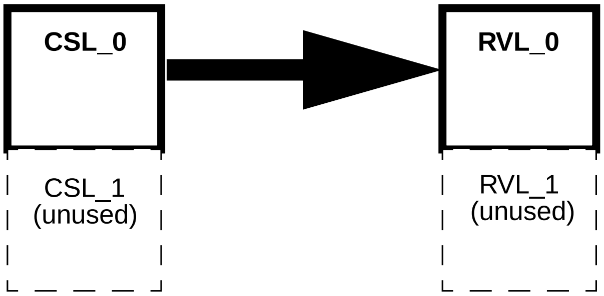


Figure 9-35. CSL Single Buffer Mode — RVL Single Buffer Mode Diagram

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.

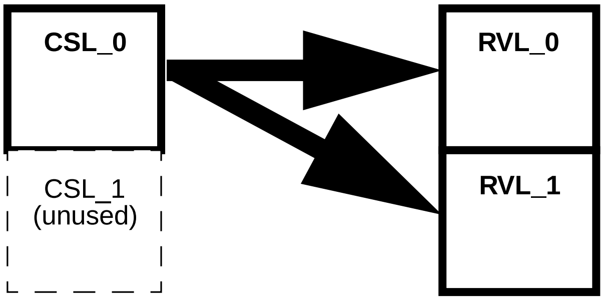


Figure 9-36. CSL Single Buffer Mode — RVL Single Buffer Mode Diagram

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.

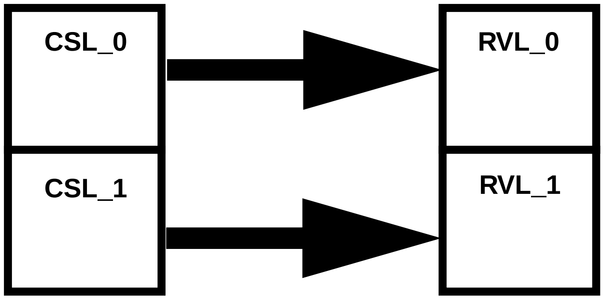


Figure 9-37. CSL Double Buffer Mode — RVL Double Buffer Mode Diagram

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.

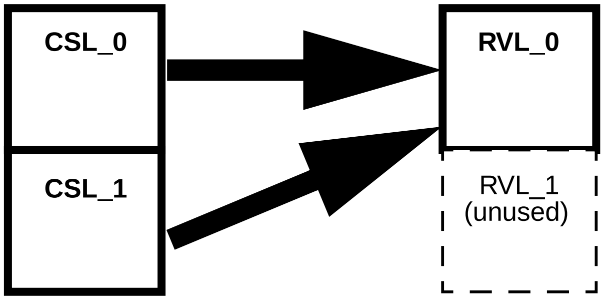


Figure 9-38. CSL Double Buffer Mode — RVL Single Buffer Mode Diagram

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.

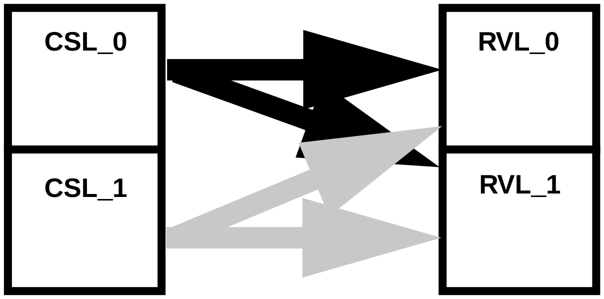
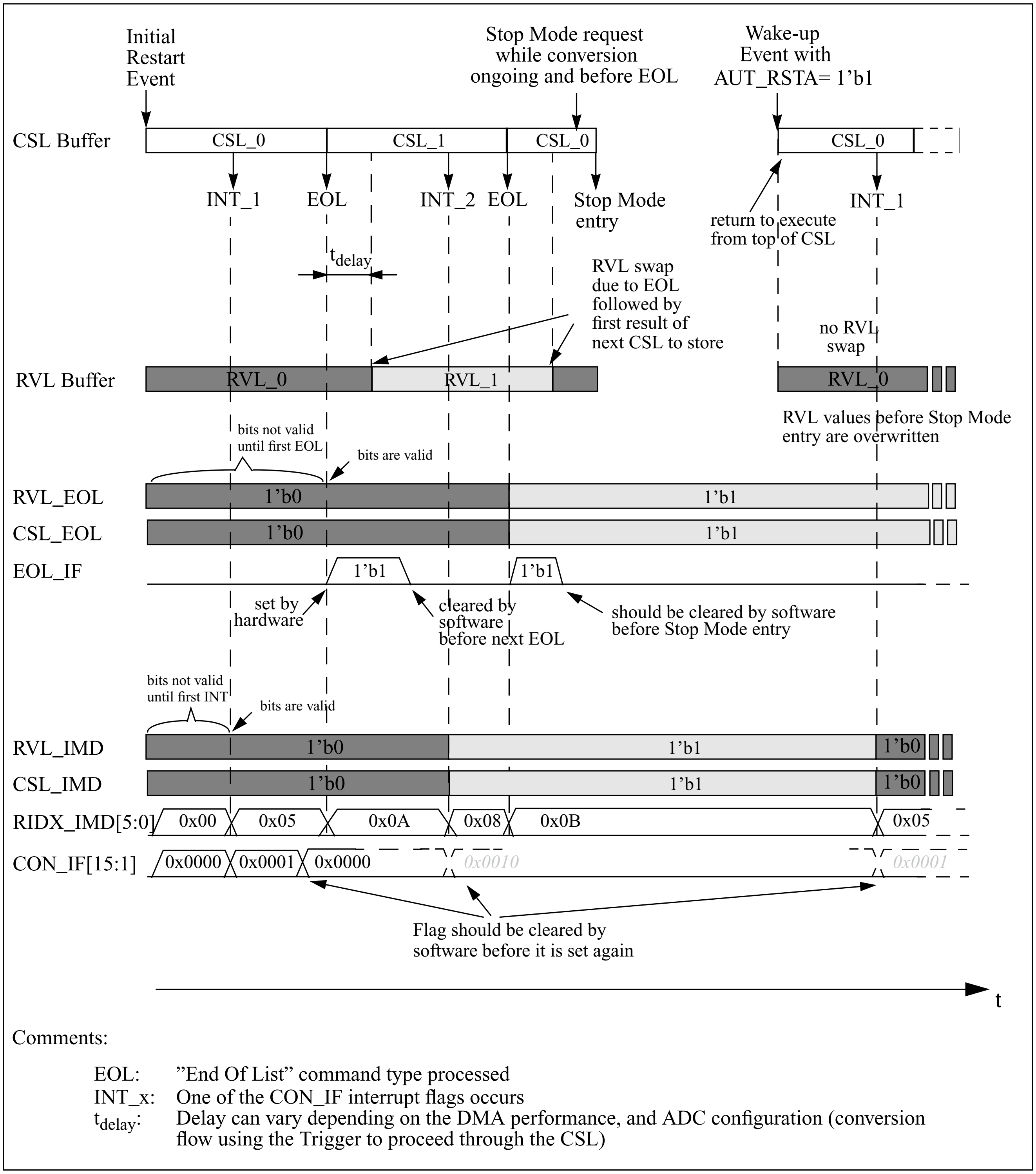


Figure 9-39. CSL Double Buffer Mode — RVL Double Buffer Mode Diagram

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.

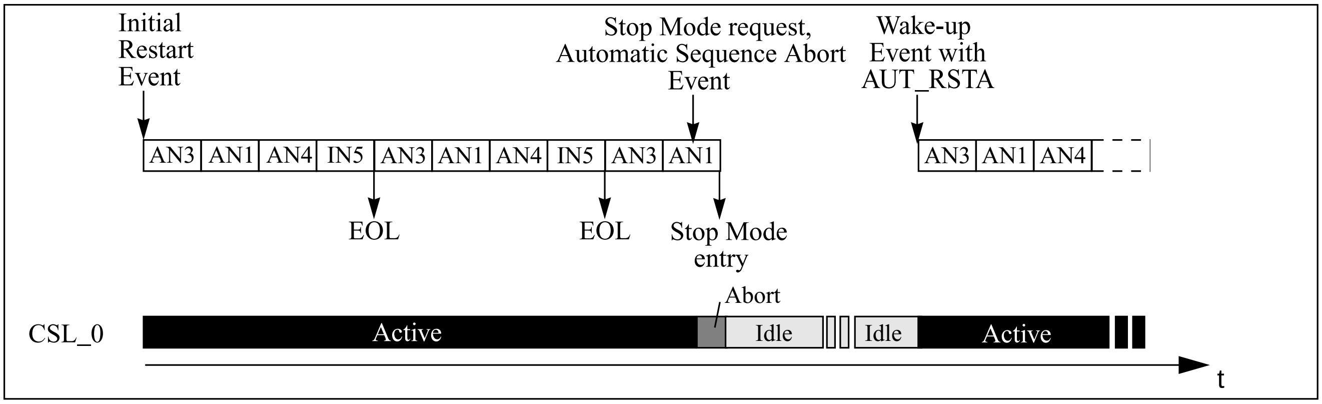


Figure 9-41. Conversion Flow Control Diagram — Continuous Conversion (with Stop Mode)

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.

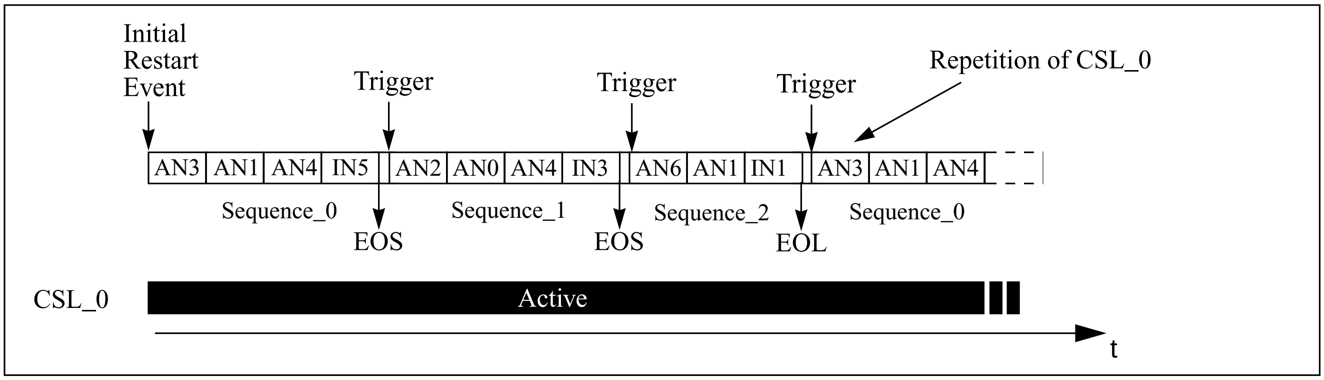


Figure 9-42. Conversion Flow Control Diagram — Triggered Conversion (CSL Repetition)

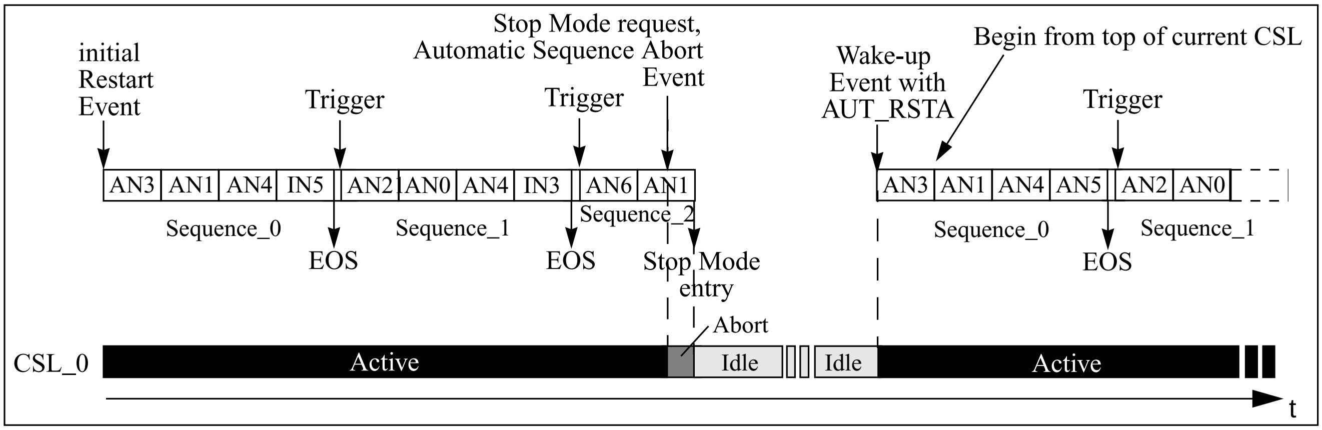


Figure 9-43. Conversion Flow Control Diagram — Triggered Conversion (with Stop Mode)

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.

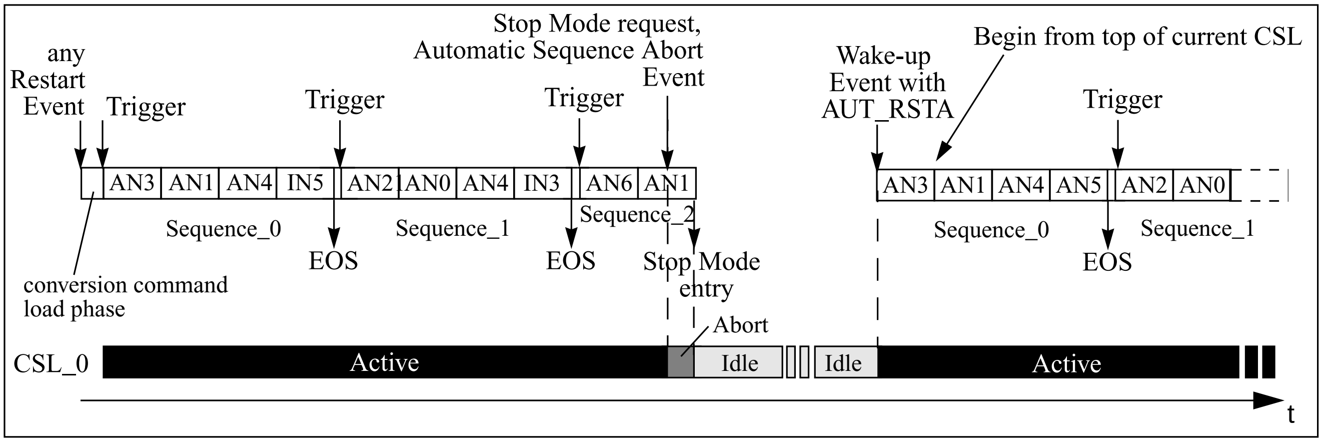


Figure 9-44. Conversion Flow Control Diagram — Fully Timing Controlled Conversion (with Stop Mode)

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.

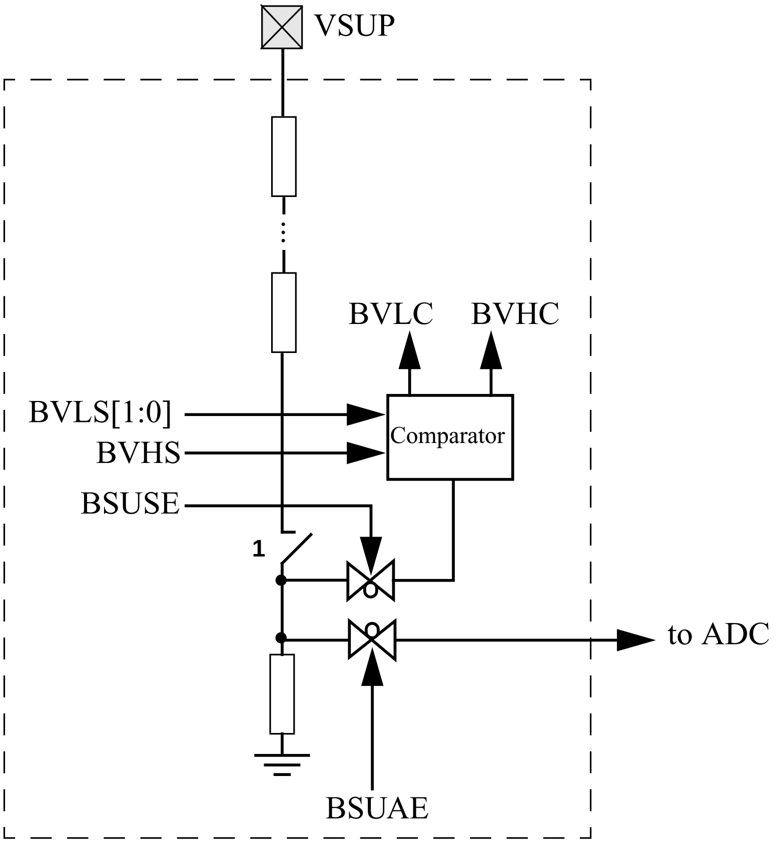
2. Stop mode

   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



1 automatically 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

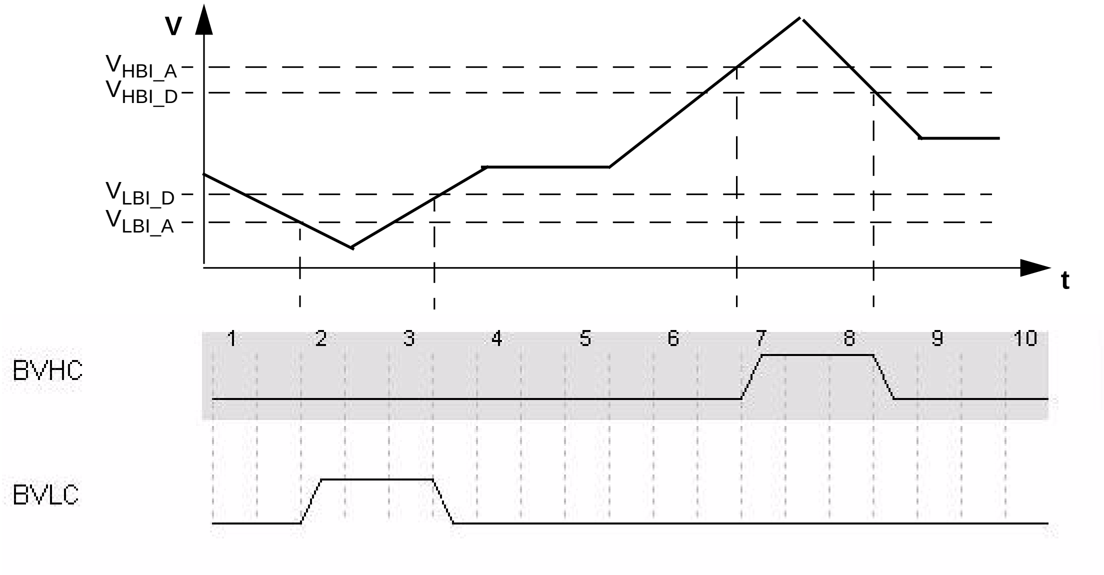
Figure 10-4. BATS Module Status Register (BATSR)

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 (TIM16B4CV3) Block Description

Table 11-1.

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.

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.

11.1.1 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.

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

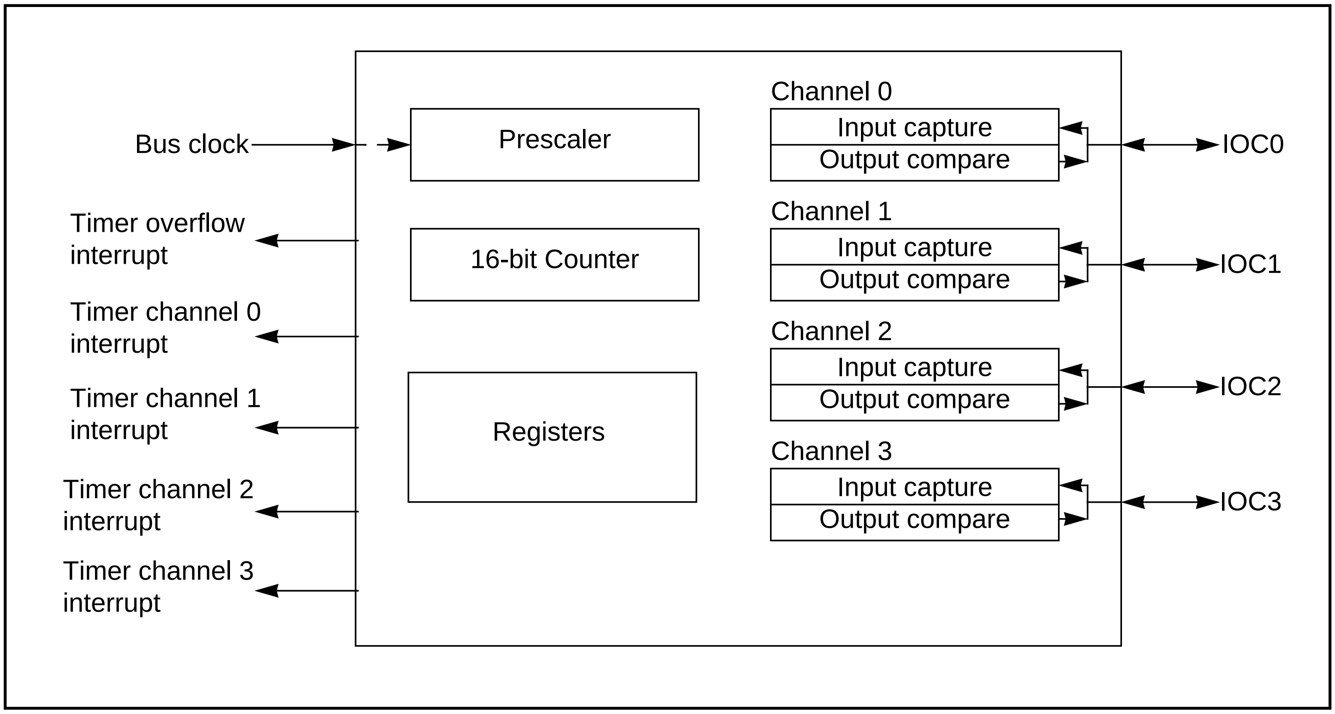


Figure 11-1. TIM16B4CV3 Block Diagram

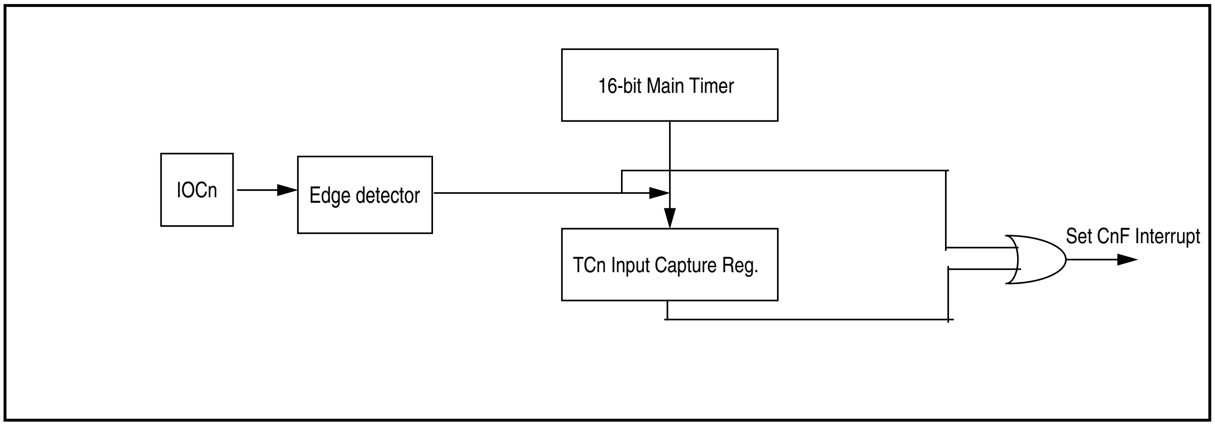


Figure 11-2. Interrupt Flag Setting

11.2 External Signal Description

The TIM16B4CV3 module has a selected number of external pins. Refer to device specification for exact number.

11.2.1 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 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 TIM16B4CV3 module is given below in Figure 11-3. 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.

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 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

Figure 11-3. TIM16B4CV3 Register Summary (Sheet 1 of 2)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
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 11-3. TIM16B4CV3 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	RESERVED	RESERVED	RESERVED	RESERVED	IOS3	IOS2	IOS1	IOS0
Reset	0	0	0	0	0	0	0	0

Figure 11-4. 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
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.

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	RESERVED	RESERVED	RESERVED	RESERVED	FOC3	FOC2	FOC1	FOC0
Reset	0	0	0	0	0	0	0	0

Figure 11-5. 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
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.

11.3.2.3 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 11-6. 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 11-7. 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.

11.3.2.4 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 11-8. Timer System Control Register 1 (TSCR1)

Read: Anytime

Write: Anytime

Table 11-4. 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.

Table 11-4. 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. 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.

11.3.2.5 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 11-9. Timer Toggle On Overflow Register 1 (TTOV)

Read: Anytime

Write: Anytime

Table 11-5. 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.

11.3.2.6 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 11-10. 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 11-11. Timer Control Register 2 (TCTL2)

Read: Anytime

Write: Anytime

Table 11-6. 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 11-7. 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

11.3.2.7 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 11-12. 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 11-13. Timer Control Register 4 (TCTL4)

Read: Anytime

Write: Anytime.

Table 11-8. 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 11-9. 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)

11.3.2.8 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 11-14. Timer Interrupt Enable Register (TIE)

Read: Anytime

Write: Anytime.

Table 11-10. 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.

11.3.2.9 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 11-15. Timer System Control Register 2 (TSCR2)

Read: Anytime

Write: Anytime.

Table 11-11. 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 Bus Clock as shown in Table 11-12.

Table 11-12. 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.

11.3.2.10 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 11-16. 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 11-13. 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.

11.3.2.11 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 11-17. 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 11-14. 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 (See also TCRE control bit explanation.)

11.3.2.12 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 11-18. 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 11-19. 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.

11.3.2.13 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 11-20. Output Compare Pin Disconnect Register (OCPD)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 11-15. 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 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.

11.3.2.14 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 11-21. Precision Timer Prescaler Select Register (PTPSR)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 11-16. 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 11-17 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 11-17. 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

11.4 Functional Description

This section provides a complete functional description of the timer TIM16B4CV3 block. Please refer to the detailed timer block diagram in Figure 11-22 as necessary.

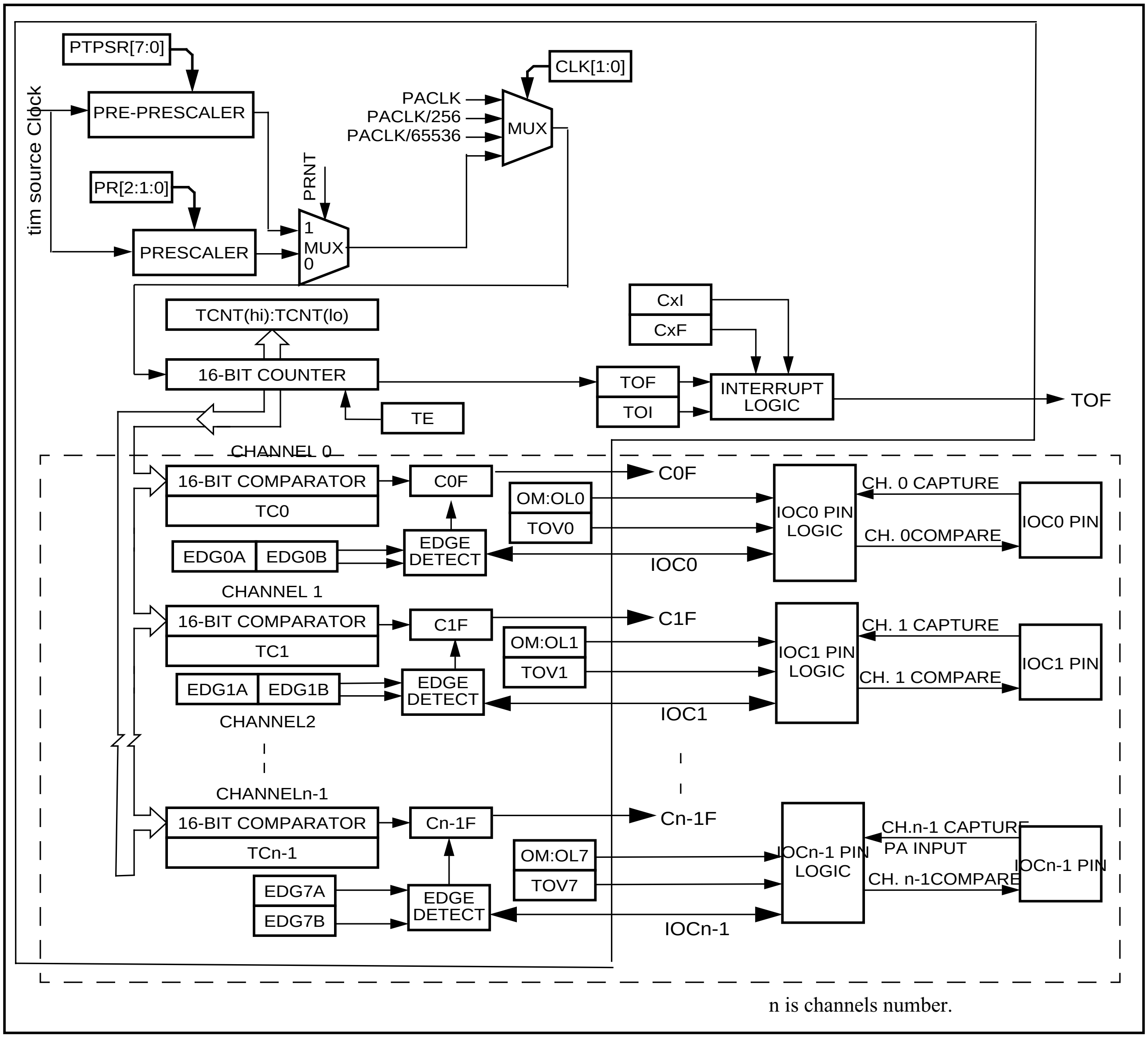


Figure 11-22. Detailed Timer Block Diagram

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 must stay enabled (TEN bit of TSCR1 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 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.

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.

11.5 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 TIM16B4CV3 block. Table 11-18 lists the interrupts generated by the TIM16B4CV3 to communicate with the MCU.

Table 11-18. 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.

11.6.1 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.

11.6.2 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 12
Freescale's Scalable Controller Area Network (S12MSCANV3)

Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V03.14	12 Nov 2012	Table 12-10	• Corrected RxWRN and TxWRN threshold values
V03.15	12 Jan 2013	Table 12-2 Table 12-25 Figure 12-37 12.1/12-403 12.3.2.15/12-425	• 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

12.1 Introduction

Freescale'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.

12.1.1 Glossary

Table 12-1. 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

12.1.2 Block Diagram

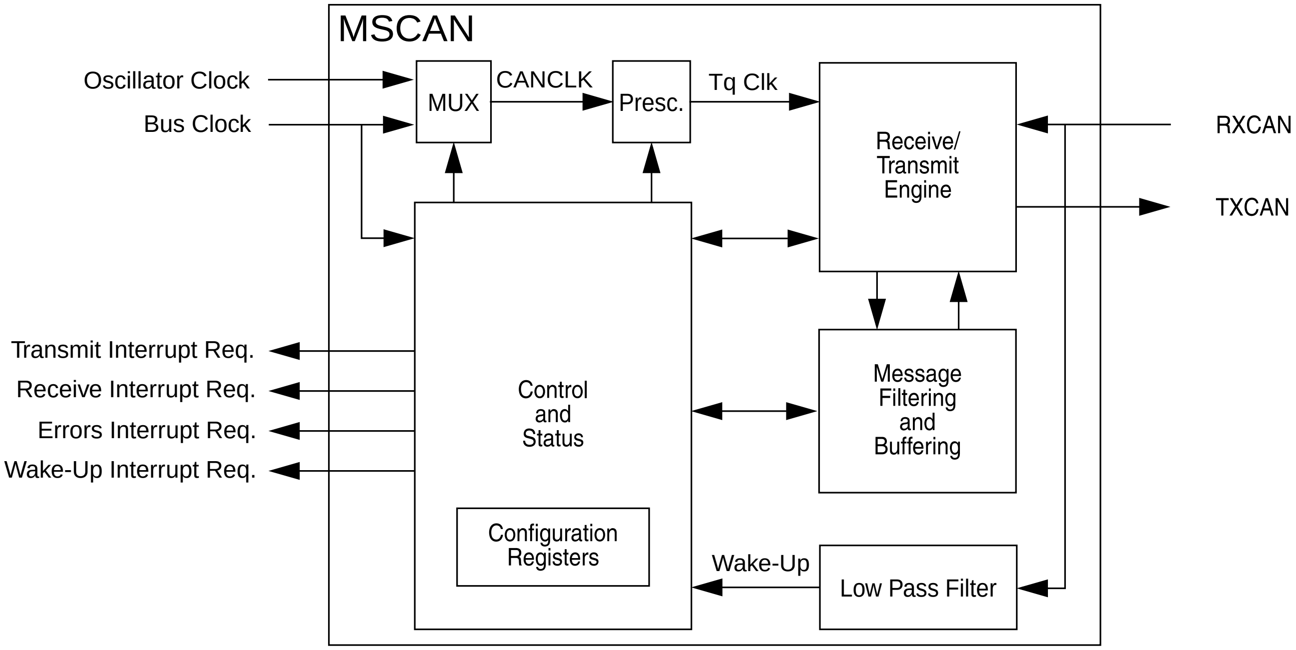


Figure 12-1. MSCAN Block Diagram

12.1.3 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

12.1.4 Modes of Operation

For a description of the specific MSCAN modes and the module operation related to the system operating modes refer to Section 12.4.4, "Modes of Operation".

---

[1] Depending on the actual bit timing and the clock jitter of the PLL.

12.2 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.

12.2.1 RXCAN — CAN Receiver Input Pin

RXCAN is the MSCAN receiver input pin.

12.2.2 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

12.2.3 CAN System

A typical CAN system with MSCAN is shown in Figure 12-2. 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.

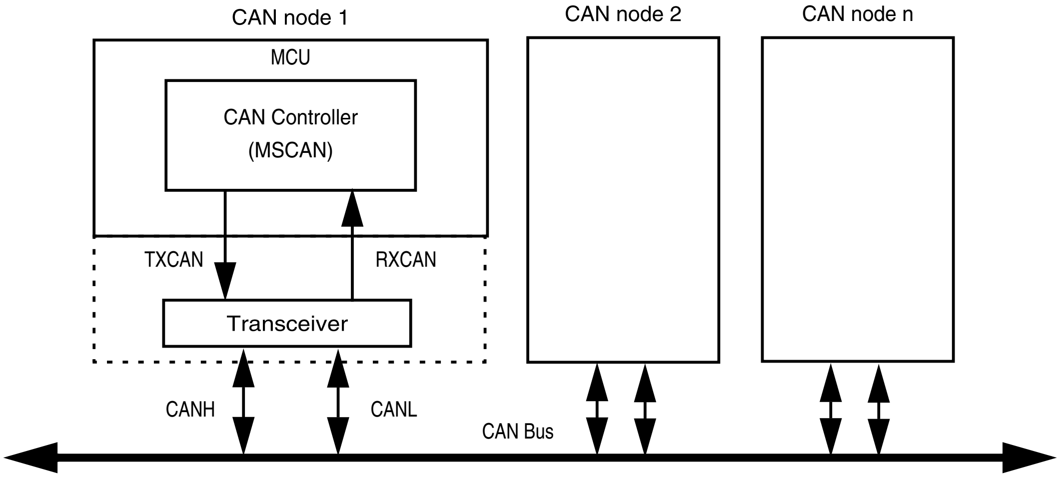


Figure 12-2. CAN System

12.3 Memory Map and Register Definition

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

12.3.1 Module Memory Map

Figure 12-3 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 12-3. 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 12.3.3, “Programmer’s Model of Message Storage”							
	W								
0x0030–0x003F CANTXFG	R	See Section 12.3.3, “Programmer’s Model of Message Storage”							
	W								

▨ = Unimplemented or Reserved

Figure 12-3. MSCAN Register Summary (continued)

12.3.2 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.

12.3.2.1 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 12-4. 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 12-2. 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 12.3.3, "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 12.4.5.5, "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 12-2. 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 12.4.5.5, “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 12.3.2.2, “MSCAN Control Register 1 (CANCTL1)”). SLPRQ cannot be set while the WUPIF flag is set (see Section 12.3.2.5, “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 12.4.4.5, “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 12.3.2.2, “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 12.4.5.2, “Operation in Wait Mode” and Section 12.4.5.3, “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 12.3.2.6, “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.

12.3.2.2 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 12-5. 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 12-3. 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 12.4.3.2, “Clock System,” and Section Figure 12-43., “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 12.4.4.4, “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 12.5.2, “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 12.4.5.5, “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 12-3. 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 12.4.5.5, “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 12.4.4.5, “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

12.3.2.3 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 12-6. MSCAN Bus Timing Register 0 (CANBTR0)

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-4. 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 12-5).
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 12-6).

Table 12-5. 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 12-6. 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

12.3.2.4 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 12-7. MSCAN Bus Timing Register 1 (CANBTR1)

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-7. 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 12-44). Time segment 2 (TSEG2) values are programmable as shown in Table 12-8.
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 12-44). Time segment 1 (TSEG1) values are programmable as shown in Table 12-9.

1. In this case, PHASE_SEG1 must be at least 2 time quanta (Tq).

Table 12-8. 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 12-36 for valid settings.

Table 12-9. 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 12-36 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 12-8 and Table 12-9).

Eqn. 12-1

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{\text{CANCLK}}} \bullet (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

12.3.2.5 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 12-8. 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 12-10. 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 12.4.5.5, “MSCAN Sleep Mode,”) and WUPE = 1 in CANTCTL0 (see Section 12.3.2.1, “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 12.3.2.6, “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 ≤ receive error counter < 96 01 RxWRN: 96 ≤ receive error counter < 128 10 RxERR: 128 ≤ receive error counter *11 Bus-off*[1]*: 256 ≤ transmit error counter*

---

[1] The RSTAT[1:0], TSTAT[1:0] bits are not affected by initialization mode.

Table 12-10. CANRFLG Register Field Descriptions (continued)

Field	Description
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 ≤ transmit error counter < 96 01 TxWRN: 96 ≤ transmit error counter < 128 10 TxERR: 128 ≤ transmit error counter < 256 11 Bus-Off: 256 ≤ transmit error counter
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.

12.3.2.6 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 12-9. 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 12-11. 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 12.3.2.1, “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 12.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)”).

12.3.2.7 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 12-10. 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 12-12. 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 12.3.2.9, “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 12.3.2.10, “MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)”). When a TXEx flag is set, the corresponding ABTRQx bit is cleared (see Section 12.3.2.9, “MSCAN Transmitter Message Abort Request Register (CANTARQ)”). When listen-mode is active (see Section 12.3.2.2, “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)

12.3.2.8 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 12-11. 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 12-13. 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.

12.3.2.9 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 12-12. 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 12-14. 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 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and abort acknowledge flags (ABTAK, see Section 12.3.2.10, “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

12.3.2.10 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 12-13. 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 12-15. 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.

12.3.2.11 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 12-14. 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 12-16. 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 12.3.2.7, “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.

12.3.2.12 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 12-15. 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 12-17. 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 12.4.3, “Identifier Acceptance Filter”). Table 12-18 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 12.4.3, “Identifier Acceptance Filter”). Table 12-19 summarizes the different settings.

Table 12-18. 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 12-19. 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.

12.3.2.13 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 12-16. 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.

12.3.2.14 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 12-17. MSCAN Miscellaneous Register (CANMISC)

1. Read: Anytime
   Write: Anytime; write of ‘1’ clears flag; write of ‘0’ ignored

Table 12-20. 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 12.5.2, “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

12.3.2.15 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 12-18. 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.

12.3.2.16 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 12-19. 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.

12.3.2.17 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 12.3.3.1, “Identifier Registers (IDR0–IDR3)”) of incoming messages in a bit by bit manner (see Section 12.4.3, “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 12-20. MSCAN Identifier Acceptance Registers (First Bank) — CANIDAR0–CANIDAR3

1. Read: Anytime
Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-21. 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 12-21. MSCAN Identifier Acceptance Registers (Second Bank) — CANIDAR4–CANIDAR7

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-22. 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.

12.3.2.18 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 12-22. MSCAN Identifier Mask Registers (First Bank) — CANIDMR0–CANIDMR3

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-23. 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 12-23. MSCAN Identifier Mask Registers (Second Bank) — CANIDMR4–CANIDMR7

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 12-24. 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

12.3.3 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 12.3.2.1, "MSCAN Control Register 0 (CANCTL0)").

The time stamp register is written by the MSCAN. The CPU can only read these registers.

Table 12-25. 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 12-24 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 12-25.

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 12-24. 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 12-24. 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 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).
- For receive buffers, only when RXF flag is set (see Section 12.3.2.5, “MSCAN Receiver Flag Register (CANRFLG)”).

Write:

- For transmit buffers, anytime when TXEx flag is set (see Section 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).
- Unimplemented for receive buffers.

Reset: Undefined because of RAM-based implementation

Figure 12-25. 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’

12.3.3.1 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.

12.3.3.1.1 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 12-26. Identifier Register 0 (IDR0) — Extended Identifier Mapping

Table 12-26. 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 12-27. Identifier Register 1 (IDR1) — Extended Identifier Mapping

Table 12-27. 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 12-28. Identifier Register 2 (IDR2) — Extended Identifier Mapping

Table 12-28. 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 12-29. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 12-29. 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

12.3.3.1.2 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 12-30. Identifier Register 0 — Standard Mapping

Table 12-30. 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 12-31.

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 12-31. Identifier Register 1 — Standard Mapping

Table 12-31. 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 12-30.
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 12-32. 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 12-33. Identifier Register 3 — Standard Mapping

12.3.3.2 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 12-34. Data Segment Registers (DSR0–DSR7) — Extended Identifier Mapping

Table 12-32. DSR0–DSR7 Register Field Descriptions

Field	Description
7-0 DB[7:0]	**Data bits 7-0**

12.3.3.3 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 12-35. Data Length Register (DLR) — Extended Identifier Mapping

Table 12-33. 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 12-34 shows the effect of setting the DLC bits.

Table 12-34. 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

12.3.3.4 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 12-36. Transmit Buffer Priority Register (TBPR)

1. Read: Anytime when TXEx flag is set (see Section 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”)
   Write: Anytime when TXEx flag is set (see Section 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”)

12.3.3.5 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 12.3.2.1, “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 12-37. Time Stamp Register — High Byte (TSRH)

1. Read: For transmit buffers: Anytime when TXEx flag is set (see Section 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “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 12-38. Time Stamp Register — Low Byte (TSRL)

1. Read: or transmit buffers: Anytime when TXEx flag is set (see Section 12.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”). For receive buffers: Anytime when RXF is set.
   Write: Unimplemented

12.4 Functional Description

12.4.1 General

This section provides a complete functional description of the MSCAN.

12.4.2 Message Storage

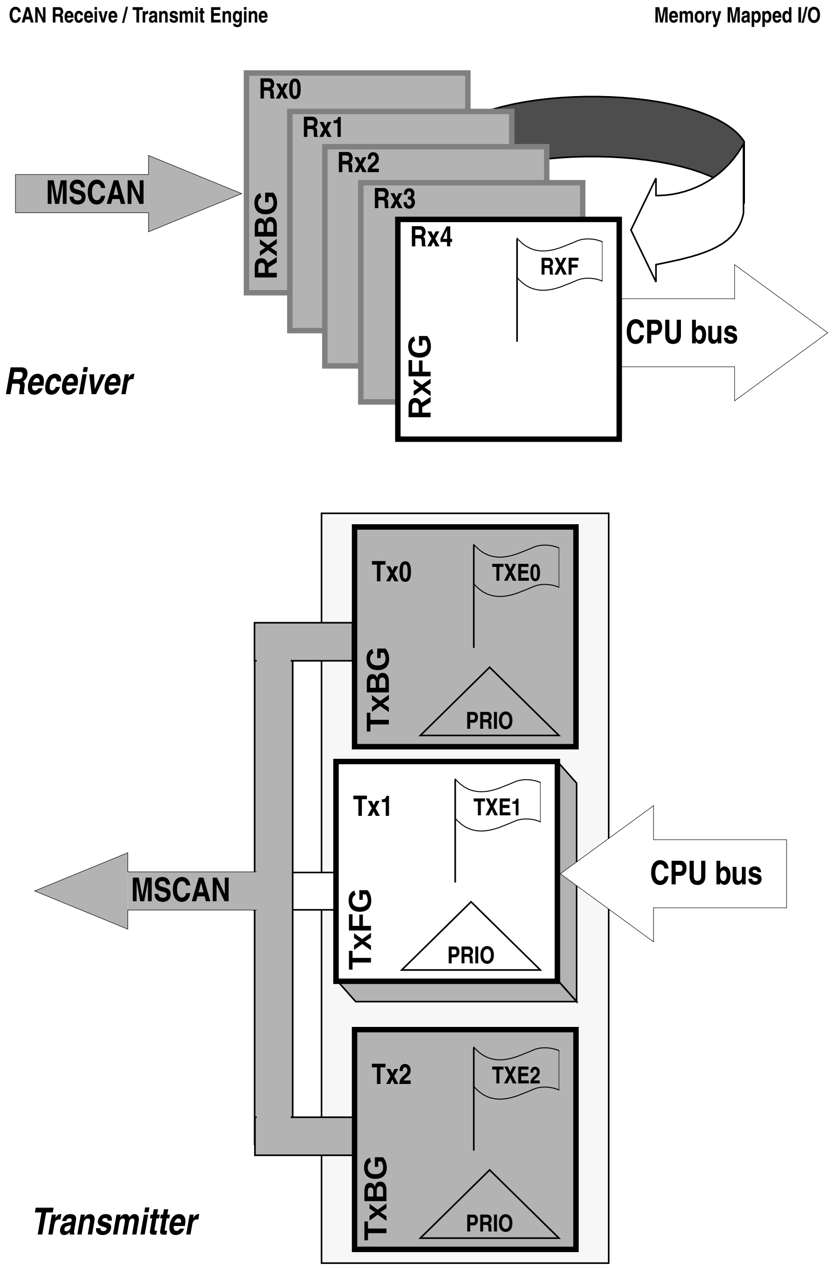


Figure 12-39. User Model for Message Buffer Organization

The MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

12.4.2.1 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 12.4.2.2, "Transmit Structures."

12.4.2.2 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 12-39.

All three buffers have a 13-byte data structure similar to the outline of the receive buffers (see Section 12.3.3, "Programmer's Model of Message Storage"). An additional Transmit Buffer Priority Register (TBPR) contains an 8-bit local priority field (PRIO) (see Section 12.3.3.4, "Transmit Buffer Priority Register (TBPR)"). The remaining two bytes are used for time stamping of a message, if required (see Section 12.3.3.5, "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 12.3.2.7, "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 12.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)"). This makes the respective buffer accessible within the CANTXFG address space (see Section 12.3.3, "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 12.4.7.2, “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 12.3.2.9, “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).

12.4.2.3 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 12-39). The background receive buffer (RxBG) is exclusively associated with the MSCAN, but the foreground receive buffer (RxFG) is addressable by the CPU (see Figure 12-39). 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 12.3.3, “Programmer’s Model of Message Storage”).

The receiver full flag (RXF) (see Section 12.3.2.5, “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 12.4.3, “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 12.4.7.3, "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 12.3.2.2, "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 12.4.7.5, "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.

12.4.3 Identifier Acceptance Filter

The MSCAN identifier acceptance registers (see Section 12.3.2.12, "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 12.3.2.18, "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 12.3.2.12, "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 eight identifier acceptance filters.

---

[1] The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.

Figure 12-40 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 12-41 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 12-42 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.

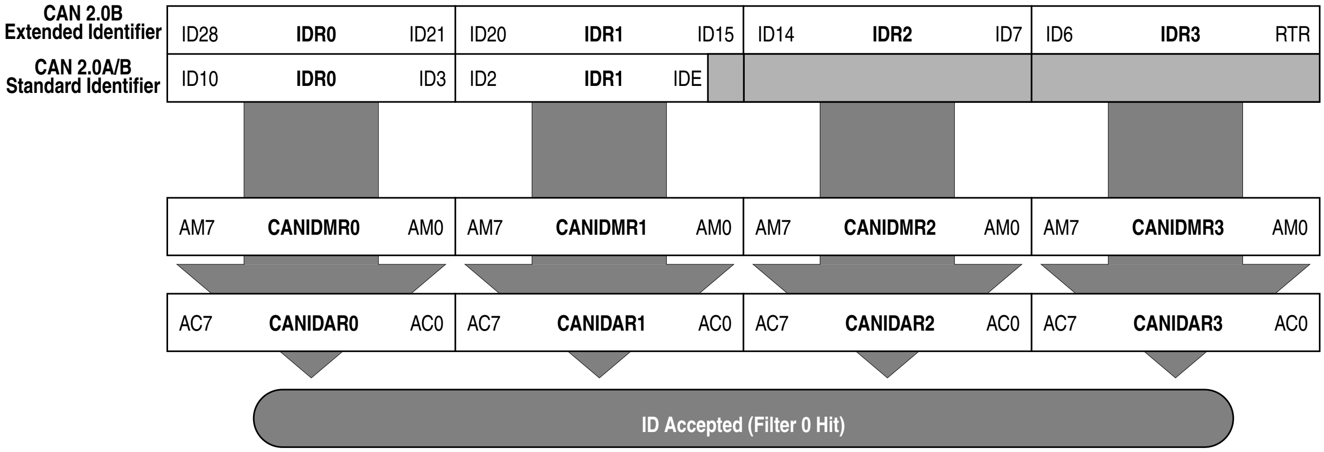


Figure 12-40. 32-bit Maskable Identifier Acceptance Filter

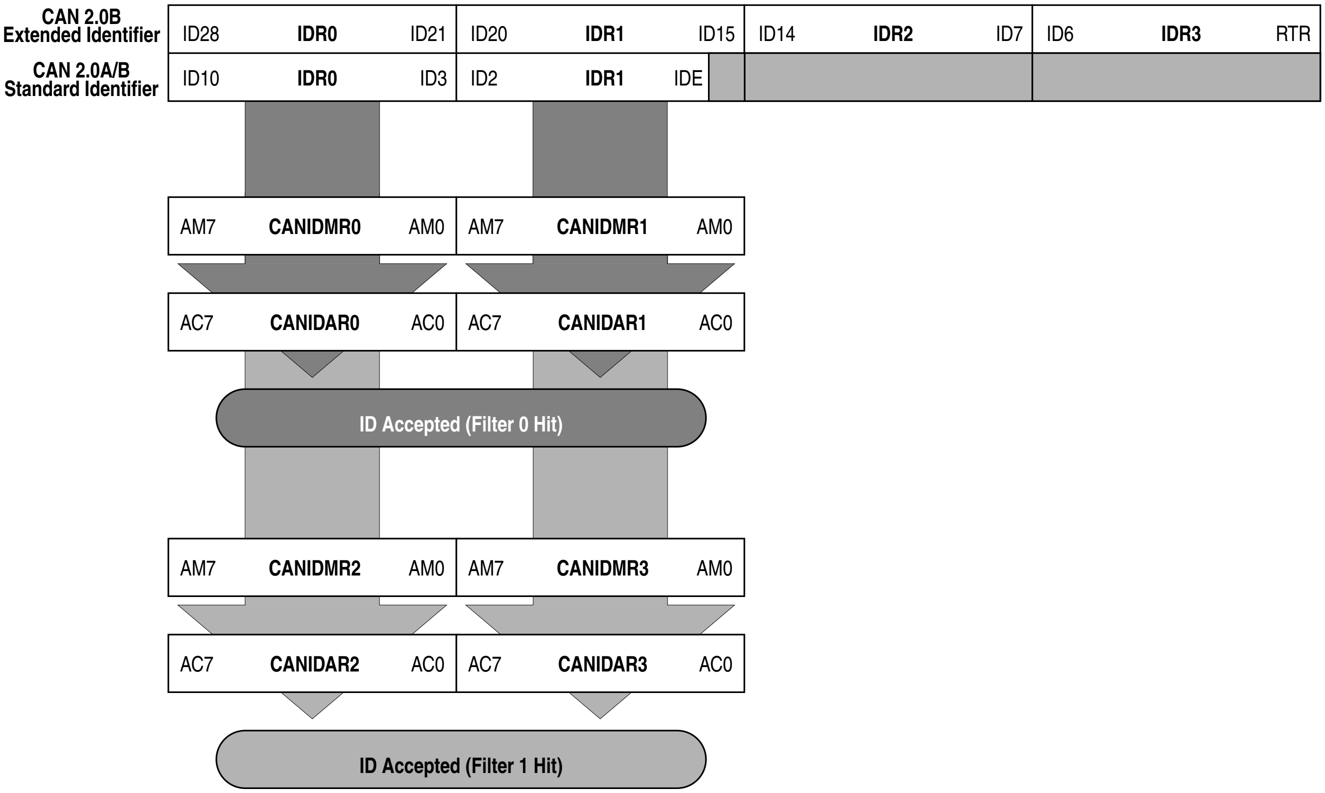


Figure 12-41. 16-bit Maskable Identifier Acceptance Filters

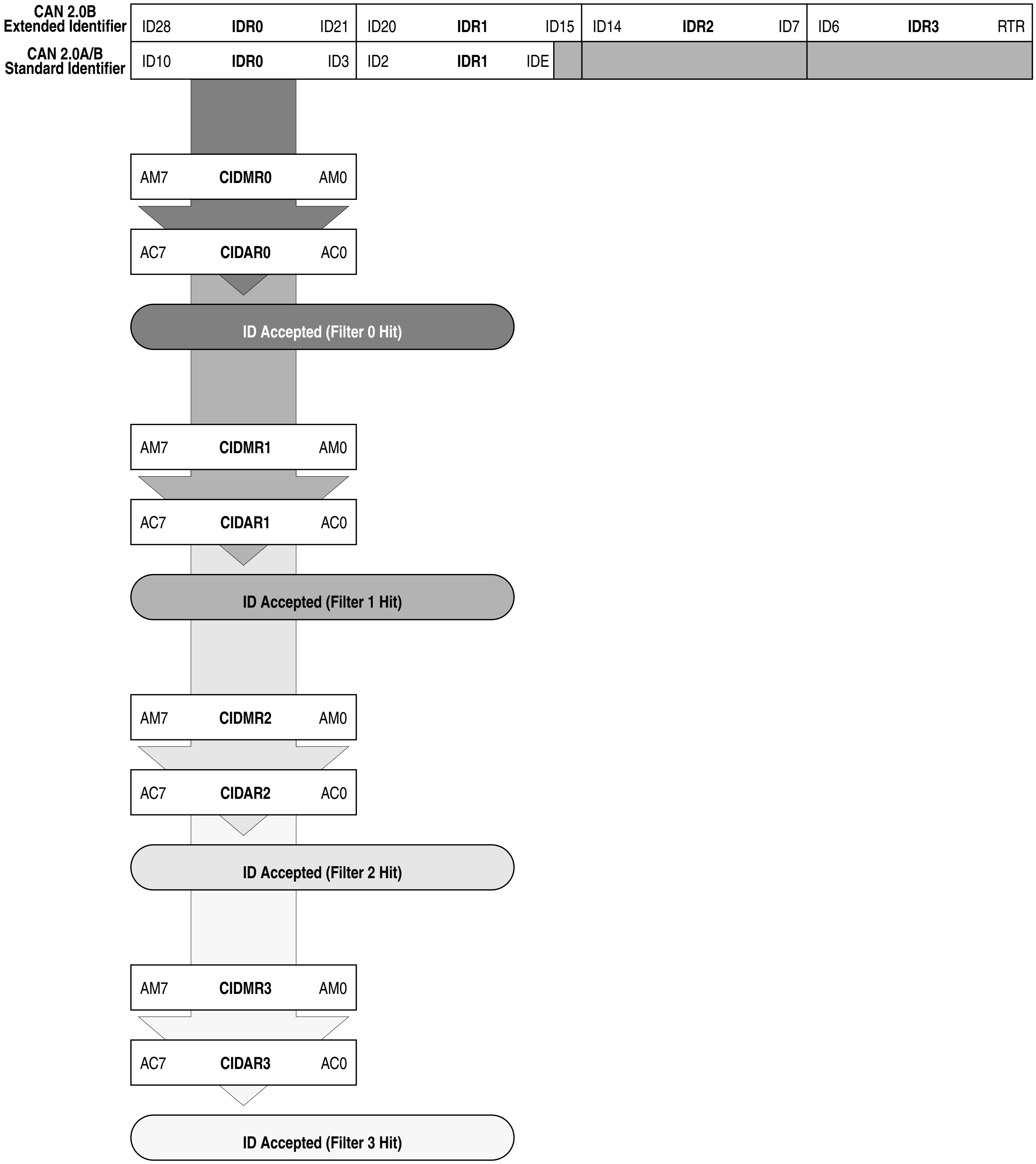


Figure 12-42. 8-bit Maskable Identifier Acceptance Filters

12.4.3.1 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 12.3.2.1, "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 12.4.5.6, "MSCAN Power Down Mode," and Section 12.4.4.5, "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.

12.4.3.2 Clock System

Figure 12-43 shows the structure of the MSCAN clock generation circuitry.

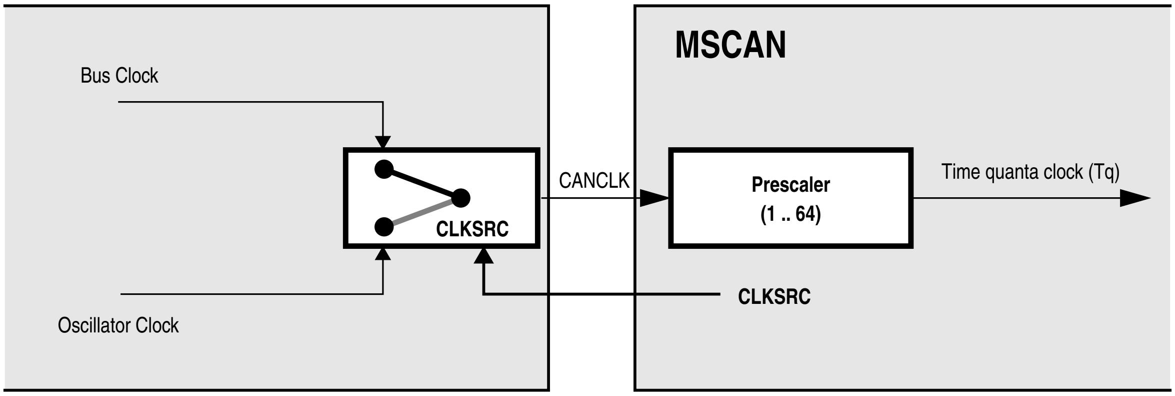


Figure 12-43. MSCAN Clocking Scheme

The clock source bit (CLKSRC) in the CANCTL1 register (12.3.2.2/12-411) 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. 12-2

$$\mathrm{Tq} = \frac{f_{\mathrm{CANCLK}}}{(\mathrm{Prescaler\ value})}$$

A bit time is subdivided into three segments as described in the Bosch CAN 2.0A/B specification. (see Figure 12-44):

- 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. 12-3

$$\mathrm{Bit\ Rate} = \frac{f_{\mathrm{Tq}}}{(\mathrm{number\ of\ Time\ Quanta})}$$

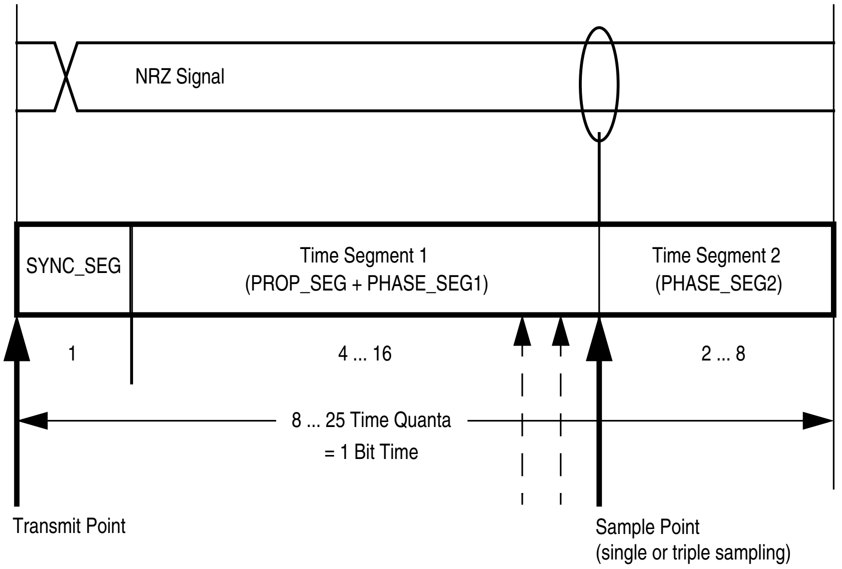


Figure 12-44. Segments within the Bit Time

Table 12-35. 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 12.3.2.3, "MSCAN Bus Timing Register 0 (CANBTR0)" and Section 12.3.2.4, "MSCAN Bus Timing Register 1 (CANBTR1)").

Table 12-36 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 12-36. 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

12.4.4 Modes of Operation

12.4.4.1 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.

12.4.4.2 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.

12.4.4.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like in normal system operating modes as described within this specification.

12.4.4.4 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.

12.4.4.5 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 12.3.2.1, "MSCAN Control Register 0 (CANCTL0)," for a detailed description of the initialization mode.

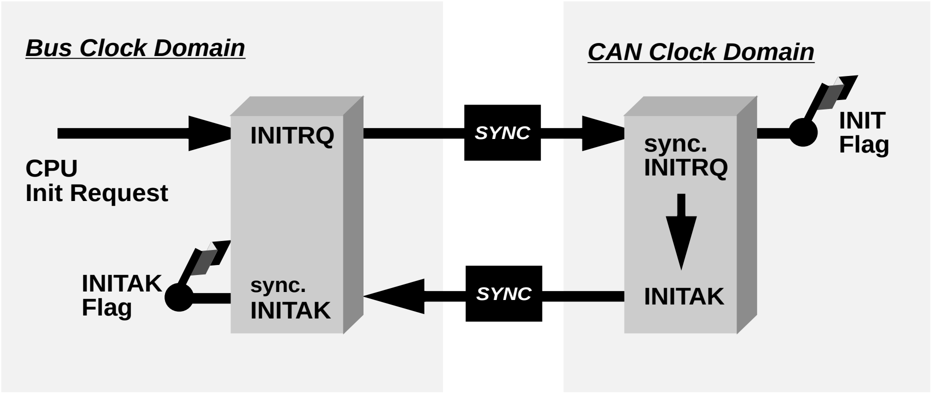


Figure 12-45. Initialization Request/Acknowledge Cycle

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 12-45).

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.

12.4.5 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 12-37 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 12-37. 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.

12.4.5.1 Operation in Run Mode

As shown in Table 12-37, only MSCAN sleep mode is available as low power option when the CPU is in run mode.

12.4.5.2 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).

12.4.5.3 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 12-37).

12.4.5.4 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 12.4.4.5, "MSCAN Initialization Mode".

12.4.5.5 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.

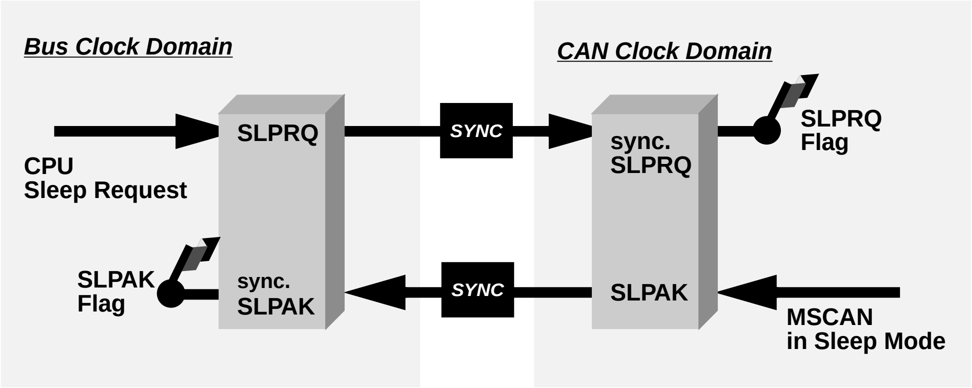


Figure 12-46. Sleep Request / Acknowledge Cycle

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 12-46). 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.

12.4.5.6 MSCAN Power Down Mode

The MSCAN is in power down mode (Table 12-37) 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.

12.4.5.7 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.

12.4.5.8 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 12.3.2.2, "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.

12.4.6 Reset Initialization

The reset state of each individual bit is listed in Section 12.3.2, "Register Descriptions," which details all the registers and their bit-fields.

12.4.7 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.

12.4.7.1 Description of Interrupt Operation

The MSCAN supports four interrupt vectors (see Table 12-38), any of which can be individually masked (for details see Section 12.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)" to Section 12.3.2.8, "MSCAN Transmitter Interrupt Enable Register (CANTIER)").

Refer to the device overview section to determine the dedicated interrupt vector addresses.

Table 12-38. 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])

12.4.7.2 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.

12.4.7.3 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.

12.4.7.4 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).

12.4.7.5 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 12.4.2.3, "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 12.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)" and Section 12.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)").

12.4.7.6 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.

12.5 Initialization/Application Information

12.5.1 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

12.5.2 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 13
Programmable Trigger Unit (PTUV2)

Table 13-1. Revision History Table

Rev. No. (Item No.)	Data	Sections Affected	Substantial Change(s)
01.00	21 Oct. 2011	all	Initial Version
02.00	22. Mar. 2012	13.3.2.1, 13.3.2.7, 13.3.2.10, 13.3.2.14 - 13.3.2.17	- removed PTUWP bit (now: PTUPTR is write protected if both TGs are disabled, TGxLxIDX is write protected if the associated TG is disabled) - TGxLIST bits are writeable if associated TG is disabled - PTULDOK bit is writable if both TGs are disabled - TGxLIST swap at every reload with LDOK set

Table 13-2. Terminology

Term	Meaning
TG	Trigger Generator
EOL	End of trigger list

13.1 Introduction

In PWM driven systems it is important to schedule the acquisition of the state variables with respect to PWM cycle.

The Programmable Trigger Unit (PTU) is intended to completely avoid CPU involvement in the time acquisitions of state variables during the control cycle that can be half, full, multiple PWM cycles.

All acquisition time values are stored inside the global memory map, basically inside the system memory; see the MMC section for the supported memory area. In such cases the pre-setting of the acquisition times needs to be completed during the previous control cycle to where the actual acquisitions are to be made.

13.1.1 Features

The PTU module includes these distinctive features:

- One 16 bit counter as time base for all trigger events
- Two independent trigger generators (TG0 and TG1)
- Up to 32 trigger events per trigger generator
- Global Load OK support, to guarantee coherent update of all control loop modules

- Trigger values stored inside the global memory map, basically inside system memory
- Software generated reload event and Trigger event generation for debugging

13.1.2 Modes of Operation

The PTU module behaves as follows in the system power modes:

1. Run mode

   All PTU features are available.
2. Wait mode
   All PTU features are available.
3. Freeze Mode

   Depends on the PTUFRZ register bit setting the internal counter is stopped and no trigger events will be generated.
4. Stop mode

   The PTU is disabled and the internal counter is stopped; no trigger events will be generated. The content of the configuration register is unchanged.

13.1.3 Block Diagram

Figure 13-1 shows a block diagram of the PTU module.

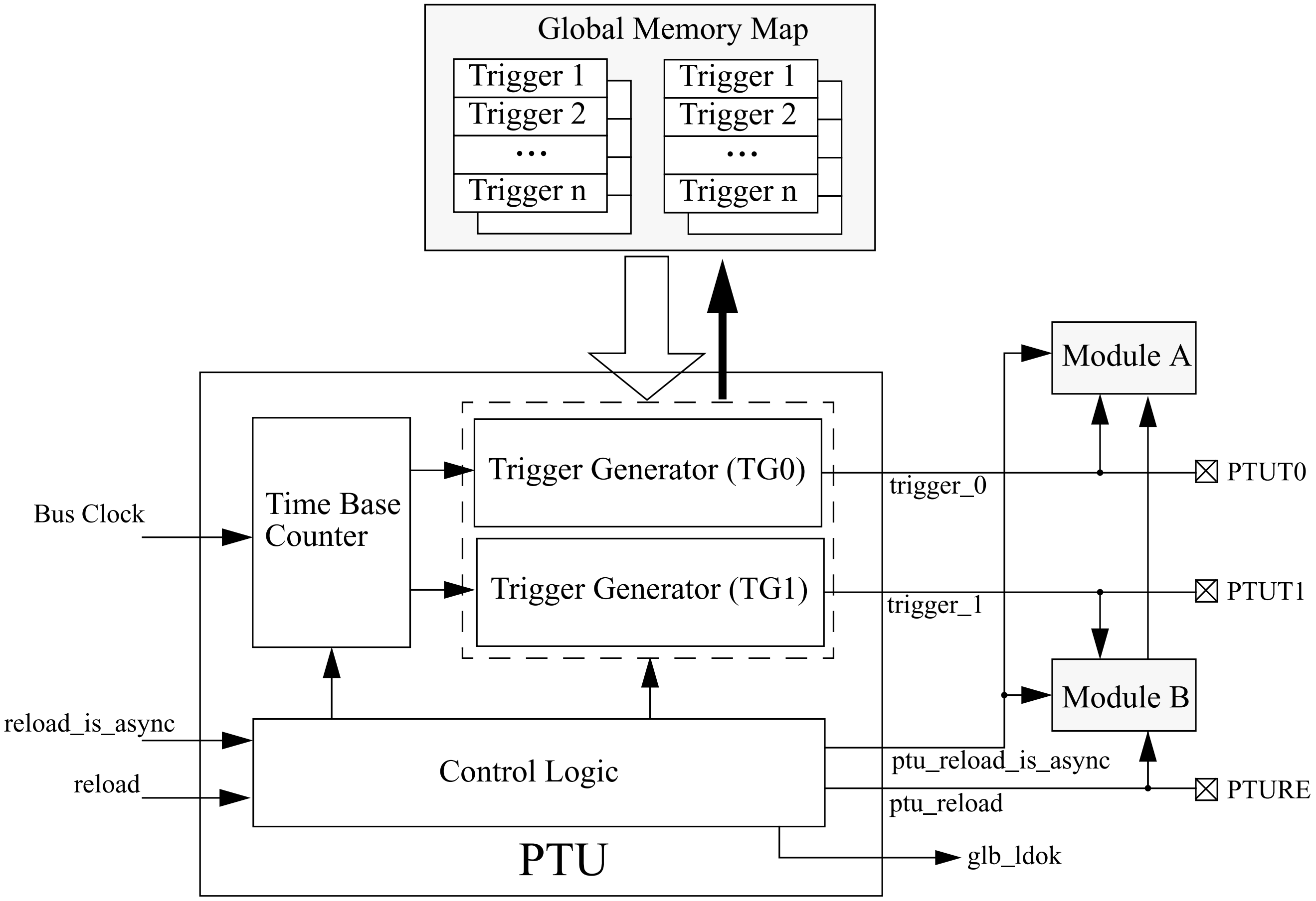


Figure 13-1. PTU Block Diagram

13.2 External Signal Description

This section lists the name and description of all external ports.

13.2.1 PTUT0 — PTU Trigger 0

If enabled (PTUT0PE is set) this pin shows the internal trigger_0 event.

13.2.2 PTUT1 — PTU Trigger 1

If enabled (PTUT1PE is set) this pin shows the internal trigger_1 event.

13.2.3 PTURE — PTUE Reload Event

If enabled (PTUREPE is set) this pin shows the internal reload event.

13.3 Memory Map and Register Definition

This section provides the detailed information of all registers for the PTU module.

13.3.1 Register Summary

Figure 13-2 shows the summary of all implemented registers inside the PTU 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 PTUE	R	0	PTUFRZ	0	0	0	0	TG1EN	TG0EN
	W	▨		▨	▨	▨	▨		
0x0001 PTUC	R	0	0	0	0	0	0	0	PTULDOK
	W	▨	▨	▨	▨	▨	▨	▨	
0x0002 PTUIEH	R	0	0	0	0	0	0	0	PTUROIE
	W	▨	▨	▨	▨	▨	▨	▨	
0x0003 PTUIEL	R	TG1AEIE	TG1REIE	TG1TEIE	TG1DIE	TG0AEIE	TG0REIE	TG0TEIE	TG0DIE
	W								
0x0004 PTUIFH	R	0	0	0	0	0	0	PTUDEEF	PTUROIF
	W	▨	▨	▨	▨	▨	▨		
0x0005 PTUIFL	R	TG1AEIF	TG1REIF	TG1TEIF	TG1DIF	TG0AEIF	TG0REIF	TG0TEIF	TG0DIF
	W								
0x0006 TG0LIST	R	0	0	0	0	0	0	0	TG0LIST
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0007 TG0TNUM	R	0	0	0	TG0TNUM[4:0]				
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0008 TG0TVH	R	TG0TV[15:8]							
	W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented

Figure 13-2. PTU Register Summary

Address Offset **Register Name**		**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
0x0009 TG0TVL	R	TG0TV[7:0]							
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000A TG1LIST	R	0	0	0	0	0	0	0	TG1LIST
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000B TG1TNUM	R	0	0	0	TG1TNUM4:0]				
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000C TG1TVH	R	TG1TV[15:8]							
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000D TG1TVL	R	TG1TV[7:0]							
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000E PTUCNTH	R	PTUCNT[15:8]							
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000F PTUCNTL	R	PTUCNT[7:0]							
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0010 Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0011 PTUPTRH	R	PTUPTR[23:16]							
	W								
0x0012 PTUPTRM	R	PTUPTR[15:8]							
	W								
0x0013 PTUPTRL	R	PTUPTR[7:1]							0
	W								▨
0x0014 TG0L0IDX	R	0	TG0L10DX[6:0]						
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0015 TG0L1IDX	R	0	TG0L1IDX[6:0]						
	W	▨							
0x0016 TG1L0IDX	R	0	TG1L0IDX[6:0]						
	W	▨							
0x0017 TG1L1IDX	R	0	TG1L1IDX[6:0]						
	W	▨							

▨ = Unimplemented

Figure 13-2. PTU Register Summary

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0018 - 0x001E Reserved	R	0	0	0	0	0	0	0	0
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x001F PTUDEBUG	R	0	PTUREPE	PTUT1PE	PTUT0PE	0	0	0	0
	W	▨				▨	PTUFRE	TG1FTE	TG0FTE

▨ = Unimplemented

Figure 13-2. PTU Register Summary

13.3.2 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.

13.3.2.1 PTU Module Enable Register (PTUE)

Module Base + 0x0000 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	PTUFRZ	0	0	0	0	TG1EN	TG0EN
W	▨		▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-3. PTU Module Enable Register (PTUE)

1. Read: Anytime
   Write: Anytime

Table 13-3. PTUE Register Field Description

Field	Description
6 PTUFRZ	**PTU Stop in Freeze Mode** — In freeze mode, there is an option to disable the input clock to the PTU time base counter. If this bit is set, whenever the MCU is in freeze mode, the input clock to the time base counter is disabled. In this way, the counters can be stopped while in freeze mode so that once normal program flow is continued, the counter is re-enabled. 0 Allow time base counter to continue while in freeze mode 1 Disable time base counter clock whenever the part is in freeze mode
1 TG1EN	**Trigger Generator 1 Enable** — This bit enables trigger generator 1. 0 Trigger generator 1 is disabled 1 Trigger generator 1 is enabled
0 TG0EN	**Trigger Generator 0 Enable** — This bit enables trigger generator 0. 0 Trigger generator 0 is disabled 1 Trigger generator 0 is enabled

13.3.2.2 PTU Module Control Register (PTUC)

Module Base + 0x0001 Access: User read/write(1)

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

▨ = Unimplemented

Figure 13-4. PTU Module Control Register (PTUC)

1. Read: Anytime
   Write: write 1 anytime, write 0 if TG0EN and TG1EN is cleared

Table 13-4. PTUC Register Field Descriptions

Field	Description
0 PTULDOK	**Load Okay** — When this bit is set by the software, this allows the trigger generator to switch to the alternative list and load the trigger time values at the next reload event from the new list. If the reload event occurs when the PTULDOK bit is not set then the trigger generator generates a reload overrun event and uses the previously used list. At the next reload event this bit is cleared by control logic. Write 0 is only possible if TG0EN and TG1EN is cleared. The PTULDOK can be used by other module as global load OK (glb_ldok).

13.3.2.3 PTU Interrupt Enable Register High (PTUIEH)

Module Base + 0x0002 Access: User read/write[1]

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

▨ = Unimplemented

Figure 13-5. PTU Interrupt Enable Register High (PTUIEH)

1. Read: Anytime
   Write: Anytime

Table 13-5. PTUIEH Register Field Descriptions

Field	Description
0 PTUROIE	**PTU Reload Overrun Interrupt Enable** — Enables PTU reload overrun interrupt. 0 No interrupt will be requested whenever PTUROIF is set 1 Interrupt will be requested whenever PTUROIF is set

13.3.2.4 PTU Interrupt Enable Register Low (PTUIEL)

Module Base + 0x0003 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	TG1AEIE	TG1REIE	TG1TEIE	TG1DIE	TG0AEIE	TG0REIE	TG0TEIE	TG0DIE
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-6. PTU Interrupt Enable Register Low (PTUIEL)

1. Read: Anytime
   Write: Anytime

Table 13-6. PTUIEL Register Field Descriptions

Field	Description
7 TG1AEIE	**Trigger Generator 1 Memory Access Error Interrupt Enable** — Enables trigger generator memory access error interrupt. 0 No interrupt will be requested whenever TG1AEIF is set 1 Interrupt will be requested whenever TG1AEIF is set
6 TG1REIE	**Trigger Generator 1 Reload Error Interrupt Enable** — Enables trigger generator reload error interrupt. 0 No interrupt will be requested whenever TG1REIF is set 1 Interrupt will be requested whenever TG1REIF is set

Table 13-6. PTUIEL Register Field Descriptions

Field	Description
5 TG1TEIE	**Trigger Generator 1 Timing Error Interrupt Enable** — Enables trigger generator timing error interrupt. 0 No interrupt will be requested whenever TG1TEIF is set 1 Interrupt will be requested whenever TG1TEIF is set
4 TG1DIE	**Trigger Generator 1 Done Interrupt Enable** — Enables trigger generator done interrupt. 0 No interrupt will be requested whenever TG1DIF is set 1 Interrupt will be requested whenever TG1DIF is set
3 TG0AEIE	**Trigger Generator 0 Memory Access Error Interrupt Enable** — Enables trigger generator memory access error interrupt. 0 No interrupt will be requested whenever TG0AEIF is set 1 Interrupt will be requested whenever TG0AEIF is set
2 TG0REIE	**Trigger Generator 0 Reload Error Interrupt Enable** — Enables trigger generator reload error interrupt. 0 No interrupt will be requested whenever TG0REIF is set 1 Interrupt will be requested whenever TG0REIF is set
1 TG0TEIE	**Trigger Generator 0 Timing Error Interrupt Enable** — Enables trigger generator timing error interrupt. 0 No interrupt will be requested whenever TG0TEIF is set 1 Interrupt will be requested whenever TG0TEIF is set
0 TG0DIE	**Trigger Generator 0 Done Interrupt Enable** — Enables trigger generator done interrupt. 0 No interrupt will be requested whenever TG0DIF is set 1 Interrupt will be requested whenever TG0DIF is set

13.3.2.5 PTU Interrupt Flag Register High (PTUIFH)

Module Base + 0x0004 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PTUDEEF	PTUROIF
W	▨	▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-7. PTU Interrupt Flag Register High (PTUIFH)

1. Read: Anytime
Write: Anytime, write 1 to clear

Table 13-7. PTUIFH Register Field Descriptions

Field	Description
1 PTUDEEF	**PTU Double bit ECC Error Flag** — This bit is set if the read data from the memory contains double bit ECC errors. While this bit is set the trigger generation of both trigger generators stops. 0 No double bit ECC error occurs 1 Double bit ECC error occurs
0 PTUROIF	**PTU Reload Overrun Interrupt Flag** — If reload event occurs when the PTULDOK bit is not set then this bit will be set. This bit is not set if the reload event was forced by an asynchronous commutation event. 0 No reload overrun occurs 1 Reload overrun occurs

13.3.2.6 PTU Interrupt Flag Register Low (PTUIFL)

Module Base + 0x0005 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	TG1AEIF	TG1REIF	TG1EIF	TG1DIF	TG0AEIF	TG0REIF	TG0EIF	TG0DIF
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-8. PTU Interrupt Flag Register Low (PTUIFL)

1. Read: Anytime
Write: Anytime, write 1 to clear

Table 13-8. PTUIFL Register Field Descriptions

Field	Description
7 TG1AEIF	**Trigger Generator 1 Memory Access Error Interrupt Flag** — This bit is set if trigger generator 1 uses a read address outside the memory address range, see the MMC section for the supported memory area. 0 No trigger generator 1 memory access error occurs 1 Trigger generator 1 memory access error occurs
6 TG1REIF	**Trigger Generator 1 Reload Error Interrupt Flag** — This bit is set if a new reload event occurs when the trigger generator has neither reached the end of list symbol nor the maximum possible triggers. This bit is not set if the reload event was forced by an asynchronous commutation event. 0 No trigger generator 1 reload error occurs 1 Trigger generator 1 reload error occurs
5 TG1TEIF	**Trigger Generator 1 Timing Error Interrupt Flag** — This bit is set if the trigger generator receives a time value which is below the current counter value. 0 No trigger generator 1 error occurs 1 Trigger generator 1 error occurs
4 TG1DIF	**Trigger Generator 1 Done Interrupt Flag** —This bit is set if the trigger generator receives the end of list symbol or the maximum number of generated trigger events was reached. 0 Trigger generator 1 is running 1 Trigger generator 1 is done
3 TG0AEIF	**Trigger Generator 0 Memory Access Error Interrupt Flag** — This bit is set if trigger generator 0 uses a read address outside the memory address range, see the MMC section for the supported memory area. 0 No trigger generator 0 memory access error occurred 1 Trigger generator 0 memory access error occurred
2 TG0REIF	**Trigger Generator 0 Reload Error Interrupt Flag** — This bit is set if a new reload event occurs when the trigger generator has neither reached the end of list symbol nor the maximum possible triggers. This bit is not set if the reload event was forced by an asynchronous commutation event. 0 No trigger generator 0 reload error occurs 1 Trigger generator 0 reload error occurs
1 TG0TEIF	**Trigger Generator 0 Timing Error Interrupt Flag** — This bit is set if the trigger generator receives a time value which is below the current counter value. 0 No trigger generator 0 error occurs 1 Trigger generator 0 error occurs
0 TG0DIF	**Trigger Generator 0 Done Interrupt Flag** —This bit is set if the trigger generator receives the end of list symbol or the maximum number of generated trigger events was reached. 0 Trigger generator 0 is running 1 Trigger generator 0 is done

13.3.2.7 Trigger Generator 0 List Register (TG0LIST)

Module Base + 0x0006 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	TG0LIST
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-9. Trigger Generator 0 List Register (TG0LIST)

1. Read: Anytime
   Write: Anytime, if TG0EN bit is cleared

Table 13-9. TG0LIST Register Field Descriptions

Field	Description
0 TG0LIST	**Trigger Generator 0 List** — This bit shows the number of the current used list. 0 Trigger generator 0 is using list 0 1 Trigger generator 0 is using list 1

13.3.2.8 Trigger Generator 0 Trigger Number Register (TG0TNUM)

Module Base + 0x0007 Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	TG0TNUM[4:0]				
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-10. Trigger Generator 0 Trigger Number Register (TG0TNUM)

1. Read: Anytime
   Write: Never

Table 13-10. TG0TNUM Register Field Descriptions

Field	Description
4:0 TG0TNUM[4:0]	**Trigger Generator 0 Trigger Number** — This register shows the number of generated triggers since the last reload event. After the generation of 32 triggers this register shows zero. The next reload event clears this register. See also Figure 13-22.

13.3.2.9 Trigger Generator 0 Trigger Value (TG0TVH, TG0TVL)

Module Base + 0x0008 Access: User read only(1)

	7	6	5	4	3	2	1	0
R	TG0TV[15:8]							
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

Module Base + 0x0009 Access: User read only

	7	6	5	4	3	2	1	0
R	TG0TV[7:0]							
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-11. Trigger Generator 0 Trigger Value Register (TG0TVH, TG0TVL)

1. Read: Anytime
   Write: Never

Table 13-11. TG0TV Register Field Descriptions

Field	Description
TG0TV[15:0]	**Trigger Generator 0 Trigger Value** — This register contains the trigger value to generate the next trigger. If the time base counter reach this value the next trigger event is generated. If the trigger generator reached the end of list (EOL) symbol then this value is visible inside this register. If the last generated trigger was trigger number 32 then the last used trigger value is visible inside this register. See also Figure 13-22.

13.3.2.10 Trigger Generator 1 List Register (TG1LIST)

Module Base + 0x000A Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	TG1LIST
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-12. Trigger Generator 1 List Register (TG1LIST)

1. Read: Anytime
   Write: Anytime, if TG1EN bit is cleared

Table 13-12. TG1LIST Register Field Descriptions

Field	Description
0 TG1LIST	**Trigger Generator 1 List** — This bit shows the number of the current used list. 0 Trigger generator 1 is using list 0 1 Trigger generator 1 is using list 1

13.3.2.11 Trigger Generator 1 Trigger Number Register (TG1TNUM)

Module Base + 0x000B — Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	TG1TNUM[4:0]				
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-13. Trigger Generator 1 Trigger Number Register (TG1TNUM)

1. Read: Anytime
   Write: Never

Table 13-13. TG1TNUM Register Field Descriptions

Field	Description
4:0 TG1TNUM[4:0]	**Trigger Generator 1 Trigger Number** — This register shows the number of generated triggers since the last reload event. After the generation of 32 triggers this register shows zero. The next reload event clears this register. See also Figure 13-22.

13.3.2.12 Trigger Generator 1 Trigger Value (TG1TVH, TG1TVL)

Module Base + 0x000C — Access: User read only[1]

	7	6	5	4	3	2	1	0
R	TG1TV[15:8]							
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

Module Base + 0x000D — Access: User read only

	7	6	5	4	3	2	1	0
R	TG1TV[7:0]							
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-14. Trigger Generator 1 Trigger Value Register (TG1TVH, TG1TVL)

1. Read: Anytime
   Write: Never

Table 13-14. TG1TV Register Field Descriptions

Field	Description
TG1TV[15:0]	**Trigger Generator 1 Next Trigger Value** — This register contains the currently used trigger value to generate the next trigger. If the time base counter reach this value the next trigger event is generated. If the trigger generator reached the end of list (EOL) symbol then this value is visible inside this register. If the last generated trigger was trigger number 32 then the last used trigger value is visible inside this register. See also Figure 13-22.

13.3.2.13 PTU Counter Register (PTUCNTH, PTUCNTL)

Module Base + 0x000E Access: User read only[1]

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

Module Base + 0x000F Access: User read only

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

▨ = Unimplemented

Figure 13-15. PTU Counter Register (PTUCNTH, PTUCNTL)

[1] Read: Anytime
Write: Never

Table 13-15. PTUCNT Register Field Descriptions

Field	Description
PTUCNT[15:0]	**PTU Time Base Counter value** — This register contains the current status of the internal time base counter. If both TG are done with the execution of the trigger list then the counter also stops. The counter is restarted by the next reload event.

13.3.2.14 PTU Pointer Register (PTUPTRH, PTUPTRM, PTUPTRL)

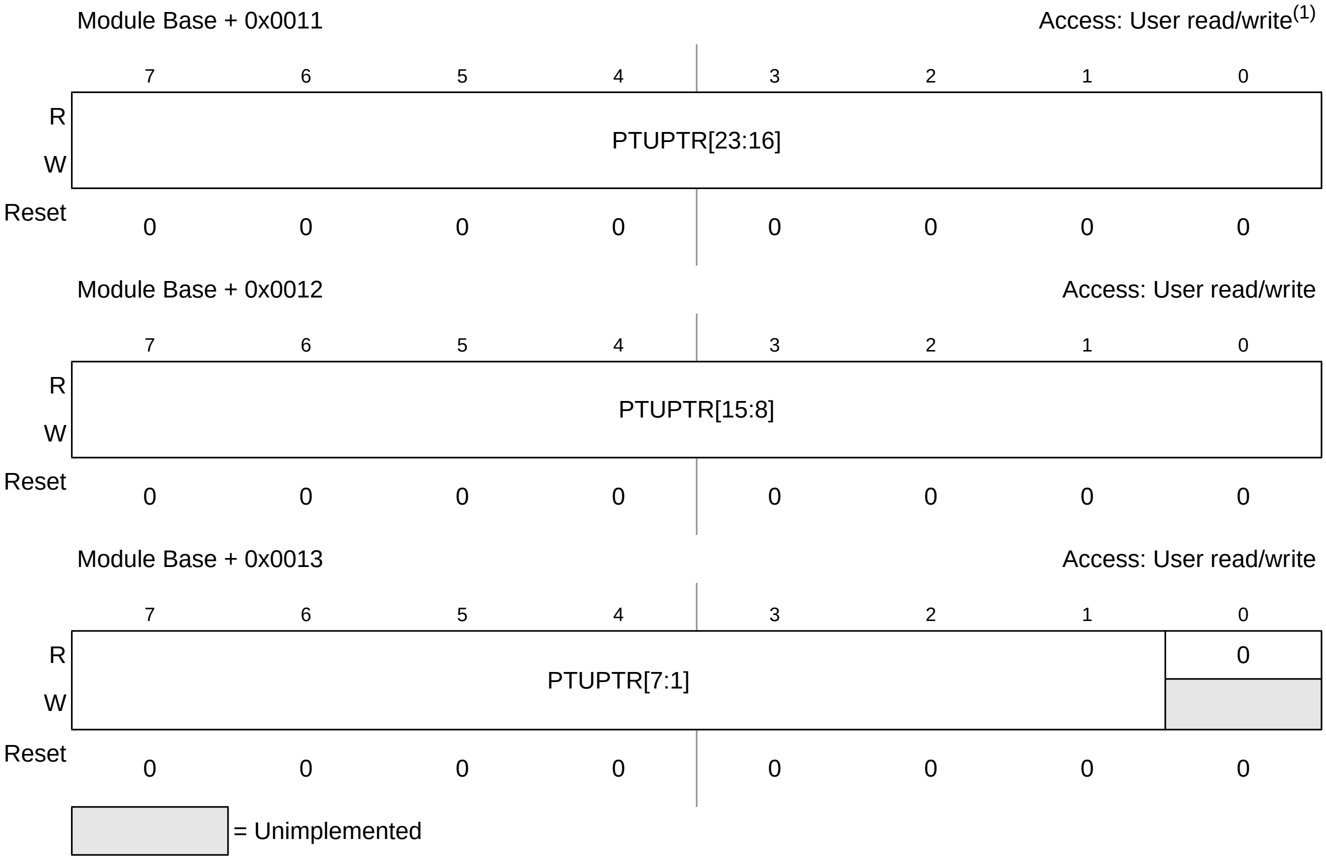


Figure 13-16. PTU List Add Register (PTUPTRH, PTUPTRM, PTUPTRL)

1. Read: Anytime
   Write: Anytime, if TG0En and TG1EN bit are cleared

Table 13-16. PTUPTR Register Field Descriptions

Field	Description
PTUPTR [23:0]	**PTU Pointer** — This register cannot be modified if TG0EN or TG1EN bit is set. This register defines the start address of the used list area inside the global memory map. For more information see Section 13.4.2, “Memory based trigger event list”.

13.3.2.15 Trigger Generator 0 List 0 Index (TG0L0IDX)

Module Base + 0x0014

Access: User read only(1)

Figure 13-17. Trigger Generator 0 List 0 Index (TG0L0IDX)

1. Read: Anytime
   Write: Never

Table 13-17. TG0L0IDX Register Field Descriptions

Field	Description
6:0 TG0L0IDX [6:0]	**Trigger Generator 0 List 0 Index Register** — This register defines offset of the start point for the trigger event list 0 used by trigger generator 0. This register is read only, so the list 0 for trigger generator 0 will start at the PTUPTR address. For more information see Section 13.4.2, "Memory based trigger event list".

13.3.2.16 Trigger Generator 0 List 1 Index (TG0L1IDX)

Module Base + 0x0015

Access: User read/write(1)

Figure 13-18. Trigger Generator 0 List 1 Index (TG0L1IDX)

1. Read: Anytime
   Write: Anytime, if TG0EN bit is cleared

Table 13-18. TG0L1IDX Register Field Descriptions

Field	Description
6:0 TG0L1IDX [6:0]	**Trigger Generator 0 List 1 Index Register** — This register cannot be modified after the TG0EN bit is set. This register defines offset of the start point for the trigger event list 1 used by trigger generator 0. For more information see Section 13.4.2, "Memory based trigger event list".

13.3.2.17 Trigger Generator 1 List 0 Index (TG1L0IDX)

Module Base + 0x0016 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	TG1L0IDX[6:0]						
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-19. Trigger Generator 1 List 0 Index (TG1L0IDX)

1. Read: Anytime
   Write: Anytime, if TG1EN bit is cleared

Table 13-19. TG0L1IDX Register Field Descriptions

Field	Description
6:0 TG1L0IDX [6:0]	**Trigger Generator 1 List 0 Index Register** — This register cannot be modified after the TG1EN bit is set. This register defines offset of the start point for the trigger event list 0 used by trigger generator 1. For more information see Section 13.4.2, “Memory based trigger event list”.

Trigger Generator 1 List 1 Index (TG1L1IDX)

Module Base + 0x0017 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	TG1L1IDX[6:0]						
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-20. Trigger Generator 1 List 1 Index (TG1L1IDX)

1. Read: Anytime
   Write: Anytime, if TG1EN bit is cleared

Table 13-20. TG1L1IDX Register Field Descriptions

Field	Description
6:0 TG1L1IDX [6:0]	**Trigger Generator 1 List 1 Index Register** — This register cannot be modified after the TG1EN bit is set. This register defines offset of the start point for the trigger event list 1 used by trigger generator 1. For more information see Section 13.4.2, “Memory based trigger event list”.

13.3.2.18 PTU Debug Register (PTUDEBUG)

Module Base + 0x001F Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	PTUREPE	PTUT1PE	PTUT0PE	0	0	0	0
W	▨				▨	PTUFRE	TG1FTE	TG0FTE
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-21. PTU Debug Register (PTUDEBUG)

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

Table 13-21. PTUDEBUG Register Field Descriptions

Field	Description
6 PTUREPE	**PTURE Pin Enable** — This bit enables the output port for pin PTURE. 0 PTURE output port are disabled 1 PTURE output port are enabled
5 PTUT1PE	**PTU PTUT1 Pin Enable** — This bit enables the output port for pin PTUT1. 0 PTUT1 output port are disabled 1 PTUT1 output port are enabled
4 PTUT0PE	**PTU PTUT0 Pin Enable** — This bit enables the output port for pin PTUT0. 0 PTUT0 output port are disabled 1 PTUT0 output port are enabled
2 PTUFRE	**Force Reload event generation** — If one of the TGs is enabled then writing 1 to this bit will generate a reload event. The reload event forced by PTUFRE does not set the PTUROIF interrupt flag. Also the ptu_reload signal asserts for one bus clock cyclet. Writing 0 to this bit has no effect. Always reads back as 0. This behavior is not available during stop or freeze mode.
1 TG1FTE	**Trigger Generator 1 Force Trigger Event** — If TG1 is enabled then writing 1 to this bit will generate a trigger event independent on the list based trigger generation. Writing 0 to this bit has no effect. Always reads back as 0.This behavior is not available during stop or freeze mode.
0 TG0FTE	**Trigger Generator 0 Force Trigger Event** — If TG0 is enabled then writing 1 to this bit will generate a trigger event independent on the list based trigger generation. Writing 0 to this bit has no effect. Always reads back as 0. This behavior is not available during stop or freeze mode.

13.4 Functional Description

13.4.1 General

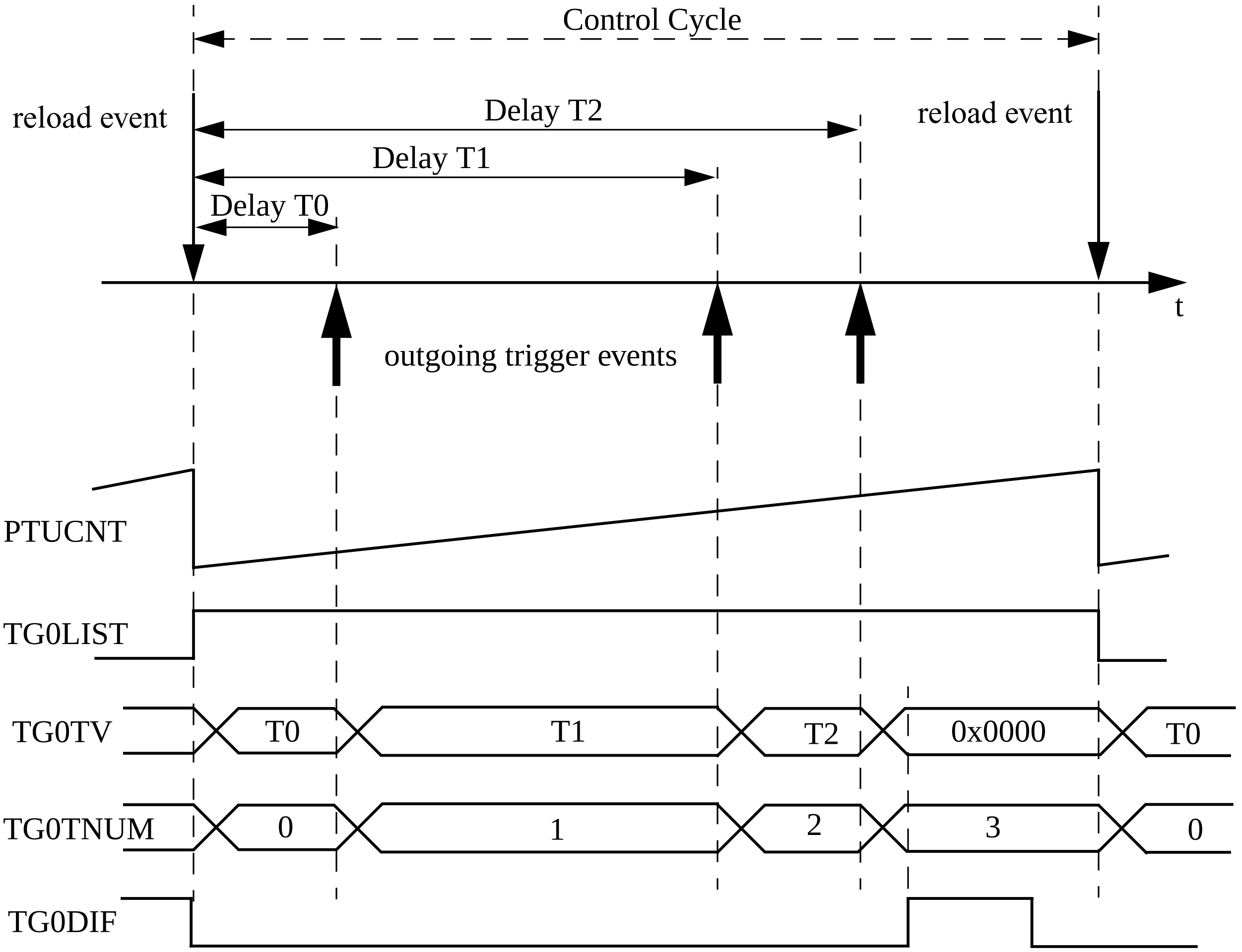
The PTU module consists of two trigger generators (TG0 and TG1). For each TG a separate enable bit is available, so that both TGs can be enabled independently.

If both trigger generators are disabled then the PTU is disabled, the trigger generation stops and the memory accesses are disabled.

The trigger generation of the PTU module is synchronized to the incoming reload event. This reload event resets and restarts the internal time base counter and makes sure that the first trigger value from the actual trigger list is loaded. Furthermore the corresponding module is informed that a new control cycle has started.

If the counter value matches the current trigger value then a trigger event is generated. In this way, the reload event is delayed by the number of bus clock cycles defined by the current trigger value. After this, a new trigger value is loaded from the memory and the TG waits for the next match. So up to 32 trigger events per control cycle can be generated. If the trigger value is 0x0000 or 32 trigger events have been generated during this control cycle, the TGxDIF bit is set and the TG waits for the next reload event. Figure 13-22 shows an example of the trigger generation using the trigger values shown in Figure 13-23.

Figure 13-22. TG0 trigger generation example



NOTE

If the trigger list contains less than 32 trigger values a delay between the generation of the last trigger and the assertion of the done interrupt flag will be visible. During this time the PTU loads the next trigger value from the memory to evaluate the EOL symbol.

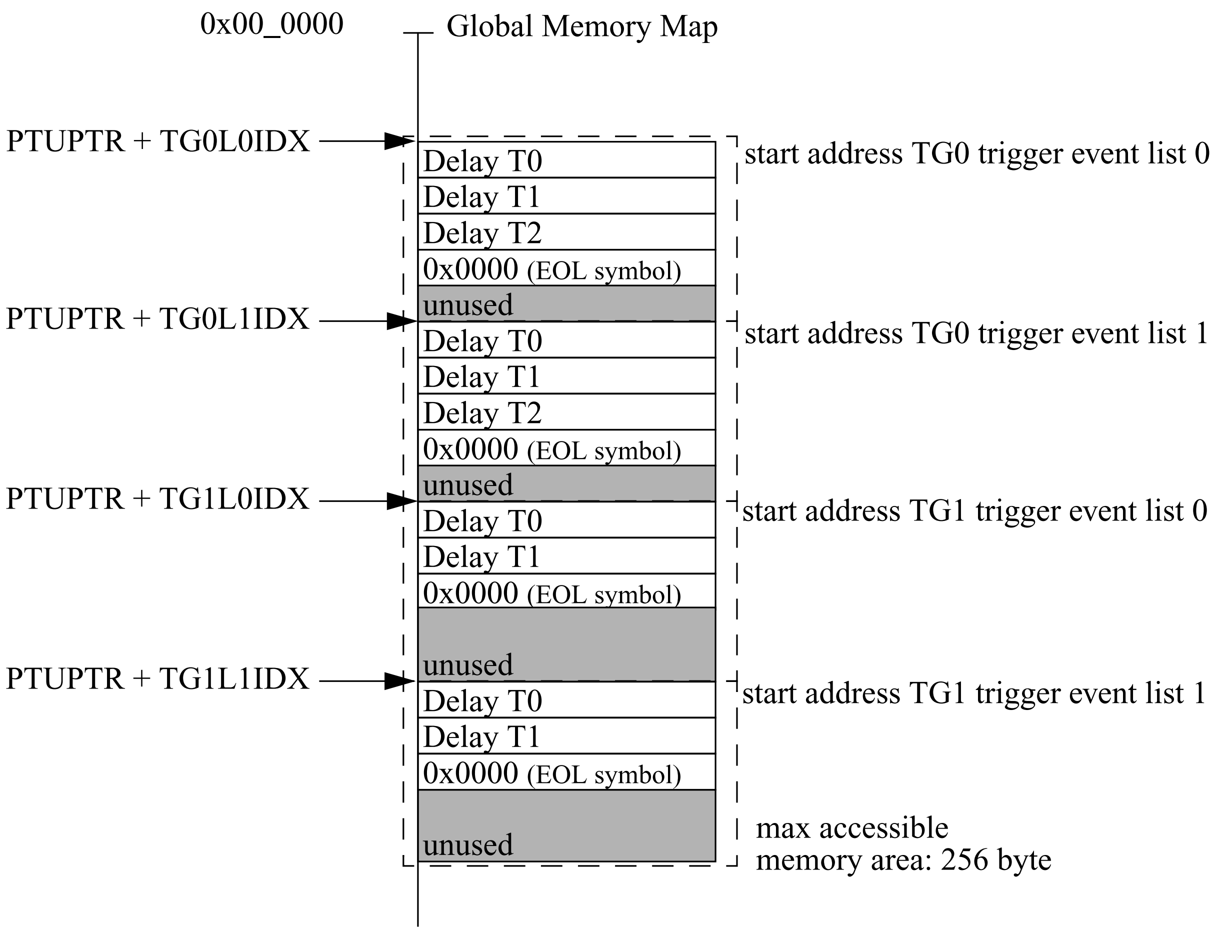
13.4.2 Memory based trigger event list

The lists with the trigger values are located inside the global memory map. The location of the trigger lists in the memory map is configured with registers PTUPTR and TGxLxIDX. If one of the TGs is enabled then the PTUPTR register is locked. If the TG is enabled then the associated TGxLxIDX registers are locked.

The trigger values inside the trigger list are 16 bit values. Each 16 bit value defines the delay between the reload event and the trigger event in bus clock cycles. A delay value of 0x0000 will be interpreted as End Of trigger List (EOL) symbol. The list must be sorted in ascending order. If a subsequent value is smaller than the previous value or the loaded trigger value is smaller than the current counter value then the TGxTEIF error indication is generated and the trigger generation of this list is stopped until the next reload event. For more information about these error scenario see Section 13.4.5.5, “Trigger Generator Timing Error”.

The module is not able to access memory area outside the 256 byte window starting at the memory address defined by PTUPTR.

Figure 13-23. Global Memory map usage



13.4.3 Reload mechanism

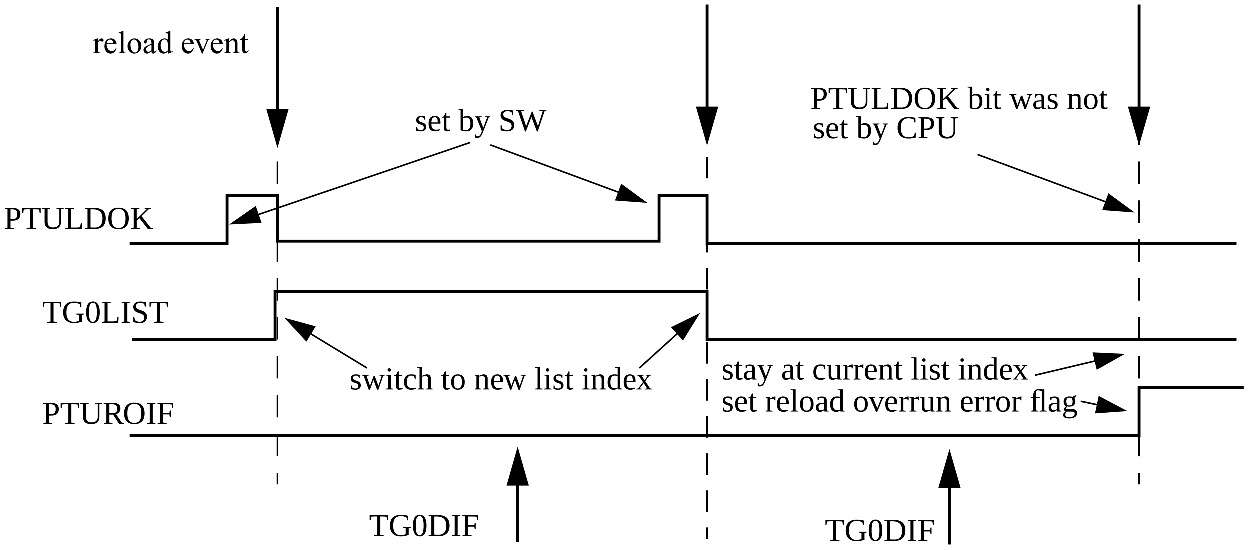
Each trigger generator uses two lists to load the trigger values from the memory. One list can be updated by the CPU while the other list is used to generate the trigger events. After enabling, the TG uses the lists in alternate order. When the update of alternate trigger list is done, the SW must set the PTULDOK bit. If the load OK bit is set at the time of reload event, the TG switches to the alternate list and loads the first trigger value from this trigger event list. The reload event clears the PTULDOK bit.

The TG0LIST and TG1LIST bits shows the currently use list number. These bits are writeable if the associated TG is disabled.

If the PTULDOK bit was not set before the reload event then the reload overrun error flag is set (PTUROIF)and both TGs do not switch to the alternative list. The current trigger list is used to load the trigger values. Figure 13-24 shows an example. The PTULDOK bit can be used by other modules as glb_ldok.

To reduce the used memory size, it is also possible to set TG0L0IDX equal to TG0L1IDX or to set TG1L0IDX equal to TG1L1IDX. In this case the trigger generator is using only one physical list of trigger events even if the trigger generator logic is switching between both pointers. The SW must make sure, that the CPU does not update the trigger list before the execution of the trigger list is done. The time window to update the trigger list starts at the trigger generator done interrupt flag (TGxDIF) and ends with the next reload event. Even if only one physical trigger event list is used the TGxLIST shows a swap between list 0 and 1 at every reload event with set PTULDOK bit.

Figure 13-24. TG0 Reload behavior with local PTULDOK



13.4.4 Async reload event

If the reload and reload_is_async are active at the same time then an async reload event happens. The PTU behavior on an async reload event is the same like on the reload event described in Section 13.4.3, "Reload mechanism" above. The only difference is, that during an async reload event the error interrupt flags PTUROIF and TGxREIF are not generated.

13.4.5 Interrupts and error handling

This sections describes the interrupts generated by the PTU module and their individual sources, Vector addresses and interrupt priority are defined by MCU level.

Table 13-22. PTU Interrupt Sources

Module Interrupt Sources	Local Enable
PTU Reload Overrun Error	PTUIEH[PTUROIE]
TG0 Error	PTUIEL[TG0AEIE,TG0REIE,TG0TEIE]
TG1 Error	PTUIEL[TG1AEIE,TG1REIE,TG1TEIE]
TG0 Done	PTUIEL[TG0DIE]
TG1 Done	PTUIEL[TG1DIE]

13.4.5.1 PTU Double Bit ECC Error

If one trigger generator reads trigger values from the memory which contains double bit ECC errors then the PTUDEEF is set. These read data are ignored and the execution of both trigger generators is stopped until the PTUDEEF flag was cleared. To make sure the trigger generator starts in a define state it is required to execute follow sequence:

1. disable both trigger generators
2. configure the PTU if required
3. clear the PTUDEEF
4. enable the desired trigger generators

13.4.5.2 PTU Reload Overrun Error

If the PTULDOK bit is not set during the reload event then the PTUROIF bit is set. If enabled (PTUROIE is set) an interrupt is generated. For more information see Section 13.4.3, "Reload mechanism". During an async reload event the PTUROIF interrupt flag is not set.

13.4.5.3 Trigger Generator Memory Access Error

The trigger generator memory access error flag (TGxAEIF) is set if the used read address is outside the accessible memory address area; see the MMC section for the supported memory area. The loaded trigger values are ignored and the execution of this trigger list is stopped until the next reload event. If enabled (TGxAEIE is set) an interrupt will be generated.

13.4.5.4 Trigger Generator Reload Error

The trigger generator reload error flag (TGxREIF) is set if a new reload event occurs before the trigger generator reaches the EOL symbol or the maximum number of generated triggers. Independent from this error condition the trigger generator reloads the new data from the trigger list and starts to generate the trigger. During an async reload event the TGxREIF interrupt flag is not set.

If the trigger value loaded from the memory contains double bit ECC errors (PTUDEEF flag is set) then the data are ignored and the trigger generator reload error flag (TGxREIF) is not set.

13.4.5.5 Trigger Generator Timing Error

The PTU module requires minimum 6 core clock cycles to reload the next trigger values from the memory. This reload time defines the minimum possible distance between two consecutive trigger values within a trigger list or the distance between the reload event and the first trigger value.

The trigger generator timing error flag (TGxTEIF) is set if the loaded trigger value is smaller than the current counter value. The execution of this trigger list is stopped until the next reload event. There are different reasons for the trigger generator error condition:

- reload time exceeds time of next trigger event
- reload time exceeds the time between two consecutive trigger values
- a subsequent trigger value is smaller than the predecessor trigger value

If the trigger value loaded from the memory contains double bit ECC errors (PTUDEEF flag is set) then the data are ignored and the trigger generator timing error flag (TGxTEIF) is not set.

If enabled (TGxEIE is set) an interrupt will be generated.

13.4.5.6 Trigger Generator Done

The trigger generator done flag (TGxDIF) is set if the loaded trigger value contains 0x0000 or if the number of maximum trigger events (32) was reached. Please note, that the time which is required to load the next trigger value defines the delay between the generation of the last trigger and the assertion of the done flag. If enabled (TGxDIE is set) an interrupt is generated.If the trigger value loaded from the memory contains double bit ECC errors (PTUDEEF flag is set) then the data are ignored and the trigger generator done flag (TGxDIF) is not set.

13.4.6 Debugging

To see the internal status of the trigger generator the register TGxLIST, TGxTNUM, and TGxTV can be used. The TGxLIST register shows the number of currently used list. The TGxTNUM shows the number of generated triggers since the last reload event. If the maximum number of triggers was generated then this register shows zero. The trigger value loaded from the memory to generate the next trigger event is visible inside the TGxTV register. If the execution of the trigger list is done then these registers are unchanged until the next reload event. The next PWM reload event clears the TGxTNUM register and toggles the used trigger list if PTULDOK was set.

To generate a reload event or trigger event independent from the PWM status the debug register bits PTUFRE or TGxFTE can be used. A write one to this bits will generate the associated event. This behavior is not available during stop or freeze mode.

Chapter 14
Pulse Width Modulator with Fault Protection (PMF15B6CV3)

Table 14-1. Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V03.23	10 Oct 2013	14.2.8/14-490 14.3.2.18/14-509 14.3.2.22/14-513 14.8.1.1/14-553	• Corrected pmf_reload_is_async signal description • Enhanced note at PMFCINV register • Corrected write value limitations for PMFMODx registers • Corrected register write protection bit names • Orthographical corrections after review
V03.24	08 Nov 2013	14.3.2.8/14-502 Table 14-14 14.4.7/14-537	• Updated PMFFIF bit description • Updated note to QSMP table • Updated Asymmetric PWM output description • Replaced ‘fault clearing’ with ‘fault recovery’ to avoid ambiguity with flags • Various minor corrections. •
V03.25	03 Dec 2013	14.3.2.18/14-509	• Updated note at PMFCINV register

Glossary

Table 14-2. Glossary of Terms

Term	Definition
Set	Discrete signal is in active logic state.
Clear	A discrete signal is in inactive logic state.
Pin	External physical connection.
Signal	Electronic construct whose state or change in state conveys information.
PWM active state Normal output Positive polarity	PWM logic level high causing external power device to conduct
PWM inactive or disabled state Inverted output Negative polarity	PWM logic level low causing external power device not to conduct
PWM clock	Clock supplied to PWM and deadtime generators. Based on core clock. Rate depends on prescaler setting.

Table 14-2. Glossary of Terms

Term	Definition
PWM cycle	PWM period determined by modulus register and PWM clock rate. Note the differences in edge- or center-aligned mode.
PWM reload cycle	A.k.a. control cycle. Determined by load frequency which is 1 to n-times the PWM cycle. PWM reload cycle triggered double-buffered registers take effect at the next PWM reload event.
Commutation cycle	For 6-step motor control only. Started by an event external to the PMF module (async_event). This may be a delayed Hall effect or back-EMF zero crossing event determining the rotor position. Commutation cycle triggered double-buffered registers take effect at the next commutation event and optionally the PWM counters are restarted.
Index *x*	Related to time bases. *x* = A, B or C
Index *n*	Related to PWM channels. *n* = 0, 1, 2, 3, 4, or 5
Index *m*	Related to fault inputs. *m* = 0, 1, 2, 3, 4, or 5

14.1 Introduction

The Pulse width Modulator with Fault protection (PMF) module can be configured for one, two, or three complementary pairs. For example:

- One complementary pair and four independent PWM outputs
- Two complementary pairs and two independent PWM outputs
- Three complementary pairs and zero independent PWM outputs
- Zero complementary pairs and six independent PWM outputs

All PWM outputs can be generated from the same counter, or each pair can have its own counter for three independent PWM frequencies. Complementary operation permits programmable deadtime insertion, distortion correction through current sensing by software, and separate top and bottom output polarity control. Each counter value is programmable to support a continuously variable PWM frequency. Both edge- and center-aligned synchronous pulse width-control and full range modulation from 0 percent to 100 percent, are supported. The PMF is capable of controlling most motor types: AC induction motors (ACIM), both brushless (BLDC) and brush DC motors (BDC), switched (SRM) and variable reluctance motors (VRM), and stepper motors.

14.1.1 Features

- Three complementary PWM signal pairs, or six independent PWM signals
- Edge-aligned or center-aligned mode
- Features of complementary channel operation:
    - Deadtime insertion
    - Separate top and bottom pulse width correction via current status inputs or software
    - Three variants of PWM output:

  - Asymmetric in center-aligned mode
  - Variable edge placement in edge-aligned mode
  - Double switching in center-aligned mode
- Three 15-bit counters based on core clock
- Separate top and bottom polarity control
- Half-cycle reload capability
- Integral reload rates from 1 to 16
- Programmable fault protection
- Link to timer output compare for 6-step BLDC commutation support with optional counter restart Reload overrun interrupt
- PWM compare output polarity control Software-controlled PWM outputs, complementary or independent

14.1.2 Modes of Operation

Care must be exercised when using this module in the modes listed in Table 14-3. Some applications require regular software updates for proper operation. Failure to do so could result in destroying the hardware setup. Because of this, PWM outputs are placed in their inactive states in STOP mode, and optionally under WAIT and FREEZE modes. PWM outputs will be reactivated (assuming they were active to begin with) when these modes are exited.

Table 14-3. Modes When PWM Operation is Restricted

Mode	Description
STOP	PWM outputs are disabled
WAIT	PWM outputs are disabled as a function of the PMFWAI bit
FREEZE	PWM outputs are disabled as a function of the PMFFRZ bit

14.1.3 Block Diagram

Figure 14-1 provides an overview of the PMF module.

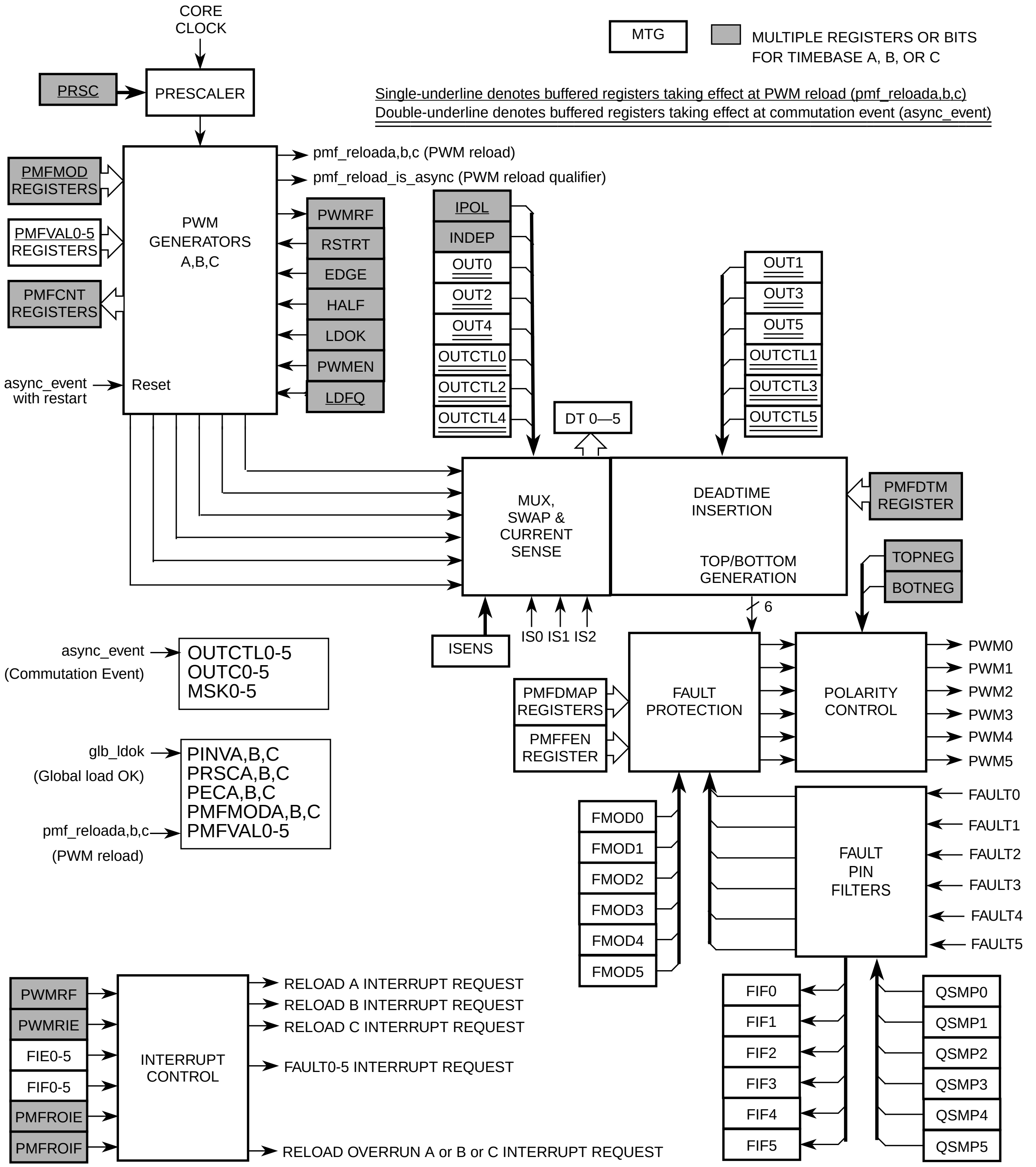


Figure 14-1. PMF Block Diagram

14.2 Signal Descriptions

If the signals are not used exclusively internally, the PMF has external pins named PWM0–5, FAULT0–5, and $\overline{\text{IS0}}$–$\overline{\text{IS2}}$. Refer to device overview section.

14.2.1 PWM0–PWM5 Pins

PWM0–PWM5 are the output signals of the six PWM channels.

NOTE

> On MCUs with an integrated gate drive unit the PWM outputs are connected internally to the GDU inputs. In these cases the PWM signals may optionally be available on pins for monitoring purposes. Refer to the device overview section for routing options and pin locations.

14.2.2 FAULT0–FAULT5 Pins

FAULT0–FAULT5 are input signals for disabling selected PWM outputs (FAULT0-3) or drive the outputs to a configurable active/inactive state (FAULT4-5).

NOTE

> On MCUs with an integrated gate drive unit (GDU) either one or more FAULT inputs may be connected internally or/and available on an external pin. Refer to the device overview section for availability and pin locations.

14.2.3 $\overline{\text{IS0}}$–$\overline{\text{IS2}}$ Pins

$\overline{\text{IS0}}$–$\overline{\text{IS2}}$ are current status signals for top/bottom pulse width correction in complementary channel operation while deadtime is asserted.

NOTE

> Refer to the device overview section for signal availability on pins.

14.2.4 Global Load OK Signal — glb_ldok

This device-internal PMF input signal is connected to the global load OK bit at integration level. For each of the three PWM generator time bases the use of the global load OK input can be enabled individually (GLDOKA,B,C).

14.2.5 Commutation Event Signal — async_event

This device-internal PMF input signal is connected to the source of the asynchronous event generator (preferably timer output compare channel) at integration level.

The commutation event input must be enabled to take effect (ENCE=1). When this bit is set the PMFOUTC, PMFOUT, and MSKx registers switch from non-buffered to async_event triggered double buffered mode. In addition, if restart is enabled (RSTRTx=1), the commutation event generates both "PWM reload event" and "PWM reload-is-asynchronous event" simultaneously.

14.2.6 Commutation Event Edge Select Signal — async_event_edge_sel[1:0]

These device-internal PMF input signals select the active edge for the async_event input. Refer to the device overview section to determine if the selection is user configurable or tied constant at integration level.

Table 14-4. Commutation Event Edge Selection

async_event_edge-sel[1:0]	async_event active edge
00	direct input
01	rising edge
10	falling edge
11	both edges

14.2.7 PWM Reload Event Signals — pmf_reloada,b,c

These device-internal PMF output signals assert once per control cycle and can serve as triggers for other implemented IP modules. Signal pmf_reloadb and pmf_reloadc are related to time base B and C, respectively, while signal pmf_reloada is off out of reset and can be programmed for time base A, B, or C. Refer to the device overview section to determine the signal connections.

14.2.8 PWM Reload-Is-Asynchronous Signal — pmf_reload_is_async

This device-internal PMF output signal serves as a qualifier to the PMF reload event signal pmf_reloada. Whenever the async_event signal causes pmf_reloada output to assert also the pmf_reload_is_async output asserts for the same duration, except if asynchronous event and generated PWM reload event occur in the same cycle.

14.3 Memory Map and Registers

14.3.1 Module Memory Map

A summary of the registers associated with the PMF module is shown in Figure 14-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

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	PMFCFG0	R	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
		W								
0x0001	PMFCFG1	R	0	ENCE	BOTNEGC	TOPNEGC	BOTNEGB	TOPNEGB	BOTNEGA	TOPNEGA
		W	▨							
0x0002	PMFCFG2	R	REV1	REV0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		W								
0x0003	PMFCFG3	R	PMFWAI	PMFFRZ	0	VLMODE		PINVC	PINVB	PINVA
		W			▨					
0x0004	PMFFEN	R	0	FEN5	0	FEN4	FEN3	FEN2	FEN1	FEN0
		W	▨		▨					
0x0005	PMFFMOD	R	0	FMOD5	0	FMOD4	FMOD3	FMOD2	FMOD1	FMOD0
		W	▨		▨					
0x0006	PMFFIE	R	0	FIE5	0	FIE4	FIE3	FIE2	FIE1	FIE0
		W	▨		▨					
0x0007	PMFFIF	R	0	FIF5	0	FIF4	FIF3	FIF2	FIF1	FIF0
		W	▨		▨					
0x0008	PMFQSMP0	R	0	0	0	0	QSMP5		QSMP4	
		W	▨	▨	▨	▨				
0x0009	PMFQSMP1	R	QSMP3		QSMP2		QSMP1		QSMP0	
		W								
0x000A- 0x000B	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 14-2. Quick Reference to PMF Registers (Sheet 1 of 5)

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000C	PMFOUTC	R	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
		W	▨	▨						
0x000D	PMFOUTB	R	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
		W	▨	▨						
0x000E	PMFDTMS	R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
		W	▨	▨	▨	▨	▨	▨	▨	▨
0x000F	PMFCCTL	R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
		W	▨	▨			▨			
0x0010	PMFVAL0	R	PMFVAL0							
		W								
0x0011	PMFVAL0	R	PMFVAL0							
		W								
0x0012	PMFVAL1	R	PMFVAL1							
		W								
0x0013	PMFVAL1	R	PMFVAL1							
		W								
0x0014	PMFVAL2	R	PMFVAL2							
		W								
0x0015	PMFVAL2	R	PMFVAL2							
		W								
0x0016	PMFVAL3	R	PMFVAL3							
		W								
0x0017	PMFVAL3	R	PMFVAL3							
		W								
0x0018	PMFVAL4	R	PMFVAL4							
		W								
0x0019	PMFVAL4	R	PMFVAL4							
		W								
0x001A	PMFVAL5	R	PMFVAL5							
		W								

▨ = Unimplemented or Reserved

Figure 14-2. Quick Reference to PMF Registers (Sheet 2 of 5)

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001B	PMFVAL5	R	PMFVAL5							
		W								
0x001C	PMFROIE	R	0	0	0	0	0	PMFROIEC	PMFROIEB	PMFROIEA
		W	▨	▨	▨	▨	▨			
0x001D	PMFROIF	R	0	0	0	0	0	PMFROIFC	PMFROIFB	PMFROIFA
		W	▨	▨	▨	▨	▨			
0x001E	PMFICCTL	R	0	0	PECC	PECB	PECA	ICCC	ICCB	ICCA
		W	▨	▨						
0x001F	PMFCINV	R	0	0	CINV5	CINV4	CINV3	CINV2	CINV1	CINV0
		W	▨	▨						
0x0020	PMFENCA	R	PWMENA	GLDOKA	0	0	0	RSTRTA	LDOKA	PWMRIEA
		W			▨	▨	▨			
0x0021	PMFFQCA	R	LDFQA				HALFA	PRSCA		PWMRFA
		W								
0x0022	PMFCNTA	R	0	PMFCNTA						
		W	▨	▨						
0x0023	PMFCNTA	R	PMFCNTA							
		W	▨							
0x0024	PMFMODA	R	0	PMFMODA						
		W	▨							
0x0025	PMFMODA	R	PMFMODA							
		W								
0x0026	PMFDTMA	R	0	0	0	0	PMFDTMA			
		W	▨	▨	▨	▨				
0x0027	PMFDTMA	R	PMFDTMA							
		W								
0x0028	PMFENCB	R	PWMENB	GLDOKB	0	0	0	RSTRTB	LDOKB	PWMRIEB
		W			▨	▨	▨			
0x0029	PMFFQCB	R	LDFQB				HALFB	PRSCB		PWMRFB
		W								

▨ = Unimplemented or Reserved

Figure 14-2. Quick Reference to PMF Registers (Sheet 3 of 5)

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x002A	PMFCNTB	R	0	PMFCNTB						
		W	▨	▨						
0x002B	PMFCNTB	R	PMFCNTB							
		W	▨							
0x002C	PMFMODB	R	0	PMFMODB						
		W	▨							
0x002D	PMFMODB	R	PMFMODB							
		W								
0x002E	PMFDTMB	R	0	0	0	0	PMFDTMB			
		W	▨	▨	▨	▨				
0x002F	PMFDTMB	R	PMFDTMB							
		W								
0x0030	PMFENCC	R	PWMENC	GLDOKC	0	0	0	RSTRTC	LDOKC	PWMRIEC
		W			▨	▨	▨			
0x0031	PMFFQCC	R	LDFQC				HALFC	PRSCC		PWMRFC
		W								

▨ = Unimplemented or Reserved

Figure 14-2. Quick Reference to PMF Registers (Sheet 4 of 5)

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0032	PMFCNTC	R	0	PMFCNTC						
		W	▨	▨						
0x0033	PMFCNTC	R	PMFCNTC							
		W	▨							
0x0034	PMFMODC	R	0	PMFMODC						
		W	▨							
0x0035	PMFMODC	R	PMFMODC							
		W								
0x0036	PMFDTMC	R	0	0	0	0	PMFDTMC			
		W	▨	▨	▨	▨				
0x0037	PMFDTMC	R	PMFDTMC							
		W								
0x0038	PMFDMP0	R	DMP05		DMP04		DMP03	DMP02	DMP01	DMP00
		W								
0x0039	PMFDMP1	R	DMP15		DMP14		DMP13	DMP12	DMP11	DMP10
		W								
0x003A	PMFDMP2	R	DMP25		DMP24		DMP23	DMP22	DMP21	DMP20
		W								
0x003B	PMFDMP3	R	DMP35		DMP34		DMP33	DMP32	DMP31	DMP30
		W								
0x003C	PMFDMP4	R	DMP45		DMP44		DMP43	DMP42	DMP41	DMP40
		W								
0x003D	PMFDMP5	R	DMP55		DMP54		DMP53	DMP52	DMP51	DMP50
		W								
0x003E	PMFOUTF	R	0	0	OUTF5	OUTF4	OUTF3	OUTF2	OUTF1	OUTF0
		W	▨	▨						
0x003F	Reserved	R	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented or Reserved

Figure 14-2. Quick Reference to PMF Registers (Sheet 5 of 5)

14.3.2 Register Descriptions

14.3.2.1 PMF Configure 0 Register (PMFCFG0)

Address: Module Base + 0x0000 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
Reset	0	0	0	0	0	0	0	0

Figure 14-3. PMF Configure 0 Register (PMFCFG0)

1. Read: Anytime
Write: This register cannot be modified after the WP bit is set

Table 14-5. PMFCFG0 Field Descriptions

Field	Description
7 WP	**Write Protect**— This bit enables write protection to be used for all write-protectable registers. While clear, WP allows write-protected registers to be written. When set, WP prevents any further writes to write-protected registers. Once set, WP can be cleared only by reset. 0 Write-protectable registers may be written 1 Write-protectable registers are write-protected
6 MTG	**Multiple Timebase Generators** — This bit determines the number of timebase counters used. This bit cannot be modified after the WP bit is set. If MTG is set, PWM generators B and C and registers 0x0028 – 0x0037 are availabled.The three generators have their own variable frequencies and are not synchronized. If MTG is cleared, PMF registers from 0x0028 – 0x0037 can not be written and read zeroes, and bits EDGEC and EDGEB are ignored. Pair A, Pair B, and Pair C PWMs are synchronized to PWM generator A and use registers from 0x0020 – 0x0027. 0 Single timebase generator 1 Multiple timebase generators
5 EDGEC	**Edge-Aligned or Center-Aligned PWM for Pair C** — This bit determines whether PWM4 and PWM5 channels will use edge-aligned or center-aligned waveforms. This bit has no effect if MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM4 and PWM5 are center-aligned PWMs 1 PWM4 and PWM5 are edge-aligned PWMs
4 EDGEB	**Edge-Aligned or Center-Aligned PWM for Pair B** — This bit determines whether PWM2 and PWM3 channels will use edge-aligned or center-aligned waveforms. This bit has no effect if MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM2 and PWM3 are center-aligned PWMs 1 PWM2 and PWM3 are edge-aligned PWMs
3 EDGEA	**Edge-Aligned or Center-Aligned PWM for Pair A**— This bit determines whether PWM0 and PWM1 channels will use edge-aligned or center-aligned waveforms. It determines waveforms for Pair B and Pair C if the MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM0 and PWM1 are center-aligned PWMs 1 PWM0 and PWM1 are edge-aligned PWMs
2 INDEPC	**Independent or Complementary Operation for Pair C**— This bit determines if the PWM channels 4 and 5 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM4 and PWM5 are complementary PWM pair 1 PWM4 and PWM5 are independent PWMs

Table 14-5. PMFCFG0 Field Descriptions (continued)

Field	Description
1 INDEPB	**Independent or Complementary Operation for Pair B**— This bit determines if the PWM channels 2 and 3 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM2 and PWM3 are complementary PWM pair 1 PWM2 and PWM3 are independent PWMs
0 INDEPA	**Independent or Complementary Operation for Pair A**— This bit determines if the PWM channels 0 and 1 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM0 and PWM1 are complementary PWM pair 1 PWM0 and PWM1 are independent PWMs

14.3.2.2 PMF Configure 1 Register (PMFCFG1)

Address: Module Base + 0x0001 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	ENCE	BOTNEGC	TOPNEGC	BOTNEGB	TOPNEGB	BOTNEGA	TOPNEGA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-4. PMF Configure 1 Register (PMFCFG1)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set

A normal PWM output or positive polarity means that the PWM channel outputs high when the counter value is smaller than or equal to the pulse width value and outputs low otherwise. An inverted output or negative polarity means that the PWM channel outputs low when the counter value is smaller than or equal to the pulse width value and outputs high otherwise.

NOTE

The TOPNEGx and BOTNEGx are intended for adapting to the polarity of external predrivers on devices driving the PWM output directly to pins. If an integrated GDU is driven it must be made sure to keep the reset values of these bits in order not to violate the deadtime insertion.

Table 14-6. PMFCFG1 Field Descriptions

Field	Description
6 ENCE	**Enable Commutation Event** — This bit enables the commutation event input and activates buffering of registers PMFOUTC and PMFOUTB and MSKx bits.This bit cannot be modified after the WP bit is set.If set to zero the commutation event input is ignored and writes to the above registers and bits will take effect immediately. If set to one, the commutation event input is enabled and the value written to the above registers and bits does not take effect until the next commutation event occurs. 0 Commutation event input disabled and PMFOUTC, PMFOUTB and MSK*n* not buffered 1 Commutation event input enabled and PMFOUTC, PMFOUTB and MSK*n* buffered
5 BOTNEGC	**Pair C Bottom-Side PWM Polarity** — This bit determines the polarity for Pair C bottom-side PWM (PWM5). This bit cannot be modified after the WP bit is set. 0 Positive PWM5 polarity 1 Negative PWM5 polarity

Table 14-6. PMFCFG1 Field Descriptions (continued)

Field	Description
4 TOPNEGC	**Pair C Top-Side PWM Polarity** — This bit determines the polarity for Pair C top-side PWM (PWM4). This bit cannot be modified after the WP bit is set. 0 Positive PWM4 polarity 1 Negative PWM4 polarity
3 BOTNEGB	**Pair B Bottom-Side PWM Polarity** — This bit determines the polarity for Pair B bottom-side PWM (PWM3). This bit cannot be modified after the WP bit is set. 0 Positive PWM3 polarity 1 Negative PWM3 polarity
2 TOPNEGB	**Pair B Top-Side PWM Polarity** — This bit determines the polarity for Pair B top-side PWM (PWM2). This bit cannot be modified after the WP bit is set. 0 Positive PWM2 polarity 1 Negative PWM2 polarity
1 BOTNEGA	**Pair A Bottom-Side PWM Polarity** — This bit determines the polarity for Pair A bottom-side PWM (PWM1). This bit cannot be modified after the WP bit is set. 0 Positive PWM1 polarity 1 Negative PWM1 polarity
0 TOPNEGA	P**air A Top-Side PWM Polarity** — This bit determines the polarity for Pair A top-side PWM (PWM0). This bit cannot be modified after the WP bit is set. 0 Positive PWM0 polarity 1 Negative PWM0 polarity

14.3.2.3 PMF Configure 2 Register (PMFCFG2)

Address: Module Base + 0x0002

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	REV1	REV0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
Reset	0	0	0	0	0	0	0	0

Figure 14-5. PMF Configure 2 Register (PMFCFG2)

1. Read: Anytime
   Write: Anytime

Table 14-7. PMFCFG2 Field Descriptions

Field	Description
7-6 REV[1:0]	**Select timebase counter to output reload event on pmf_reloada** These bits select if timebase generator A, B or C provides the reload event on output signal pmf_reloada. 00 Reload event generation disabled 01 PWM generator A generates reload event 10 PWM generator B generates reload event 11 PWM generator C generates reload event

Table 14-7. PMFCFG2 Field Descriptions (continued)

Field	Description
5–0 MSK[5:0]	**Mask PWM*n* —** **Note:** MSK*n* are buffered if ENCE is set. The value written does not take effect until the next commutation cycle begins. Reading MSK*n* returns the value in the buffer and not necessarily the value the output control is currently using. 0 PWM*n* is unmasked 1 PWM*n* is masked and the channel is set to a value of 0 percent duty cycle *n* is 0, 1, 2, 3, 4, and 5.

WARNING

When using the TOPNEG/BOTNEG bits and the MSK*n* bits at the same time, when in complementary mode, it is possible to have both PMF channel outputs of a channel pair set to one.

14.3.2.4 PMF Configure 3 Register (PMFCFG3)

Address: Module Base + 0x0003 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PMFWAI	PMFFRZ	0	VLMODE		PINVC	PINVB	PINVA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-6. PMF Configure 3 Register (PMFCFG3)

1. Read: Anytime
Write: This register cannot be modified after the WP bit is set, except for bits PINVA, PINVB and PINVC

Table 14-8. PMFCFG3 Field Descriptions

Field	Description
7 PMFWAI	**PMF Stops While in WAIT Mode** — When set to zero, the PWM generators will continue to run while the chip is in WAIT mode. In this mode, the peripheral clock continues to run but the CPU clock does not. If the device enters WAIT mode and this bit is one, then the PWM outputs will be switched to their inactive state until WAIT mode is exited. At that point the PWM outputs will resume operation as programmed in the PWM registers. This bit cannot be modified after the WP bit is set. 0 PMF continues to run in WAIT mode 1 PMF is disabled in WAIT mode
6 PMFFRZ	**PMF Stops While in FREEZE Mode** — When set to zero, the PWM generators will continue to run while the chip is in FREEZE mode. If the device enters FREEZE mode and this bit is one, then the PWM outputs will be switched to their inactive state until FREEZE mode is exited. At that point the PWM outputs will resume operation as programmed in the PWM registers. This bit cannot be modified after the WP bit is set. 0 PMF continues to run in FREEZE mode 1 PMF is disabled in FREEZE mode

Table 14-8. PMFCFG3 Field Descriptions (continued)

Field	Description
4–3 VLMODE [1:0]	**Value Register Load Mode** — This field determines the way the value registers are being loaded. This register cannot be modified after the WP bit is set. 00 Each value register is accessed independently 01 Writing to value register zero also writes to value registers one to five 10 Writing to value register zero also writes to value registers one to three 11 Reserved (defaults to independent access)
2 PINVC	**PWM Invert Complement Source Pair C** — This bit controls PWM4/PWM5 pair. When set, this bit inverts the COMPSRCC signal. This bit has no effect in independent mode. **Note:** PINVC is buffered. The value written does not take effect until the LDOK bit or global load OK is set and the next PWM load cycle begins. Reading PINVC returns the value in the buffer and not necessarily the value in use. 0 No inversion 1 COMPSRCC inverted only in complementary mode
1 PINVB	**PWM Invert Complement Source Pair B** — This bit controls PWM2/PWM3 pair. When set, this bit inverts the COMPSRCB signal. This bit has no effect in independent mode. **Note:** PINVB is buffered. The value written does not take effect until the LDOK bit or global load OK is set and the next PWM load cycle begins. Reading PINVB returns the value in the buffer and not necessarily the value in use. 0 No inversion 1 COMPSRCB inverted only in complementary mode
0 PINVA	**PWM Invert Complement Source Pair A** — This bit controls PWM0/PWM1 pair. When set, this bit inverts the COMPSRCA signal. This bit has no effect on in independent mode. **Note:** PINVA is buffered. The value written does not take effect until the LDOKA bit or global load OK is set and the next PWM load cycle begins. Reading PINVA returns the value in the buffer and not necessarily the value in use. 0 No inversion 1 COMPSRCA inverted only in complementary mode

14.3.2.5 PMF Fault Enable Register (PMFFEN)

Address: Module Base + 0x0004

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	FEN5	0	FEN4	FEN3	FEN2	FEN1	FEN0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-7. PMF Fault Enable Register (PMFFEN)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set

Table 14-9. PMFFEN Field Descriptions

Field	Description
6,4-0 FEN[5:0]	**Fault *m* Enable** — This register cannot be modified after the WP bit is set. 0 FAULT*m* input is disabled 1 FAULT*m* input is enabled for fault protection *m* is 0, 1, 2, 3, 4 and 5

14.3.2.6 PMF Fault Mode Register (PMFFMOD)

Address: Module Base + 0x0005 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	FMOD5	0	FMOD4	FMOD3	FMOD2	FMOD1	FMOD0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-8. PMF Fault Mode Register (PMFFMOD)

1. Read: Anytime
   Write: Anytime

Table 14-10. PMFFMOD Field Descriptions

Field	Description
6,4-0 FMOD[5:0]	**Fault *m* Pin Recovery Mode** — This bit selects automatic or manual recovery of FAULT*m* input faults. See Section 14.4.13.2, “Automatic Fault Recovery” and Section 14.4.13.3, “Manual Fault Recovery” for more details. 0 Manual fault recovery of FAULT*m* input faults 1 Automatic fault recovery of FAULT*m* input faults *m* is 0, 1, 2, 3, 4 and 5.

14.3.2.7 PMF Fault Interrupt Enable Register (PMFFIE)

Address: Module Base + 0x0006 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	FIE5	0	FIE4	FIE3	FIE2	FIE1	FIE0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-9. PMF Fault Interrupt Enable Register (PMFFIE)

1. Read: Anytime
   Write: Anytime

Table 14-11. PMFFIE Field Descriptions

Field	Description
6,4-0 FIE[5:0]	**Fault *m* Pin Interrupt Enable** — This bit enables CPU interrupt requests to be generated by the FAULT*m* input. The fault protection circuit is independent of the FIEm bit and is active when FEN*m* is set. If a fault is detected, the PWM outputs are disabled or switched to output control according to the PMF Disable Mapping registers. 0 FAULT*m* CPU interrupt requests disabled 1 FAULT*m* CPU interrupt requests enabled *m* is 0, 1, 2, 3, 4 and 5.

14.3.2.8 PMF Fault Interrupt Flag Register (PMFFIF)

Address: Module Base + 0x0007 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	FIF5	0	FIF4	FIF3	FIF2	FIF1	FIF0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-10. PMF Fault Interrupt Flag Register (PMFFIF)

1. Read: Anytime
   Write: Anytime. Write 1 to clear.

Table 14-12. PMFFIF Field Descriptions

Field	Description
6,4-0 FIF[5:0]	**Fault *m* Interrupt Flag** — This flag is set after the required number of samples have been detected after an edge to the active level(1) on the FAULT*m* input. Writing a logic one to FIF*m* clears it. Writing a logic zero has no effect. If a set flag is attempted to be cleared and a flag setting event occurs in the same cycle, then the flag remains set. The fault protection is enabled when FEN*m* is set even when the PWMs are not enabled; therefore, a fault will be latched in, requiring to be cleared in order to prevent an interrupt. 0 No fault on the FAULT*m* input 1 Fault on the FAULT*m* input **Note:** Clearing FIF*m* satisfies pending FIF*m* CPU interrupt requests. *m* is 0, 1, 2, 3, 4 and 5.

1. The active input level may be defined or programmable at SoC level. The default for internally connected resources is active-high. For availability and configurability of fault inputs on pins refer to the device overview section.

14.3.2.9 PMF Fault Qualifying Samples Register 0-1 (PMFQSMP0-1)

Address: Module Base + 0x0008 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	QSMP5		QSMP4	
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-11. PMF Fault Qualifying Samples Register (PMFQSMP0)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set.

Address: Module Base + 0x0009 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	QSMP3		QSMP2		QSMP1		QSMP0	
Reset	0	0	0	0	0	0	0	0

Figure 14-12. PMF Fault Qualifying Samples Register (PMFQSMP1)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set.

Table 14-13. PMFQSMP0-1 Field Descriptions

Field	Description
7–0 QSMP*m*[1:0]	**Fault *m* Qualifying Samples** — This field indicates the number of consecutive samples taken at the FAULT*m* input to determine if a fault is detected. The first sample is qualified after two bus cycles from the time the fault is present and each sample after that is taken every four core clock cycles. See Table 14-14. This register cannot be modified after the WP bit is set. *m* is 0, 1, 2, 3, 4 and 5.

Table 14-14. Qualifying Samples

QSMP*m*[1:0]	Number of Samples
00	1 sample(1)
01	5 samples
10	10 samples
11	15 samples

1. There is an asynchronous path from fault inputs FAULT2-0, FAULT4 if DMP*n*4=b10, and FAULT5 if DMP*n*5=b10 to disable PWMs immediately but the fault is qualified in two bus cycles.

14.3.2.10 PMF Output Control Register (PMFOUTC)

Address: Module Base + 0x000C

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-13. PMF Output Control Register (PMFOUTC)

1. Read: Anytime
   Write: Anytime

Table 14-15. PMFOUTC Field Descriptions

Field	Description
5–0 OUTCTL[5:0]	**OUTCTL*n* Bits** — These bits enable software control of their corresponding PWM output. When OUTCTL*n* is set, the OUT*n* bit directly controls the level of the PWM*n* output. **Note:** OUTCTL*n* is buffered if ENCE is set. The value written does not take effect until the next commutation cycle begins. Reading OUTCTLn returns the value in the buffer and not necessarily the value the output control is currently using. When operating the PWM in complementary mode, these bits must be switched in pairs for proper operation. That is OUTCTL0 and OUTCTL1 must have the same value; OUTCTL2 and OUTCTL3 must have the same value; and OUTCTL4 and OUTCTL5 must have the same value. 0 Software control disabled 1 Software control enabled *n* is 0, 1, 2, 3, 4 and 5.

14.3.2.11 PMF Output Control Bit Register (PMFOUTB)

Address: Module Base + 0x000D Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-14. PMF Output Control Bit Register (PMFOUTB)

1. Read: Anytime, if no FAULT input is enabled to be driven from an asynchronous source (FEN=1 and QSMP=0), otherwise only if OUTCTL*n*=1, else read data are undefined. (Note: Use PMFOUTC value to mask related read data bits.)
Write: Anytime

Table 14-16. PMFOUTB Field Descriptions

Field	Description
5–0 OUT[5:0]	**OUT*n* Bits** — If the corresponding OUTCTL*n* bit is set, these bits control the PWM outputs, illustrated in Table 14-17. **Note:** OUT*n* is buffered if ENCE is set. The value written does not take effect until the next commutation cycle begins. Reading OUT*n* (with OUTCTL*n*=1) returns the value in the buffer and not necessarily the value the output control is currently using. *n* is 0, 1, 2, 3, 4 and 5.

Table 14-17. Software Output Control

OUT*n* Bit	Complementary Channel Operation	Independent Channel Operation
OUT0	1 — PWM0 is active 0 — PWM0 is inactive	1 — PWM0 is active 0 — PWM0 is inactive
OUT1	1 — PWM1 is complement of PWM0 0 — PWM1 is inactive	1 — PWM1 is active 0 — PWM1 is inactive
OUT2	1 — PWM2 is active 0 — PWM2 is inactive	1 — PWM2 is active 0 — PWM2 is inactive
OUT3	1 — PWM3 is complement of PWM2 0 — PWM3 is inactive	1 — PWM3 is active 0 — PWM3 is inactive
OUT4	1 — PWM4 is active 0 — PWM4 is inactive	1 — PWM4 is active 0 — PWM4 is inactive
OUT5	1 — PWM5 is complement of PWM4 0 — PWM5 is inactive	1 — PWM5 is active 0 — PWM5 is inactive

14.3.2.12 PMF Deadtime Sample Register (PMFDTMS)

Address: Module Base + 0x000E Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-15. PMF Deadtime Sample Register (PMFDTMS)

1. Read: Anytime
   Write: Never

Table 14-18. PMFDTMS Field Descriptions

Field	Description
5–0 DT[5:0]	**DT*n* Bits** — The DT*n* bits are grouped in pairs, DT0 and DT1, DT2 and DT3, DT4 and DT5. Each pair reflects the corresponding $\overline{\text{IS}}$ input value as sampled at the end of deadtime. *n* is 0, 1, 2, 3, 4 and 5.

14.3.2.13 PMF Correction Control Register (PMFCCTL)

Address: Module Base + 0x000F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-16. PMF Correction Control Register (PMFCCTL)

1. Read: Anytime
   Write: Anytime

Table 14-19. PMFCCTL Field Descriptions

Field	Description
5–4 ISENS[1:0]	**Current Status Sensing Method** — This field selects the top/bottom correction scheme, illustrated in Table 14-20. **Note:** The user must provide current sensing circuitry causing the voltage at the corresponding input to be low for positive current and high for negative current. The top PWMs are PWM 0, 2, and 4 and the bottom PWMs are PWM 1, 3, and 5. **Note:** The ISENS bits are not buffered. Changing the current status sensing method can affect the present PWM cycle.
2 IPOLC	**Current Polarity** — This buffered bit selects the PMF Value register for PWM4 and PWM5 in top/bottom software correction in complementary mode. 0 PMF Value 4 register in next PWM cycle 1 PMF Value 5 register in next PWM cycle
1 IPOLB	**Current Polarity** — This buffered bit selects the PMF Value register for PWM2 and PWM3 in top/bottom software correction in complementary mode. 0 PMF Value 2 register in next PWM cycle 1 PMF Value 3 register in next PWM cycle
0 IPOLA	**Current Polarity** — This buffered bit selects the PMF Value register for PWM0 and PWM1 in top/bottom software correction in complementary mode. 0 PMF Value 0 register in next PWM cycle 1 PMF Value 1 register in next PWM cycle

Table 14-20. Correction Method Selection

ISENS	Correction Method
00	No correction(1)
01	Manual correction
10	Current status sample correction on inputs $\overline{\text{IS0}}$, $\overline{\text{IS1}}$, and $\overline{\text{IS2}}$ during deadtime(2)
11	Current status sample on inputs $\overline{\text{IS0}}$, $\overline{\text{IS1}}$, and $\overline{\text{IS2}}$(3) At the half cycle in center-aligned operation At the end of the cycle in edge-aligned operation

1. The current status inputs can be used as general purpose input/output ports.
2. The polarity of the related $\overline{\text{IS}}$ input is latched when both the top and bottom PWMs are off. At the 0% and 100% duty cycle boundaries, there is no deadtime, so no new current value is sensed.
3. Current is sensed even with 0% or 100% duty cycle.

NOTE

> The IPOL*x* bits take effect at the beginning of the next PWM cycle, regardless of the state of the LDOK bit or global load OK. Select top/bottom software correction by writing 01 to the current select bits, ISENS[1:0], in the PWM control register. Reading the IPOL*x* bits read the buffered value and not necessarily the value currently in effect.

14.3.2.14 PMF Value 0-5 Register (PMFVAL0-PMFVAL5)

Address: Module Base + 0x0010 PMFVAL0
Module Base + 0x0012 PMFVAL1
Module Base + 0x0014 PMFVAL2
Module Base + 0x0016 PMFVAL3
Module Base + 0x0018 PMFVAL4
Module Base + 0x001A PMFVAL5

Access: User read/write(1)

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

Figure 14-17. PMF Value *n* Register (PMFVAL*n*)

1. Read: Anytime
   Write: Anytime

Table 14-21. PMFVAL*n* Field Descriptions

Field	Description
15–0 PMFVAL*n*	**PMF Value n Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM clock periods. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 14-39**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL*n* is buffered. The value written does not take effect until the related or global load OK bit is set and the next PWM load cycle begins. Reading PMFVAL*n* returns the value in the buffer and not necessarily the value the PWM generator is currently using. *n* is 0, 1, 2, 3, 4 and 5.

14.3.2.15 PMF Reload Overrun Interrupt Enable Register (PMFROIE)

Address: Module Base + 0x001C

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PMFROIEC	PMFROIEB	PMFROIEA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-18. PMF Interrupt Enable Register (PMFROIE)

1. Read: Anytime
   Write: Anytime

Table 14-22. PMFROIE Descriptions

Field	Description
2 PMFROIEC	**Reload Overrun Interrupt Enable C** — 0 Reload Overrun Interrupt C disabled 1 Reload Overrun Interrupt C enabled
1 PMFROIEB	**Reload Overrun Interrupt Enable B** — 0 Reload Overrun Interrupt B disabled 1 Reload Overrun Interrupt B enabled
0 PMFROIEA	**Reload Overrun Interrupt Enable A** — 0 Reload Overrun Interrupt A disabled 1 Reload Overrun Interrupt A enabled

14.3.2.16 PMF Interrupt Flag Register (PMFROIF)

Address: Module Base + 0x001D

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	PMFROIFC	PMFROIFB	PMFROIFA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-19. PMF Interrupt Flag Register (PMFROIF)

1. Read: Anytime
   Write: Anytime. Write 1 to clear.

Table 14-23. PMFROIF Field Descriptions

Field	Description
2 PMFROIFC	**Reload Overrun Interrupt Flag C** — If a reload event occurs when the LDOKC or global load OK bit is not set then this flag will be set. 0 No Reload Overrun C occurred 1 Reload Overrun C occurred
1 PMFROIFB	**Reload Overrun Interrupt Flag B** — If a reload event occurs when the LDOKB or global load OK bit is not set then this flag will be set. 0 No Reload Overrun B occurred 1 Reload Overrun B occurred
0 PMFROIFA	**Reload Overrun Interrupt Flag A** — If PMFCFG2[REV1:REV0]=01 and a reload event occurs when the LDOKA or global load OK bit is not set then this flag will be set. If PMFCFG2[REV1:REV0]=10 and a reload event occurs when the LDOKB or global load OK bit is not set then this flag will be set. If PMFCFG2[REV1:REV0]=11 and a reload event occurs when the LDOKC or global load OK bit is not set then this flag will be set. If PMFCFG2[REV1:REV0]=00 no flag will be generated. 0 No Reload Overrun A occurred 1 Reload Overrun A occurred

14.3.2.17 PMF Internal Correction Control Register (PMFICCTL)

Address: Module Base + 0x001E Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	PECC	PECB	PECA	ICCC	ICCB	ICCA
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

Figure 14-20. PMF Internal Correction Control Register (PMFICCTL)

1. Read: Anytime
   Write: Anytime

This register is used to control PWM pulse generation for various applications, such as a power-supply phase-shifting application.

ICC*x* bits apply only in center-aligned operation during complementary mode. These control bits determine whether values set in the IPOL*x* bits control or the whether PWM count direction controls which PWM value register is used.

NOTE

> The ICC*x* bits are buffered. The value written does not take effect until the next PWM load cycle begins regardless of the state of the LDOK bit or global load OK. Reading ICC*x* returns the value in a buffer and not necessarily the value the PWM generator is currently using.

The PEC*x* bits apply in edge-aligned and center-aligned operation during complementary mode. Setting the PEC*x* bits overrides the ICC*x* settings. This allows the PWM pulses generated by both the odd and even PWM value registers to be ANDed together prior to the complementary logic and deadtime insertion.

NOTE

The PEC*x* bits are buffered. The value written does not take effect until the related LDOK bit or global load OK is set and the next PWM load cycle begins. Reading PEC*n* returns the value in a buffer and not necessarily the value the PWM generator is currently using.

Figure 14-21. PMF Internal Correction Control Register (PMFICCTL) Descriptions

Field	Description
5 PECC	**Pulse Edge Control** — This bit controls PWM4/PWM5 pair. 0 Normal operation 1 Allow one of PMFVAL4 and PMFVAL5 to activate the PWM pulse and the other to deactivate the pulse
4 PECB	**Pulse Edge Control** — This bit controls PWM2/PWM3 pair. 0 Normal operation 1 Allow one of PMFVAL2 and PMFVAL3 to activate the PWM pulse and the other to deactivate the pulse
3 PECA	**Pulse Edge Control** — This bit controls PWM0/PWM1 pair. 0 Normal operation 1 Allow one of PMFVAL0 and PMFVAL1 to activate the PWM pulse and the other to deactivate the pulse
2 ICCC	**Internal Correction Control** — This bit controls PWM4/PWM5 pair. 0 IPOLC setting determines whether to use the PMFVAL4 or PMFVAL5 register 1 Use PMFVAL4 register when the PWM counter is counting up. Use PMFVAL5 register when counting down.
1 ICCB	**Internal Correction Control** — This bit controls PWM2/PWM3 pair. 0 IPOLB setting determines whether to use the PMFVAL2 or PMFVAL3 register 1 Use PMFVAL2 register when the PWM counter is counting up. Use PMFVAL3 register when counting down.
0 ICCA	**Internal Correction Control** — This bit controls PWM0/PWM1 pair. 0 IPOLA setting determines whether to use the PMFVAL0 or PMFVAL1 register 1 Use PMFVAL0 register when the PWM counter is counting up. Use PMFVAL1 register when counting down.

14.3.2.18 PMF Compare Invert Register (PMFCINV)

Address: Module Base + 0x001F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	CINV5	CINV4	CINV3	CINV2	CINV1	CINV0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-22. PMF Compare Invert Register (PMFCINV)

1. Read: Anytime
   Write: Anytime

Figure 14-23. PMF Compare Invert Register (PMFCINV) Descriptions

Field	Description
5 CINV5	**PWM Compare Invert 5** — This bit controls the polarity of PWM compare output 5. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 5 is high when PMFCNTC (PMFCNTA if MTG=1) is less than PMFVAL5 1 PWM output 5 is high when PMFCNTC (PMFCNTA if MTG=1) is greater than PMFVAL5
4 CINV4	**PWM Compare Invert 4** — This bit controls the polarity of PWM compare output 4. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 4 is high when PMFCNTC (PMFCNTA if MTG=1) is less than PMFVAL4 1 PWM output 4 is high when PMFCNTC (PMFCNTA if MTG=1) is greater than PMFVAL4
3 CINV3	**PWM Compare Invert 3** — This bit controls the polarity of PWM compare output 3. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 3 is high when PMFCNTB (PMFCNTA if MTG=1) is less than PMFVAL3 1 PWM output 3 is high when PMFCNTB (PMFCNTA if MTG=1) is greater than PMFVAL3
2 CINV2	**PWM Compare Invert 2** — This bit controls the polarity of PWM compare output 2. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 2 is high when PMFCNTB (PMFCNTA if MTG=1) is less than PMFVAL2 1 PWM output 2 is high when PMFCNTB (PMFCNTA if MTG=1) is greater than PMFVAL2
1 CINV1	**PWM Compare Invert 1** — This bit controls the polarity of PWM compare output 1. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 1 is high when PMFCNTA is less than PMFVAL1 1 PWM output 1 is high when PMFCNTA is greater than PMFVAL1.
0 CINV0	**PWM Compare Invert 0** — This bit controls the polarity of PWM compare output 0. Please see the output operations in Figure 14-42 and Figure 14-43. 0 PWM output 0 is high when PMFCNTA is less than PMFVAL0. 1 PWM output 0 is high when PMFCNTA is greater than PMFVAL0

NOTE

Changing the CINV*n* status can affect the PWM output if the related PWM channel is disabled. Similarly on an enabled PWM channel the present PWM cycle can be affected, if the related PMFVAL*n* is zero.

14.3.2.19 PMF Enable Control A Register (PMFENCA)

Address: Module Base + 0x0020

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PWMENA	GLDOKA	0	0	0	RSTRTA	LDOKA	PWMRIEA
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-24. PMF Enable Control A Register (PMFENCA)

1. Read: Anytime
   Write: Anytime except GLDOKA and RSTRTA which cannot be modified after the WP bit is set.

Table 14-24. PMFENCA Field Descriptions

Field	Description
7 PWMENA	**PWM Generator A Enable** — When MTG is clear, this bit when set enables the PWM generators A, B and C and PWM0–5 outputs. When PWMENA is clear, PWM generators A, B and C are disabled, and the PWM0–5 outputs are in their inactive states unless the corresponding OUTCTL bits are set. When MTG is set, this bit when set enables the PWM generator A and the PWM0 and PWM1 outputs.When PWMENA is clear, the PWM generator A is disabled and PWM0 and PWM1 outputs are in their inactive states unless the OUTCTL0 and OUTCTL1 bits are set. After setting this bit a reload event is generated at the beginning of the PWM cycle. 0 PWM generator A and PWM0-1 (2–5 if MTG = 0) outputs disabled unless the respective OUTCTL bit is set 1 PWM generator A and PWM0-1 (2–5 if MTG = 0) outputs enabled
6 GLDOKA	**Global Load Okay A** — When this bit is set, a PMF external global load OK defined on device level replaces the function of LDOKA. This bit cannot be modified after the WP bit is set. 0 LDOKA controls reload of double buffered registers 1 PMF external global load OK controls reload of double buffered registers
2 RSTRTA	**Restart Generator A** — When this bit is set, PWM generator A will be restarted at the next commutation event. This bit cannot be modified after the WP bit is set. 0 No PWM generator A restart at the next commutation event. 1 PWM generator A restarts at the next commutation event
1 LDOKA	**Load Okay A** — When MTG is clear, this bit allows loads of the PRSCA bits, the PMFMODA register, and the PMFVAL0-5 registers into a set of buffers. The buffered prescaler A divisor, PWM counter modulus A value, and all PWM pulse widths take effect at the next PWM reload. When MTG is set, this bit allows loads of the PRSCA bits, the PMFMODA register, and the PMFVAL0–1 registers into a set of buffers. The buffered prescaler divisor A, PWM counter modulus A value, and PWM0–1 pulse widths take effect at the next PWM reload. Set LDOKA by reading it when it is logic zero and then writing a logic one to it. LDOKA is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKA. 0 Do not load new modulus A, prescaler A, and PWM0–1 (2–5 if MTG = 0) values 1 Load prescaler A, modulus A, and PWM0–1 (2–5 if MTG = 0) values **Note:** Do not set PWMENA bit before setting the LDOKA bit and do not clear the LDOKA bit at the same time as setting the PWMENA bit.
0 PWMRIEA	**PWM Reload Interrupt Enable A** — This bit enables the PWMRFA flag to generate CPU interrupt requests. 0 PWMRFA CPU interrupt requests disabled 1 PWMRFA CPU interrupt requests enabled

14.3.2.20 PMF Frequency Control A Register (PMFFQCA)

Address: Module Base + 0x0021 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	LDFQA				HALFA	PRSCA		PWMRFA
Reset	0	0	0	0	0	0	0	0

Figure 14-25. PMF Frequency Control A Register (PMFFQCA)

1. Read: Anytime
   Write: Anytime

Table 14-25. PMFFQCA Field Descriptions

Field	Description
7–4 LDFQA[3:0]	**Load Frequency A** — This field selects the PWM load frequency according to Table 14-26. See Section 14.4.12.3, “Load Frequency” for more details. **Note:** The LDFQA field takes effect when the current load cycle is complete, regardless of the state of the LDOKA bit or global load OK. Reading the LDFQA field reads the buffered value and not necessarily the value currently in effect.
3 HALFA	**Half Cycle Reload A** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. It takes effect immediately. When set, reload opportunities occur also when the counter matches the modulus in addition to the start of the PWM period at count zero. See Section 14.4.12.3, “Load Frequency” for more details. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCA[1:0]	**Prescaler A** — This buffered field selects the PWM clock frequency illustrated in Table 14-27. **Note:** Reading the PRSCA field reads the buffered value and not necessarily the value currently in effect. The PRSCA field takes effect at the beginning of the next PWM cycle and only when the LDOKA bit or global load OK is set.
0 PWMRFA	**PWM Reload Flag A** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKA bit or global load OK. Clear PWMRFA by reading PMFFQCA with PWMRFA set and then writing a logic one to the PWMRFA bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFA has no effect. 0 No new reload cycle since last PWMRFA clearing 1 New reload cycle since last PWMRFA clearing **Note:** Clearing PWMRFA satisfies pending PWMRFA CPU interrupt requests.

Table 14-26. PWM Reload Frequency A

LDFQA[3:0]	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 14-27. PWM Prescaler A

PRSCA[1:0]	Prescaler Value P_A	PWM Clock Frequency f_{PWM_A}
00	1	f_{core}
01	2	$f_{core}/2$
10	4	$f_{core}/4$
11	8	$f_{core}/8$

14.3.2.21 PMF Counter A Register (PMFCNTA)

Address: Module Base + 0x0022 Access: User read/write[1]

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

Figure 14-26. PMF Counter A Register (PMFCNTA)

1. Read: Anytime
   Write: Never

This register displays the state of the 15-bit PWM A counter.

14.3.2.22 PMF Counter Modulo A Register (PMFMODA)

Address: Module Base + 0x0024 Access: User read/write[1]

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

Figure 14-27. PMF Counter Modulo A Register (PMFMODA)

1. Read: Anytime
   Write: Anytime. Do not write a modulus value of zero for center-aligned operation. Do not write a modulus of zero or one in edge-aligned mode.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods.

NOTE

> The PWM counter modulo register is buffered. The value written does not take effect until the LDOKA bit or global load OK is set and the next PWM load cycle begins. Reading PMFMODA returns the value in the buffer. It is not necessarily the value the PWM generator A is currently using.

14.3.2.23 PMF Deadtime A Register (PMFDTMA)

Address: Module Base + 0x0026 Access: User read/write[1]

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

Figure 14-28. PMF Deadtime A Register (PMFDTMA)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to the maximum value of 0x0FFF, selecting a deadtime of 4095 PWM clock cycles. Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows:

$$T_{DEAD\_A} = PMFDTMA / f_{PWM\_A} = PMFDTMA \times P_A \times T_{core} \qquad \textit{Eqn. 14-1}$$

14.3.2.24 PMF Enable Control B Register (PMFENCB)

Address: Module Base + 0x0028 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PWMENB	GLDOKB	0	0	0	RSTRTB	LDOKB	PWMRIEB
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-29. PMF Enable Control B Register (PMFENCB)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set.GLDOKB and RSTRTB cannot be modified after the WP bit is set.

Table 14-28. PMFENCB Field Descriptions

Field	Description
7 PWMENB	**PWM Generator B Enable** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit when set enables the PWM generator B and the PWM2 and PWM3 outputs. When PWMENB is clear, PWM generator B is disabled, and the PWM2 and PWM3 outputs are in their inactive states unless the corresponding OUTCTL bits are set. After setting this bit a reload event is generated at the beginning of the PWM cycle. 0 PWM generator B and PWM2–3 outputs disabled unless the respective OUTCTL bit is set 1 PWM generator B and PWM2–3 outputs enabled
6 GLDOKB	**Global Load Okay B** — When this bit is set, a PMF external global load OK defined on device level replaces the function of LDOKB. This bit cannot be modified after the WP bit is set. 0 LDOKB controls double reload of buffered registers 1 PMF external global load OK controls reload of double buffered registers
2 RSTRTB	**Restart Generator B** — When this bit is set, PWM generator B will be restarted at the next commutation event. This bit cannot be modified after the WP bit is set. 0 No PWM generator B restart at the next commutation event 1 PWM generator B restart at the next commutation event
1 LDOKB	**Load Okay B** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit loads the PRSCB bits, the PMFMODB register and the PMFVAL2-3 registers into a set of buffers. The buffered prescaler divisor B, PWM counter modulus B value, PWM2–3 pulse widths take effect at the next PWM reload. Set LDOKB by reading it when it is logic zero and then writing a logic one to it. LDOKB is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKB. 0 Do not load new modulus B, prescaler B, and PWM2–3 values 1 Load prescaler B, modulus B, and PWM2–3 values **Note:** Do not set PWMENB bit before setting the LDOKB bit and do not clear the LDOKB bit at the same time as setting the PWMENB bit.
0 PWMRIEB	**PWM Reload Interrupt Enable B** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit enables the PWMRFB flag to generate CPU interrupt requests. 0 PWMRFB CPU interrupt requests disabled 1 PWMRFB CPU interrupt requests enabled

14.3.2.25 PMF Frequency Control B Register (PMFFQCB)

Address: Module Base + 0x0029 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	LDFQB				HALFB	PRSCB		PWMRFB
Reset	0	0	0	0	0	0	0	0

Figure 14-30. PMF Frequency Control B Register (PMFFQCB)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set.

Table 14-29. PMFFQCB Field Descriptions

Field	Description
7–4 LDFQB[3:0]	**Load Frequency B** — This field selects the PWM load frequency according to Table 14-30. See Section 14.4.12.3, "Load Frequency" for more details. **Note:** The LDFQB field takes effect when the current load cycle is complete, regardless of the state of the LDOKB bit or global load OK. Reading the LDFQB field reads the buffered value and not necessarily the value currently in effect.
3 HALFB	**Half Cycle Reload B** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. It takes effect immediately. When set, reload opportunities occur also when the counter matches the modulus in addition to the start of the PWM period at count zero. See Section 14.4.12.3, "Load Frequency" for more details. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCB[1:0]	**Prescaler B** — This buffered field selects the PWM clock frequency illustrated in Table 14-31. **Note:** Reading the PRSCB field reads the buffered value and not necessarily the value currently in effect. The PRSCB field takes effect at the beginning of the next PWM cycle and only when the LDOKB bit or global load OK is set.
0 PWMRFB	**PWM Reload Flag B** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKB bit. Clear PWMRFB by reading PMFFQCB with PWMRFB set and then writing a logic one to the PWMRFB bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFB has no effect. 0 No new reload cycle since last PWMRFB clearing 1 New reload cycle since last PWMRFB clearing **Note:** Clearing PWMRFB satisfies pending PWMRFB CPU interrupt requests.

Table 14-30. PWM Reload Frequency B

LDFQB[3:0]	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities

Table 14-30. PWM Reload Frequency B

LDFQB[3:0]	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 14-31. PWM Prescaler B

PRSCB[1:0]	Prescaler Value P_B	PWM Clock Frequency f_{PWM_B}
00	1	f_{core}
01	2	$f_{core}/2$
10	4	$f_{core}/4$
11	8	$f_{core}/8$

14.3.2.26 PMF Counter B Register (PMFCNTB)

Address: Module Base + 0x002A Access: User read/write[1]

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

Figure 14-31. PMF Counter B Register (PMFCNTB)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Never

This register displays the state of the 15-bit PWM B counter.

14.3.2.27 PMF Counter Modulo B Register (PMFMODB)

Address: Module Base + 0x002C Access: User read/write[1]

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

Figure 14-32. PMF Counter Modulo B Register (PMFMODB)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set.Do not write a modulus value of zero for center-aligned operation. Do not write a modulus of zero or one in edge-aligned mode.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods.

NOTE

> The PWM counter modulo register is buffered. The value written does not take effect until the LDOKB bit or global load OK is set and the next PWM load cycle begins. Reading PMFMODB returns the value in the buffer. It is not necessarily the value the PWM generator B is currently using.

14.3.2.28 PMF Deadtime B Register (PMFDTMB)

Address: Module Base + 0x002E

Access: User read/write(1)

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

Figure 14-33. PMF Deadtime B Register (PMFDTMB)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set. This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to the maximum value of 0x0FFF, selecting a deadtime of 4095 PWM clock cycles. Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows:

$$T_{DEAD\_B} = PMFDTMB / f_{PWM\_B} = PMFDTMB \times P_B \times T_{core} \qquad \textit{Eqn. 14-2}$$

14.3.2.29 PMF Enable Control C Register (PMFENCC)

Address: Module Base + 0x0030

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PWMENC	GLDOKC	0	0	0	RSTRTC	LDOKC	PWMRIEC
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-34. PMF Enable Control C Register (PMFENCC)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set. GLDOKC and RSTRTC cannot be modified after the WP bit is set.

Table 14-32. PMFENCC Field Descriptions

Field	Description
7 PWMENC	**PWM Generator C Enable** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit when set enables the PWM generator C and the PWM4 and PWM5 outputs. When PWMENC is clear, PWM generator C is disabled, and the PWM4 and PWM5 outputs are in their inactive states unless the corresponding OUTCTL bits are set. After setting this bit a reload event is generated at the beginning of the PWM cycle. 0 PWM generator C and PWM4–5 outputs disabled unless the respective OUTCTL bit is set 1 PWM generator C and PWM4–5 outputs enabled
6 GLDOKC	**Global Load Okay C** — When this bit is set, a PMF external global load OK defined on device level replaces the function of LDOKC. This bit cannot be modified after the WP bit is set. 0 LDOKC controls reload of double buffered registers 1 PMF external global load OK controls reload of double buffered registers
2 RSTRTC	**Restart Generator C** — When this bit is set, PWM generator C will be restarted at the next commutation event. This bit cannot be modified after the WP bit is set. 0 No PWM generator C restart at the next commutation event 1 PWM generator C restart at the next commutation event

Table 14-32. PMFENCC Field Descriptions (continued)

Field	Description
1 LDOKC	**Load Okay C** — If MTG is clear, this bit reads zero and can not be written. If MTG is set, this bit loads the PRSCC bits, the PMFMODC register and the PMFVAL4–5 registers into a set of buffers. The buffered prescaler divisor C, PWM counter modulus C value, PWM4–5 pulse widths take effect at the next PWM reload. Set LDOKC by reading it when it is logic zero and then writing a logic one to it. LDOKC is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKC. 0 Do not load new modulus C, prescaler C, and PWM4–5 values 1 Load prescaler C, modulus C, and PWM4–5 values **Note:** Do not set PWMENC bit before setting the LDOKC bit and do not clear the LDOKC bit at the same time as setting the PWMENC bit.
0 PWMRIEC	**PWM Reload Interrupt Enable C** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit enables the PWMRFC flag to generate CPU interrupt requests. 0 PWMRFC CPU interrupt requests disabled 1 PWMRFC CPU interrupt requests enabled

14.3.2.30 PMF Frequency Control C Register (PMFFQCC)

Address: Module Base + 0x0031 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	LDFQC				HALFC	PRSCC		PWMRFC
Reset	0	0	0	0	0	0	0	0

Figure 14-35. PMF Frequency Control C Register (PMFFQCC)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set.

Table 14-33. PMFFQCC Field Descriptions

Field	Description
7–4 LDFQC[3:0]	**Load Frequency C** — This field selects the PWM load frequency according to Table 14-34. See Section 14.4.12.3, “Load Frequency” for more details. **Note:** The LDFQC field takes effect when the current load cycle is complete, regardless of the state of the LDOKC bit or global load OK. Reading the LDFQC field reads the buffered value and not necessarily the value currently in effect.
3 HALFC	**Half Cycle Reload C** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. It takes effect immediately. When set, reload opportunities occur also when the counter matches the modulus in addition to the start of the PWM period at count zero. See Section 14.4.12.3, “Load Frequency” for more details. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCC[1:0]	**Prescaler C** — This buffered field selects the PWM clock frequency illustrated in Table 14-35. **Note:** Reading the PRSCC field reads the buffered value and not necessarily the value currently in effect. The PRSCC field takes effect at the beginning of the next PWM cycle and only when the LDOKC bit or global load OK is set.

Table 14-33. PMFFQCC Field Descriptions (continued)

Field	Description
0 PWMRFC	**PWM Reload Flag C** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKC bit or global load OK. Clear PWMRFC by reading PMFFQCC with PWMRFC set and then writing a logic one to the PWMRFC bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFC has no effect. 0 No new reload cycle since last PWMRFC clearing 1 New reload cycle since last PWMRFC clearing **Note:** Clearing PWMRFC satisfies pending PWMRFC CPU interrupt requests.

Table 14-34. PWM Reload Frequency C

LDFQC[3:0]	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 14-35. PWM Prescaler C

PRSCC[1:0]	Prescaler Value P_C	PWM Clock Frequency f_{PWM_C}
00	1	f_{core}
01	2	$f_{core}/2$
10	4	$f_{core}/4$
11	8	$f_{core}/8$

14.3.2.31 PMF Counter C Register (PMFCNTC)

Address: Module Base + 0x0032

Access: User read/write(1)

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

Figure 14-36. PMF Counter C Register (PMFCNTC)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Never

This register displays the state of the 15-bit PWM C counter.

14.3.2.32 PMF Counter Modulo C Register (PMFMODC)

Address: Module Base + 0x0034 Access: User read/write[1]

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

Figure 14-37. PMF Counter Modulo C Register (PMFMODC)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set. Do not write a modulus value of zero for center-aligned operation. Do not write a modulus of zero or one in edge-aligned mode.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods.

NOTE

> The PWM counter modulo register is buffered. The value written does not take effect until the LDOKC bit or global load OK is set and the next PWM load cycle begins. Reading PMFMODC returns the value in the buffer. It is not necessarily the value the PWM generator A is currently using.

14.3.2.33 PMF Deadtime C Register (PMFDTMC)

Address: Module Base + 0x0036 Access: User read/write[1]

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

Figure 14-38. PMF Deadtime C Register (PMFDTMC)

1. Read: Anytime. Returns zero if MTG is clear.
   Write: Anytime if MTG is set.This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to the maximum value of 0x0FFF, selecting a deadtime of 4095 PWM clock cycles. Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows:

$$T_{DEAD\_C} = PMFDTMC / f_{PWM\_C} = PMFDTMC \times P_C \times T_{core} \qquad \textit{Eqn. 14-3}$$

14.3.2.34 PMF Disable Mapping Registers (PMFDMP0-5)

Address: Module Base + 0x0038 PMFDMP0
Module Base + 0x0039 PMFDMP1
Module Base + 0x003A PMFDMP2
Module Base + 0x003B PMFDMP3
Module Base + 0x003C PMFDMP4
Module Base + 0x003D PMFDMP5

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R W	DMP*n*5		DMP*n*4		DMP*n*3	DMP*n*2	DMP*n*1	DMP*n*0
Reset	0	0	0	0	0	0	0	0

Figure 14-39. PMF Disable Mapping Register (PMFDMP0-5)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set.

Table 14-36. PMFDMP0-5 Field Descriptions

Field	Description
7-6 DMP*n*5	**PWM Disable Mapping Channel** *n* **FAULT5** — This bit selects for PWM*n* whether the output is disabled or forced to OUTF*n* at a FAULT5 event. Disabling PWM*n* has priority over forcing PWM*n* to OUTF*n*. This register cannot be modified after the WP bit is set. 00 PWM*n* unaffected by FAULT5 event (interrupt flag setting only) 01 PWM*n* unaffected by FAULT5 event (interrupt flag setting only) 10 PWM*n* disabled on FAULT5 event 11 PWM*n* forced to OUTF*n* on FAULT5 event *n* is 0, 1, 2, 3, 4 and 5.
5-4 DMP*n*4	**PWM Disable Mapping Channel** *n* **FAULT4** — This bit selects for PWM*n* whether the output is disabled or forced to OUTF*n* at a FAULT4 event. Disabling PWM*n* has priority over forcing PWM*n* to OUTF*n*. This register cannot be modified after the WP bit is set. 00 PWM*n* unaffected by FAULT4 event (interrupt flag setting only) 01 PWM*n* unaffected by FAULT4 event (interrupt flag setting only) 10 PWM*n* disabled on FAULT4 event 11 PWM*n* forced to OUTF*n* on FAULT4 event *n* is 0, 1, 2, 3, 4 and 5.
3-0 DMP*n*	**PWM Disable Mapping Channel *n* FAULT3-0** — This bit selects for PWM*n* if the output is disabled at a FAULT3-0 event. Disabling PWM*n* has priority over forcing PWM*n* to OUTF*n*. This bit cannot be modified after the WP bit is set. FAULT3-0 have priority over FAULT5-4. 0 PWM*n* unaffected by FAULT3-0 event 1 PWM*n* disabled on FAULT3-0 event *n* is 0, 1, 2, 3, 4 and 5.

14.3.2.35 PMF Output Control on Fault Register (PMFOUTF)

Address: Module Base + 0x003E

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	OUTF5	OUTF4	OUTF3	OUTF2	OUTF1	OUTF0
W								
Reset	0	0	0	0	0	0	0	0

Figure 14-40. PMF Output Control on Fault Register (PMFOUTF)

1. Read: Anytime
   Write: This register cannot be modified after the WP bit is set.

Table 14-37. PMFOUTF Field Descriptions

Field	Description
5–0 OUTF[5:0]	**OUTF Bits** — When the corresponding DMP*n*4 or DMP*n*5 bits are set to switch to output control on a related FAULT4 or FAULT5 event, these bits control the PWM outputs, illustrated in Table 14-38.This register cannot be modified after the WP bit is set.

Table 14-38. Software Output Control on FAULT4 or FAULT5 Event

OUTF*n* Bit	Complementary Channel Operation	Independent Channel Operation
OUTF0	1 — PWM0 is active 0 — PWM0 is inactive	1 — PWM0 is active 0 — PWM0 is inactive
OUTF1	1 — PWM1 is complement of PWM0 0 — PWM1 is inactive	1 — PWM1 is active 0 — PWM1 is inactive
OUTF2	1 — PWM2 is active 0 — PWM2 is inactive	1 — PWM2 is active 0 — PWM2 is inactive
OUTF3	1 — PWM3 is complement of PWM2 0 — PWM3 is inactive	1 — PWM3 is active 0 — PWM3 is inactive
OUTF4	1 — PWM4 is active 0 — PWM4 is inactive	1 — PWM4 is active 0 — PWM4 is inactive
OUTF5	1 — PWM5 is complement of PWM4 0 — PWM5 is inactive	1 — PWM5 is active 0 — PWM5 is inactive

14.4 Functional Description

14.4.1 Block Diagram

A block diagram of the PMF is shown in Figure 14-1**.** The MTG bit allows the use of multiple PWM generators (A, B, and C) or just a single generator (A). PWM0 and PWM1 constitute Pair A, PWM2 and PWM3 constitute Pair B, and PWM4 and PWM5 constitute Pair C.

Figure 14-41 depicts Pair A signal paths of PWM0 and PWM1. Pairs B and C have the same structure.

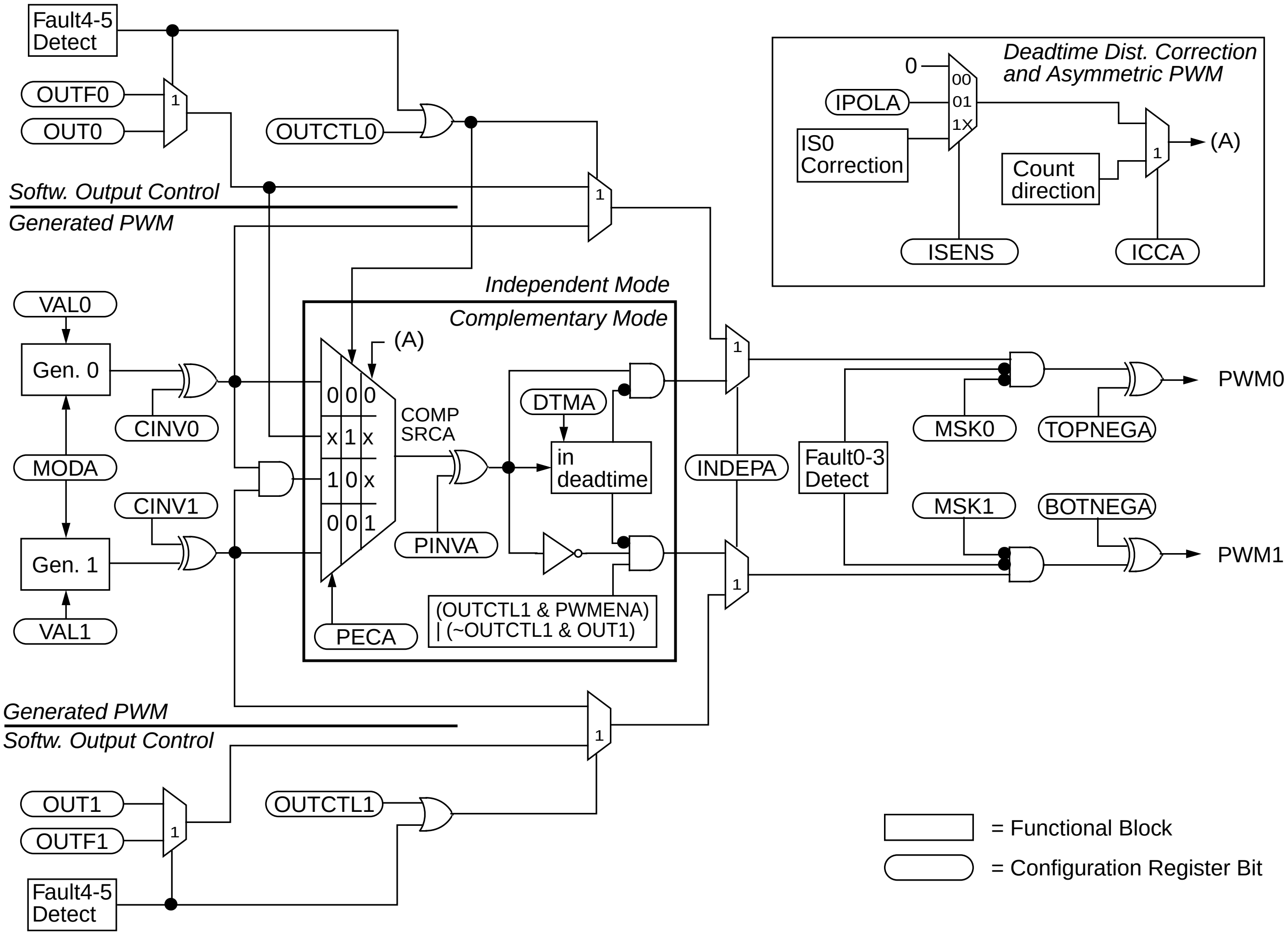


Figure 14-41. Detail of PWM0 and PWM1 Signal Paths

NOTE

It is possible to have both channels of a complementary pair to be high. For example, if the TOPNEGA (negative polarity for PWM0), BOTNEGA (negative polarity for PWM1), MSK0 and MSK1 bits are set, both the PWM complementary outputs of generator A will be high. See Section 14.3.2.2, “PMF Configure 1 Register (PMFCFG1)” for the description of TOPNEG and BOTNEG bits, and Section 14.3.2.3, “PMF Configure 2 Register (PMFCFG2)” for the description of the MSK0 and MSK1 bits.

14.4.2 Prescaler

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the core clock frequency by one, two, four, and eight. Each PWM generator has its own prescaler divisor. Each prescaler is buffered and will not be used by its PWM generator until the corresponding Load OK bit is set and a new PWM reload cycle begins.

14.4.3 PWM Generator

Each PWM generator contains a 15-bit up/down PWM counter producing output signals with software-selectable

- Alignment — The logic state of each pair EDGE bit determines whether the PWM pair outputs are edge-aligned or center-aligned
- Period — The value written to each pair PWM counter modulo register is used to determine the PWM pair period. The period can also be varied by using the prescaler
- With edge-aligned output, the modulus is the period of the PWM output in clock cycles
- With center-aligned output, the modulus is one-half of the PWM output period in clock cycles
- Pulse width — The number written to the PWM value register determines the pulse width duty cycle of the PWM output in clock cycles
  - With center-aligned output, the pulse width is twice the value written to the PWM value register
  - With edge-aligned output, the pulse width is the value written to the PWM value register

14.4.3.1 Alignment and Compare Output Polarity

Each edge-align bit, EDGEx, selects either center-aligned or edge-aligned PWM generator outputs.

PWM compare output polarity is selected by the CINV*n* bit field in the source control (PMFCINV) register. Please see the output operations in Figure 14-42 and Figure 14-43.

The PWM compare output is driven to a high state when the value of PWM value (PMFVAL*n*) register is greater than the value of PWM counter, and PWM compare is counting downwards if the corresponding channel CINV*n*=0. Or, the PWM compare output is driven to low state if the corresponding channel CINV*n*=1.

The PWM compare output is driven to low state when the value of PWM value (PMFVAL*n*) register matches the value of PWM counter, and PWM counter is counting upwards if the corresponding channel CINV*n*=0. Or, the PWM compare output is driven to high state if the corresponding channel CINV*n*=1.

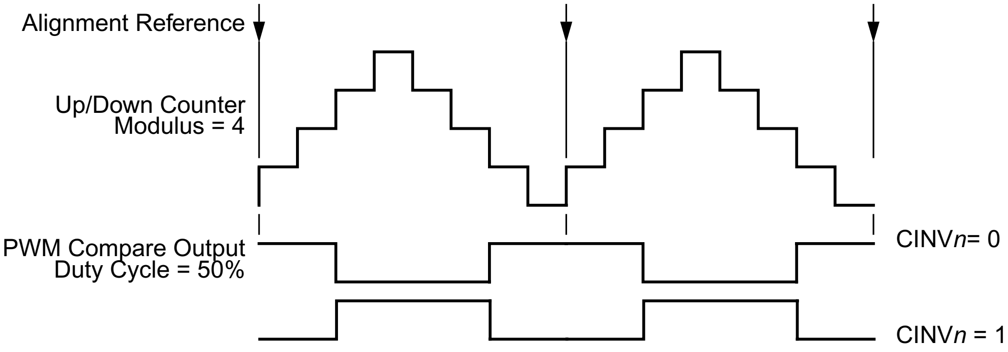


Figure 14-42. Center-Aligned PWM Output

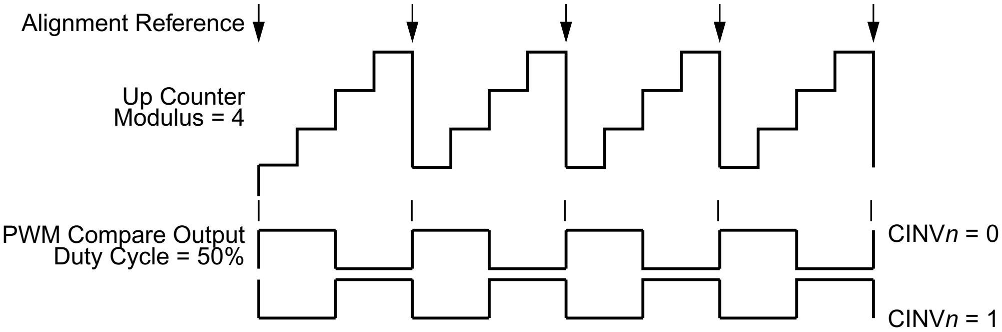


Figure 14-43. Edge-Aligned PWM Output

14.4.3.2 Period

A PWM period is determined by the value written to the PWM counter modulo registers PMFMOD*x*.

The PWM counter is an up/down counter in center-aligned mode. In this mode the PWM highest output resolution is two core clock cycles.

$$\textbf{PWM period = (PWM modulus)} \times \textbf{(PWM clock period)} \times \textbf{2} \qquad \textit{Eqn. 14-4}$$

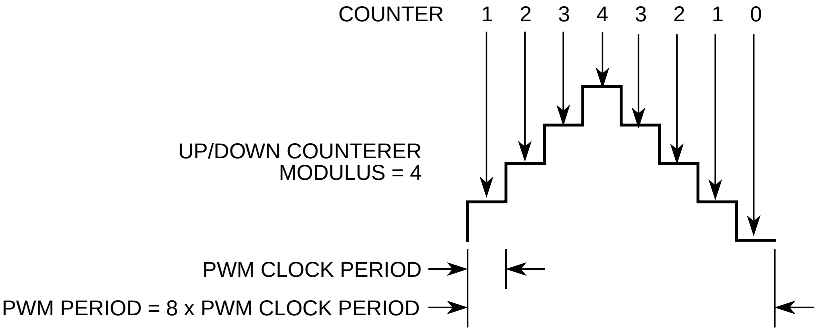


Figure 14-44. Center-Aligned PWM Period

NOTE

Because of the equals-comparator architecture of this PMF, the modulus equals zero case is considered illegal in center-aligned mode. Therefore, the modulus register does not return to zero, and a modulus value of zero will result in waveforms inconsistent with the other modulus waveforms. If a modulus of zero is loaded, the counter will continually count down from 0x7FFF. This operation will not be tested or guaranteed. Consider it illegal. However, the deadtime constraints and fault conditions will still be guaranteed.

In edge-aligned mode, the PWM counter is an up counter. The PWM output resolution is one core clock cycle.

$$\textbf{PWM period} = \textbf{PWM modulus} \times \textbf{PWM clock period} \qquad \textit{Eqn. 14-5}$$

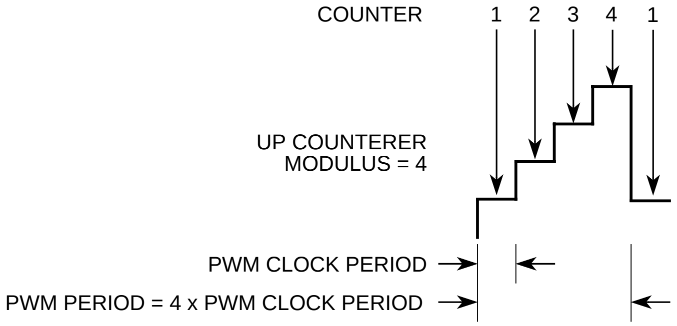


Figure 14-45. Edge-Aligned PWM Period

NOTE

In edge-aligned mode the modulus equals zero and one cases are considered illegal.

14.4.3.3 Duty Cycle

The signed 16-bit number written to the PMF value registers (PMFVAL*n*) is the pulse width in PWM clock periods of the PWM generator output (or period minus the pulse width if CINV*n*=1).

$$\text{Duty cycle} = \frac{\text{PMFVAL}}{\text{PMFMOD}} \times 100$$

NOTE

A PWM value less than or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than or equal to the modulus activates the PWM output for the entire PWM period when CINV*n*=0, and vice versa if CINV*n*=1.

Table 14-39. PWM Value and Underflow Conditions

PMFVALn	Condition	PWM Value Used
0x0000–0x7FFF	Normal	Value in registers
0x8000–0xFFFF	Underflow	0x0000

Center-aligned operation is illustrated in Figure 14-46.

$$\text{PWM pulse width} = (\text{PWM value}) \times (\text{PWM clock period}) \times 2 \qquad \textit{Eqn. 14-6}$$

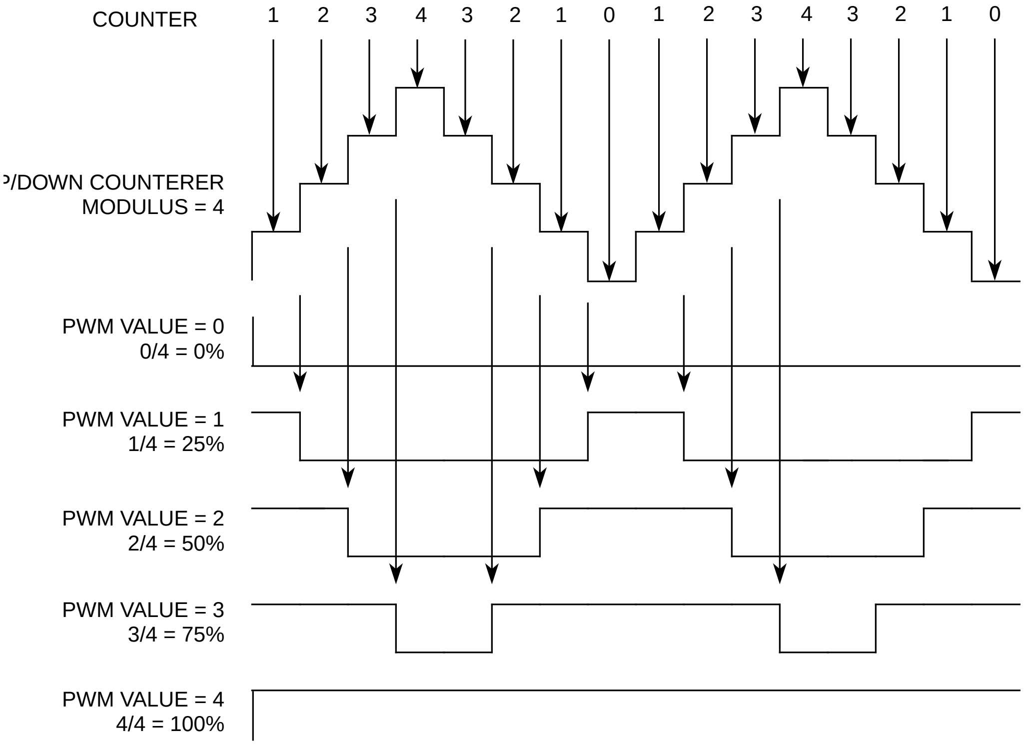


Figure 14-46. Center-Aligned PWM Pulse Width

Edge-aligned operation is illustrated in Figure 14-47.

$$\text{PWM pulse width} = (\text{PWM value}) \times (\text{PWM clock period}) \qquad \textit{Eqn. 14-7}$$

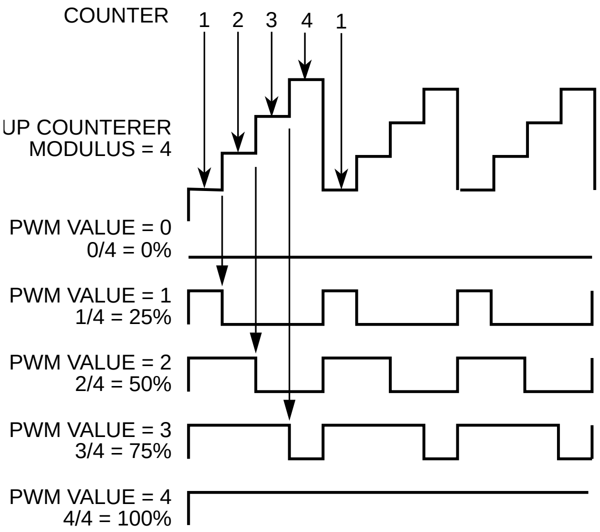


Figure 14-47. Edge-Aligned PWM Pulse Width

14.4.4 Independent or Complementary Channel Operation

Writing a logic one to an INDEP*x* bit configures a pair of the PWM outputs as two independent PWM channels. Each PWM output has its own PWM value register operating independently of the other channels in independent channel operation.

Writing a logic zero to a INDEP*x* bit configures the PWM output as a pair of complementary channels. The PWM outputs are paired as shown in Figure 14-48 in complementary channel operation.

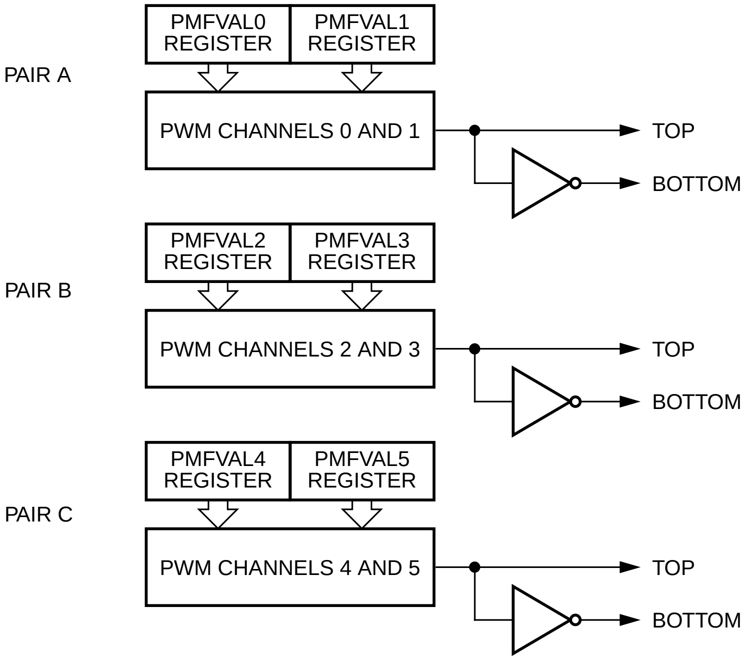


Figure 14-48. Complementary Channel Pairs

The complementary channel operation is for driving top and bottom transistors in a motor drive circuit, such as the one in Figure 14-49.

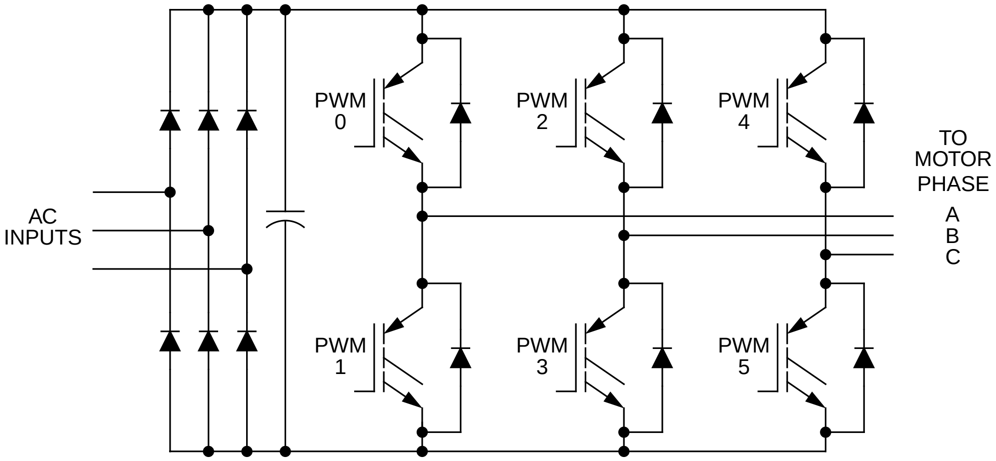


Figure 14-49. Typical 3-Phase AC Motor Drive

In complementary channel operation following additional features exist:

- Deadtime insertion
- Separate top and bottom pulse width correction via current status inputs or software
- Three variants of PWM output:
  - Asymmetric in center-aligned mode
  - Variable edge placement in edge-aligned mode
  - Double switching in center-aligned mode

14.4.5 Deadtime Generators

While in complementary operation, each PWM pair can be used to drive top/bottom transistors, as shown in Figure 14-50. Ideally, the PWM pairs are an inversion of each other. When the top PWM channel is active, the bottom PWM channel is inactive, and vice versa.

NOTE

> To avoid a short-circuit on the DC bus and endangering the transistor, there must be no overlap of conducting intervals between the top and bottom transistor. But the transistor’s characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of the top and bottom transistors, deadtime needs to be inserted in the switching period.

Deadtime generators automatically insert software-selectable activation delays into each pair of PWM outputs. The deadtime register (PMFDTMx) specifies the number of PWM clock cycles to use for deadtime delay. Every time the deadtime generator inputs changes state, deadtime is inserted. Deadtime forces both PWM outputs in the pair to the inactive state.

A method of correcting this, adding to or subtracting from the PWM value used, is discussed next.

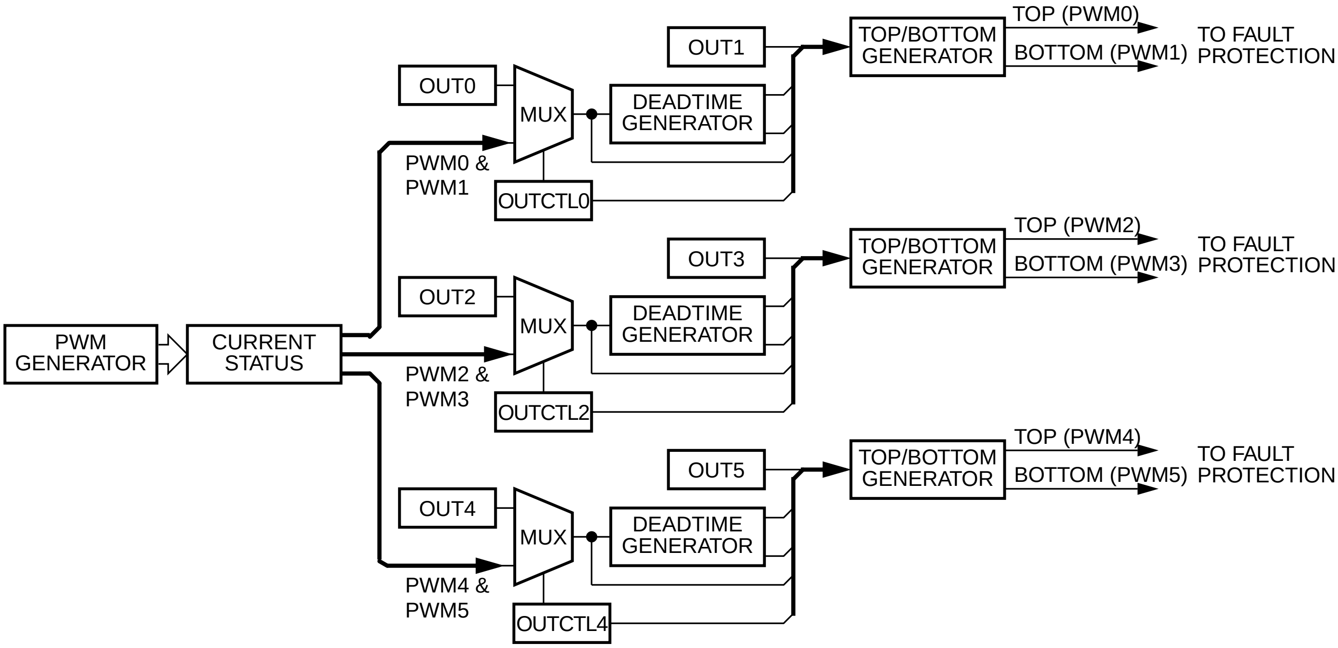


Figure 14-50. Deadtime Generators

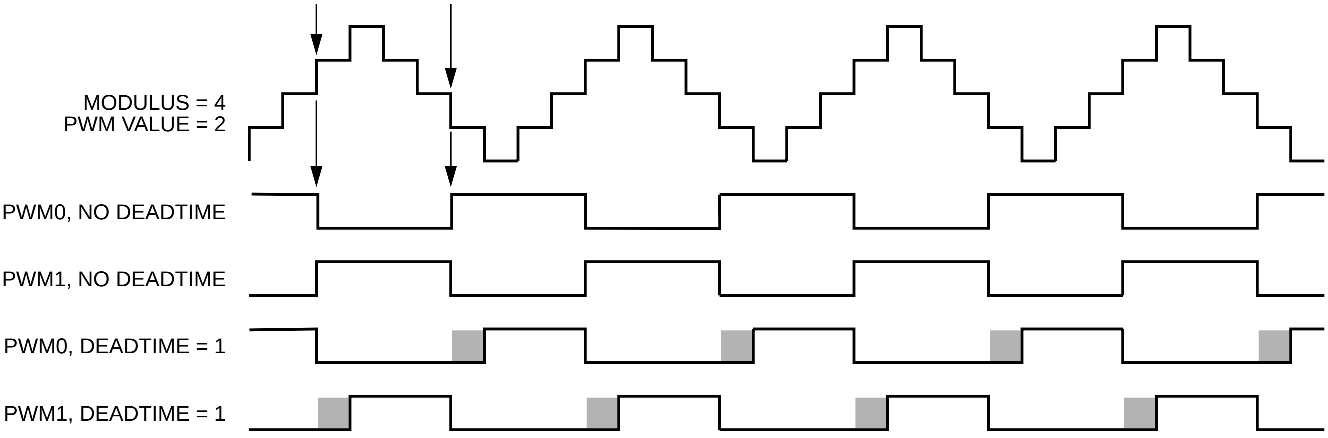


Figure 14-51. Deadtime Insertion, Center Alignment

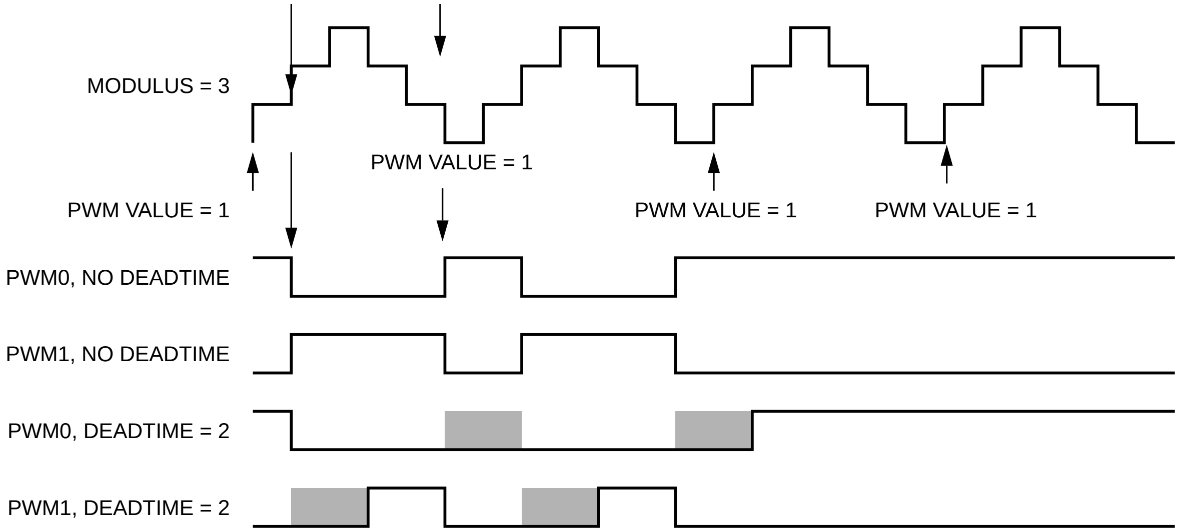


Figure 14-52. Deadtime at Duty Cycle Boundaries

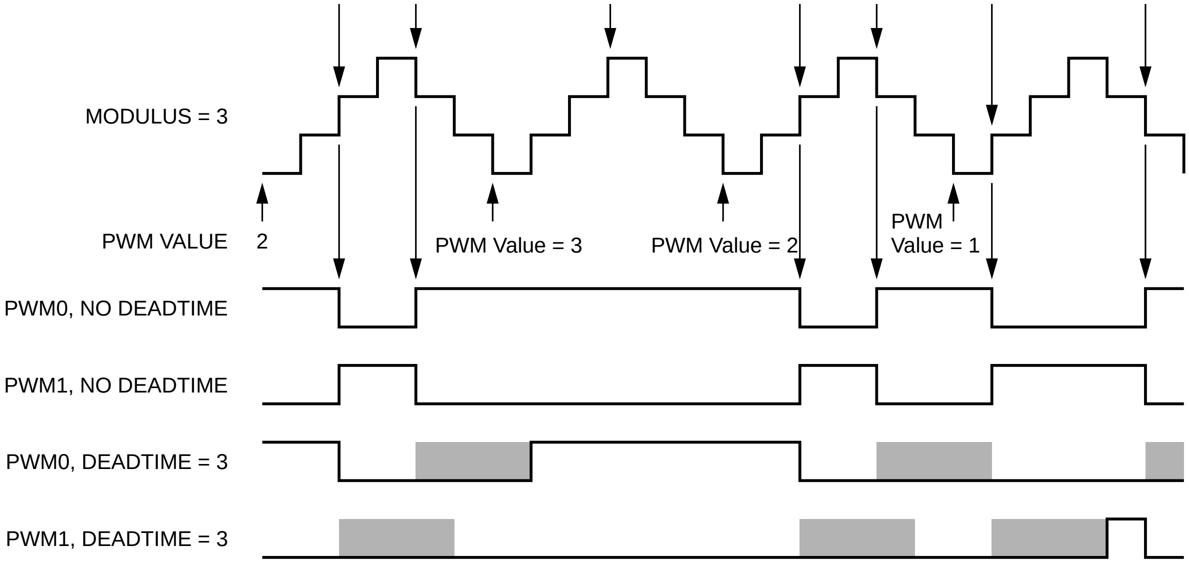


Figure 14-53. Deadtime and Small Pulse Widths

NOTE

The waveform at the output is delayed by two core clock cycles for deadtime insertion.

14.4.6 Top/Bottom Correction

In complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transistor. Both transistors in complementary mode are off during deadtime, allowing the output voltage to be determined by the current status of the load and introduce distortion in the output voltage. See Figure 14-54. On AC induction motors running open-loop, the distortion typically manifests itself as poor low-speed performance, such as torque ripple and rough operation.

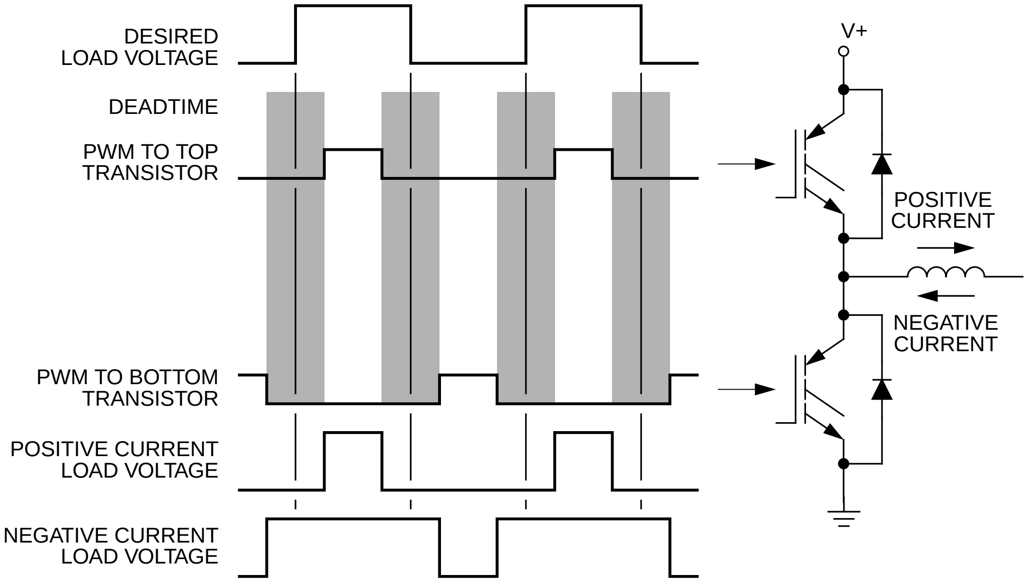


Figure 14-54. Deadtime Distortion

During deadtime, load inductance distorts output voltage by keeping current flowing through the diodes. This deadtime current flow creates a load voltage that varies with current direction. With a positive current flow, the load voltage during deadtime is equal to the bottom supply, putting the top transistor in control. With a negative current flow, the load voltage during deadtime is equal to the top supply putting the bottom transistor in control.

Remembering that the original PWM pulse widths were shortened by deadtime insertion, the averaged sinusoidal output will be less than the desired value. However, when deadtime is inserted, it creates a distortion in motor current waveform. This distortion is aggravated by dissimilar turn-on and turn-off delays of each of the transistors. By giving the PWM module information on which transistor is controlling at a given time, this distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors will be effective in controlling the output voltage at any given time. This depends on the direction of the motor current for that pair. See Figure 14-54. To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values prior to placing them in an odd-numbered/even numbered PWM register pair. Either the odd or the even PMFVAL register controls the pulse width at any given time. For a given PWM pair, whether the odd or even PMFVAL register is active depends on either:

- The state of the current status input, $\overline{\text{IS}}$, for that driver
- The state of the odd/even correction bit, IPOLx, for that driver if ICC bits in the PMFICCTL register are set to zeros
- The direction of PWM counter if ICC bits in the PMFICCTL register are set to ones

To correct deadtime distortion, software can decrease or increase the value in the appropriate PMFVAL register.

- In edge-aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In center-aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.

In the complementary channel operation, ISENS selects one of three correction methods:

- Manual correction
- Automatic current status correction during deadtime
- Automatic current status correction when the PWM counter value equals the value in the PWM counter modulus registers

Table 14-40. Correction Method Selection

ISENS	Correction Method
00	No correction(1)
01	Manual correction
10	Current status sample correction on inputs $\overline{\text{IS0}}$, $\overline{\text{IS1}}$, and $\overline{\text{IS2}}$ during deadtime(2)
11	Current status sample on inputs $\overline{\text{IS0}}$, $\overline{\text{IS1}}$, and $\overline{\text{IS2}}$(3) At the half cycle in center-aligned operation At the end of the cycle in edge-aligned operation

1. The current status inputs can be used as general purpose input/output ports.
2. The polarity of the $\overline{\text{IS}}$ input is latched when both the top and bottom PWMs are off. At the 0% and 100% duty cycle boundaries, there is no deadtime, so no new current value is sensed.
3. Current is sensed even with 0% or 100% duty cycle.

NOTE

External current status sensing circuitry is required at the corresponding inputs which produces a logic zero level for positive current and logic one for negative current. PWM 0, 2, and 4 are considered the top PWMs while the bottom PWMs are PWM 1, 3, and 5.

14.4.6.1 Manual Correction

The IPOL*x* bits select either the odd or the even PWM value registers to use in the next PWM cycle.

Table 14-41. Top/Bottom Manual Correction

Bit	Logic state	Output Control
IPOLA	0	PMFVAL0 controls PWM0/PWM1 pair
	1	PMFVAL1 controls PWM0/PWM1 pair
IPOLB	0	PMFVAL2 controls PWM2/PWM3 pair
	1	PMFVAL3 controls PWM2/PWM3 pair
IPOLC	0	PMFVAL4 controls PWM4/PWM5 pair
	1	PMFVAL5 controls PWM4/PWM5 pair

NOTE

> IPOL*x* bits are buffered so only one PWM register is used per PWM cycle. If an IPOL*x* bit changes during a PWM period, the new value does not take effect until the next PWM period.
>
> IPOL*x* bits take effect at the end of each PWM cycle regardless of the state of the related LDOK bit or global load OK.

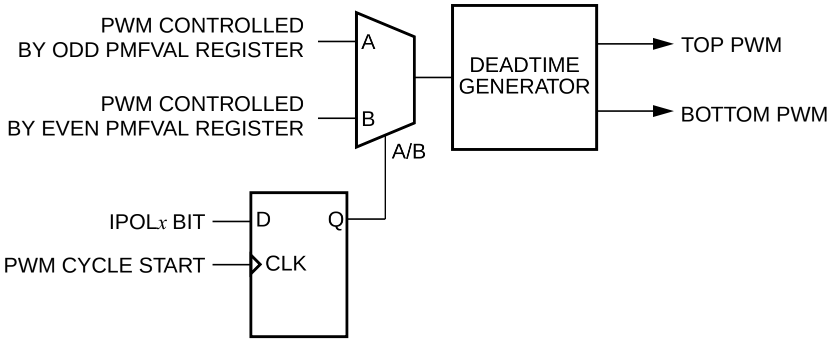


Figure 14-55. Internal Correction Logic when ISENS = 01

To detect the current status, the voltage on each $\overline{\text{IS}}$ input is sampled twice in a PWM period, at the end of each deadtime. The value is stored in the DT*n* bits in the PMF Deadtime Sample register (PMFDTMS). The DT*n* bits are a timing marker especially indicating when to toggle between PWM value registers. Software can then set the IPOL*x* bit to toggle PMFVAL registers according to DT*n* values.

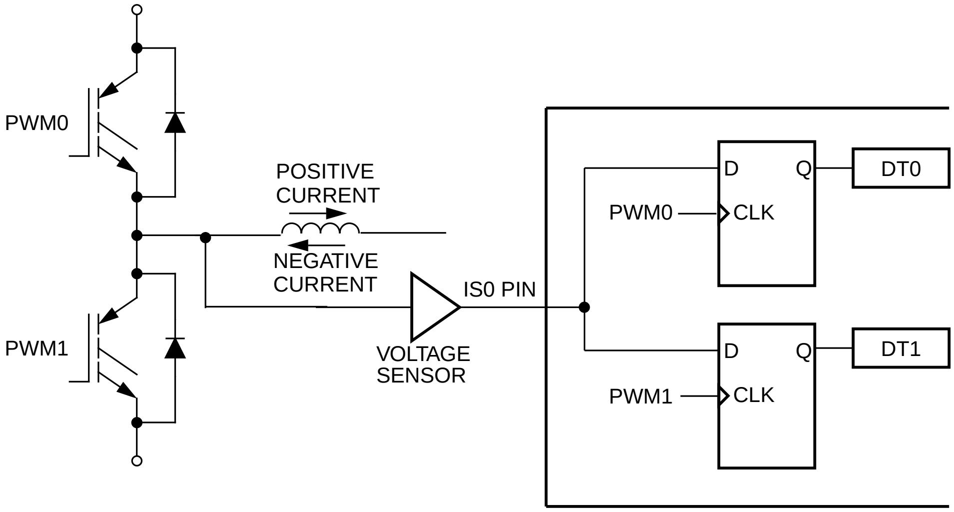


Figure 14-56. Current Status Sense Scheme for Deadtime Correction

Both D flip-flops latch low, DT0 = 0, DT1 = 0, during deadtime periods if current is large and flowing out of the complementary circuit. See Figure 14-56. Both D flip-flops latch the high, DT0 = 1, DT1 = 1, during deadtime periods if current is also large and flowing into the complementary circuit.

However, under low-current, the output voltage of the complementary circuit during deadtime is somewhere between the high and low levels. The current cannot free-wheel through the opposition anti-body diode, regardless of polarity, giving additional distortion when the current crosses zero.

Sampled results will be DT0 = 0 and DT1 = 1. Thus, the best time to change one PWM value register to another is just before the current zero crossing.

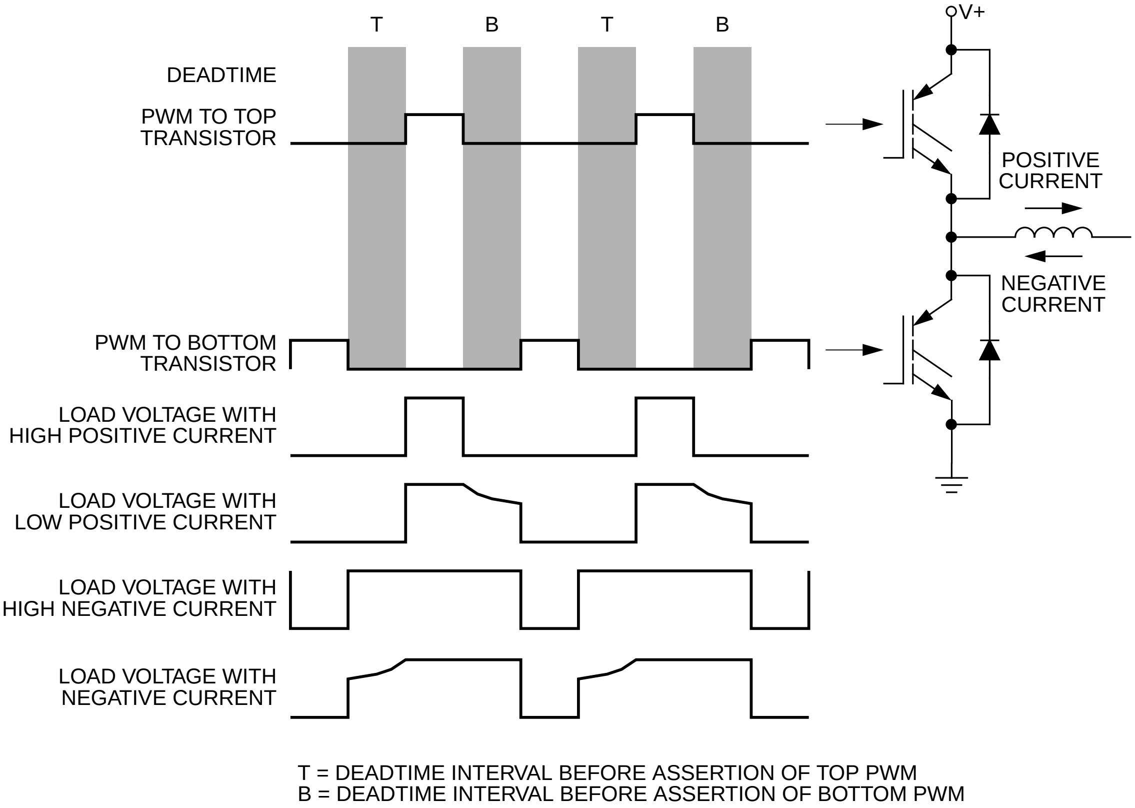


Figure 14-57. Output Voltage Waveforms

14.4.6.2 Current-Sensing Correction

A current sense input, $\overline{\text{IS}}$, for a PWM pair selects either the odd or the even PWM value registers to use in the next PWM cycle. The selection is based on user-provided current sense circuitry driving the related $\overline{\text{IS}}$ input high for negative current and low for positive current.

Table 14-42. Top/Bottom Current Sense Correction

Pin	Logic State	Output Control
IS0	0	PMFVAL0 controls PWM0/PWM1 pair
	1	PMFVAL1 controls PWM0/PWM1 pair
IS1	0	PMFVAL2 controls PWM2/PWM3 pair
	1	PMFVAL3 controls PWM2/PWM3 pair
IS2	0	PMFVAL4 controls PWM4/PWM5 pair
	1	PMFVAL5 controls PWM4/PWM5 pair

Previously shown, the current direction can be determined by the output voltage during deadtime. Thus, a simple external voltage sensor can be used when current status is completed during deadtime, ISENS = 10. Deadtime does not exist at the 100 percent and zero percent duty cycle boundaries. Therefore, the second automatic mode must be used for correction, ISENS = 11, where current status is sampled at the half cycle in center-aligned operation and at the end of cycle in edge-aligned operation. Using this mode requires external circuitry to sense current direction.

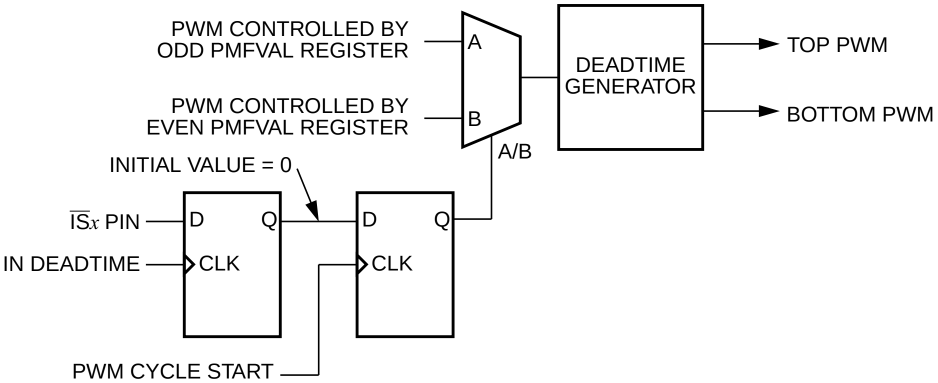


Figure 14-58. Internal Correction Logic when ISENS = 10

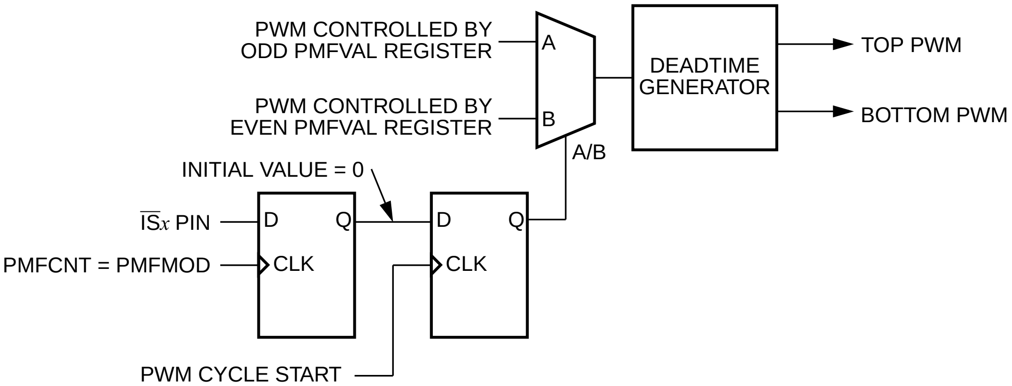


Figure 14-59. Internal Correction Logic when ISENS = 11

NOTE

Values latched on the $\overline{\text{IS}}x$ inputs are buffered so only one PWM register is used per PWM cycle. If a current status changes during a PWM period, the new value does not take effect until the next PWM period.

When initially enabled by setting the PWMEN bit, no current status has previously been sampled. PWM value registers one, three, and five initially control the three PWM pairs when configured for current status correction.

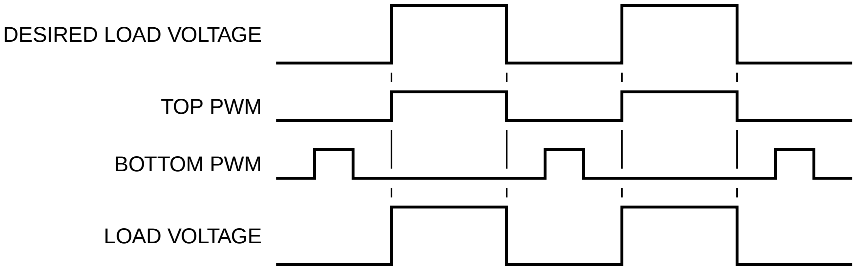


Figure 14-60. Correction with Positive Current

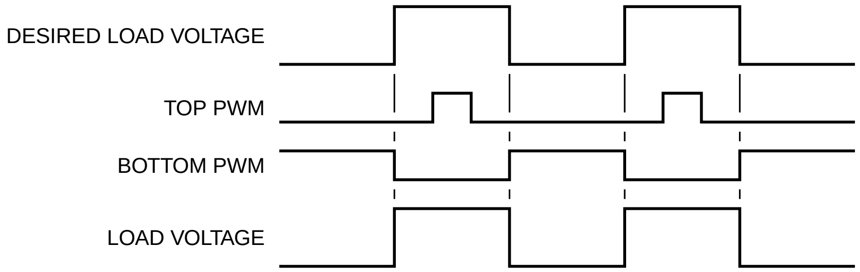


Figure 14-61. Correction with Negative Current

14.4.7 Asymmetric PWM Output

In complementary center-aligned mode, the PWM duty cycle is able to change alternatively at every half cycle. The count direction of the PWM counter selects either the odd or the even PWM value registers to use in the PWM cycle. For counting up, select even PWM value registers to use in the PWM cycle. For counting down, select odd PWM value registers to use in the PWM cycle. The related CINV*n* bits of the PWM pair must select the same polarity for both generators.

Table 14-43. Top/Bottom Corrections Selected by ICC*n* Bits

Bit	Logic State	Output Control
ICCA	0	IPOLA Controls PWM0/PWM1 Pair
	1	PWM Count Direction Controls PWM0/PWM1 Pair
ICCB	0	IPOLB Controls PWM2/PWM3 Pair
	1	PWM Count Direction Controls PWM2/PWM3 Pair
ICCC	0	IPOLC Controls PWM4/PWM5 Pair
	1	PWM Count Direction Controls PWM4/PWM5 Pair

NOTE

If an ICC*x* bit in the PMFICCTL register changes during a PWM period, the new value does not take effect until the next PWM period. ICC*x* bits take effect at the end of each PWM cycle regardless of the state of the related LDOK*x* bit or global load OK.

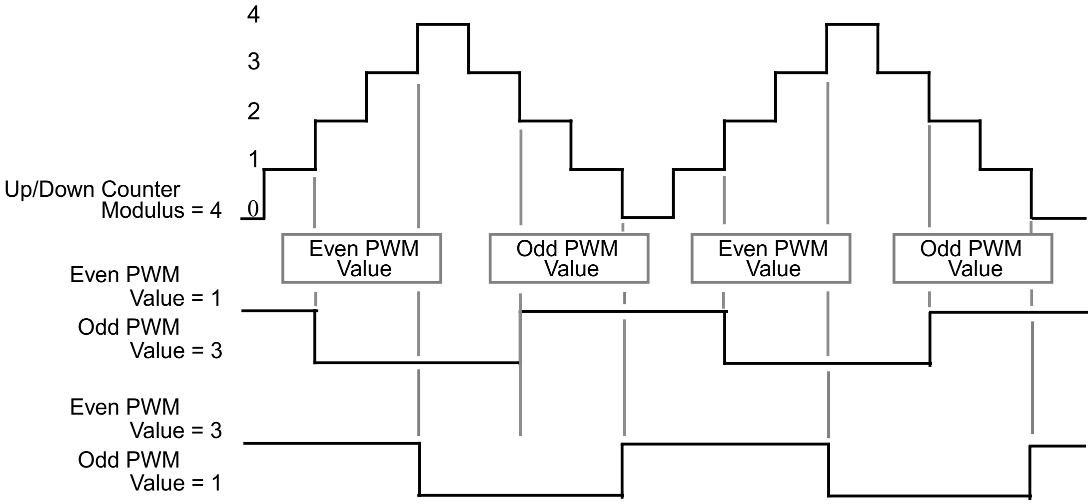


Figure 14-62. Asymmetric Waveform - Phase Shift PWM Output

14.4.8 Variable Edge Placement PWM Output

In complementary edge-aligned mode, the timing of both edges of the PWM output can be controlled using the PEC*x* bits in the PMFICCTL register and the CINV*n* bits in the PMFCINV register.

The edge-aligned signal created by the even value register and the associated CINV*n* bit is ANDed with the signal created by the odd value register and its associated CINV*n* bit. The resulting signal can optionally be negated by PINV*x* and is then fed into the complement and deadtime logic (Figure 14-63). If the value of the inverted register exceeds the non-inverted register value, no output pulse is generated (0% or 100% duty cycle). See right half of Figure 14-64.

In contrast to asymmetric PWM output mode, the PWM phase shift can pass the PWM cycle boundary.

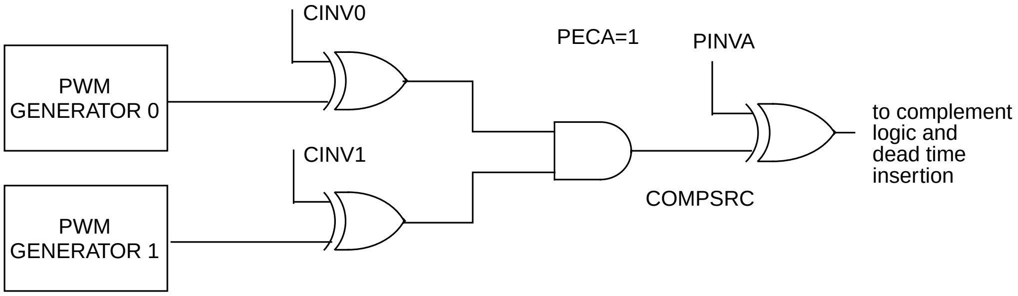


Figure 14-63. Logic AND Function with Signal Inversions

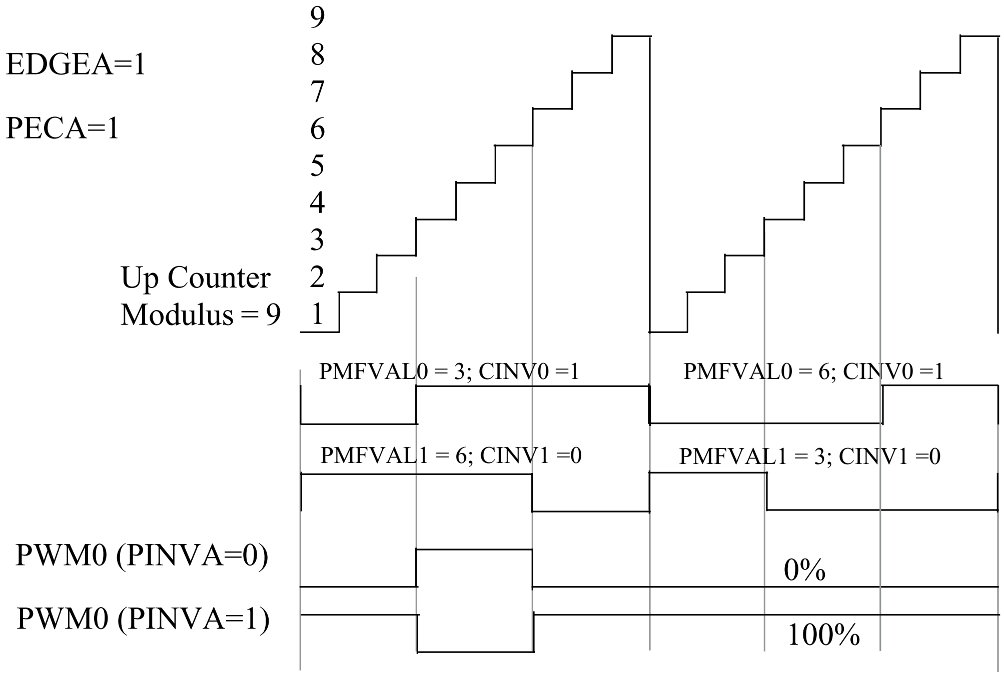


Figure 14-64. Variable Edge Placement Waveform - Phase Shift PWM Output (Edge-Aligned)

14.4.9 Double Switching PWM Output

By using the AND function in Figure 14-63 in complementary center-aligned mode, the PWM output can be configured for double switching operation (Figure 14-65, Figure 14-66). By setting the non-inverted value register greater or equal to the PWM modulus the output function can be switched to single pulse generation on PWM reload cycle basis.

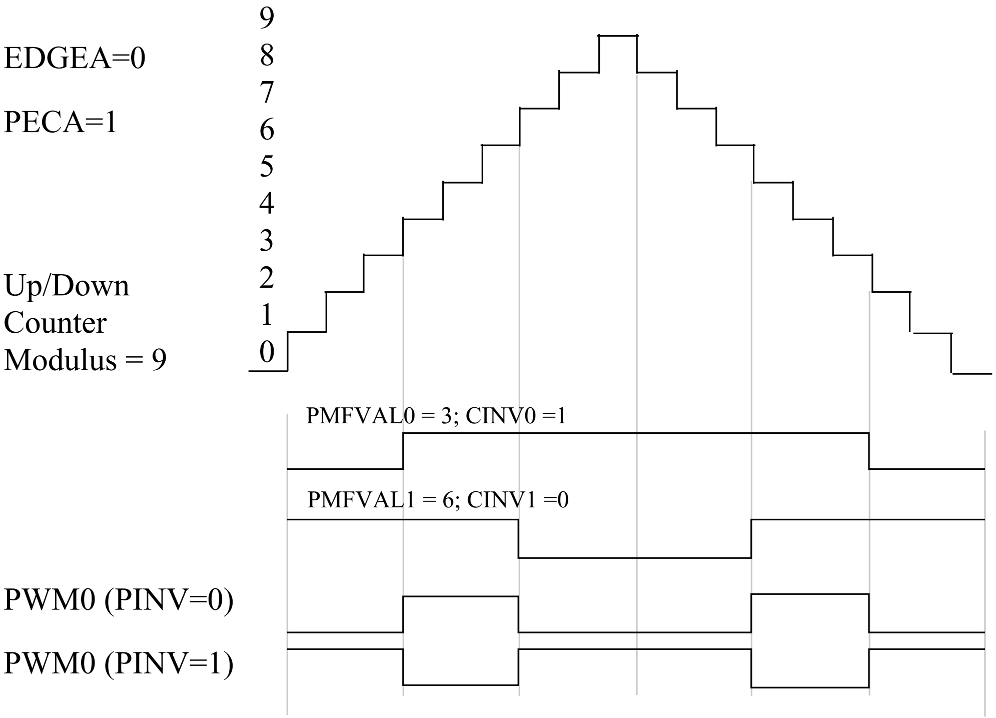


Figure 14-65. Double-Switching PWM Output VAL0<VAL1 (Center-Aligned)

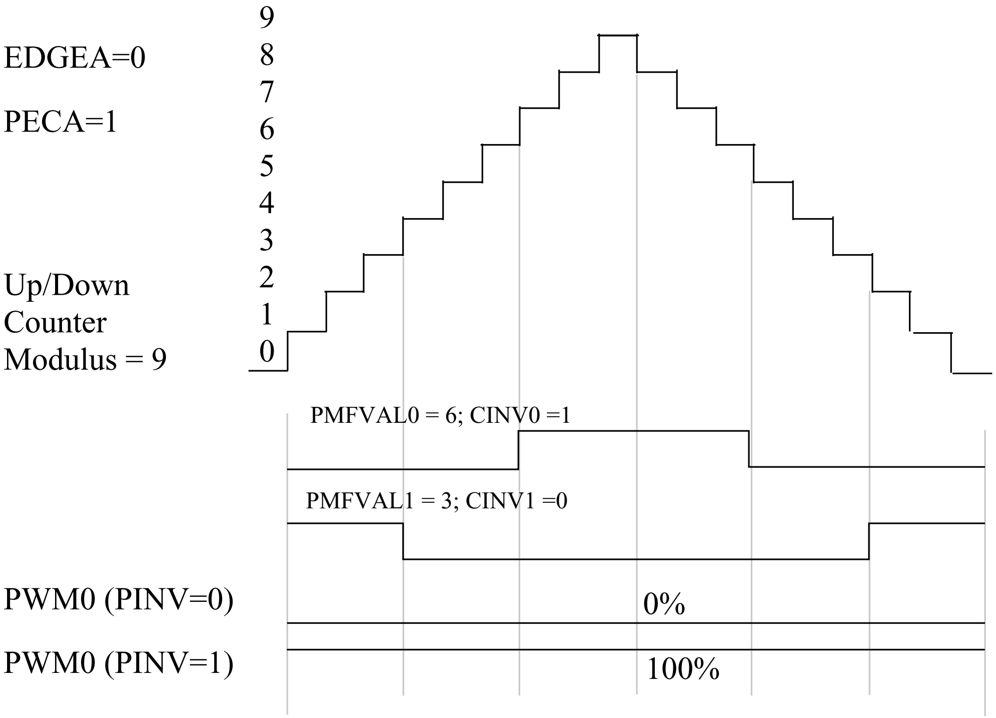


Figure 14-66. Double-Switching PWM Output VAL0>VAL1 (Center-Aligned)

14.4.10 Output Polarity

Output polarity of the PWMs is determined by two options: TOPNEG and BOTNEG. The top polarity option, TOPNEG, controls the polarity of PWM0, PWM2, and PWM4. The bottom polarity option, BOTNEG, controls the polarity of PWM1, PWM3, and PWM5.

Positive polarity means when the PWM is an active level its output is high. Conversely, *negative* polarity means when the PWM is driving an active level its output is low.

If TOPNEG is set, PWM0, PWM2, and PWM4 outputs become *active-low*. When BOTNEG is set, PWM1, PWM3, and PWM5 outputs are *active-low*. When these bits are clear, their respective PWM outputs are *active-high*. See Figure 14-67.

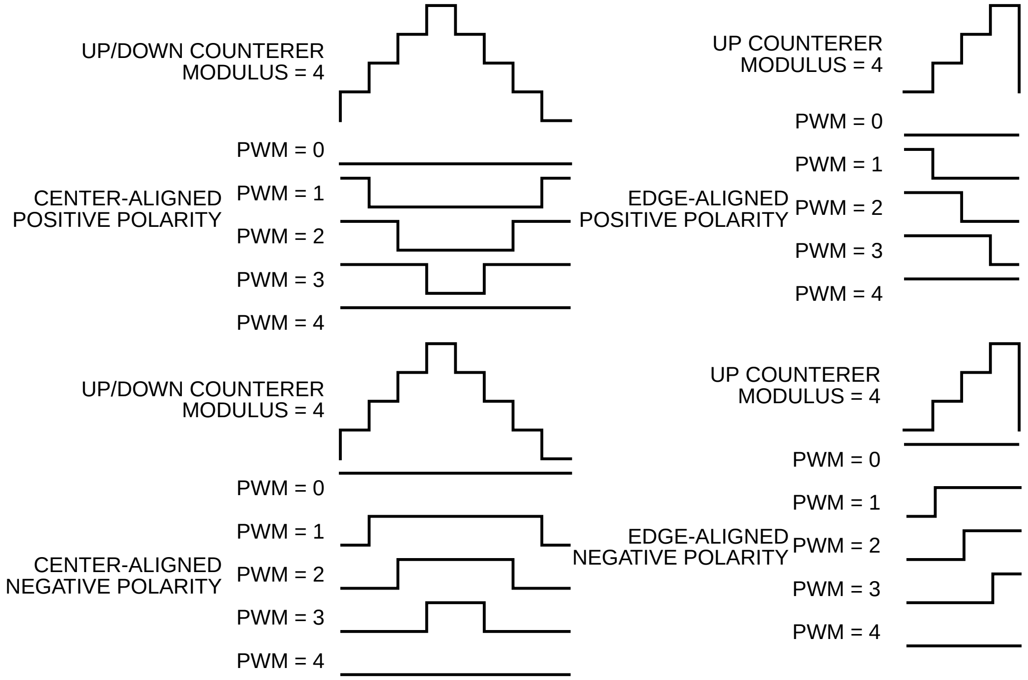


Figure 14-67. PWM Polarity

14.4.11 Software Output Control

Setting output control enable bit, OUTCTL*n*, enables software to drive the PWM outputs instead of the PWM generator. In independent mode, with OUTCTL*n* = 1, the output bit OUT*n*, controls the PWM*n* channel. In complementary channel operation the even OUTCTL*n* bit is used to enable software output control for the pair. The OUTCTL*n* bits must be switched in pairs for proper operation. The OUTCTL*n* and OUT*n* bits are in the PWM output control register.

NOTE

During software output control, TOPNEG and BOTNEG still control output polarity. It will take up to 3 core clock cycles to see the effect of output control on the PWM outputs.

In independent PWM operation, setting or clearing the OUT*n* bit activates or deactivates the PWM*n* output.

In complementary channel operation, the even-numbered OUT*n* bits replace the PWM generator outputs as inputs to the deadtime generators. Complementary channel pairs still cannot drive active level simultaneously, and the deadtime generators continue to insert deadtime in both channels of that pair, whenever an even OUT*n* bit toggles. Even OUT*n* bits control the top PWM signals while the odd OUT bits control the bottom PWM signals with respect to the even OUT*n* bits. Setting the odd OUT*n* bit makes its corresponding PWM the complement of its even pair, while clearing the odd OUT*n* bit deactivates the odd PWM.

Setting the OUTCTL*n* bits does not disable the PWM generators and current status sensing circuitry. They continue to run, but no longer control the outputs. When the OUTCTL*n* bits are cleared, the outputs of the PWM generator become the inputs to the deadtime generators at the beginning of the next PWM cycle. Software can drive the PWM outputs even when PWM enable bit (PWMEN*x*) is set to zero.

NOTE

Avoid an unexpected deadtime insertion by clearing the OUT*n* bits before setting and after clearing the OUTCTL*n* bits.

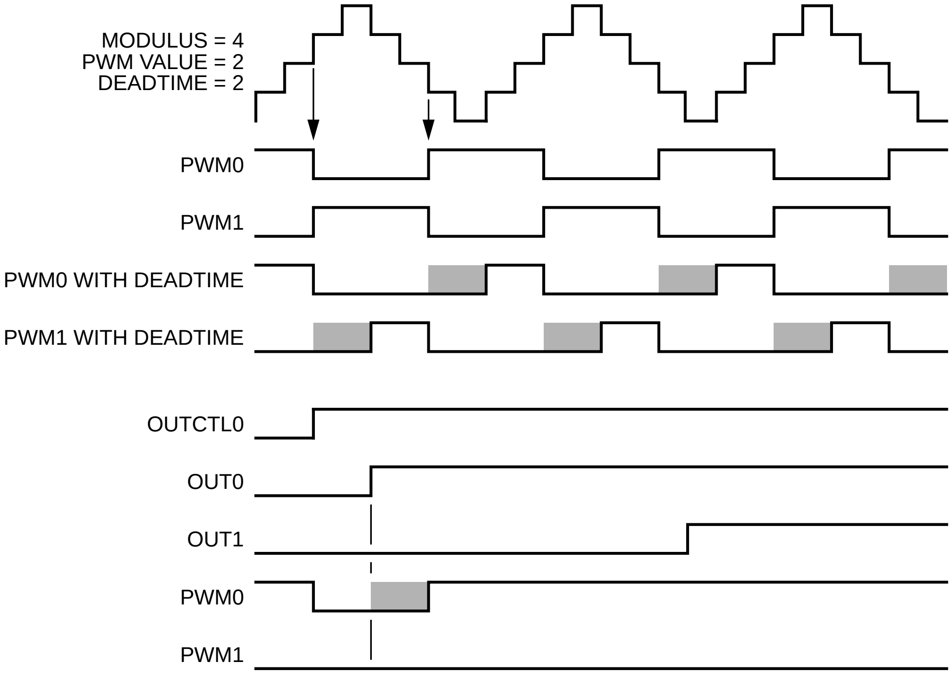


Figure 14-68. Setting OUT0 with OUTCTL Set in Complementary Mode

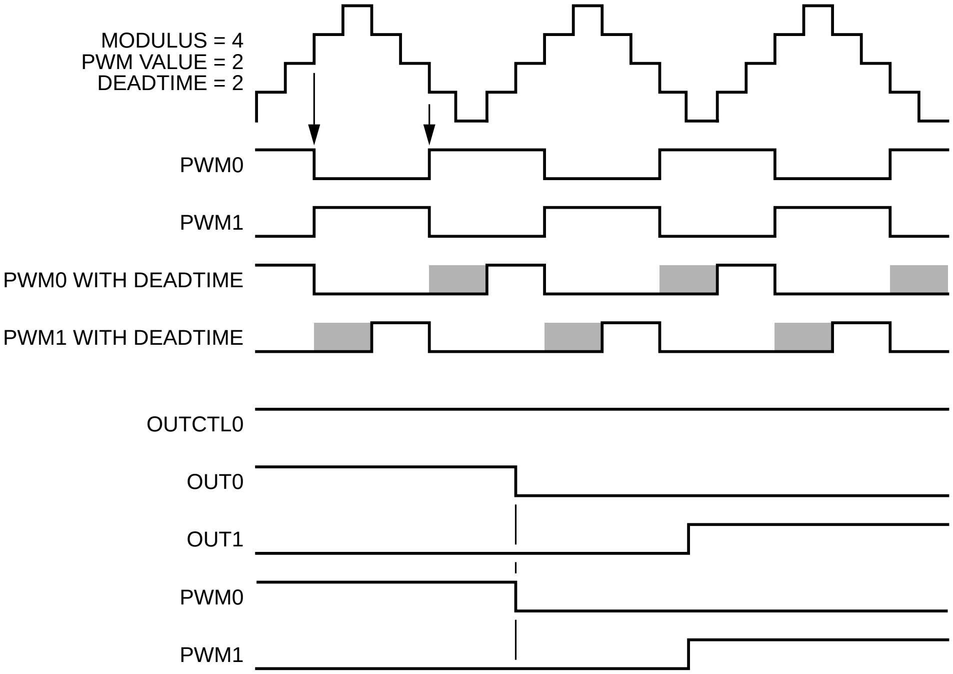


Figure 14-69. Clearing OUT0 with OUTCTL Set in Complementary Mode

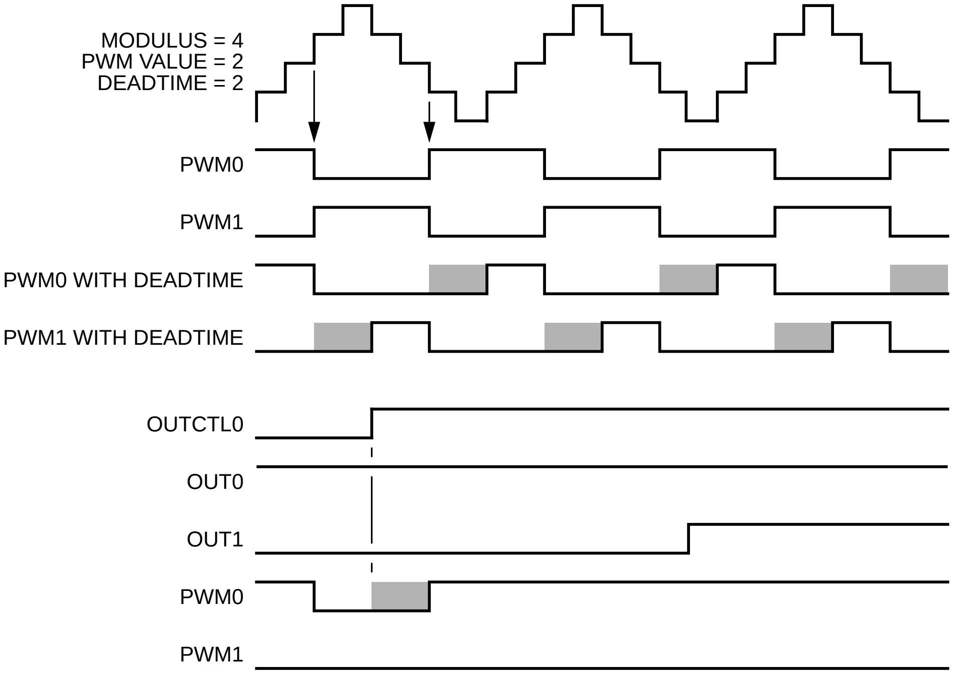


Figure 14-70. Setting OUTCTL with OUT0 Set in Complementary Mode

14.4.12 PWM Generator Loading

14.4.12.1 Load Enable

The load okay bit, LDOK, enables loading the PWM generator with:

- A prescaler divisor—from the PRSC bits in PMFFQC register
- A PWM period—from the PWM counter modulus registers
- A PWM pulse width—from the PWM value registers

LDOK prevents reloading of these PWM parameters before software is finished calculating them. Setting LDOK allows the prescaler bits, PMFMOD and PMFVAL registers to be loaded into a set of buffers. The loaded buffers are used by the PWM generator at the beginning of the next PWM reload cycle. Set LDOK by reading it when it is a logic zero and then writing a logic one to it. After the PWM reload event, LDOK is automatically cleared.

If LDOK is set in the same cycle as the PWM reload event occurs, then the current buffers will be used and the LDOK is valid at the next PWM reload event. See Figure 14-71.

If an asserted LDOK bit is attempted to be set again one cycle prior to the PWM reload event, then the buffers will loaded and LDOK will be cleared automatically. Else if the write access to the set LDOK bit occurs in the same cycle with the reload event, the buffers will also be loaded but the LDOK remains valid also for the next PWM reload event. See Figure 14-72.

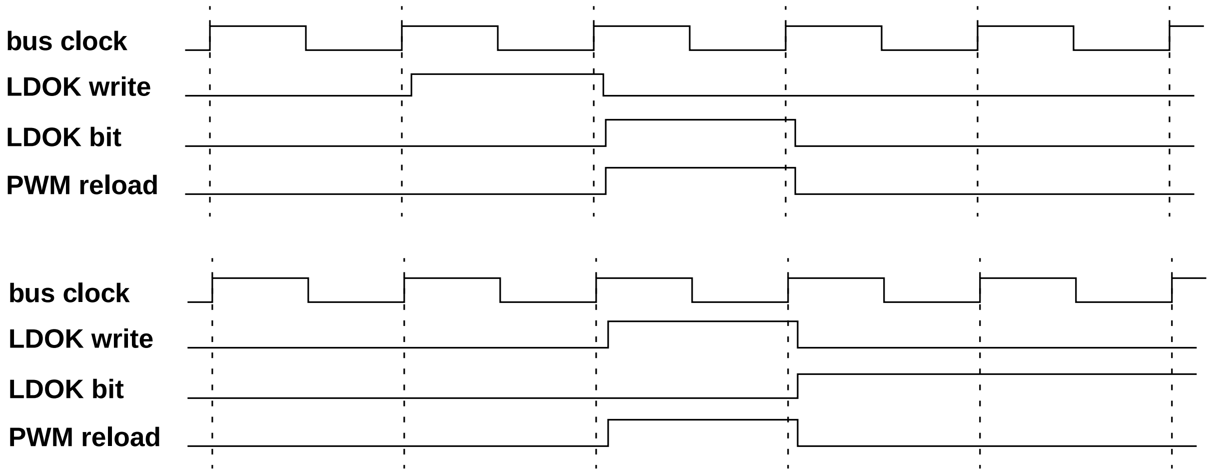


Figure 14-71. Setting cleared LDOK bit at PWM reload event

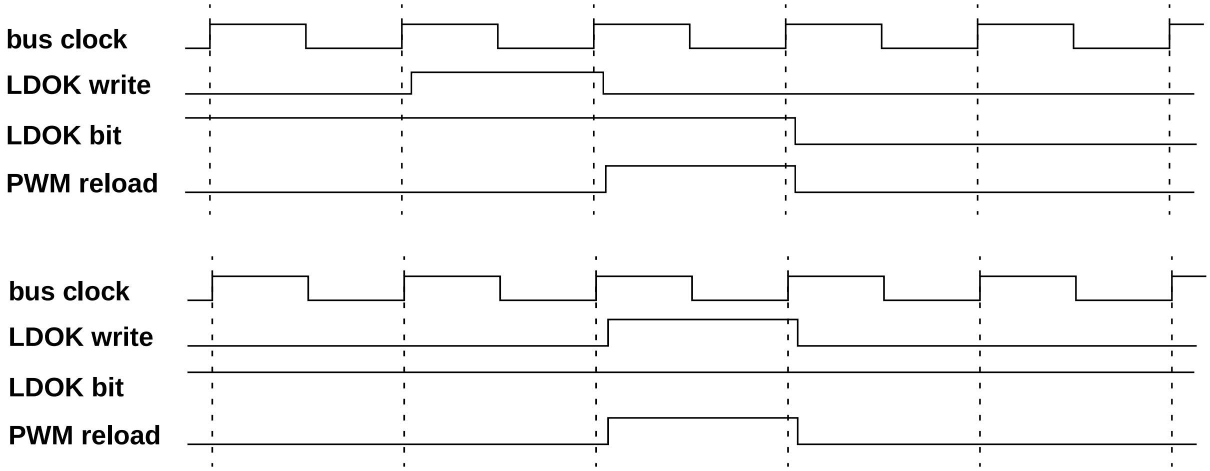


Figure 14-72. Setting asserted LDOK bit at PWM reload event

14.4.12.2 Global Load Enable

If a global load enable bit GLDOKA, B, or C is set, the global load OK bit defined on device level as input to the PMF replaces the function of the related local LDOKA, B, or C bits. The global load OK signal is typically shared between multiple IP blocks with the same double buffer scheme. Software handling must be transferred to the global load OK bit at the chip level.

14.4.12.3 Load Frequency

The LDFQ3, LDFQ2, LDFQ1, and LDFQ0 bits in the PWM control register (PMFFQCx) select an integral loading frequency of 1 to 16-PWM reload opportunities. The LDFQ bits take effect at every PWM reload opportunity, regardless the state of the related load okay bit or global load OK. The *half* bit in the PMFFQC register controls half-cycle reloads for center-aligned PWMs. If the *half* bit is set, a reload opportunity occurs at the beginning of every PWM cycle and half cycle when the count equals the modulus. If the half bit is not set, a reload opportunity occurs only at the beginning of every cycle. Reload opportunities can only occur at the beginning of a PWM cycle in edge-aligned mode.

NOTE

Setting the half bit takes effect immediately. Depending on whether the counter is incrementing or decrementing at this point in time, reloads at even-numbered reload frequencies (every 2, 4, 6,... reload opportunities) will occur only when the counter matches the modulus or only when the counter equals zero, respectively (refer to example of reloading at every two opportunities in Figure 14-74).

NOTE

Loading a new modulus on a half cycle will force the count to the new modulus value minus one on the next clock cycle. Half cycle reloads are possible only in center-aligned mode. Enabling or disabling half-cycle reloads in edge-aligned mode will have no effect on the reload rate.

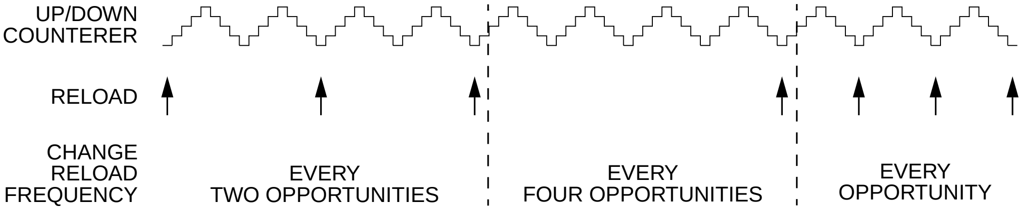


Figure 14-73. Full Cycle Reload Frequency Change

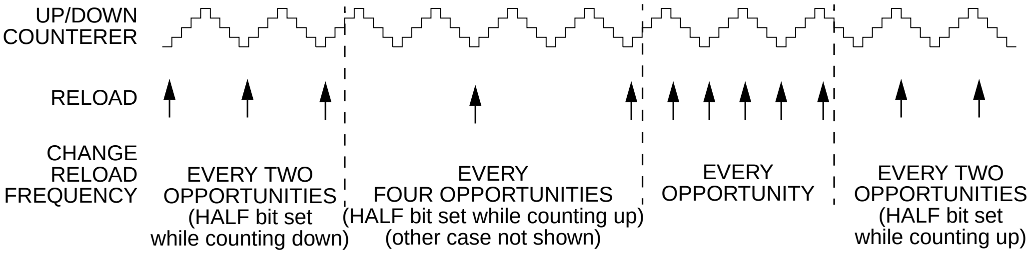


Figure 14-74. Half Cycle Reload Frequency Change

14.4.12.4 Reload Flag

The PWMRF reload flag is set at every reload opportunity, regardless of whether an actual reload occurs (as determined by the related LDOK bit or global load OK). If the PWM reload interrupt enable bit PWMRIE is set, the PWMRF flag generates CPU interrupt requests allowing software to calculate new PWM parameters in real time. When PWMRIE is not set, reloads still occur at the selected reload rate without generating CPU interrupt requests.

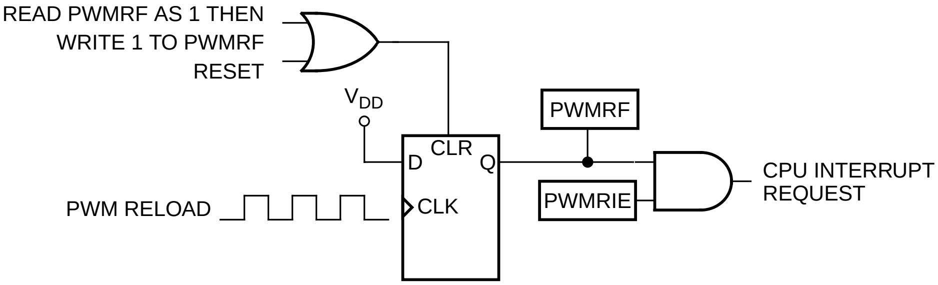


Figure 14-75. PWMRF Reload Interrupt Request

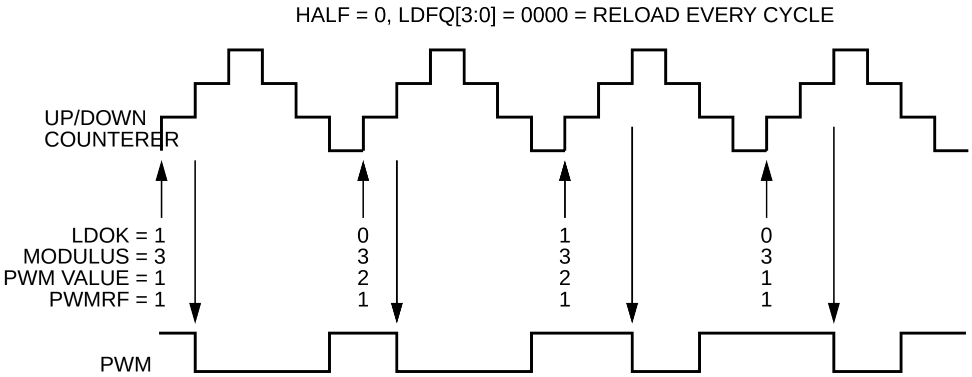


Figure 14-76. Full-Cycle Center-Aligned PWM Value Loading

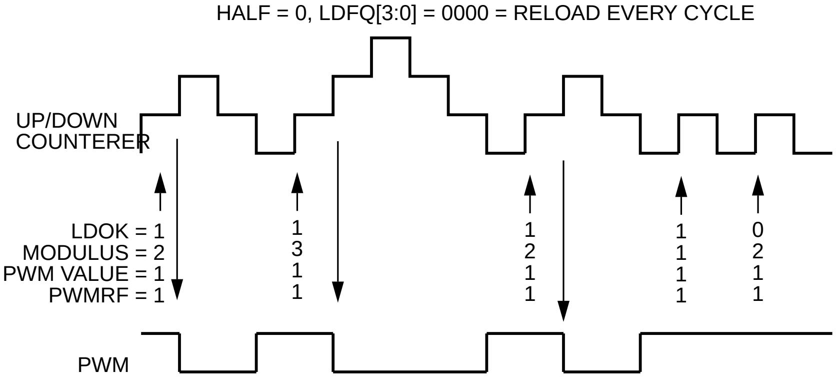


Figure 14-77. Full-Cycle Center-Aligned Modulus Loading

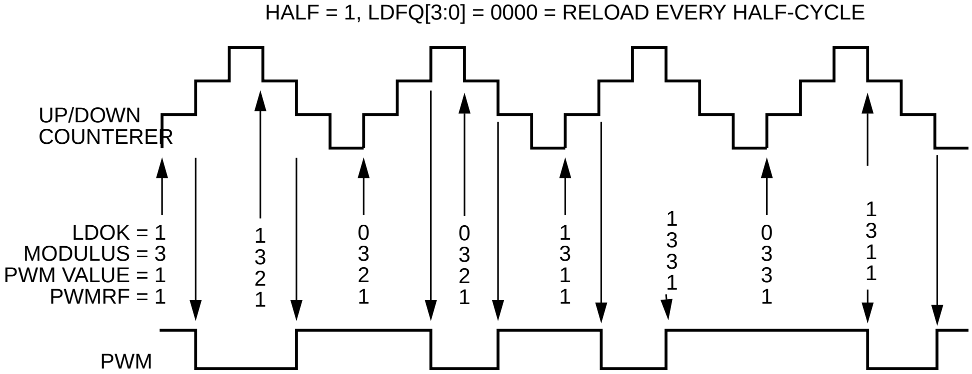


Figure 14-78. Half-Cycle Center-Aligned PWM Value Loading

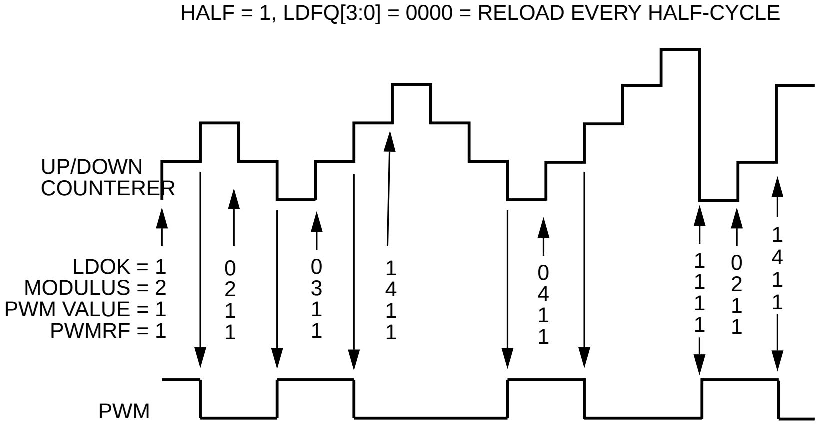


Figure 14-79. Half-Cycle Center-Aligned Modulus Loading

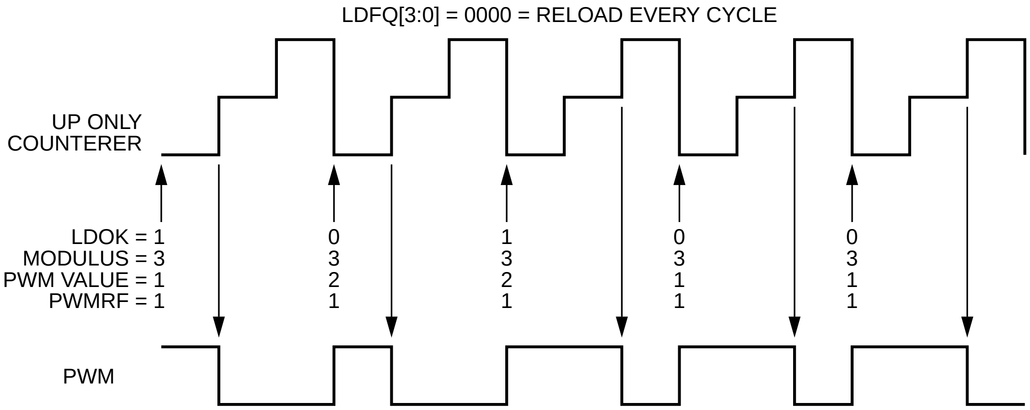


Figure 14-80. Edge-Aligned PWM Value Loading

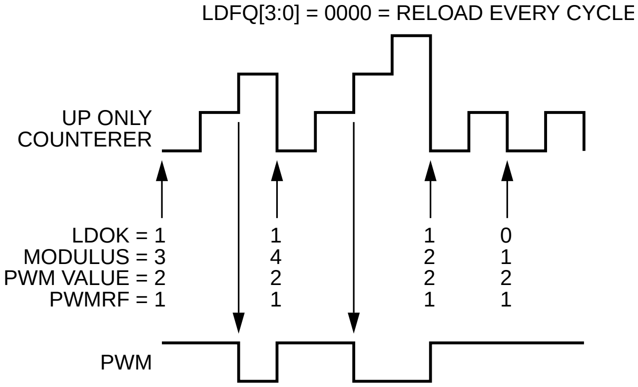


Figure 14-81. Edge-Aligned Modulus Loading

14.4.12.5 Reload Overrun Flag

If a LDOK bit was not set before the PWM reload event, then the related reload overrun error flag is set (PMFROIFx). If the PWM reload overrun interrupt enable bit PMFROIE*x* is set, the PMFROIF*x* flag generates a CPU interrupt request allowing software to handle the error condition.

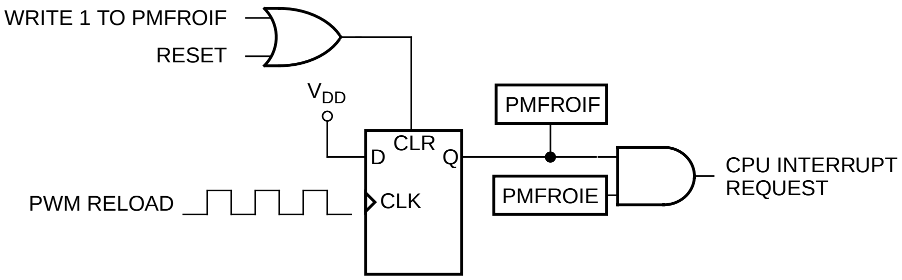


Figure 14-82. PMFROIF Reload Overrun Interrupt Request

14.4.12.6 Synchronization Output (pmf_reload)

The PMF uses reload events to output a synchronization pulse, which can be used as an input to the timer module. A high-true pulse occurs for each PWM cycle start of the PWM, regardless of the state of the related LDOK bit or global load OK and load frequency.

14.4.13 Fault Protection

Fault protection can disable any combination of PWM outputs (for all FAULT0-5 inputs) or switch to output control register PMFOUTF on a fault event (for FAULT4-5 only). Faults are generated by an active level[1] on any of the FAULT inputs. Each FAULT input can be mapped arbitrarily to any of the PWM outputs.

In complementary mode, if a FAULT4 or FAULT5 event is programmed to switch to output control on a fault event resulting in a PWM active state on a particular output, then the transition will take place after deadtime insertion. Thus an asynchronous path to disable the PWM output is not available.

On a fault event the PWM generator continues to run.

The fault decoder affects the PWM outputs selected by the fault logic and the disable mapping register.

The fault protection is enabled even when the PWM is not enabled; therefore, a fault will be latched in and will be cleared in order to prevent an interrupt when the PWM is enabled.

14.4.13.1 Fault Input Sample Filter

Each fault input has a sample filter to test for fault conditions. After every bus cycle setting the FAULT*m* input at logic zero, the filter synchronously samples the input once every four bus cycles. QSMP determines the number of consecutive samples that must be logic one for a fault to be detected. When a fault is detected, the corresponding FAULT*m* flag, FIF*m*, is set. FIF*m* can only be cleared by writing a logic one to it.

If the FIE*m*, FAULT*m* interrupt enable bit is set, the FIF*m* flag generates a CPU interrupt request. The interrupt request latch remains set until:

- Software clears the FIF*m* flag by writing a logic one to it
- Software clears the FIE*m* bit by writing a logic zero to it
- A reset occurs

14.4.13.2 Automatic Fault Recovery

Setting a fault mode bit, FMOD*m*, configures faults from the FAULT*m* input for automatically reenabling the PWM outputs.

When FMOD*m* is set, disabled PWM outputs are enabled when the FAULT*m* input returns to logic zero and a new PWM half cycle begins. See Figure 14-83. Clearing the FIF*m* flag does not affect disabled PWM outputs when FMOD*m* is set.

---

[1] The active input level may be defined or programmable at SoC level. The default for internally connected resources is active-high. For availability and configurability of fault inputs on pins refer to the device overview section.

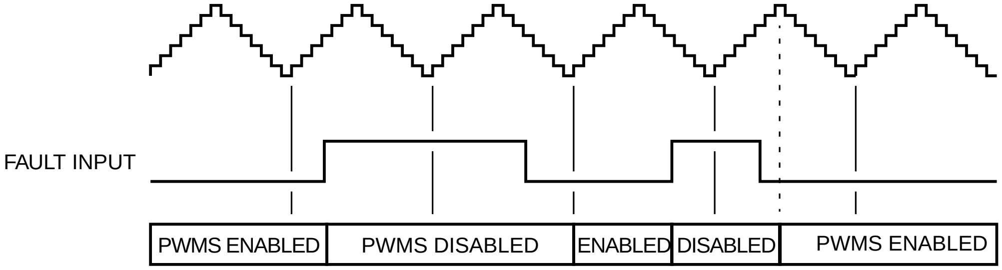


Figure 14-83. Automatic Fault Recovery

14.4.13.3 Manual Fault Recovery

Clearing a fault mode bit, FMOD*m*, configures faults from the FAULT*m* input for manually reenabling the PWM outputs:

- PWM outputs disabled by the FAULT0 input or the FAULT2 input are enabled by clearing the corresponding FIF*m* flag. The time at which the PWM outputs are enabled depends on the corresponding QSMP bit setting. If QSMP*m* = 00, the PWM outputs are enabled on the next IP bus cycle when the logic level detected by the filter at the fault input is logic zero. If QSMP*m* = 01,10 or 11, the PWMs are enabled when the next PWM half cycle begins regardless of the state of the logic level detected by the filter at the fault. See Figure 14-84 and Figure 14-85.
- PWM outputs disabled by the FAULT1 or FAULT3-5 inputs are enabled when
  - Software clears the corresponding FIF*m* flag
  - The filter detects a logic zero on the fault input at the start of the next PWM half cycle boundary. See Figure 14-86.

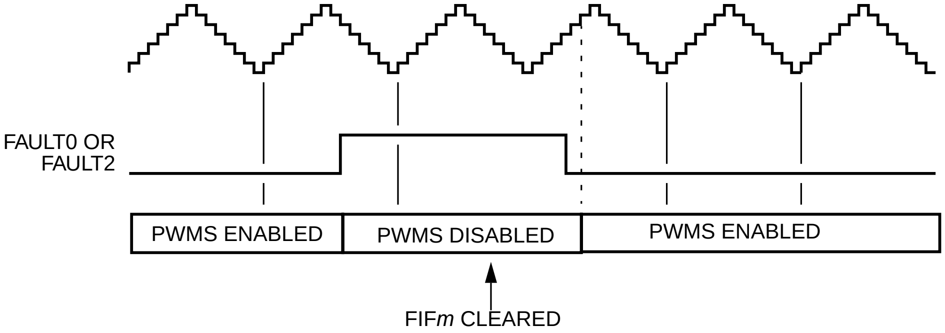


Figure 14-84. Manual Fault Recovery (Faults 0 and 2) — QSMP = 00

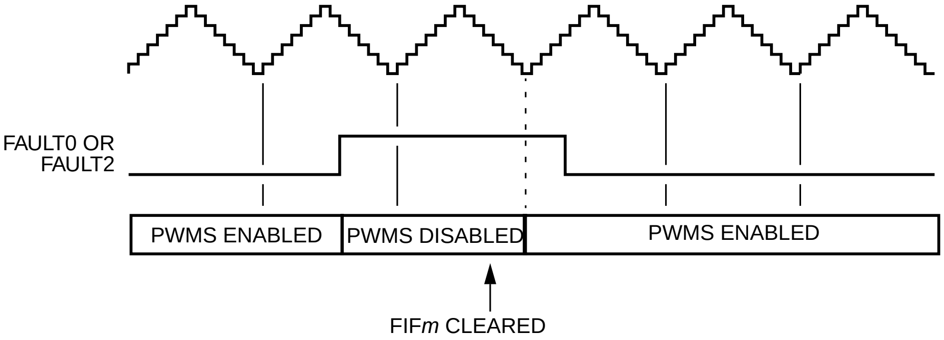


Figure 14-85. Manual Fault Recovery (Faults 0 and 2) — QSMP = 01, 10, or 11

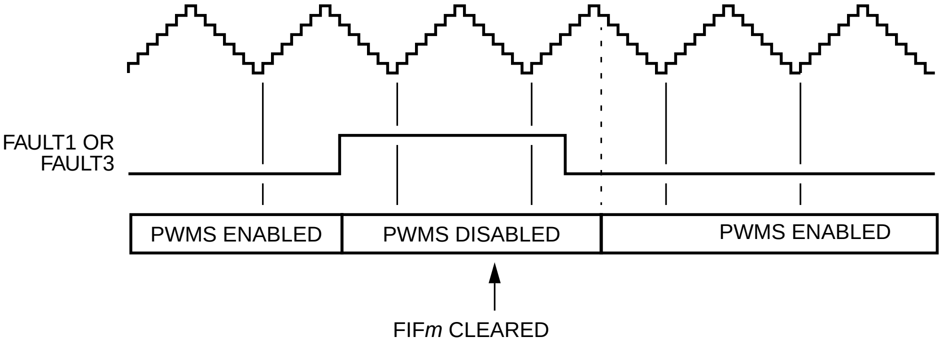


Figure 14-86. Manual Fault Recovery (Faults 1 and 3-5)

NOTE

PWM half-cycle boundaries occur at both the PWM cycle start and when the counter equals the modulus, so in edge-aligned operation full-cycles and half-cycles are equal.

NOTE

Fault protection also applies during software output control when the OUTCTL*n* bits are set. Fault recovery still occurs at half PWM cycle boundaries while the PWM generator is engaged, PWMEN equals one. But the OUT*n* bits can control the PWM outputs while the PWM generator is off, PWMEN equals zero. Thus, fault recovery occurs at IPbus cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

14.5 Resets

All PMF registers are reset to their default values upon any system reset.

14.6 Clocks

The gated system core clock is the clock source for all PWM generators. The system clock is used as a clock source for any other logic in this module. The system bus clock is used as clock for specific control registers and flags (LDOK*x*, PWMRF*x*, PMFOUTB).

14.7 Interrupts

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

Table 14-44. PMF Interrupt Sources

Module Interrupt Sources (Interrupt Vector)	Associated Flags	Local Enable
PMF reload A	PWMRFA	PMFENCA[PWMRIEA]
PMF reload B(1)	PWMRFB	PMFENCB[PWMRIEB]
PMF reload C[1]	PWMRFC	PMFENCC[PWMRIEC]
PMF fault	PMFFIF[FIF0] PMFFIF[FIF1] PMFFIF[FIF2] PMFFIF[FIF3]	PMFFIE[FIE0] PMFFIE[FIE1] PMFFIE[FIE2] PMFFIE[FIE3]
PMF reload overrun	PMFROIF[PMFROIFA] PMFROIF[PMFROIFB] PMFROIF[PMFROIFC]	PMFROIE[PMFROIEA] PMFROIE[PMFROIEB] PMFROIE[PMFROIEC]

1. If MTG=0: Interrupt mirrors PMF reload A interrupt

14.8 Initialization and Application Information

14.8.1 Initialization

Initialize all registers; read, then set the related LDOK bit or global load OK before setting the PWMEN bit. With LDOK set, setting PWMEN for the first time after reset immediately loads the PWM generator thereby setting the PWMRF flag. PWMRF generates a CPU interrupt request if the PWMRIE bit is set. In complementary channel operation with current-status correction selected, PWM value registers one, three, and five control the outputs for the first PWM cycle.

NOTE

> Even if LDOK is not set, setting PWMEN also sets the PWMRF flag. To prevent a CPU interrupt request, clear the PWMRIE bit before setting PWMEN.

Setting PWMEN for the first time after reset without first setting LDOK loads a prescaler divisor of one, a PWM value of 0x0000, and an unknown modulus.

The PWM generator uses the last values loaded if PWMEN is cleared and then set while LDOK equals zero.

Initializing the deadtime register, after setting PWMEN or OUTCTL*n*, can cause an improper deadtime insertion. However, the deadtime can never be shorter than the specified value.

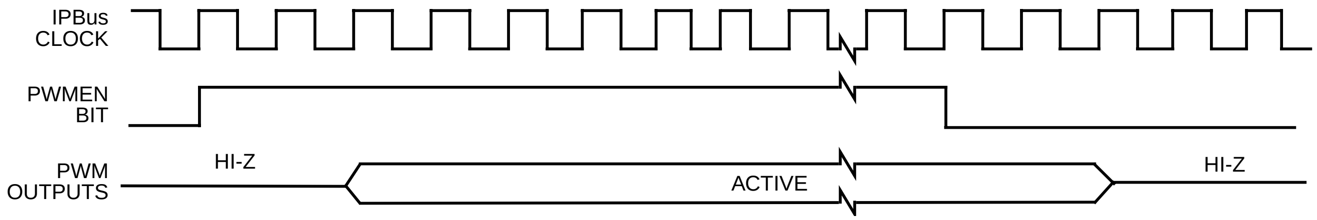


Figure 14-87. PWMEN and PWM Outputs in Independent Operation

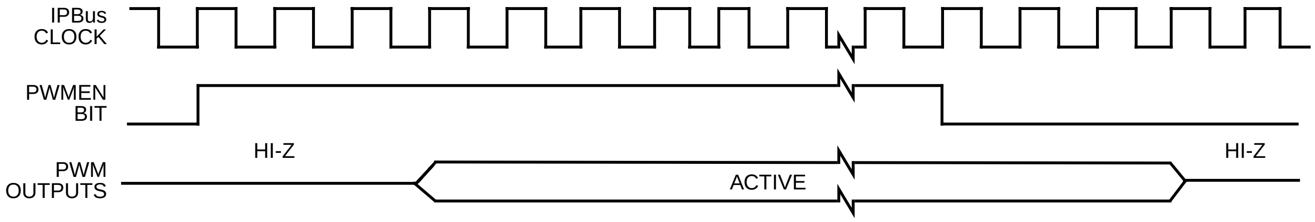


Figure 14-88. PWMEN and PWM Outputs in Complementary Operation

When the PWMEN bit is cleared:

- The PWM*n* outputs lose priority on associated outputs unless OUTCTL*n* = 1
- The PWM counter is cleared and does not count
- The PWM generator forces its outputs to zero
- The PWMRF flag and pending CPU interrupt requests are not cleared
- All fault circuitry remains active unless FEN*m* = 0
- Software output control remains active
- Deadtime insertion continues during software output control

14.8.1.1 Register Write Protection

The following configuration registers and bits can be write protected:

PMFCFG0, PMFCFG1, PMFCFG3, PMFFEN, PMFQSMP0-1, PMFENCA[RSTRTA,GLDOKA], PMFENCB[RSTRTB,GLDOKB], PMFENCC[RSTRTC,GLDOKC], PMFDTMA,B,C, PMFDMP0-5, PMFOUTF

NOTE

> Make sure to set the write protection bit WP in PMFCFG0 after configuring and prior to enabling PWM outputs and fault inputs.

14.8.2 BLDC 6-Step Commutation

14.8.2.1 Unipolar Switching Mode

Unipolar switching mode uses registers PMFOUTC and PMFOUTB to perform commutation.

Table 14-45. Effects of OUTCTL and OUT Bits on PWM Output Pair in Complementary Mode

OUTCTL (odd,even)	OUT (odd,even)	PWM (odd)	PWM (even)
00	xx	$\overline{\text{PWMgen(even)}}$	PWMgen(even)
11	10	$\overline{\text{OUTB(even)}}$=1	OUTB(even)=0
01	x0	0	OUTB(even)=0

The recommended setup is:

```
PMFCFG0[INDEPC,INDEPB,INDEPA] = 0x0;   // Complementary mode
PMFCFG1[ENCE]                 = 1;     // Enable commutation event
PMFOUTB = 0x2A;                        // Set return path pattern, high-side off, low-side on
PMFOUTC = 0x1C;                        // Branch A->B, “mask” C // 0˚
```

The commutation sequence is:

```
PMFOUTC = 0x34;                        // Branch A->C, “mask” B // 60˚
PMFOUTC = 0x31;                        // Branch B->C, “mask” A // 120˚
PMFOUTC = 0x13;                        // Branch B->A, “mask” C // 180˚
PMFOUTC = 0x07;                        // Branch C->A, “mask” B // 240˚
PMFOUTC = 0x0D;                        // Branch C->B, “mask” A // 300˚
PMFOUTC = 0x1C;                        // Branch A->B, “mask” C // 360˚
```

Table 14-46. Unipolar Switching Sequence

<table>
<tr><th>Branch</th><th>Channel</th><th>0°</th><th>60°</th><th>120°</th><th>180°</th><th>240°</th><th>300°</th></tr>
<tr><td rowspan="2">A</td><td>PWM0</td><td colspan="2">PWMgen</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>PWM1</td><td colspan="2">$\overline{\text{PWMgen}}$</td><td>0</td><td colspan="2">1</td><td>0</td></tr>
<tr><td rowspan="2">B</td><td>PWM2</td><td>0</td><td>0</td><td colspan="2">PWMgen</td><td>0</td><td>0</td></tr>
<tr><td>PWM3</td><td>1</td><td>0</td><td colspan="2">$\overline{\text{PWMgen}}$</td><td>0</td><td>1</td></tr>
<tr><td rowspan="2">C</td><td>PWM4</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">PWMgen</td></tr>
<tr><td>PWM5</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">$\overline{\text{PWMgen}}$</td></tr>
</table>

14.8.2.2 Bipolar Switching Mode

Bipolar switching mode uses register bits MSK5-0 and PINVA, B, C to perform commutation.

The recommended setup is:

```
PMFCFG0[INDEPC,INDEPB,INDEPA] = 0x0;  // Complementary mode
PMFCFG1[ENCE]                 = 1;    // Enable commutation event
PMFCFG2[MSK5:MSK0]            = 0x30; // Branch A<->B, mask C // 0˚
PMFCFG3[PINVC,PINVB,PINVA]    = 0x2;  // Invert B
```

The commutation sequence is:

```
PMFCFG2[MSK5:MSK0]            = 0x03; // Branch C<->B, mask A // 60˚
PMFCFG3[PINVC,PINVB,PINVA]    = 0x2;  // Invert B
```

```
PMFCFG2[MSK5:MSK0]            = 0x0c; // Branch C<->A, mask B // 120°
PMFCFG3[PINVC,PINVB,PINVA]    = 0x1;  // Invert A

PMFCFG2[MSK5:MSK0]            = 0x30; // Branch B<->A, mask C // 180°
PMFCFG3[PINVC,PINVB,PINVA]    = 0x1;  // Invert A

PMFCFG2[MSK5:MSK0]            = 0x03; // Branch B<->C, mask A // 240°
PMFCFG3[PINVC,PINVB,PINVA]    = 0x4;  // Invert C

PMFCFG2[MSK5:MSK0]            = 0x0c; // Branch A<->C, mask B // 300°
PMFCFG3[PINVC,PINVB,PINVA]    = 0x4;  // Invert C

PMFCFG2[MSK5:MSK0]            = 0x30; // Branch A<->B, mask A // 360°
PMFCFG3[PINVC,PINVB,PINVA]    = 0x2;  // Invert B
```

Table 14-47. Bipolar Switching Sequence

Branch	Channel	0°	60°	120°	180°	240°	300°
A	PWM0	PWMgen	Masked	$\overline{\text{PWMgen}}$		Masked	PWMgen
	PWM1	$\overline{\text{PWMgen}}$	Masked	PWMgen		Masked	$\overline{\text{PWMgen}}$
B	PWM2	$\overline{\text{PWMgen}}$		Masked	PWMgen		Masked
	PWM3	PWMgen		Masked	$\overline{\text{PWMgen}}$		Masked
C	PWM4	Masked	PWMgen		Masked	$\overline{\text{PWMgen}}$	
	PWM5	Masked	$\overline{\text{PWMgen}}$		Masked	PWMgen	

Chapter 15
Serial Communication Interface (S12SCIV6)

Table 15-1. Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
06.01	05/29/2012			update register map, change BD,move IREN to SCIACR2
06.02	10/17/2012			fix typo on page 15-562 and on page 15-562;fix typo of version V6 update fast data tolerance calculation and add notes.
06.03	10/25/2012			fix typo Table 15-2, SBR[15:4],not SBR[15:0]
06.04	12/19/2012			fix typo Table 15-6,15.4.1/15-575
06.05	02/22/2013			fix typo Figure 15-1./15-559 Figure 15-4./15-562 update Table 15-2./15-562 15.4.4/15-577 15.4.6.3/15-584
06.06	03/11/2013			fix typo of BDL reset value,Figure 15-4 fix typo of Table 15-2,Table 15-16,reword 15.4.4/15-577
06.07	09/03/2013			update Figure 15-14./15-574 Figure 15-16./15-578 Figure 15-20./15-583 update 15.4.4/15-577,more detail for two baud add note for Table 15-16./15-577 update Figure 15-2./15-561,Figure 15-12./15-572

15.1 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.

15.1.1 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

15.1.2 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

15.1.3 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

15.1.4 Block Diagram

Figure 15-1 is a high level block diagram of the SCI module, showing the interaction of various function blocks.

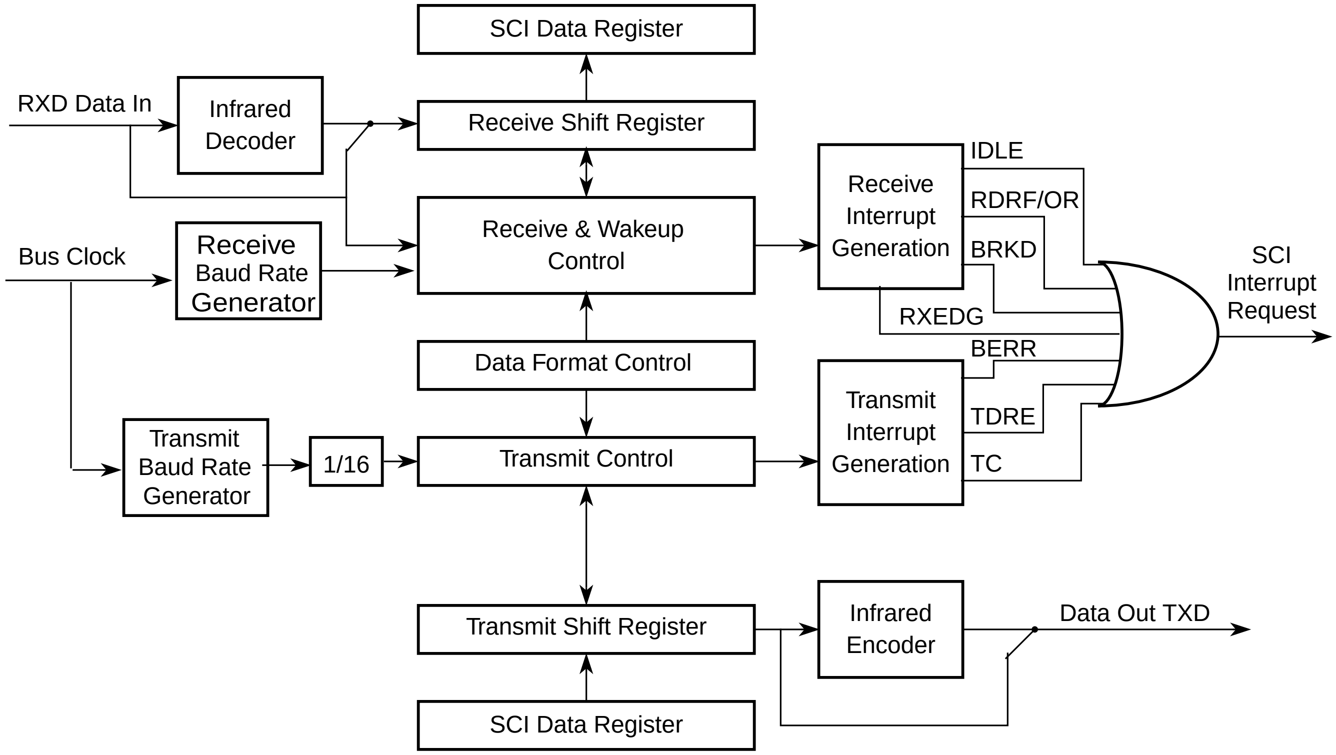


Figure 15-1. SCI Block Diagram

15.2 External Signal Description

The SCI module has a total of two external pins.

15.2.1 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.

15.2.2 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.

15.3 Memory Map and Register Definition

This section provides a detailed description of all the SCI registers.

15.3.1 Module Memory Map and Register Definition

The memory map for the SCI module is given below in Figure 15-2. 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.

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. 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		▨	▨					▨
0x0006 SCIDRH	R	R8	T8	0	0	0	Reserved	Reserved	Reserved
	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 15-2. SCI Register Summary

15.3.2.1 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 15-3. 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 15-4. 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 15-2. 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.

15.3.2.2 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 15-5. 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 15-3. 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 15-4. 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 15-3. 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 15-4. 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

15.3.2.3 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 15-6. SCI Alternative Status Register 1 (SCIASR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 15-5. 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

15.3.2.4 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 15-7. SCI Alternative Control Register 1 (SCIACR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 15-6. 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

15.3.2.5 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 15-8. SCI Alternative Control Register 2 (SCIACR2)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 15-7. 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 15-8.
2:1 BERRM[1:0]	**Bit Error Mode** — Those two bits determines the functionality of the bit error detect feature. See Table 15-9.
0 BKDFE	**Break Detect Feature Enable** — BKDFE enables the break detect circuitry. 0 Break detect circuit disabled 1 Break detect circuit enabled

Table 15-8. IRSCI Transmit Pulse Width

TNP[1:0]	Narrow Pulse Width
11	1/4
10	1/32
01	1/16
00	3/16

Table 15-9. 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 15-19)
1	0	Receive input sampling occurs during the 13th time tick of a transmitted bit (refer to Figure 15-19)
1	1	Reserved

15.3.2.6 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 15-9. SCI Control Register 2 (SCICR2)

Read: Anytime

Write: Anytime

Table 15-10. 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

15.3.2.7 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 15-10. SCI Status Register 1 (SCISR1)

Read: Anytime

Write: Has no meaning or effect

Table 15-11. 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 15-11. 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

15.3.2.8 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 15-11. SCI Status Register 2 (SCISR2)

Read: Anytime

Write: Anytime

Table 15-12. 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

15.3.2.9 SCI Data Registers (SCIDRH, SCIDRL)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	R8	T8	0	0	0	Reserved	Reserved	Reserved
W	▨		▨	▨	▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 15-12. 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 15-13. 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 15-13. 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.

15.4 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 15-14 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.

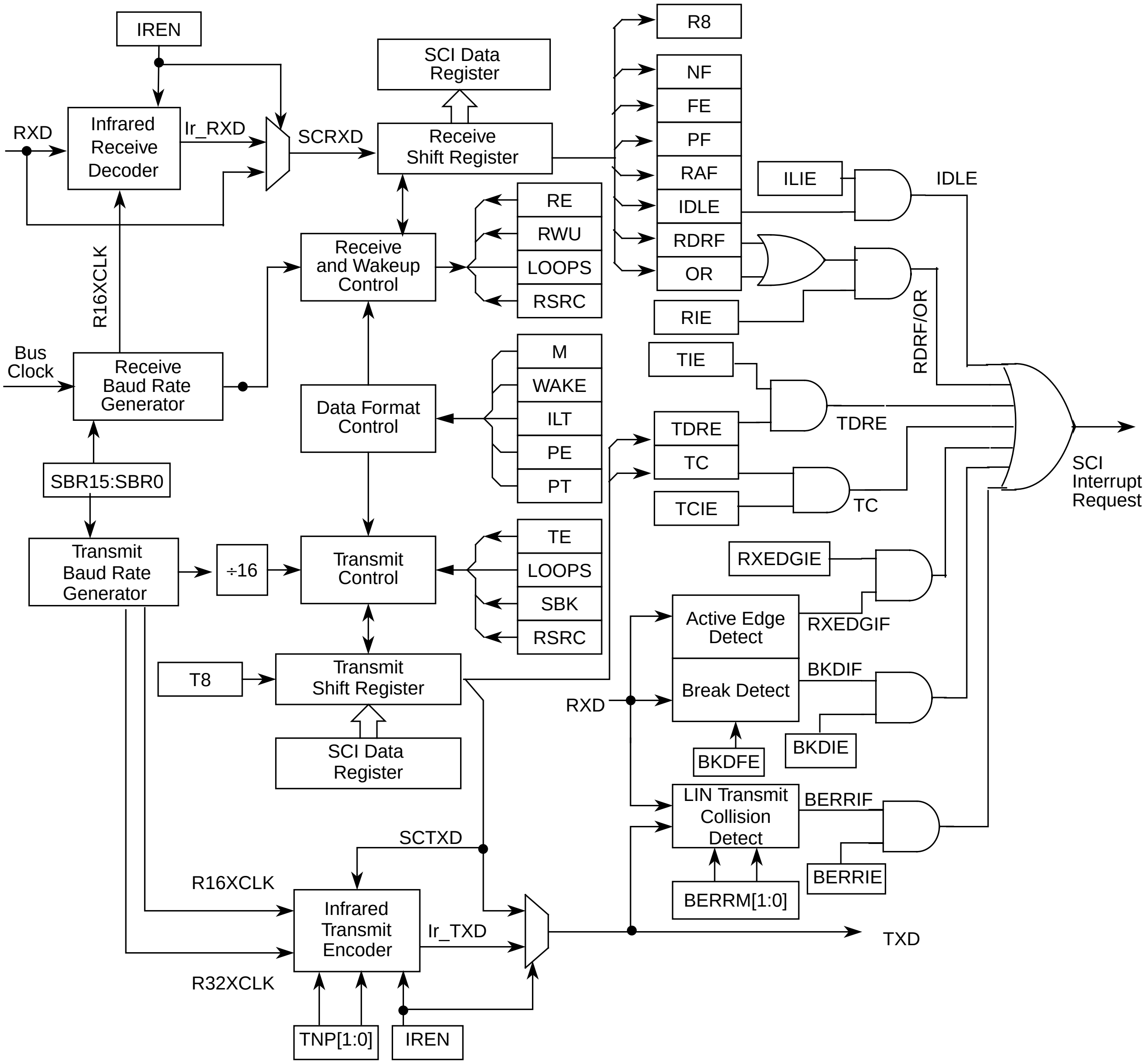


Figure 15-14. Detailed SCI Block Diagram

15.4.1 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.

15.4.1.1 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.

15.4.1.2 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.

15.4.2 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.

15.4.3 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 15-15 below.

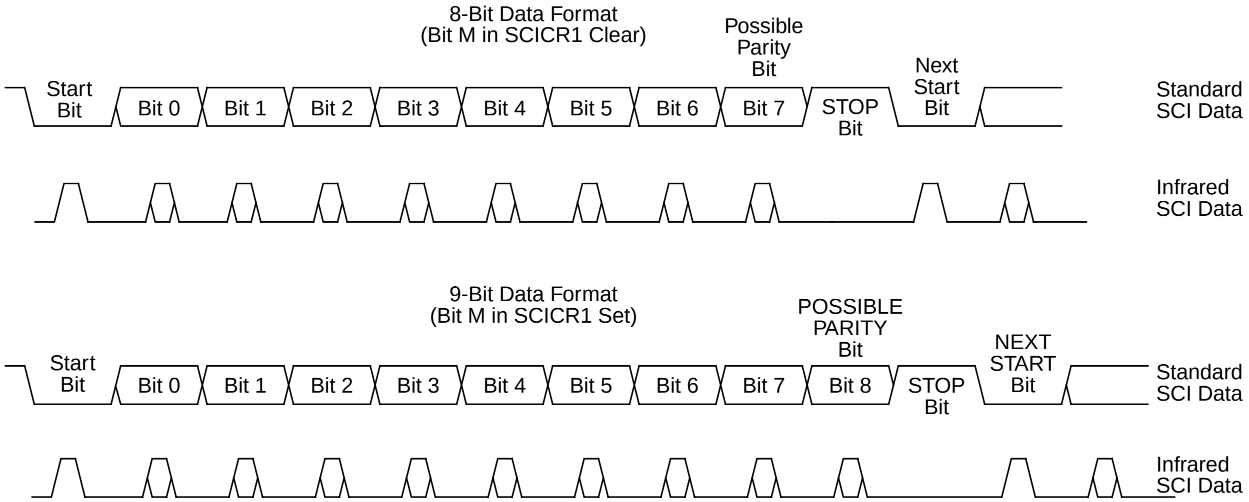


Figure 15-15. SCI Data Formats

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 15-14. 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 15.4.6.6, “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 15-15. 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 15.4.6.6, “Receiver Wakeup”.

15.4.4 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 15-16 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 15-16. 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

15.4.5 Transmitter

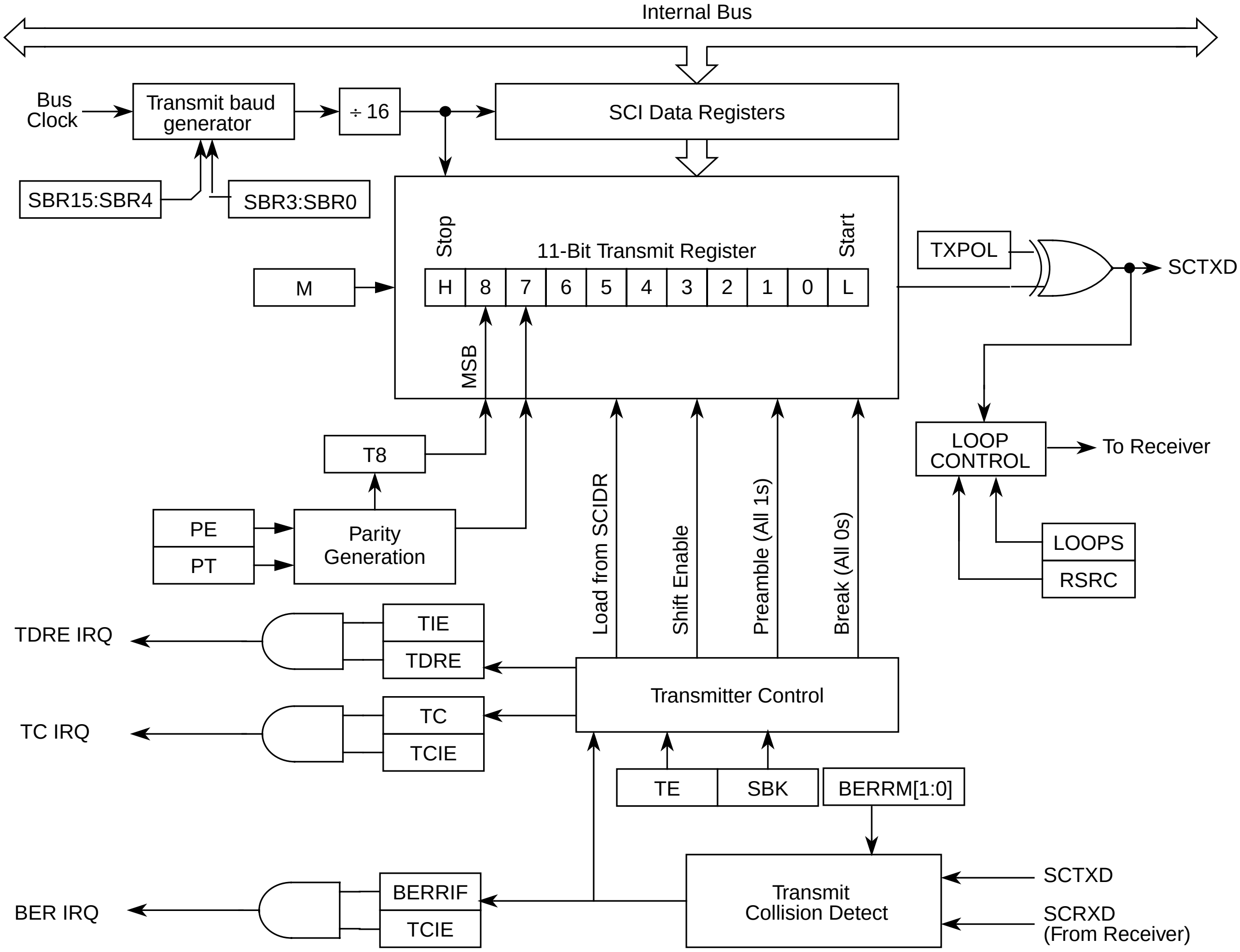


Figure 15-16. Transmitter Block Diagram

15.4.5.1 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).

15.4.5.2 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.

15.4.5.3 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 15-17 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.

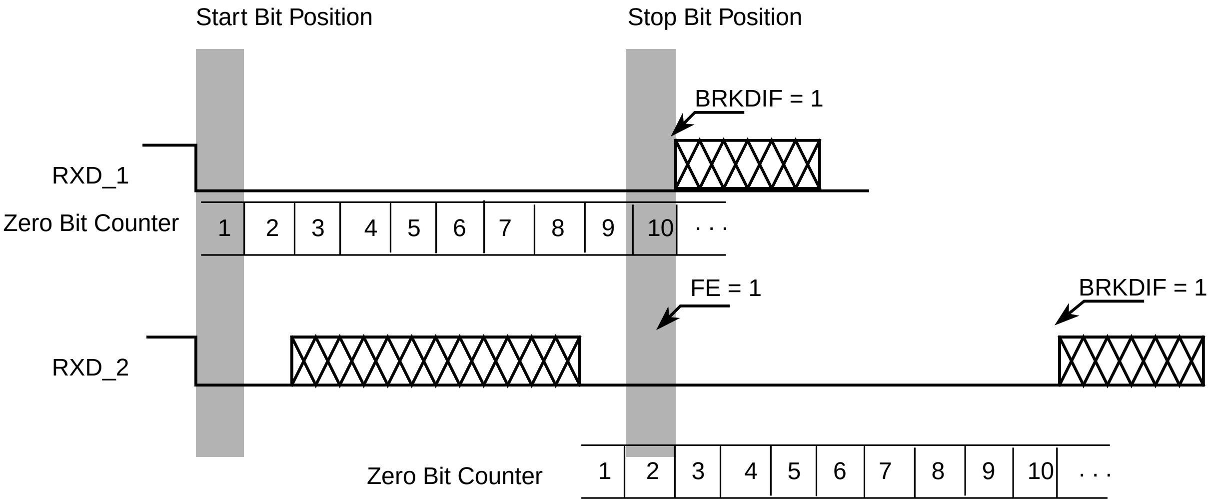


Figure 15-17. Break Detection if BRKDFE = 1 (M = 0)

15.4.5.4 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

15.4.5.5 LIN Transmit Collision Detection

This module allows to check for collisions on the LIN bus.

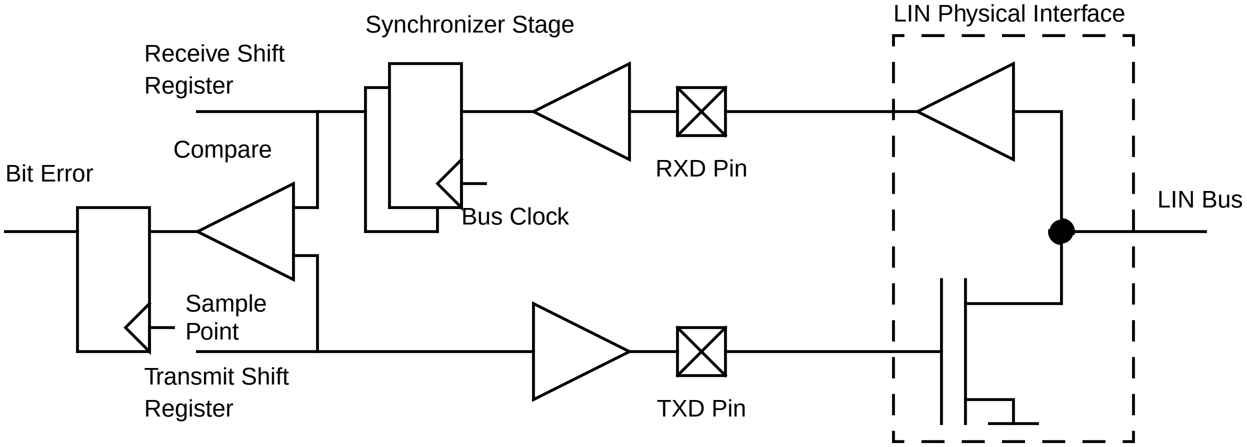


Figure 15-18. Collision Detect Principle

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.

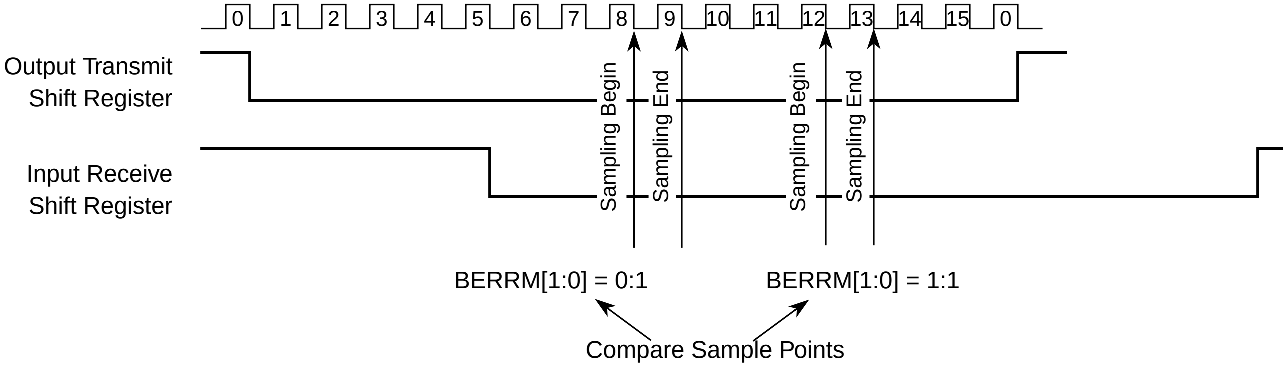


Figure 15-19. Timing Diagram Bit Error Detection

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.

15.4.6 Receiver

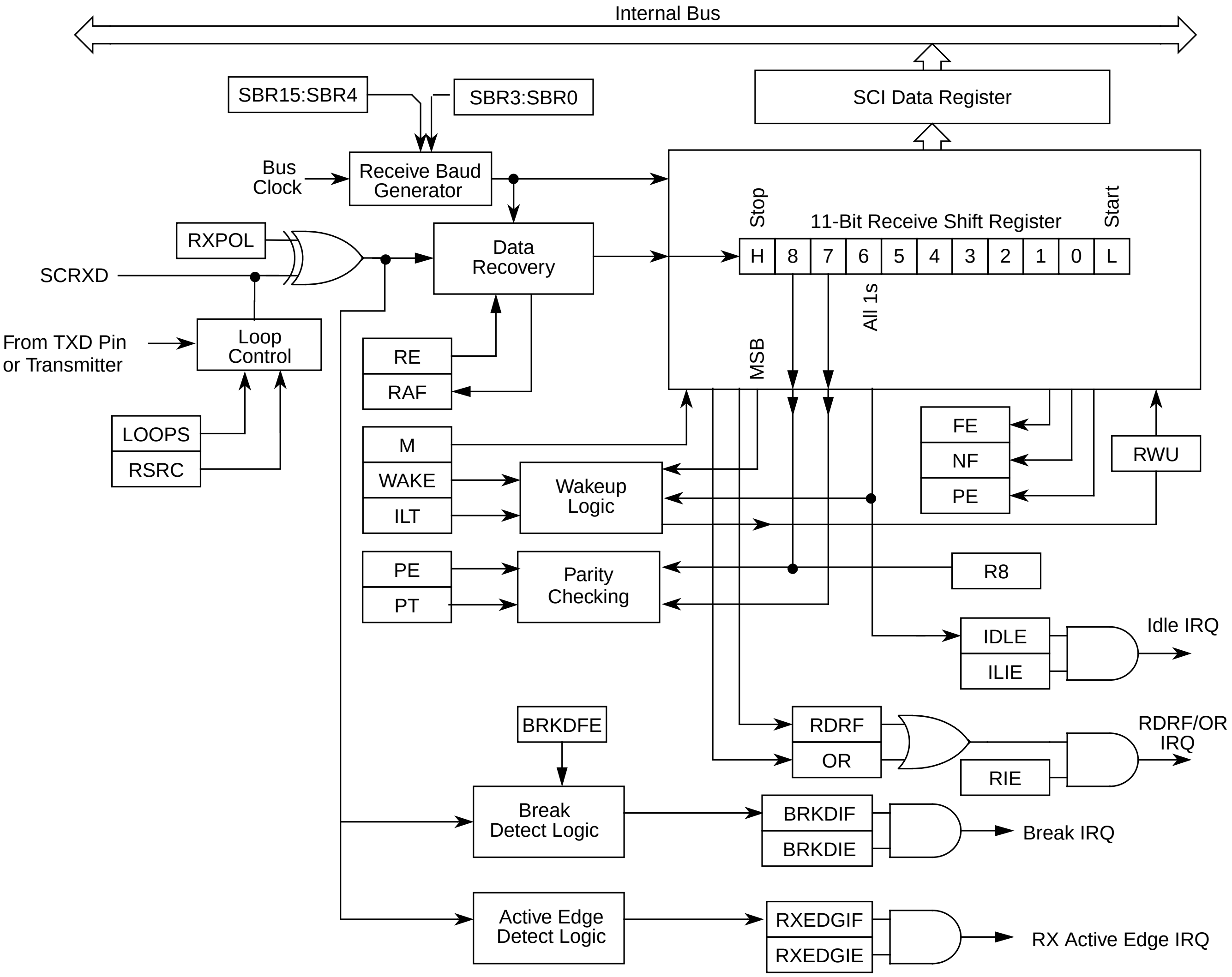


Figure 15-20. SCI Receiver Block Diagram

15.4.6.1 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).

15.4.6.2 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.

15.4.6.3 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 15-21) 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.

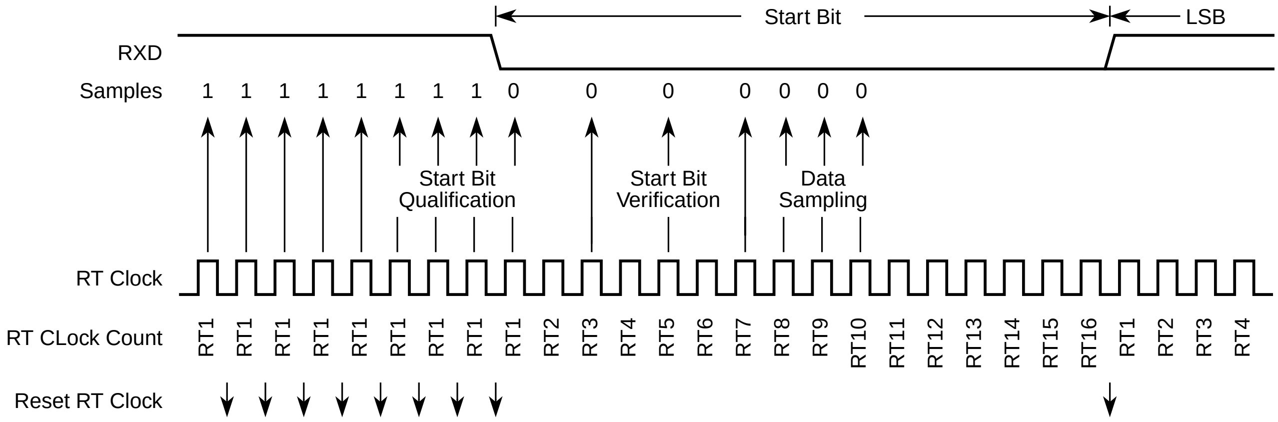


Figure 15-21. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Figure 15-17 summarizes the results of the start bit verification samples.

Table 15-17. 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 15-18 summarizes the results of the data bit samples.

Table 15-18. 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 15-19 summarizes the results of the stop bit samples.

Table 15-19. 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 15-22 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.

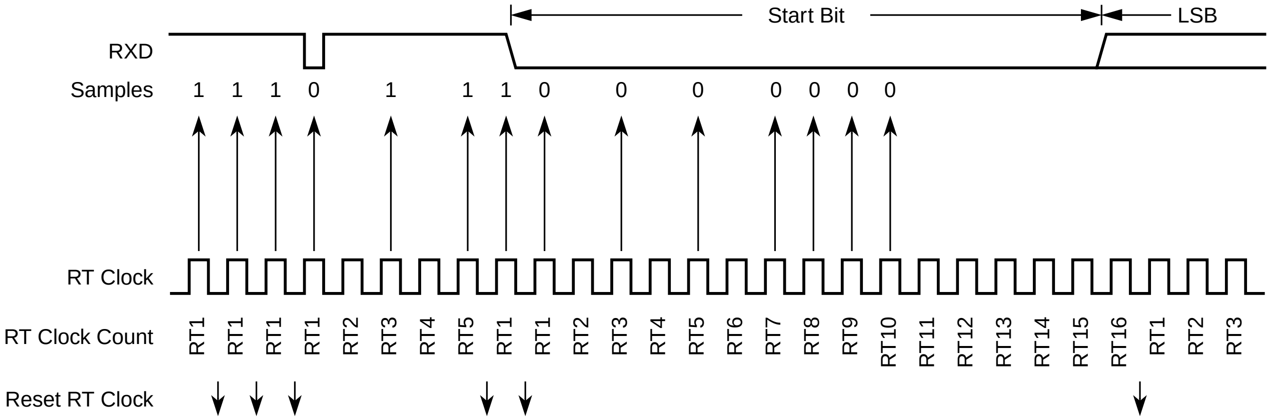


Figure 15-22. Start Bit Search Example 1

In Figure 15-23, 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.

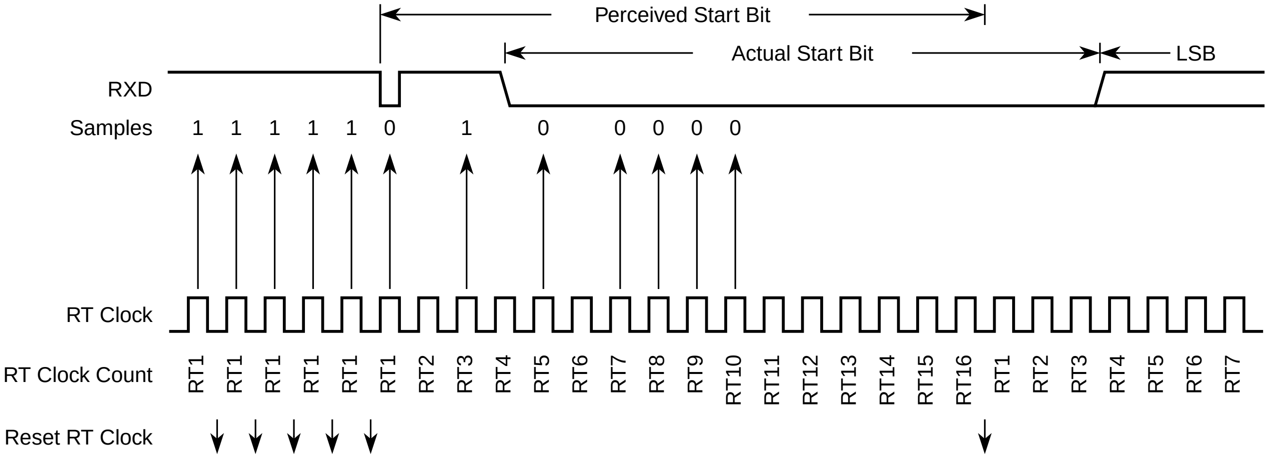


Figure 15-23. Start Bit Search Example 2

In Figure 15-24, 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.

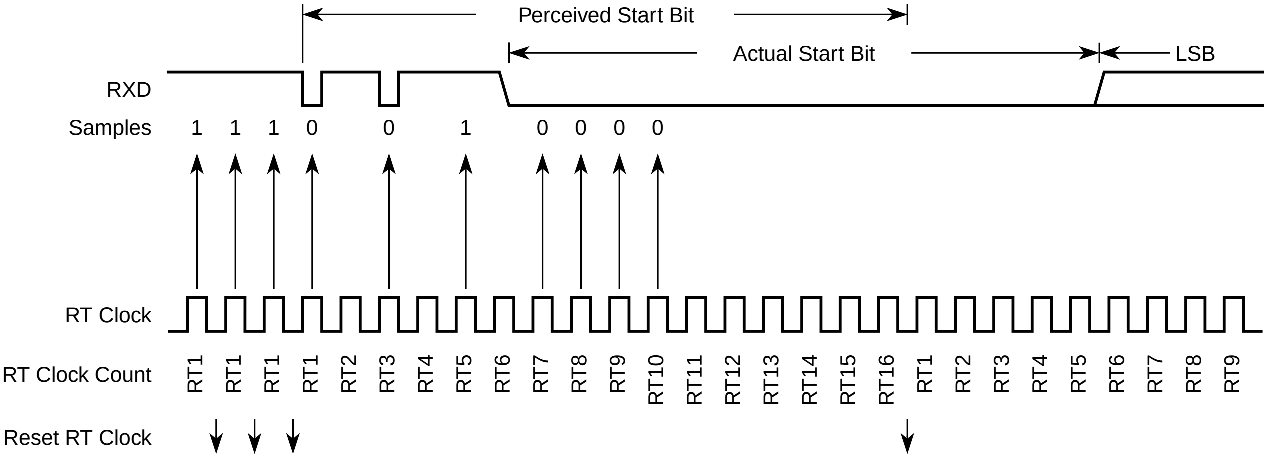


Figure 15-24. Start Bit Search Example 3

Figure 15-25 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.

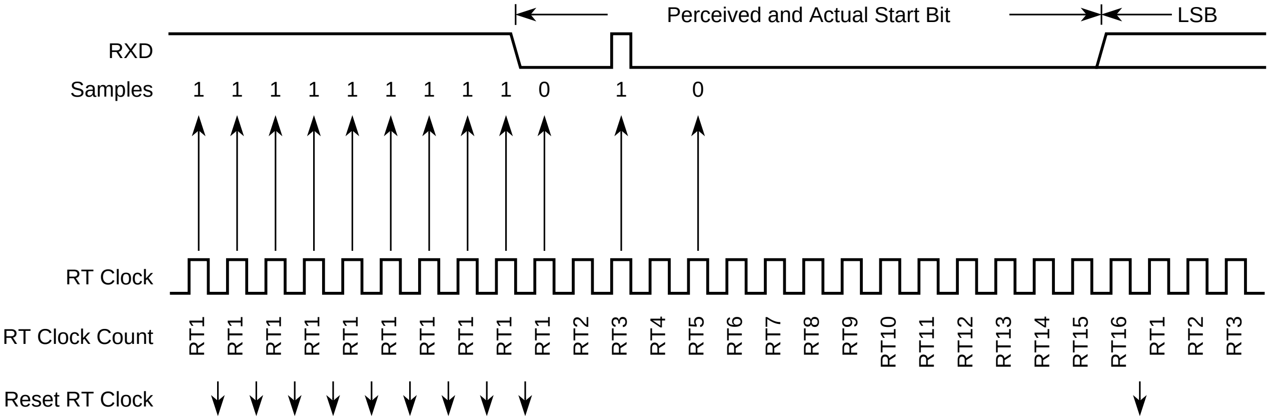


Figure 15-25. Start Bit Search Example 4

Figure 15-26 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.

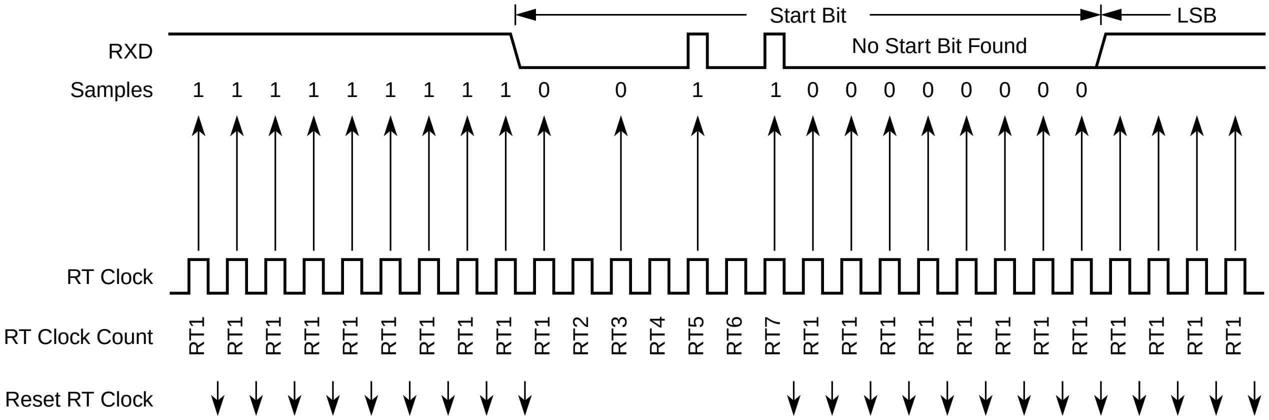


Figure 15-26. Start Bit Search Example 5

In Figure 15-27, 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.

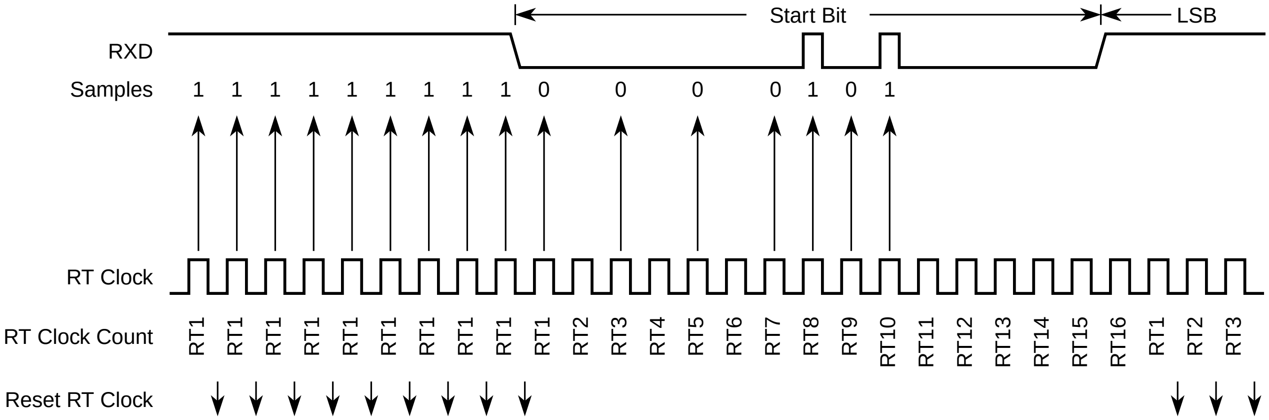


Figure 15-27. Start Bit Search Example 6

15.4.6.4 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.

15.4.6.5 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.

15.4.6.5.1 Slow Data Tolerance

Figure 15-28 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.

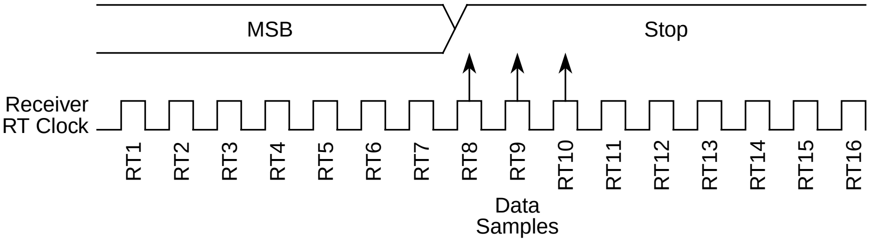


Figure 15-28. Slow Data

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 15-28, 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 15-28, 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:

$$((167 - 160) / 167) \text{ X } 100 = 4.19\%$$

15.4.6.5.2 Fast Data Tolerance

Figure 15-29 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.

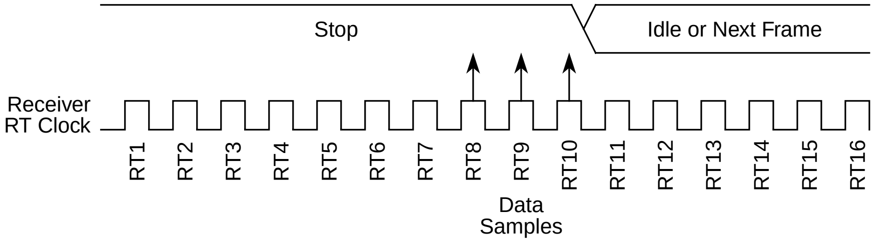


Figure 15-29. Fast Data

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 15-29, 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 15-29, 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%.

15.4.6.6 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.

15.4.6.6.1 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).

15.4.6.6.2 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.

15.4.7 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.

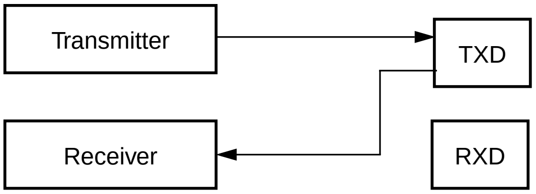


Figure 15-30. Single-Wire Operation (LOOPS = 1, RSRC = 1)

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.

15.4.8 Loop Operation

In loop operation the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI.

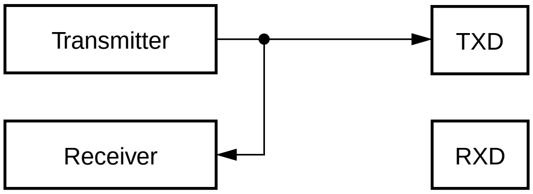


Figure 15-31. Loop Operation (LOOPS = 1, RSRC = 0)

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.

15.5 Initialization/Application Information

15.5.1 Reset Initialization

See Section 15.3.2, “Register Descriptions”.

15.5.2 Modes of Operation

15.5.2.1 Run Mode

Normal mode of operation.

To initialize a SCI transmission, see Section 15.4.5.2, “Character Transmission”.

15.5.2.2 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.

15.5.2.3 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.

15.5.3 Interrupt Operation

This section describes the interrupt originated by the SCI block.The MCU must service the interrupt requests. Table 15-20 lists the eight interrupt sources of the SCI.

Table 15-20. 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 15-20. 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.

15.5.3.1 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.

15.5.3.1.1 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).

15.5.3.1.2 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.

15.5.3.1.3 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).

15.5.3.1.4 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).

15.5.3.1.5 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).

15.5.3.1.6 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.

15.5.3.1.7 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.

15.5.3.1.8 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.

15.5.4 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

15.5.5 Recovery from Stop Mode

An active edge on the receive input can be used to bring the CPU out of stop mode.

Chapter 16
Serial Peripheral Interface (S12SPIV5)

Table 16-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V05.00	24 Mar 2005	16.3.2/16-601	- Added 16-bit transfer width feature.

16.1 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.

16.1.1 Glossary of Terms

SPI	Serial Peripheral Interface
$\overline{SS}$	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

16.1.2 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

16.1.3 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 16.4.7, “Low Power Mode Options”.

16.1.4 Block Diagram

Figure 16-1 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.

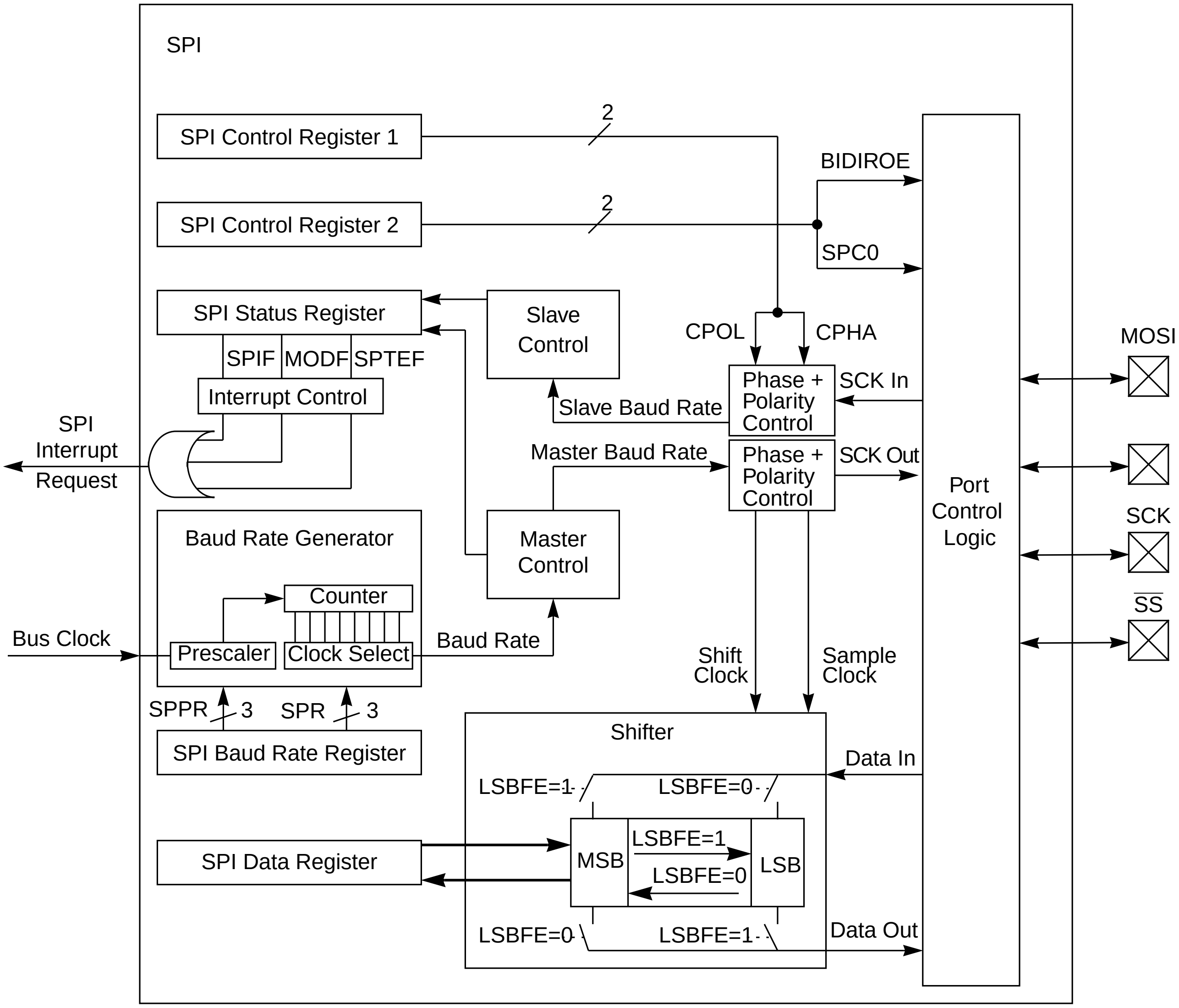


Figure 16-1. SPI Block Diagram

16.2 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.

16.2.1 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.

16.2.2 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.

16.2.3 $\overline{\text{SS}}$ — 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.

16.2.4 SCK — Serial Clock Pin

In master mode, this is the synchronous output clock. In slave mode, this is the synchronous input clock.

16.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

16.3.1 Module Memory Map

The memory map for the SPI is given in Figure 16-2. 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 16-2. SPI Register Summary

16.3.2 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.

16.3.2.1 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 16-3. SPI Control Register 1 (SPICR1)

Read: Anytime

Write: Anytime

Table 16-2. 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 16-2. SPICR1 Field Descriptions (continued)

Field	Description
1 SSOE	**Slave Select Output Enable** — The $\overline{SS}$ output feature is enabled only in master mode, if MODFEN is set, by asserting the SSOE as shown in Table 16-3. 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 16-3. $\overline{SS}$ Input / Output Selection

MODFEN	SSOE	Master Mode	Slave Mode
0	0	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
0	1	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
1	0	$\overline{SS}$ input with MODF feature	$\overline{SS}$ input
1	1	$\overline{SS}$ is slave select output	$\overline{SS}$ input

16.3.2.2 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 16-4. SPI Control Register 2 (SPICR2)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 16-4. 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 16.3.2.4, “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 16-3. 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 16-5. 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 16-5. 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	

16.3.2.3 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 16-5. SPI Baud Rate Register (SPIBR)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 16-6. 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 16-7. 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 16-7. 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:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 16-1}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor} \qquad \textit{Eqn. 16-2}$$

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

Table 16-7. 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
0	0	1	0	1	0	16	1.5625 Mbit/s

Table 16-7. 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	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
1	1	0	1	0	0	224	111.61 kbit/s

Table 16-7. 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	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

16.3.2.4 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 16-6. SPI Status Register (SPISR)

Read: Anytime

Write: Has no effect

Table 16-8. 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 16-9. 0 Transfer not yet complete. 1 New data copied to SPIDR.
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 16-10. 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 16.3.2.2, “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 16-9. SPIF Interrupt Flag Clearing Sequence

XFRW Bit	SPIF Interrupt Flag Clearing Sequence			
0	Read SPISR with SPIF == 1	**then**	Read SPIDRL	
1	Read SPISR with SPIF == 1	**then**	Byte Read SPIDRL (1)	
			or	
			Byte Read SPIDRH (2)	Byte Read SPIDRL
			or	
			Word Read (SPIDRH:SPIDRL)	

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 16-10. SPTEF Interrupt Flag Clearing Sequence

XFRW Bit	SPTEF Interrupt Flag Clearing Sequence			
0	Read SPISR with SPTEF == 1	**then**	Write to SPIDRL (1)	
1	Read SPISR with SPTEF == 1	**then**	Byte Write to SPIDRL [1](2)	
			or	
			Byte Write to SPIDRH [1](3)	Byte Write to SPIDRL [1]
			or	
			Word Write to (SPIDRH:SPIDRL) [1]	

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.

16.3.2.5 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 16-7. 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 16-8. 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 16-9).

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 16-10).

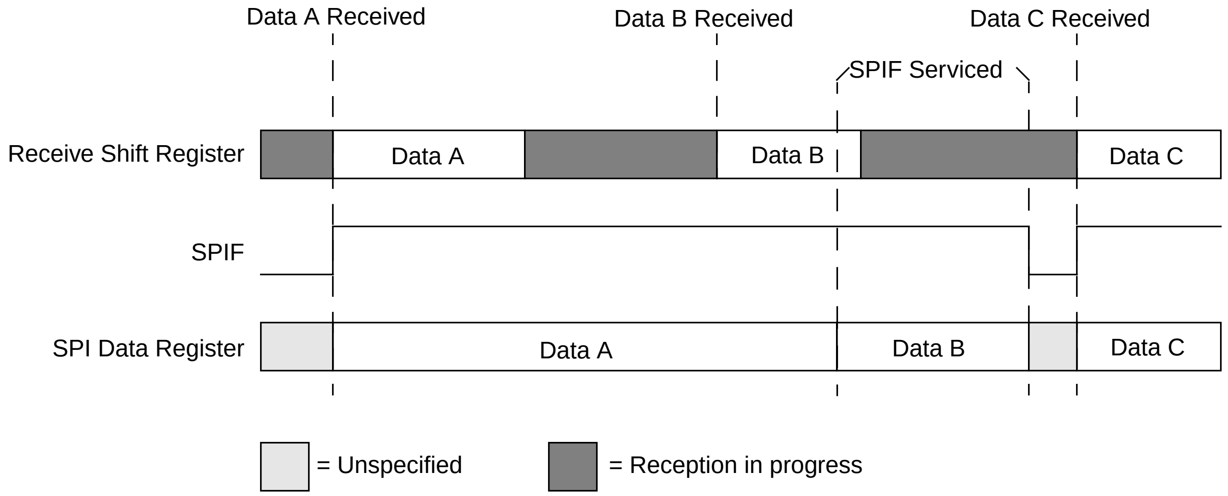


Figure 16-9. Reception with SPIF serviced in Time

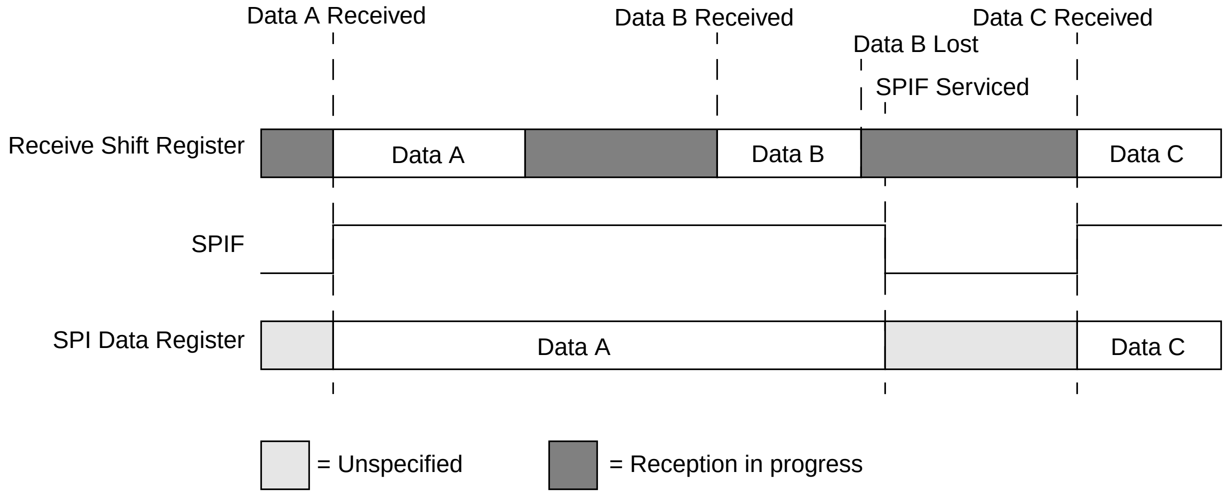


Figure 16-10. Reception with SPIF serviced too late

16.4 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 ($\overline{SS}$)
- 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 16.4.3, “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.

16.4.1 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.

  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

---

[1] n depends on the selected transfer width, please refer to Section 16.3.2.2, “SPI Control Register 2 (SPICR2)

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 16.4.3, “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.

16.4.2 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 $\overline{\text{SS}}$ 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.

16.4.3 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.

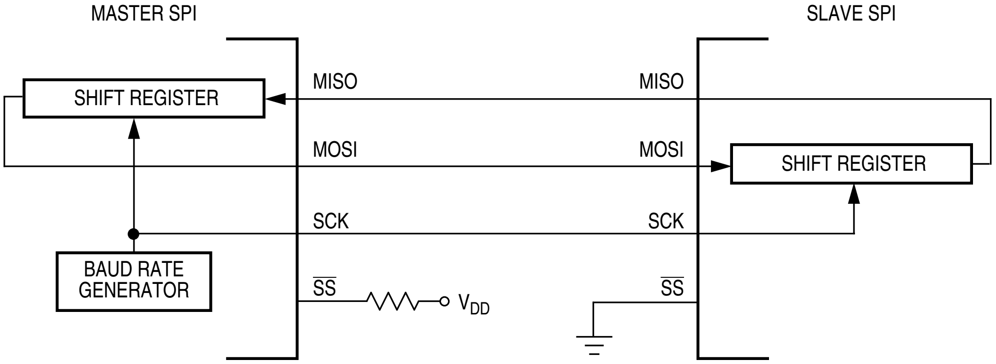


Figure 16-11. Master/Slave Transfer Block Diagram

16.4.3.1 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.

---

[1] n depends on the selected transfer width, please refer to Section 16.3.2.2, "SPI Control Register 2 (SPICR2)

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.

16.4.3.2 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.

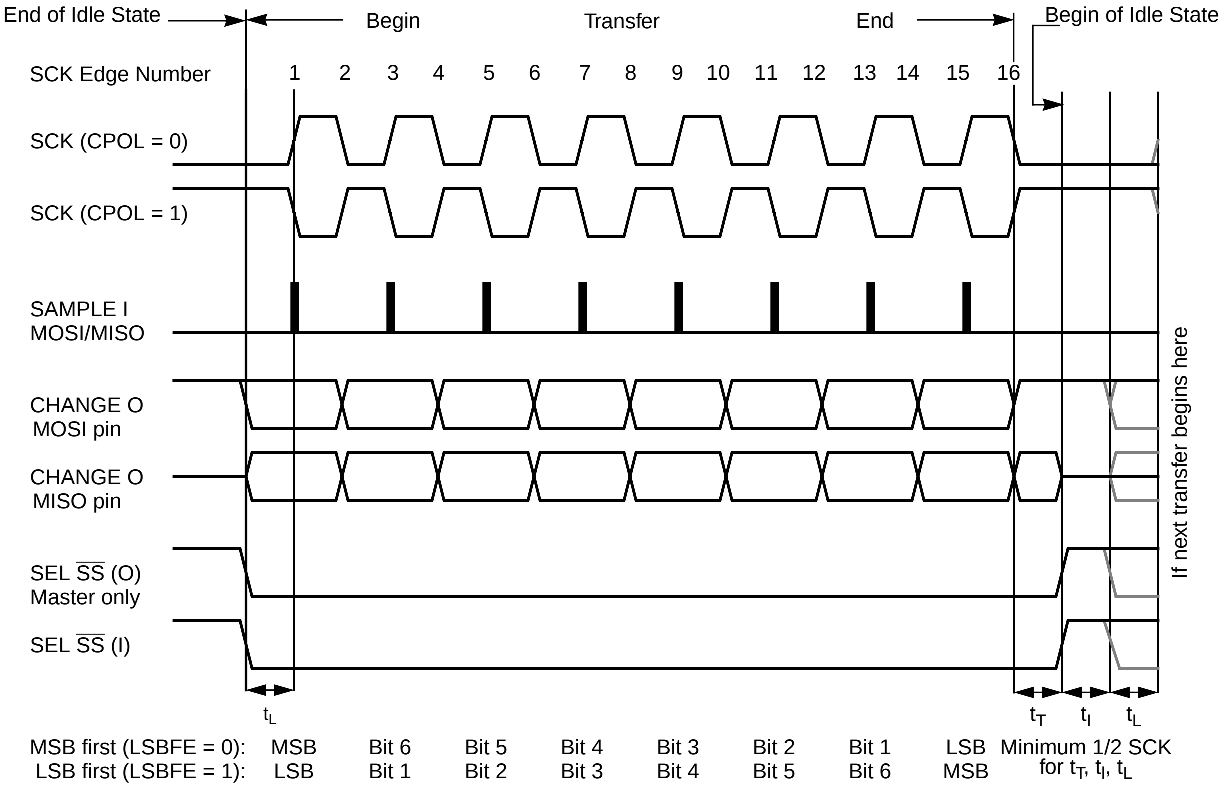
After 2n[1] (last) SCK edges:

- 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 16-12 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.

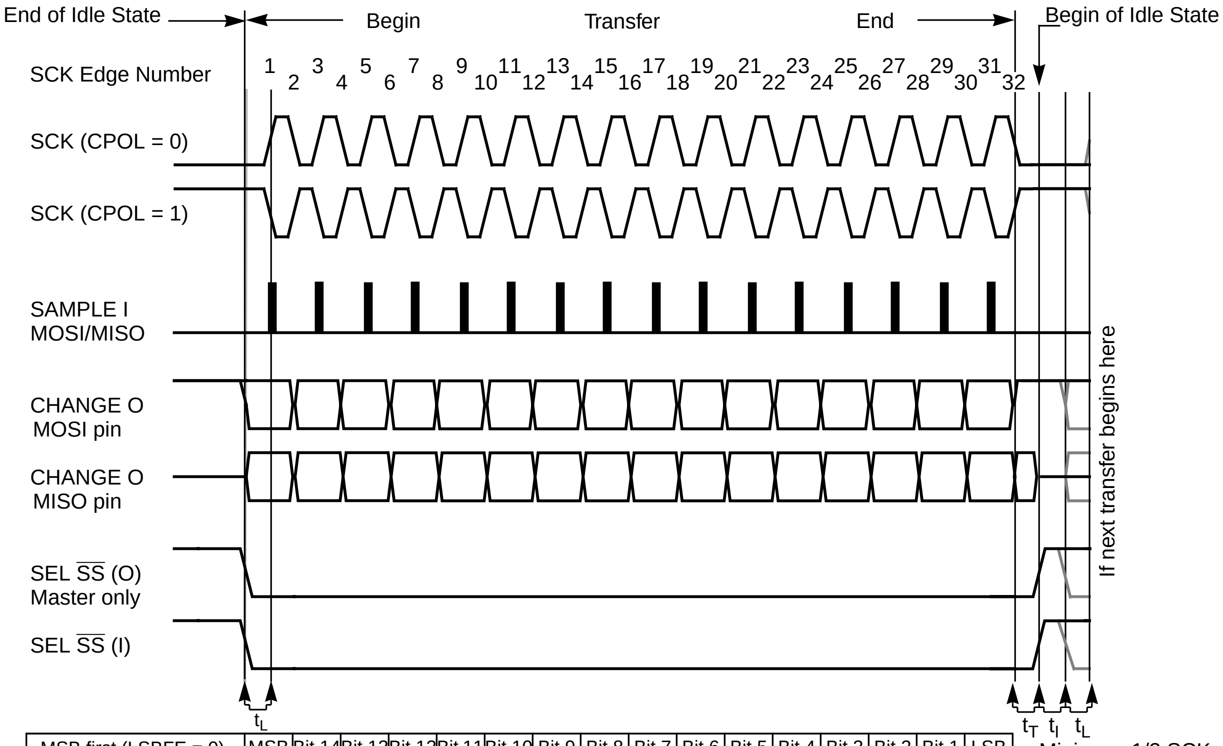
---

[1] n depends on the selected transfer width, please refer to Section 16.3.2.2, "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 $\overline{SS}$ high time)
t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 16-12. 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 16-13. 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.

16.4.3.3 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 16.3.2.2, “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.

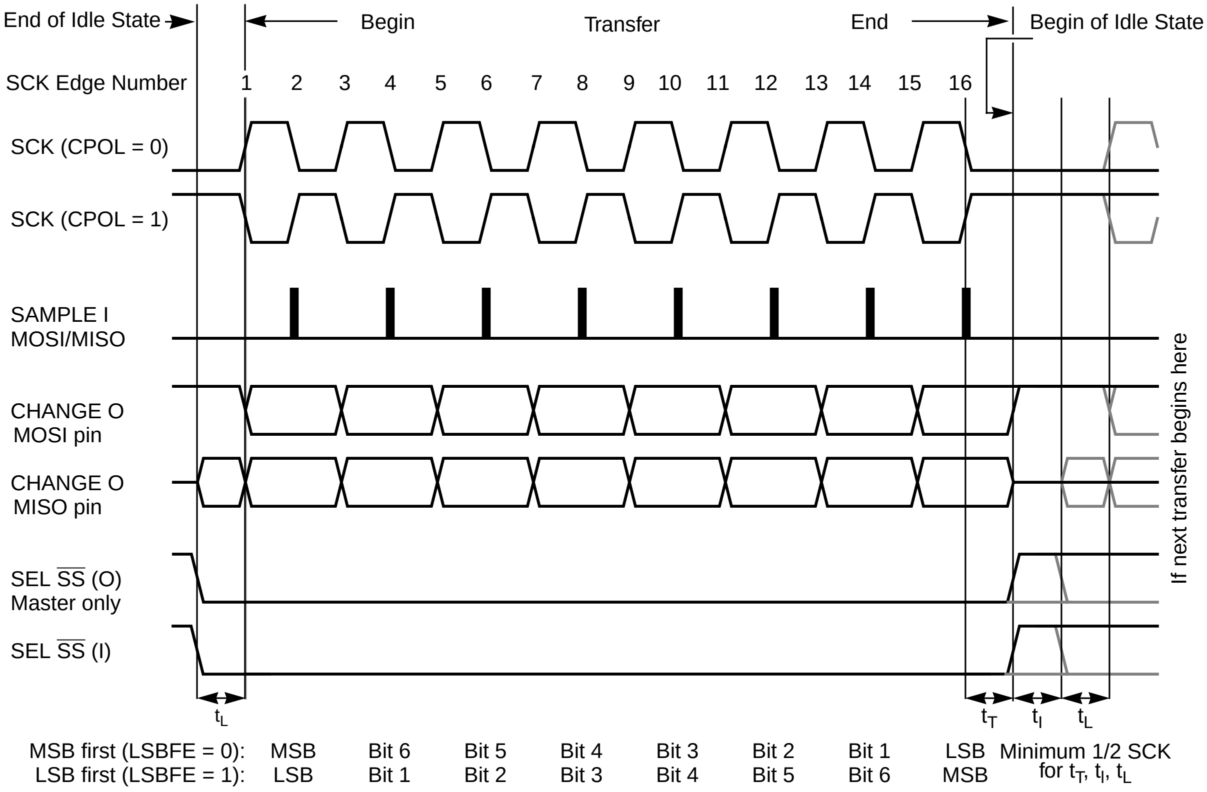
This process continues for a total of n^1 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

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 16-14 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 $\overline{\text{SS}}$ line is the slave select input to the slave. The $\overline{\text{SS}}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

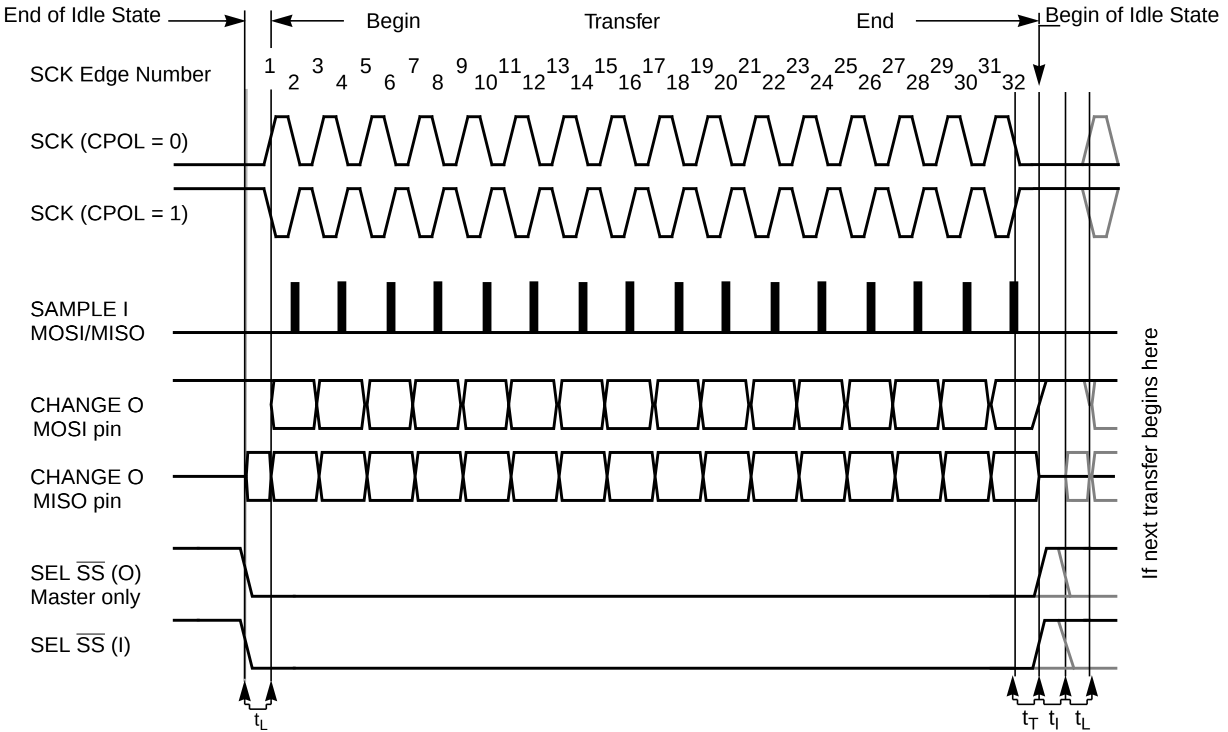


t_L = Minimum leading time before the first SCK edge, not required for back-to-back transfers

t_T = Minimum trailing time after the last SCK edge

t_I = Minimum idling time between transfers (minimum $\overline{\text{SS}}$ high time), not required for back-to-back transfers

Figure 16-14. 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 16-15. 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.

16.4.4 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 16-3.

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 16-3}$$

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 16-7 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.

16.4.5 Special Features

16.4.5.1 $\overline{\text{SS}}$ 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 16-3.

The mode fault feature is disabled while $\overline{\text{SS}}$ output is enabled.

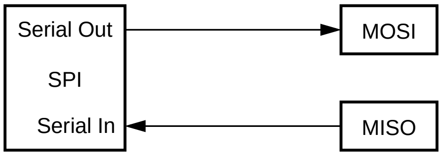
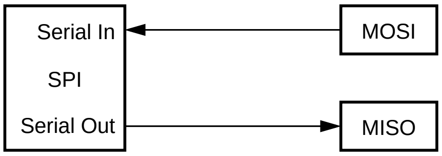
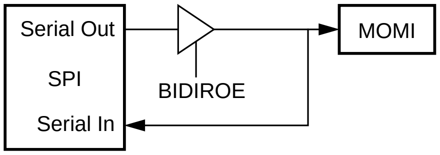
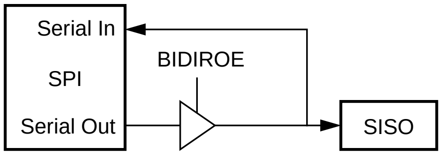
NOTE

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.

16.4.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see Table 16-11). 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 16-11. 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.

16.4.6 Error Conditions

The SPI has one error condition:

- Mode fault error

16.4.6.1 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.

16.4.7 Low Power Mode Options

16.4.7.1 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.

16.4.7.2 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.

16.4.7.3 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.

16.4.7.4 Reset

The reset values of registers and signals are described in Section 16.3, “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.

16.4.7.5 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.

16.4.7.5.1 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 16-3). 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 16.3.2.4, “SPI Status Register (SPISR)”.

16.4.7.5.2 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 16.3.2.4, “SPI Status Register (SPISR)”.

16.4.7.5.3 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 16.3.2.4, “SPI Status Register (SPISR)”.

Chapter 17
Gate Drive Unit (GDUV4)

Table 17-1. Revision History Table

4.0	27-May-2013		Changed overvoltage behaviourTable 17-22
4.1	27-June-2013		Corrected GLVLSS description Minor corrections
4.2	8-July-2013		Corrected GHHDIF description Reworked Table 1-24 fault protection summary Added Filter circiut and Note to Figure 1-19 Modified GOCA0 and GOCA1 bit descriptions
4.3	06-August-2013		Modified footnote 3 on table 1-22 (from design review) Added bulk diode to PMOS transistor in figure 1-19 (from design review) Included feedback from shared review
4.4	16-October-2013		Added 17.5.1/17-660 Minor updates from shared review
4.5	22-October-2013		Added Note to GFDE bit description Added RC filter to Figure 17-19 to avoid overshoot on VBS pin Added note to VLS and VLS_OUT pin description to add decouple capacitors
4.6	05-November-2013		changed RC filter to VBS in Figure 17-19

17.1 Introduction

The GDU module is a Field Effect Transistor (FET) pre-driver designed for three phase motor control applications.

17.1.1 Features

The GDU module includes these distinctive features:

- 11V voltage regulator for FET pre-drivers
- Boost converter option for low supply voltage condition
- 3-phase bridge FET pre-drivers
- Bootstrap circuit for high-side FET pre-drivers with external bootstrap capacitor
- Charge pump for static high-side driver operation
- Phase voltage measurement with internal ADC
- Two low-side current measurement amplifiers for DC phase current measurement
- Phase comparators for BEMF zero crossing detection in sensorless BLDC applications

- Voltage measurement on HD pin (DC-Link voltage) with internal ADC
- Desaturation comparator for high-side drivers and low-side drivers protection
- Undervoltage detection on FET pre-driver supply pin VLS
- Two overcurrent comparators with programmable voltage threshold
- Overvoltage detection on 3-phase bridge supply HD pin

17.1.2 Modes of Operation

The GDU module behaves as follows in the system power modes:

1. Run mode

   All GDU features are available.
2. Wait mode

   All GDU features are available.
3. Stop mode

   The GDU is disabled in stop mode. The high-side drivers, low-side drivers, charge pump, voltage regulator, boost circuit, and current sense amplifier are switched off. The bits in the GDUE register are cleared on entry into stop mode.The GDU will weakly pull the gates of the MOSFET to their source potential. On exit from stop mode the GDUE register bits are cleared. GFDE=0, GCPE=0, GBOE=0, GCSE1=0 and GCSE0=0.

NOTE

The device will not support putting the MOSFET in specific state during stop mode as GDU charge pump clock is not running. This means device can not be put in stop mode if FETS needs to be in specific state to protect the system from external energy supply (e.g. externally driven motor-generator).

17.1.3 Block Diagram

Figure shows a block diagram of the GDU module.

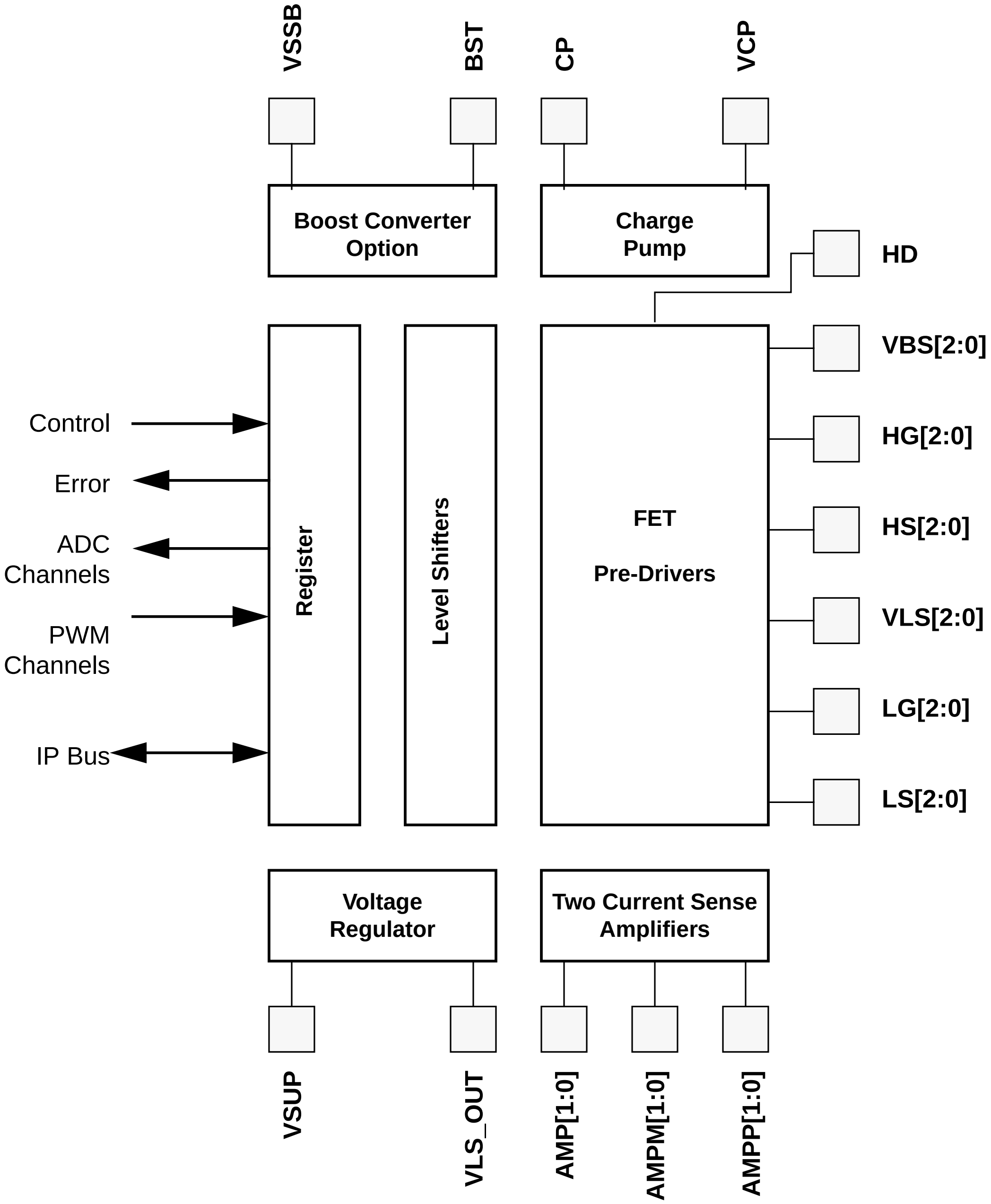


Figure 17-1. GDU Block Diagram

17.2 External Signal Description

17.2.1 HD — High-Side Drain Connection

This pin is the power supply for the 3-phase bridge (DC-link voltage).

NOTE

The HD pin should be connected as near as possible to the drain connections of the high-side MOSFETs.

17.2.2 VBS[2:0] — Bootstrap Capacitor Connection Pins

The pins are the bootstrap capacitor connections for phases HS[2:0]. The capacitor is connected between HS[2:0] and this pin. The bootstrap capacitor provides the gate voltage and current to drive the gate of the external power FET.

17.2.3 HG[2:0] — High-Side Gate Pins

The pins are the gate drives for the high-side power FETs. The drivers provide a high current with low impedance to turn on and off the high-side power FETs.

17.2.4 HS[2:0] — High-Side Source Pins

The pins are the source connection for the high-side power FETs and the drain connection for the low-side power FETs. The low voltage end of the bootstrap capacitor is also connected to this pin.

17.2.5 VLS[2:0] — Voltage Supply for Low-Side Pre-Drivers

The pins are the voltage supply pins for the three low-side FET pre-drivers. This pins should be connected to the voltage regulator output pin VLS_OUT. The output voltage on VLS_OUT pin is typically V_{VLS}=11V.

NOTE

It is recommended to add a 110nF-220nF X7R ceramic capacitor close to each VLS pin.

17.2.5.1 LG[2:0] — Low-Side Gate Pins

The pins are the gate drives for the low-side power FETs. The drivers provide a high current with low impedance to turn on and off the the low-side power FETs.

17.2.5.2 LS[2:0] — Low-Side Source Pins

The pins are the low-side source connections for the low-side power FETs. The pins are the power ground pins used to return the gate currents from the low-side power FETs.

17.2.5.3 AMPP[1:0] — Current Sense Amplifier Non-Inverting Input Pins

These pins are the non-inverting inputs to the current sense amplifiers.

17.2.5.4 AMPM[1:0] — Current Sense Amplifier Inverting Input Pins

These pins are the inverting inputs to the current sense amplifiers.

17.2.5.5 AMP[1:0] — Current Sense Amplifier Output Pins

These pins are the outputs of the current sense amplifiers. At the MCU level these pins are shared with ADC channels. For ADC channel assignment, see MCU pinout section.

17.2.5.6 CP — Charge Pump Output Pin

This pin is the switching node of the charge pump circuit. The supply voltage for charge pump driver is the output of the voltage regulator V_{VLS}. The output voltage of this pin switches typically between 0V and 11V.

17.2.5.7 VCP — Charge Pump Input for High-Side Driver Supply

This pin is the charge pump input for the high-side FET pre-driver supply VBS[2:0].

17.2.5.8 BST — Boost Converter Pin

This pin provides the basic switching elements required to implement a boost converter for low battery voltage conditions. This requires external diodes, capacitors and a coil.

17.2.5.9 VSSB — Boost Ground Pin

This pin is a separate power ground pin for the on chip boost converter switching device.

17.2.5.10 VSUP — Battery Voltage Supply Input Pin

This pin should be connected to the battery voltage. It is the input voltage to the integrated voltage regulator. The output of the voltage regulator is pin VLS_OUT.

17.2.5.11 VLS_OUT — Voltage Regulator Output Pin

This pin is the output of the integrated voltage regulator. The ouput voltage is typically V_{VLS}=11V. The input voltage to the voltage regulator is the VSUP pin.

NOTE

A 4.7uF or 10uF capacitor should be connected to this pin for stability of the the voltage regulator output.

17.3 Memory Map and Register Definition

This section provides the detailed information of all registers for the GDU module.

17.3.1 Register Summary

Figure 17-2 shows the summary of all implemented registers inside the GDU 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 GDUE	R	GWP	0	EPRES	GCSE1	GBOE	GCSE0	GCPE	GFDE
	W		▨						
0x0001 GDUCTR	R	GHHDLVL	0	GBKTIM2[3:0]				GBKTIM1[1:0]	
	W		▨						
0x0002 GDUIE	R	0	0	0	GOCIE[1:0]		GDSEIE	GHHDIE	GLVLSIE
	W	▨	▨	▨					
0x0003 GDUDSE	R	0	GDHSIF[2:0]			0	GDLSIF[2:0]		
	W	▨				▨			
0x0004 GDUSTAT	R	GPHS[2:0]			GOCS[1:0]			GHHDS	GLVLSS
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x0005 GDUSRC	R	0	GSRCHS[2:0]			0	GSRCLS[2:0]		
	W	▨				▨			
0x0006 GDUF	R	GSUF	GHHDF	GLVLSF	GOCIF[1:0]		0	GHHDIF	GLVLSIF
	W						▨		
0x0007 GDUCLK1	R	0	GBOCD[4:0]					GBODC[1:0]	
	W	▨							
0x0008 GDUBCL	R	0	0	0	0	GBCL[3:0]			
	W	▨	▨	▨	▨				
0x0009 GDUPHMUX	R	0	0	0	0	0	0	GPHMX[1:0]	
	W	▨	▨	▨	▨	▨	▨		

▨ = Unimplemented

Figure 17-2. GDU Register Summary

Address Offset Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000A GDUCSO	R	0	GCSO1[2:0]			0	GCSO0[2:0]		
	W	▨				▨			
0x000B GDUDSLVL	R	0	GDSLHS[2:0]			0	GDSLLS[2:0]		
	W	▨				▨			
0x000C GDUPHL	R	0	0	0	0	0	GPHL[2:0]		
	W	▨	▨	▨	▨	▨	▨	▨	▨
0x000D GDUCLK2	R	0	0	0	0	GCPCD[3:0]			
	W	▨	▨	▨	▨				
0x000E GDUOC0	R	GOCA0	GOCE0	0	0	GOCT0[3:0]			
	W			▨	▨				
0x000F GDUOC1	R	GOCA1	GOCE1			GOCT1[3:0]			
	W			▨	▨				
0x0010-0x001F		▨	▨	▨	▨	▨	▨	▨	▨
		▨	▨	▨	▨	▨	▨	▨	▨

▨ = Unimplemented

Figure 17-2. GDU Register Summary

17.3.2 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.

17.3.2.1 GDU Module Enable Register (GDUE)

Module Base + 0x0000

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	GWP	0	EPRES	GCSE1	GBOE	GCSE0	GCPE	GFDE
W		▨						
Reset	0	0	0	0	0	0	0	1

▨ = Unimplemented

Figure 17-3. GDU Module Enable Register (GDUE)

1. Read: Anytime
   Write: Anytime, write protected bits only if GWP=0. On entry in stop mode bits GCSE0, GCSE1, GBOE, GCPE & GFDE are cleared. After exit from stop mode write protected bits GBOE, GCPE & GFDE can be written once when GWP=1.

Table 17-2. GDUE Register Field Description

Field	Description
7 GWP	**GDU Write Protect**— This bit enables write protection to be used for the write protectable bits. While clear, GWP allows write protectable bits to be written. When set GWP prevents any further writes to write protectable bits. Once set , GWP is cleared by reset. 0 Write-protectable bits may be written 1 Write-protectable bits cannot be written
5 EPRES	**Enable High-Side Driver Preserve Functionality** — This bit controls the bootstrap charge preserve mode of the high-side drivers. If EPRES is set and the high-side driver is switched off the input current of the VBSx pins is reduced by typically 100uA in order to preserve the charge on the bootstrap capacitor. This bit cannot be modified after GWP bit is set 0 High-side driver preserve functionality disabled 1 High-side driver preserve functionality enabled
4 GCSE1	**GDU Current Sense Amplifier 1 Enable**— This bit enables the current sense amplifier. See 17.4.8Section 17.4.8, “Current Sense Amplifier and Overcurrent Comparator 0 Current sense amplifier 1 is disabled 1 Current sense amplifier 1 is enabled
3 GBOE	**GDU Boost Converter Enable** — This bit enables the boost option. This bit cannot be modified after GWP bit is set. See Section 17.4.10, “Boost Converter 0 Boost option is disabled 1 Boost option is enabled
2 GCSE0	**GDU Current Sense Amplifier 0 Enable**— This bit enables the current sense amplifier. See Section 17.4.8, “Current Sense Amplifier and Overcurrent Comparator 0 Current sense amplifier 0 is disabled 1 Current sense amplifier 0 is enabled

Table 17-2. GDUE Register Field Description

Field	Description
1 GCPE	**GDU Charge Pump Enable** — This bit enables the charge pump. This bit cannot be modified after GWP bit is set. See Section 17.4.4, “Charge Pump 0 Charge pump is disabled 1 Charge pump is enabled
0 GFDE	**GDU FET Pre-Driver Enable** — This bit enables the low-side and high-side FET pre-drivers. This bit cannot be modified after GWP bit is set.See Section 17.4.2, “Low-Side FET Pre-Drivers and Section 17.4.3, “High-Side FET Pre-Driver. 0 Low-side and high-side drivers are disabled 1 Low-side and high-side drivers are enabled **NOTE** It is not allowed to set and clear GFDE bit periodically in order to switch on and off the FET pre-drivers. In order to switch on and off the FET pre-drivers the PMF module has to be used to mask and un-mask the PWM channels.

17.3.2.2 GDU Control Register (GDUCTR)

Module Base + 0x0001 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	GHHDLVL	0	GBKTIM2[3:0]				GBKTIM1[1:0]	
W		▨						
Reset	0	0	0	1	0	0	0	0

▨ = Unimplemented

Figure 17-4. GDU Control Register (GDUCTR)

1. Read: Anytime
   Write: Only if GWP=0

Table 17-3. GDUCTR Register Field Descriptions

Field	Description
7 GHHDLVL	**GDU High HD Level Select** — Selects the voltage threshold of the overvoltage detection on HD pin. This bit cannot be modified after GWP bit is set. 0 Voltage threshold of the overvoltage detection on HD pin = V_{HVHDL} 1 Voltage threshold of the overvoltage detection on HD pin = V_{HVHDH}
5-2 GBKTIM2[3:0]	**GDU Blanking Time** — These bits adjust the blanking time t_{BLANK} of the desaturation error comparators. The resulting blanking time t_{BLANK} can be calculated from the equation below. For GBKTIM2[3:0]=$F no desaturation errors are captured and the drivers are unprotected and the charge pump will not connect to the high-side drivers. These bits cannot be modified after GWP bit is set. $$t_{BLANK} = [((GBKTIM2 + 1) \cdot 2^{GBKTIM1 + 1}) + 2] \cdot T_{BUS}$$
1-0 GBKTIM1[1:0]	**GDU Blanking Time** — These bits adjust the blanking time t_{BLANK} of the desaturation error comparators. The resulting blanking time t_{BLANK} can be calculated from the equation in the field description below.These bits cannot be modified after GWP bit is set.

NOTE

The register bits GBKTIM1 and GBKTIM2 must be set to the required values before the PWM channel is activated. Once the PWM channel is activated, the value of GBKTIM1 & GBKTIM2 must not change. If a different blanking time is required, the PWM channel has to be turned off before new values to GBKTIM1 & GBKTIM2 are written.

17.3.2.3 GDU Interrupt Enable Register (GDUIE)

Module Base + 0x0002

Access: User read only[1]

	7	6	5	4	3	2	1	0
R	0	0	0	GOCIE[1:0]		GDSEIE	GHHDIE	GLVLSIE
W	▨	▨	▨					
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-5. GDU Interrupt Enable Register (GDUIE)

1. Read: Anytime
Write: Anytime

Table 17-4. GDUIE Register Field Descriptions

Field	Description
4-3 GOCIE[1:0]	**GDU Overcurrent Interrupt Enable** — Enables overcurrent interrupt. 0 No interrupt will be requested if any of the flags GOCIF[1:0] in the GDUF register is set 1 Interrupt will be requested if any of the flags GOCIF[1:0] in the GDUF register is set
2 GDSEIE	**GDU Desaturation Error Interrupt Enable** — Enables desaturation error interrupt on low-side or high-side drivers 0 No interrupt will be requested if any of the flags in the GDUDSE register is set 1 Interrupt will be requested if any of the flags in the GDUDSE register is set
1 GHHDIE	**GDU High HD Interrupt Enable** — Enables the high HD interrupt. 0 No interrupt will be requested whenever GHHDIF flag is set 1 Interrupt will be requested whenever GHHDIF flag is set
0 GLVLSIE	**GDU Low VLS Interrupt Enable** — Enables the interrupt which indicates low VLS supply 0 No interrupt will be requested whenever GLVLSIF flag is set 1 Interrupt will be requested whenever GLVLSIF flag is set

17.3.2.4 GDU Desaturation Error Flag Register (GDUDSE)

Module Base + 0x0003

Access: User read/write[1]

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

▨ = Unimplemented

Figure 17-6. GDU Desaturation Error Flag Register (GDUDSE)

1. Read: Anytime
Write: Anytime, write 1 to clear

Table 17-5. GDUDSE Register Field Descriptions

Field	Description
6-4 GDHSIF[2:0]	**GDU High-Side Driver Desaturation Interrupt Flags** — The flag is set by hardware to “1” when a desaturation error on associated high-side driver pin HS[2:0] occurs. If the GDSEIE bit is set an interrupt is requested. Writing a logic “1” to the bit field clears the flag. 0 No desaturation error on high-side driver 1 Desaturation error on high-side driver
2-0 GDLSIF[2:0]	**GDU Low-Side Driver Desaturation Interrupt Flag** — The flag is set to “1” when a desaturation error on associated low-side driver pin LS[2:0] occurs. If the GDSEIE bit is set an interrupt is requested. Writing a logic “1” to the bit field clears the flag. 0 No desaturation error on low-side driver 1 Desaturation error on low-side driver

17.3.2.5 GDU Status Register (GDUSTAT)

Module Base + 0x0004 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	GPHS[2:0]			GOCS[1:0]		0	GHHDS	GLVLSS
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-7. GDU Status Register (GDUSTAT)

1. Read: Anytime
   Write: Never

Table 17-6. GDUSTAT Register Field Descriptions

Field	Description
7-5 GPHS[2:0]	**GDU Phase Status** — The status bits are set to 1 when the voltage on associated pin HS[2:0] is greater than $V_{HD}/2$. The flags are cleared when the voltage on associated pin HS[2:0] is less than $V_{HD}/2$. See Section 17.4.6, “Phase Comparators” 0 Voltage on pin HSx is $V_{HSx} < V_{HD}/2$ 1 Voltage on pin HSx is $V_{HSx} > V_{HD}/2$
4-3 GOCS[1:0]	**GDU Overcurrent Status** — The status bits are set to 1 when the voltage on the overcurrent comparator input is above the threshold voltage V_{OCT}. The flag is cleared when the voltage on the overcurrent comparator input is less than V_{OCT}.Section 17.4.8, “Current Sense Amplifier and Overcurrent Comparator 0 Voltage on overcurrent comparator input is is less than V_{OCT} 1 Voltage on overcurrent comparator is greater than V_{OCT}
1 GHHDS	**GDU High HD Supply Status** — The status bit is set to 1 when the voltage on HD pin is above the threshold voltage V_{HVHDLA} or V_{HVHDHA} depending on the value of the GHHDLVL bit. The flag is cleared when the voltage on HD pin is less than V_{HVHDLD} or V_{HVHDHD} depending on the value of the GHHDLVL bit. 0 Voltage on pin HD is less than V_{HVHDLD} or V_{HVHDHD} 1 Voltage on pin HD is greater than V_{HVHDLA} or V_{HVHDHA}
0 GLVLSS	**GDU Low VLS Status** — The status bit is set to 1 when the voltage on VLS_OUT pin is below the threshold voltage V_{LVLSA}. The flag is cleared when the voltage on VLS_OUT pin is greater than V_{LVLSD}. 0 Voltage on pin VLS_OUT is greater than V_{LVLSD} 1 Voltage on pin VLS_OUT is less than V_{LVLSA}

17.3.2.6 GDU Slew Rate Control Register (GDUSRC)

Module Base + 0x0005 Access: User read/write(1)

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

▨ = Unimplemented

Figure 17-8. GDU Slew Rate Control Register (GDUSRC)

1. Read: Anytime
   Write: Only if GWP=0

Table 17-7. GDU Slew Rate Control Register Field Descriptions

Field	Description
6:4 GSRCHS[2:0]	**GDU Slew Rate Control Bits High-Side FET Pre-Drivers** — These bits control the slew rate on the HG[2:0] pins (see FET Pre-Driver Details) .These bits cannot be modified after GWP bit is set. 000 : slowest . . 111 : fastest
3:0 GSRCLS[2:0]	**GDU Slew Rate Control Bits Low-Side FET Pre-Drivers** — These bits control the slew rate on the LG[2:0] pins (see FET Pre-Driver Details). These bits cannot be modified after GWP bit is set. 000 : slowest . . 111 : fastest

17.3.2.7 GDU Flag Register (GDUF)

Module Base + 0x0006 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	GSUF	GHHDF	GLVLSF	GOCIF[1:0]		0	GHHDIF	GLVLSIF
W						▨		
Reset	1	0	0(2)	0	0	0	0	0^2

▨ = Unimplemented

Figure 17-9. GDU Flag Register (GDUF)

1. Read: Anytime
   Write: Anytime, write 1 to clear flag

2. Out of power on reset the flags may be set

Table 17-8. GDUF Register Field Descriptions

Field	Description
7 GSUF	**GDU Start-up Flag** — The start-up flag is set by hardware on the first clock edge after system reset deasserts. Writing a logic “1” to the bit field clears the flag one bus cycle after writing “1”. If the flag is set all high-side FET pre-drivers are turned off and all low-side FET pre-drivers are turned on. If the flag is cleared and there is no error condition present all high-side and low-side FET pre-drivers are driven by the pwm channels. 0 High-side and low-side FET pre-drivers are driven by pwm channels 1 High-side FET pre-drivers turned off and low-side FET pre-drivers are turned on
6 GHHDF	**GDU High V_{HD} Supply Flag** — The flag controls the state of the FET pre-drivers. If the flag is set and GOCA1=0 the high-side pre-drivers are turned off and the low-side pre-drivers are turned on. If GOCA1=1 all high-side and low-side FET pre-drivers are turned off. If the flag is cleared and no other error condition is present the high-side and low-side pre-drivers are driven by the PWM channels. The flag is set by hardware if a high voltage condition on HD pin occurs. The flag is set if the voltage on pin HD is greater than the threshold voltage V_{HVHDLA} or V_{HVHDHA} . Writing a logic “1” to the bit field clears the flag. 0 Voltage on pin HD is less than V_{HVHDLD} or V_{HVHDHD} 1 Voltage on pin HD is greater than V_{HVHDLA} or V_{HVHDHA}
5 GLVLSF	**GDU Low VLS Supply Flag** — The flag controls the state of the FET pre-drivers. If the flag is set all high-side and low-side pre-drivers are turned off. If the flag is cleared and no other error condition is present the high-side and low-side pre-drivers are driven by the PWM channels. The flag is set by hardware if a low voltage condition on VLS_OUT pin occurs. The flag is set if the voltage on pin VLS drops below the threshold voltage V_{LVLSA}. Writing a logic “1” to the bit field clears the flag. 0 VLS Supply is above V_{LVLSD} 1 VLS Supply is below $V_{LVLSA,}$ all high-side and low-side FET pre-drivers are turned off
4-3 GOCIF[1:0]	**GDU Overcurrent Interrupt Flag** — The interrupt flags are set by hardware if an overcurrent condition occurs. The flags are set if the voltage on the overcurrent comparator input is greater than the threshold voltage V_{OCT}. If the GOCIE bit is set an interrupt is requested. Writing a logic “1” to the bit field clears the flag. If the GOCAx bit is cleared all high-side FET pre-drivers are turned off and fault[4] is asserted. If GOCAx is set all high-side and low-side FET pre-drivers are turned off and fault[2:0] are asserted. 0 Voltage on overcurrent comparator input is less than V_{OCT} 1 Voltage on overcurrent comparator is greater than V_{OCT}

Table 17-8. GDUF Register Field Descriptions

Field	Description
1 GHHDIF	**GDU High V_{HD} Supply Interrupt Flag**— The interrupt flag is set by hardware if GHHDF is set or if GHHDS is cleared. If the GHHDIE bit is set an interrupt is requested. Writing a logic “1” to the bit field clears the flag.
0 GLVLSIF	**GDU Low VLS Supply Interrupt Flag**— The interrupt flag is set by hardware if GLVLSF is set or GLVLSS is cleared. If the GLVLSIE bit is set an interrupt is requested.Writing a logic “1” to the bit field clears the flag.

NOTE

The purpose of the GSUF flag is to allow dissipation of the energy in the motor coils through the low side FETs in case of short reset pulses whilst the motor is spinning.

17.3.2.8 GDU Clock Control Register 1 (GDUCLK1)

Module Base + 0x0007 Access: User read/write(1)

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

Figure 17-10. GDU Clock Control Register 1 (GDUCLK1)

1. Read: Anytime
Write: Anytime if GWP=0

Table 17-9. GDUCLK1 Register Field Descriptions

Field	Description
6-2 GBOCD[4:0]	**GDU Boost Option Clock Divider** — These bits select the clock divider factor which is used to divide down the bus clock frequency f_{BUS} for the boost converter clock f_{BOOST}. These bits cannot be modified after GWP bit is set. See Table 17-10 for divider factors. See also Section 17.4.10, “Boost Converter
1-0 GBODC[1:0]	**GDU Boost Option Clock Duty Cycle**— These bits select the duty cycle of the boost option clock f_{boost}. For GBOCD[4]= 0 the duty cycle of the boost option clock is always 50%. These bits cannot be modified after GWP bit is set. 00 Duty Cycle = 50% 01 Duty Cycle = 25% 10 Duty Cycle = 50% 11 Duty Cycle = 75%

NOTE

The GBODC & GBOCD register bits must be set to the required value before GBOE bit is set. If a different boost clock frequency and duty cycle is required GBOE has to be cleared before new values to GBODC & GBOCD are written.

Table 17-10. Boost Option Clock Divider Factors k = f_{BUS} / f_{BOOST}

GBOCD[4:0]	f_{BOOST}
00000	f_{BUS} / 4
00001	f_{BUS} / 4
00010	f_{BUS} / 4
00011	f_{BUS} / 4
00100	f_{BUS} / 4
00101	f_{BUS} / 4
00110	f_{BUS} / 6
00111	f_{BUS} / 6
01000	f_{BUS} / 8
01001	f_{BUS} / 8
01010	f_{BUS} / 10
01011	f_{BUS} / 10
01100	f_{BUS} / 12
01101	f_{BUS} / 12
01110	f_{BUS} / 14
01111	f_{BUS} / 14
10000	f_{BUS} / 16
10001	f_{BUS} / 24
10010	f_{BUS} / 32
10011	f_{BUS} / 48
10100	f_{BUS} / 64
10101	f_{BUS} / 96
10110	f_{BUS} / 100
10111	f_{BUS} / 128
11000	f_{BUS} / 192
11001	f_{BUS} / 200
11010	f_{BUS} / 256

Table 17-10. Boost Option Clock Divider Factors k = f_{BUS} / f_{BOOST}

GBOCD[4:0]	f_{BOOST}
11011	f_{BUS} / 384
11100	f_{BUS} / 400
11101	f_{BUS} / 512
11110	f_{BUS} / 768
11111	f_{BUS} / 800

17.3.2.9 GDU Boost Current Limit Register (GDUBCL)

Module Base + 0x0008 Access: User read/write[1]

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

Figure 17-11. GDU Boost Current Limit Register (GDUBCL)

1. Read: Anytime
   Write: Anytime if GWP=0

Table 17-11. GDU Boost Current Limit Register Field Descriptions

Field	Description
GBCL[3:0]	**GDU Boost Current Limit Register**— This bits are used to adjust the boost coil current limit $_{ICOIL0,16}$ on the BST pin. These bits cannot be modified after GWP bit is set. See GDU electrical parameters.

17.3.2.10 GDU Phase Mux Register (GDUPHMUX)

Module Base + 0x0009 Access: User read/write[1]

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

▨ = Unimplemented

Figure 17-12. GDU Phase Mux Register (GDUPHMUX)

1. Read: Anytime
   Write: Anytime

Table 17-12. GDU Phase Mux Register Field Descriptions

Field	Description
[1:0] GPHMUX	**GDU Phase Multiplexer** — These buffered bits are used to select the voltage which is routed to internal ADC channel.The value written to the GDUPHMUX register does not take effect until the LDOK bit is set and the next PWM reload cycle begins. Reading GDUPHMUX register reads the value in the buffer. It is not necessary the value which is currently used. 00 Pin HD selected , V_{HD} / 12 connected to ADC channel 01 Pin HS0 selected , V_{HS0} / 6 connected to ADC channel 10 Pin HS1 selected , V_{HS1} / 6 connected to ADC channel 11 Pin HS2 selected, V_{HS2} / 6 connected to ADC channel

17.3.2.11 GDU Current Sense Offset Register (GDUCSO)

Module Base + 0x000A

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	GCSO1[2:0]			0	GCSO0[2:0]		
W	▨				▨			
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-13. GDU Current Sense Offset (GDUCSO)

1. Read: Anytime
   Write: Anytime

Table 17-13. GDUCSO Register Field Descriptions

Field	Description
6:4 GCSO1[2:0]	**GDU Current Sense Amplifier 1 Offset** — These bits adjust the offset of the current sense amplifier. See Section 17.4.8, “Current Sense Amplifier and Overcurrent Comparator 000 No offset 001 Offset is +5mV 010 Offset is +10mV 011 Offset is +15mV 100 No offset 101 Offset is -15mV 110 Offset is -10mV 111 Offset is -5mV
2:0 GCSO0[2:0]	**GDU Current Sense Amplifier 0 Offset** — These bits adjust the offset of the current sense amplifier. See Section 17.4.8, “Current Sense Amplifier and Overcurrent Comparator 000 No offset 001 Offset is +5mV 010 Offset is +10mV 011 Offset is +15mV 100 No offset 101 Offset is -15mV 110 Offset is -10mV 111 Offset is -5mV

17.3.2.12 GDU Desaturation Level Register (GDUDSLVL)

Module Base + 0x000B

Access: User read/write(1)

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

▨ = Unimplemented

Figure 17-14. GDU Desaturation Level Register (GDUDSLVL)

1. Read: Anytime
   Write: Only if GWP=0

Table 17-14. GDU Desaturation Level Register Field Descriptions

Field	Description
6:4 GDSLHS	**GDU Desaturation Level for High-Side Drivers** — This bits adjust the desaturation levels of the three high-side FET pre-drivers. These bits cannot be modified after GWP bit is set. See Section 17.4.5, “Desaturation Error 000 $V_{desaths}$ = V_{HD} - 0.30V 001 $V_{desaths}$ = V_{HD} - 0.45V 010 $V_{desaths}$ = V_{HD} - 0.60V 011 $V_{desaths}$ = V_{HD} - 0.75V 100 $V_{desaths}$ = V_{HD} - 0.90V 101 $V_{desaths}$ = V_{HD} - 1.05V 110 $V_{desaths}$ = V_{HD} - 1.20V 111 $V_{desaths}$ = V_{HD} - 1.35V
2:0 GDSLLS	**GDU Desaturation Level for Low-Side Drivers** — This bits adjust the desaturation level of the three low-side FET pre-drivers. These bits cannot be modified after GWP bit is set. See Section 17.4.5, “Desaturation Error 000 $V_{desatls}$ = 0.30V 001 $V_{desatls}$ = 0.45V 010 $V_{desatls}$ = 0.60V 011 $V_{desatls}$ = 0.75V 100 $V_{desatls}$ = 0.90V 101 $V_{desatls}$ = 1.05V 110 $V_{desatls}$ = 1.20V 111 $V_{desatls}$ = 1.35V

17.3.2.13 GDU Phase Log Register (GDUPHL)

Module Base + 0x000C   Access: User read/write[1]

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

▨ = Unimplemented

Figure 17-15. GDU Phase Log Register (GDUPHL)

1. Read: Anytime
   Write: never

Table 17-15. GDU Phase Log Register Field Descriptions

Field	Description
2:0 GPHL	**GDU Phase Log Bits**— If a desaturation error occurs the phase status bits GPHS[2:0] in register GDUSTAT are copied to this register. The GDUPHL register is cleared only on reset. See Section 17.4.5, “Desaturation Error

17.3.2.14 GDU Clock Control Register 2 (GDUCLK2)

Module Base + 0x000D   Access: User read/write[1]

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

Figure 17-16. GDU Clock Control Register 2 (GDUCLK2)

1. Read: Anytime
   Write: Only if GWP=0

Table 17-16. GDUCLK2 Register Field Descriptions

Field	Description
3-0 GCPCD[3:0]	**GDU Charge Pump Clock Divider** — These bits select the clock divider factor which is used to divide down the bus clock frequency f_{BUS} for the charge pump clock f_{CP}. See Table 17-17 for divider factors. These bits cannot be modified after GWP bit is set. See also Section 17.4.4, “Charge Pump

NOTE

> The GCPCD bits must be set to the required value before GCPE bit is set. If a different charge pump clock frequency is required GCPE has to be cleared before new values to GCPCD bits are written.

Table 17-17. Charge Pump Clock Divider Factors k = f_{BUS} / f_{CP}

GCPCD[3:0]	f_{CP}
0000	f_{BUS} / 16
0001	f_{BUS} / 24
0010	f_{BUS} / 32
0011	f_{BUS} / 48
0100	f_{BUS} / 64
0101	f_{BUS} / 96
0110	f_{BUS} / 100
0111	f_{BUS} / 128
1000	f_{BUS} / 192
1001	f_{BUS} / 200
1010	f_{BUS} / 256
1011	f_{BUS} / 384
1100	f_{BUS} / 400
1101	f_{BUS} / 512
1110	f_{BUS} / 768
1111	f_{BUS} / 800

17.3.2.15 GDU Overcurrent Register 0 (GDUOC0)

Module Base + 0x000E

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	GOCA0	GOCE0	0	0	GOCT0[3:0]			
W			▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-17. GDU Overcurrent Register1 (GDUOC0)

[1] Read: Anytime
Write: Only if GWP=0

Table 17-18. GDUOC0 Register Field Descriptions

Field	Description
7 GOCA0	**GDU Overcurrent Action** — This bit cannot be modified after GWP bit is set. This bit controls the action in case of an overcurrent event. See Table 17-22 and Table 17-21
6 GOCE0	**GDU Overcurrent Comparator Enable** — This bit cannot be modified after GWP bit is set. 0 Overcurrent Comparator is disabled 1 Overcurrent Comparator is enabled
3:0 GOCT0[3:0]	**GDU Overcurrent Comparator Threshold** — These bits cannot be modified after GWP bit is set. The overcurrent comparator threshold voltage is the output of a 6-bit digital-to-analog converter. The upper two bits of the digital inputs are tied to one. The other bits of the digital inputs are driven by GOCT0[3:0]. The overcurrent comparator threshold voltage can be calculated from equation below. $\mathrm{Voct0} = (49 + \mathrm{GOCT0}) \cdot \frac{\mathrm{VDDA}}{64}$

17.3.2.16 GDU Overcurrent Register 1 (GDUOC1)

Module Base + 0x000F

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	GOCA1	GOCE1	0	0	GOCT1[3:0]			
W			▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-18. GDU Overcurrent Register 1 (GDUOC1)

[1] Read: Anytime
Write: Only if GWP=0

Table 17-19. GDUOC1 Register Field Descriptions

Field	Description
7 GOCA1	**GDU Overcurrent Action** — This bit cannot be modified after GWP bit is set. This bit controls the action in case of an overcurrent event or overvoltage event. See Table 17-22 and Table 17-21
6 GOCE1	**GDU Overcurrent Enable** — This bit cannot be modified after GWP bit is set. 0 Overcurrent Comparator 1 is disabled 1 Overcurrent Comparator 1 is enabled
3:0 GOCT1[3:0]	**GDU Overcurrent Comparator Threshold** — These bits cannot be modified after GWP bit is set. The overcurrent comparator threshold voltage is the output of a 6-bit digital-to-analog converter. The upper two bits of the digital inputs are tied to one. The other bits of the digital inputs are driven by GOCT0[3:0]. The overcurrent comparator threshold voltage can be calculated from equation below. $$\mathrm{Voct1} = (49 + \mathrm{GOCT1}) \cdot \frac{\mathrm{VDDA}}{64}$$

17.4 Functional Description

17.4.1 General

The PMF module provides the values to be driven onto the outputs of the low-side and high-side FET pre-drivers. If the FET pre-drivers are enabled, the PMF channels drive their corresponding high-side or low-side FET pre-drivers according Table 17-20.

Table 17-20. PMF Channel Assignment

PMF Channel	PMF Channel Assignment
0	High-Side Gate and Source Pins HG[0], HS[0]
1	Low-Side Gate and Source Pins LG[0], LS[0]
2	High-Side Gate and Source Pins HG[1], HS[1]
3	Low-Side Gate and Source Pins LG[1], LS[1]
4	High-Side Gate and Source Pins HG[2], HS[2]
5	Low-Side Gate and Source Pins LG[2], LS[2]

17.4.2 Low-Side FET Pre-Drivers

The three low-side FET pre-drivers turn on and off the external low-side power FETs. The energy required to charge the gate capacitance of the power FET C_G is drawn from the output of the voltage regulator VLS. See Figure . The register bits GSRCLS[2:0] in the GDUSRC Register (see Figure 17-8) control the slew rate of the low-side FET pre-drivers in order to control fast voltage changes dv/dt .(see also Section 17.5.1, "FET Pre-Driver Details).

17.4.3 High-Side FET Pre-Driver

The three high-side FET pre-drivers turn on and off the external high-side power FETs. The required charge for the gate capacitance of the external power FET is delivered by the bootstrap capacitor. After the supply voltage is applied to the microcontroller or after exit from stop mode, the low-side FET pre-drivers should be activated for a short time in order to charge the bootstrap capacitor C_{BS}. Care must be taken after a long period of inactivity of the low-side FET pre-drivers to verify that the bootstrap capacitor C_{BS} is not discharged (see also Calculation of Bootstrap Capacitor).

The register bits GSRCHS[2:0] in the GDUSRC Register (see Figure 17-8) control the slew rate of the high-side FET pre-driver in order to control fast voltage changes dv/dt (see also Section 17.5.1, "FET Pre-Driver Details).

NOTE

The minimum PWM pulse on & off time must be $t_{minpulse}$.

NOTE

If the GFDE bit is cleared the high-side gate and source pins and the low-side gate and source pins are shorted with an internal resistor. The voltage differences are V_{HGx}-V_{HSx}~ 0V and V_{LGx}-V_{LSx} ~ 0V so that the external FETs are turned off.

NOTE

The PWM channel outputs for high-side and low-side drivers are delayed by two core clock cycles.

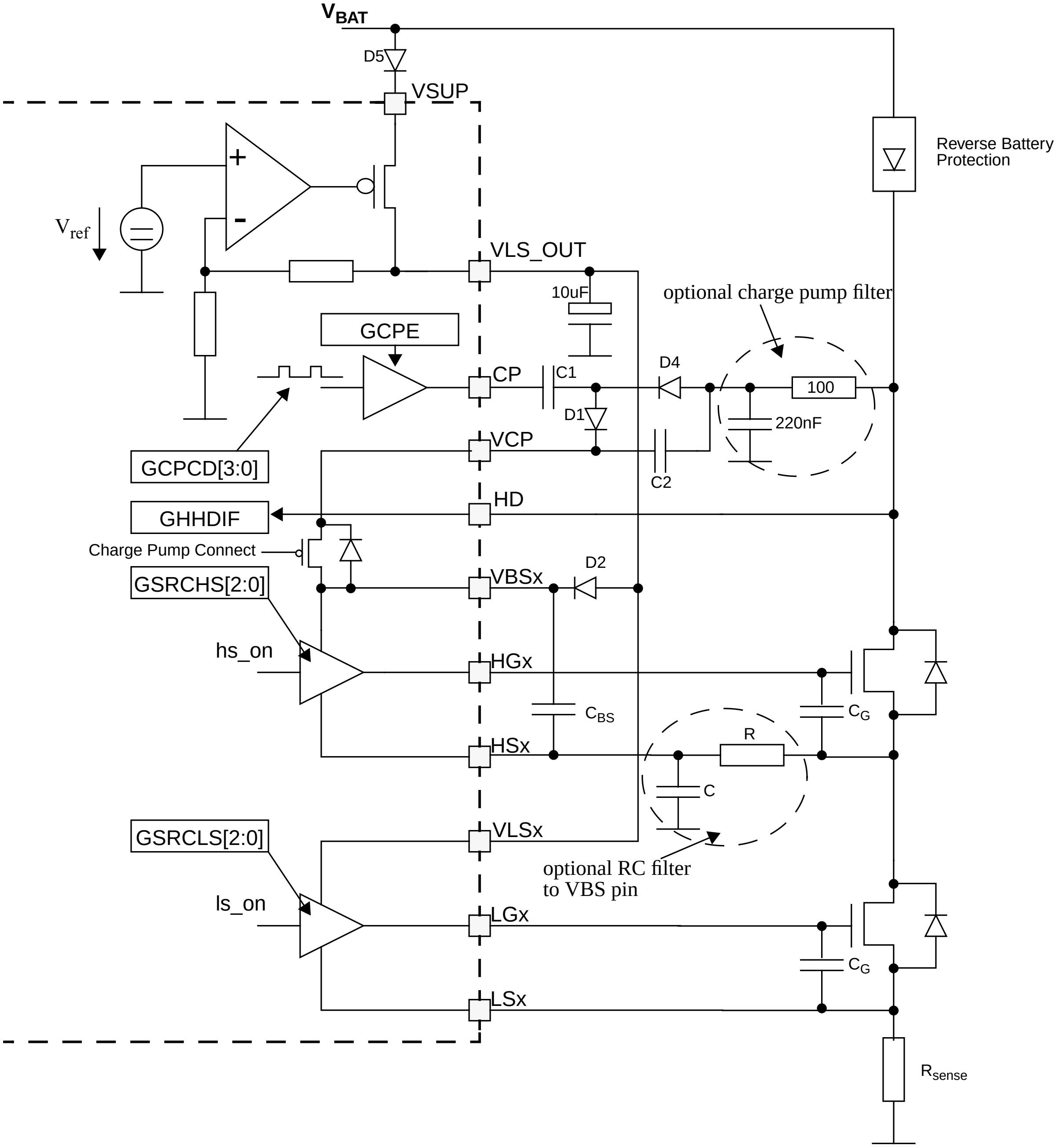


Figure 17-19. FET Pre-Driver Circuit and Voltage Regulator

NOTE

Optional charge pump input RC filter can be used to avoid overpumpimg effect when voltage spikes are present on the high-side drains.

NOTE

Optional RC filter to VBS pin should be used to avoid overshoot above maximum voltage on VBS pin. The RC filter needs to be carefully designed in order not to influence the charging time of the bootstrap capacitor C_{BS}.

17.4.4 Charge Pump

The GDU module integrates the necessary hardware to build a charge pump with external components.The charge pump is used to maintain the high-side driver gate source voltage V_{GS} when PWM is running at 100% duty cycle. The external components needed are capacitors and diodes. . The supply voltage of the charge pump driver on pin CP is $V_{VLS.}$ The output voltage on pin CP typically switches between 0 and 11V. The charge pump clock frequency depends on the setting of GCPCD bits.

17.4.5 Desaturation Error

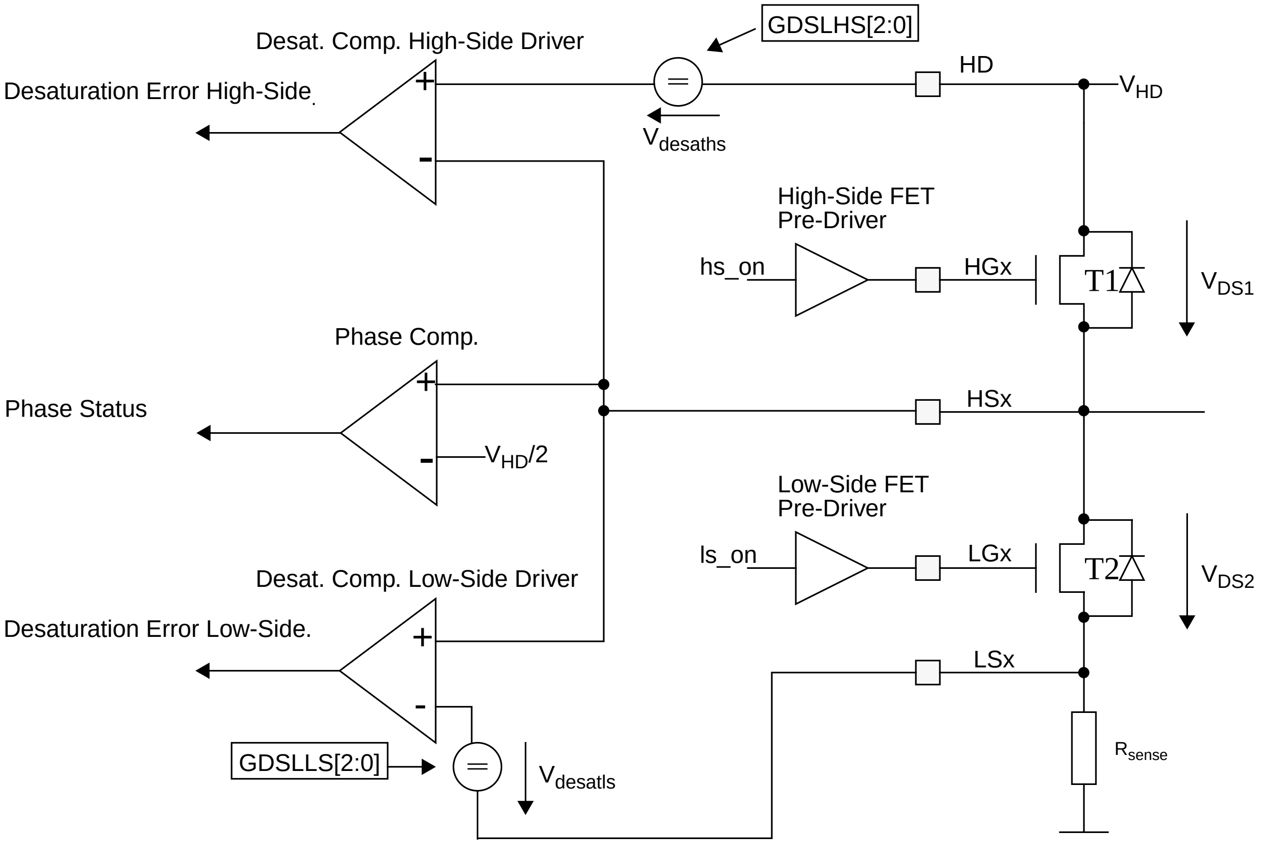
A desaturation error is generated if the output signal at HSx does not properly reflect the drive condition of the low-side and high-side FET pre-drivers. The GDU integrates three desaturation comparators for the low-side FET pre-drivers and three desaturation comparators for the high-side FET pre-drivers.

If the low-side power FET T2 (see Figure 17-20) is turned on and the drain source voltage V_{DS2} of T2 is greater than $V_{desatls}$ after the blanking time t_{BLANK} a desaturation error will be flagged. In this case the associated desaturation error flag GDLSIF[2:0] will be set (see Figure 17-6) and the low-side power FET T2 will be turned off. The level of the voltage $V_{desatls}$ can be adjusted in the range of 0.30V to 1.35V. (see Figure 17-14)

If the high-side power FET T1 (see Figure 17-20) is turned on and the drain source voltage V_{DS1} is greater than $V_{desaths}$ after the blanking time t_{BLANK} a desaturation error will be flagged.In this case the associated desaturation error flag GDHSIF[2:0] will be set (see Figure 17-6) and the high-side power FET T1 will be turned off. The level of the voltage $V_{desaths}$ can be adjusted in the range of 0.30 to 1.35V (see Figure 17-14).

The low-side and high-side desaturation interrupt flags GDHSIF and GDLSIF are cleared by writing a one to the associated flag. After the flag is cleared the associated low-side or high-side FET pre-driver is enabled again and is driven by the source selected in the PMF module.

Figure 17-20. Desaturation Comparators and Phase Comparators



17.4.6 Phase Comparators

The GDU module includes three phase comparators. The phase comparators compares the voltage on the HS[2:0] pins with one half voltage on HD pin. If V_{HSx} is greater than 0.5 V_{HD} the associated phase status bit GPHS[2:0] is set. (see Figure 17-7) If the V_{HSx} is less than 0.5 V_{HD} the associated phase status bit GPHS[2:0] is cleared. If a desaturation error is detected the state of the phase status bit GPHS[2:0] are copied to the GDUPHL register. The phase flags get unlocked when the associated desaturation interrupt flag is cleared.

17.4.7 Fault Protection Features

The GDU includes a number of fault protection features against overvoltage, overcurrent, undervoltage and power bridge faults like phase shorted to ground or supply. Fault conditions enables response by the GDU and turn on or off low-side and high-side drivers as shown in Table 17-22. In addition five fault outputs are provided to signal detected faults to other modules of the MCU. For connectivity of the fault outputs see section 1 of the reference manual. Table 17-21 shows the logic equations for the five fault outputs.

Table 17-21. Fault Outputs Logic Equations

Fault Output	Logic Equation
Fault[0]	(GDLSIF[0] | GDHSIF[0]) | (GOCIF[0] & GOCA0) | (GOCIF[1] & GOCA1)
Fault[1]	(GDLSIF[1] | GDHSIF[1]) | (GOCIF[0] & GOCA0) | (GOCIF[1] & GOCA1)
Fault[2]	(GDLSIF[2] | GDHSIF[2]) | (GOCIF[0] & GOCA0) | (GOCIF[1] & GOCA1)
Fault[3]	GLVLSIF
Fault[4]	GHHDIF | (GOCIF[0] & ~GOCA0) | (GOCIF[1] & ~GOCA1)

Table 17-22. Fault Protection Features Summary

Priority	Condition	GSUF	GHHDF	GOCIF0	GOCIF1	GLVLSF	GDHSIF [2:0]	GDLSIF [2:0]		HS2	HS1	HS0	LS2	LS1	LS0
low	normal operation,no error condition, FET pre-driver driven by PMF module	0	0	0	0	0	000	000		PWM [4][1]	PWM [2]	PWM [0]	PWM [5]	PWM [3]	PWM [1]
	startup condition after reset deassert, no error condition	1	0	0	0	0	000	000		off	off	off	on	on	on
	overvoltage on HD pin GOCA1=0	x	1	0	0	0	000	000		off	off	off	on	on	on
	overcurrent condition comparator 0 GOCA0=0	x	x	1	x	0	000	000		off	off	off	on	on	on
	overcurrent condition comparator 1 GOCA1=0	x	x	x	1	0	000	000		off	off	off	on	on	on
	undervoltage condition on VLS_OUT pin	x	x	x	x	1	000	000		off	off	off	off	off	off
	overcurrent condition comparator 0 GOCA0=1	x	x	1	x	x	000	000		off	off	off	off	off	off
	overcurrent condition comparator 1 GOCA1=1	x	x	x	1	x	000	000		off	off	off	off	off	off
	desaturation error condition on high-side FET pre-drivers	x	x	x	x	x	001	000		PWM [4]	PWM [2]	off	PWM [5]	PWM [3]	PWM [1]
		x	x	x	x	x	010	000		PWM [4]	off	PWM [0]	PWM [5]	PWM [3]	PWM [1]
		x	x	x	x	x	100	000		off	PWM [2]	PWM [0]	PWM [5]	PWM [3]	PWM [1]
	desaturation error condition on low-side FET pre-drivers	x	x	x	x	x	000	001		PWM [4]	PWM [2]	PWM [0]	PWM [5]	PWM [3]	off
		x	x	x	x	x	000	010		PWM [4]	PWM [2]	PWM [0]	PWM [5]	off	PWM [1]
		x	x	x	x	x	000	100		PWM [4]	PWM [2]	PWM [0]	off	PWM [3]	PWM [1]
high	overvoltage on HD pin GOCA1=1	x	1	x	x	x	xxx	xxx		off	off	off	off	off	off

1. PWM[i] means low or high-side driveris driven by associated PWM channel

NOTE

Since all MOSFET transistors are turned off, VBSX can reach phase voltage plus bootstrap voltage which may exceed allowable levels during high supply voltage conditions. If such operating condition exist the application must make sure that VBSX levels are clamped below maximum ratings for example by using clamping diodes.

Figure 17-21. Short to Supply Detection

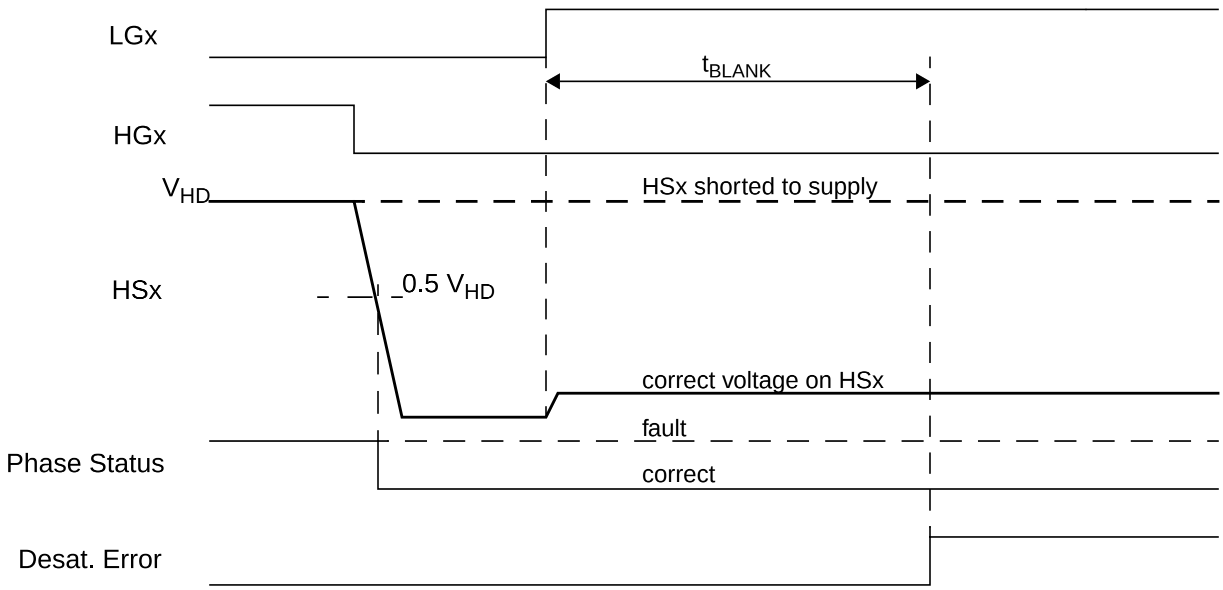
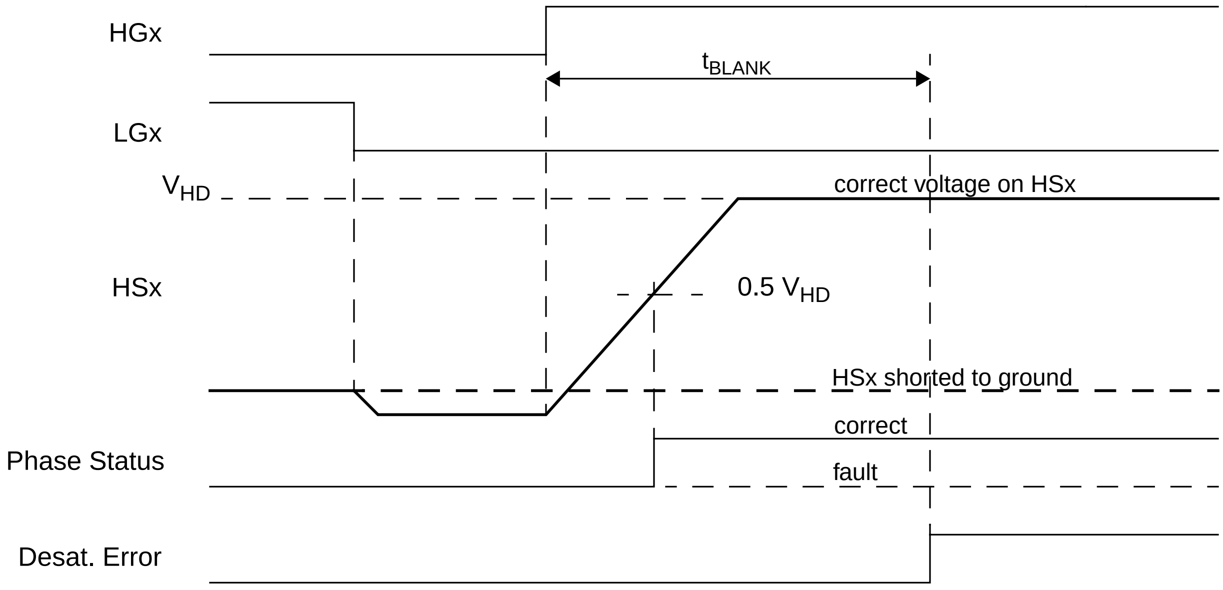


Figure 17-22. Short to Ground Detection



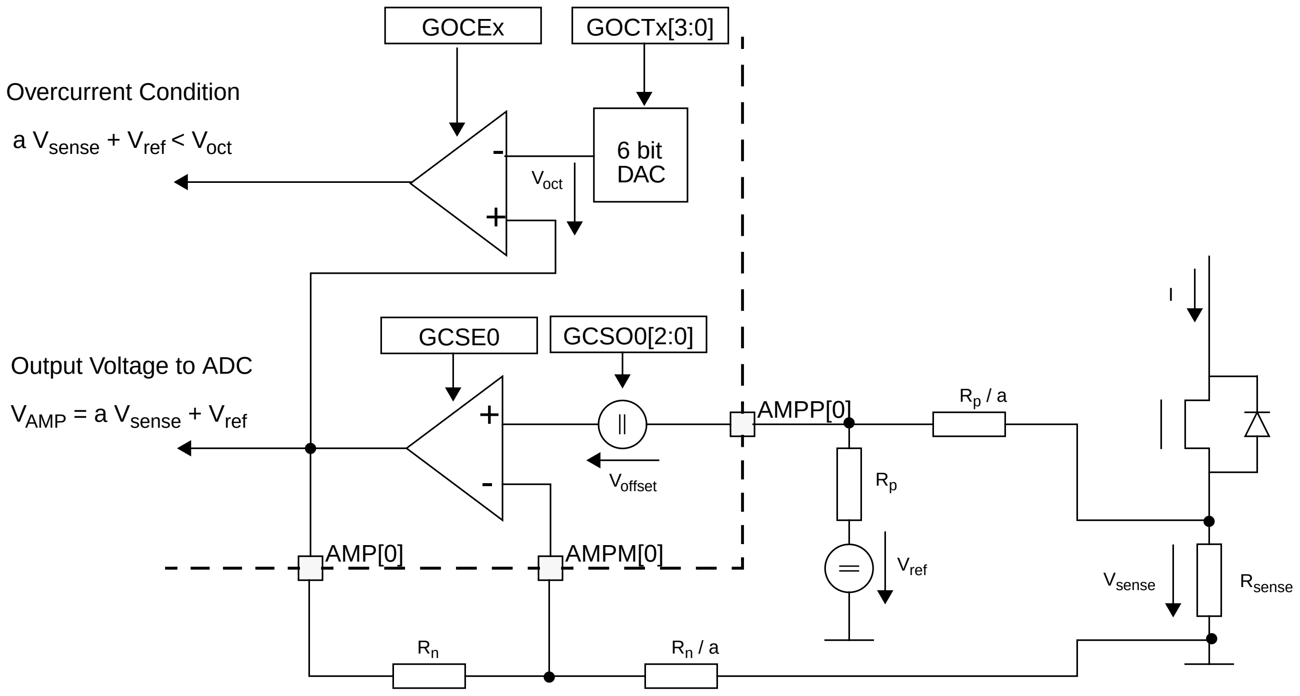
17.4.8 Current Sense Amplifier and Overcurrent Comparator

The current sense amplifier is usually connected as a differential amplifier(see Figure 17-23). It senses the current flowing through the external power FET as a voltage across the current sense resistor R_{sense}. In order to measure both positive and negative currents, an external reference has to be used. The output of the current sense amplifier can be connected to an ADC channel. For more details on ADC channel assignment, refer to Device Overview Internal Signal Mapping Section. The input offset voltage of the current sense amplifier can be adjusted with the GCSO[2:0] bits in the GDUCSO register. (see Figure 17-13) The output of the current sense amplifier is connected to the plus input of the overcurrent comparator. The minus input is driven by the output voltage of a 6 Bit DA converter. The digital input of the DA converter is {11,GOCTx[3:0]}. In order to use the overcurrent comparator GOCEx and GCSxE have to be set.

NOTE

If both overcurrent comparators are used both action bits GOCA0 and GOCA1 must have the same value. For example GOCA0=0 and GOCA1=1 is not allowed. Only GOCA0=GOCA1=1 or GOCA0=GOCA1=0 is allowed.

Figure 17-23. Current Sense Amplifier Connected as Differential Amplifier



17.4.9 GDU DC Link Voltage Monitor

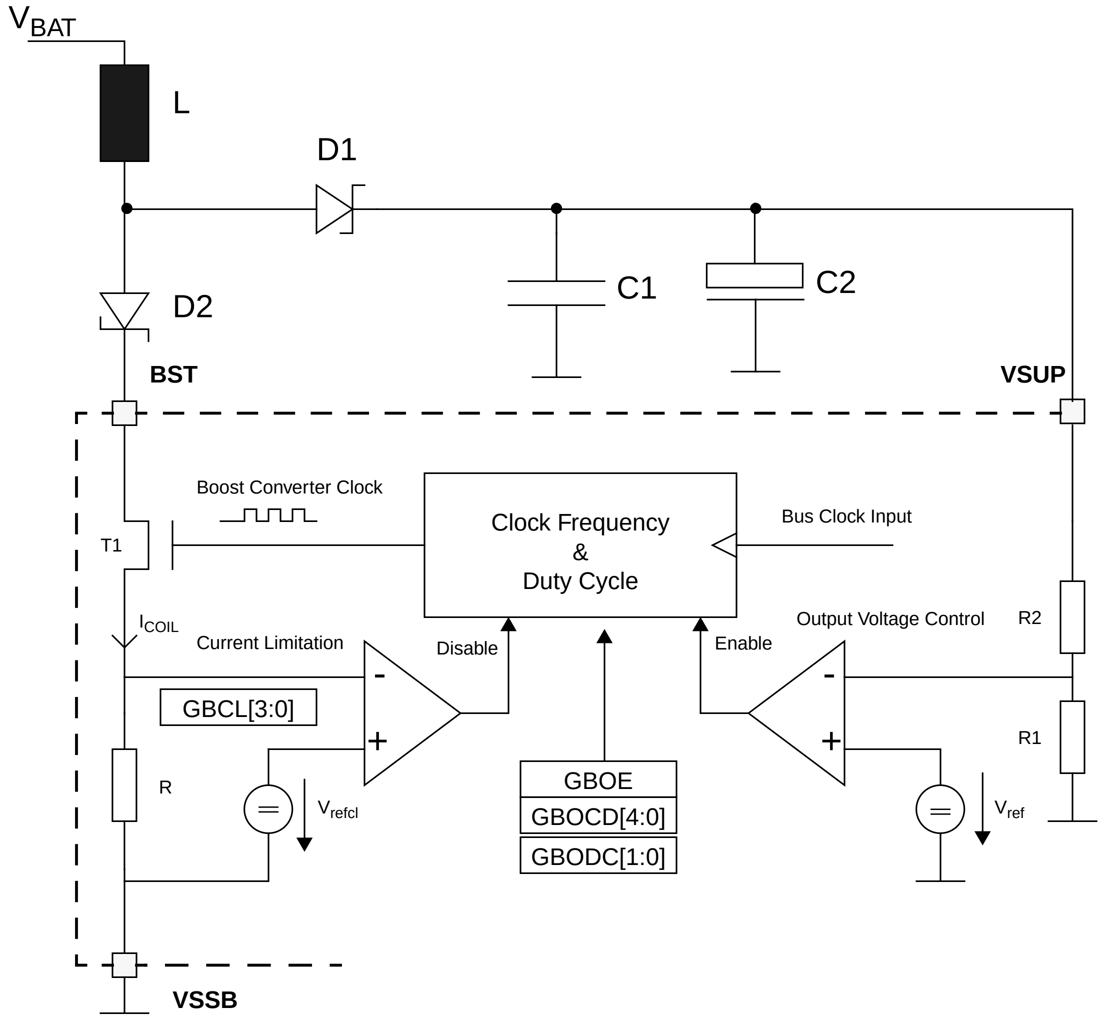
In addition to the feature described in Section 17.3.2.10, “GDU Phase Mux Register (GDUPHMUX) the voltage on pin HD divide by 5 is routed to an ADC channel. See SOC section for ADC channel number. This feature is only available if GFDE is set.

17.4.10 Boost Converter

The GDU module integrates the necessary hardware to build a boost converter with external components in case of low voltage condition. The external components needed are two Schottky diodes, one coil, and capacitors. See Figure 17-24. The boost converter clock which is driving the transistor T1 (see Figure 17-24) is derived from the bus clock. This clock can be divided down as described in Table 17-9. The boost converter also includes a circuit to limit the current through coil. This current limit can be adjusted with the bits GBCL[3:0] in the GDUBCL register. See GDU electrical parameters.

The output voltage of the boost converter on VSUP pin is divided down and compared with a reference voltage V_{ref}. As long as the divided voltage V_{VSUP} is below V_{ref} the boost converter clock is enabled assuming that GBOE (GDU Boost Option Enable) is set.

Figure 17-24. Boost Converter Option with external Components[1]



---

[1] Diode D2 shown is optional if coil is connected behind reverse battery protection.

17.4.11 Interrupts

This section describes the interrupts generated in the GDU module. The interrupts are only available in CPU run mode. Entering and exiting stop mode has no effect on the interrupt flags. The GDU module has twointerrupt vectors which are listed in Table 17-23. The low-side and high-side desaturation error flags are combined into one interrupt line and the over and under voltage detection are combined into another interrupt line. (see SOC section interrupt vector table)

Table 17-23. GDU Module Interrupt Sources

#	GDU Module Interrupt Source	Module Internal Interrupt Source	Local Enable
0	GDU desaturation error interrupt	GDU low-side and high-side desaturation error flags GDHSF[2:0] and GDLSF[2:0]	GDSEIE = 1
1	GDU over/under voltage detection and overcurrent detection interrupt	GDU low voltage condition on pin VLS (GLVLSIF)	GLVLSIE = 1
		GDU high voltage condition on pin HD (GHHDIF)	GHHDIE = 1
		GDU Overcurrent Condition (GOCIF[1:0])	GOCIE[1:0]=11

17.5 Application Information

17.5.1 FET Pre-Driver Details

The basic concept of the high-side driver is shown in Figure 17-25. If the FET pre-driver is switched on the transistor T2 is driving the output HG. For on resistance R_{gduon} of transistor T2 refer to GDU electricals. The output current is limited to I_{OUT} which is derived from the reference current I_{REF}. The current source is controlled by the slew rate control bits GSRCHS[2:0]. If the FET pre-driver is switched off transistors T3 and T4 are switched on. For on resistance $R_{gduoffn}$ and $R_{gduoffp}$ of transistors T3 and T4 refer to GDU electricals.

The reference current I_{REF} is controlled by the slew rate conrol bits GSRCHS[2:0] :

- I_{REF} = 10uA + GSRCHS 10uA, [10uA, 20uA . . . 80uA]

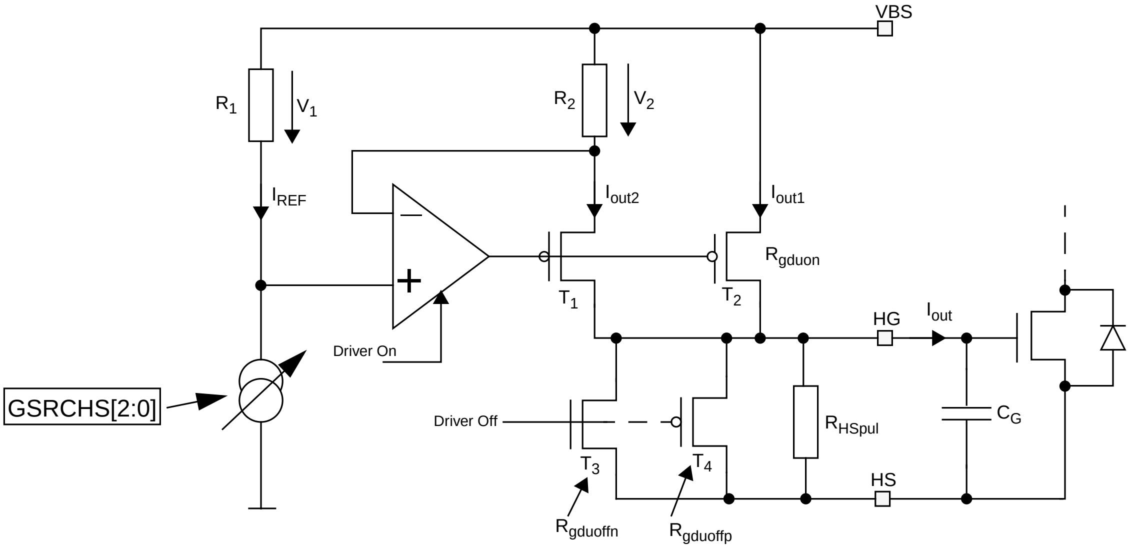
Assuming an ideal op-amp the voltage across R_1 is equal voltage across R_2 and I_{OUT2} is given by:

- $V_1 = V_2 = I_{REF} R_1 = I_{OUT2} R_2$
- $I_{OUT2} = I_{REF} (R_1/R_2)$

With the ratio of the transistor sizes of T_1 and T_2 k=450, and the ratio of the resistors R_1/R_2=36, and neglect the current through R_{HSpul} the output current I_{OUT} is:

- $I_{OUT1} = k\ I_{OUT2}$
- $I_{OUT} = I_{OUT1} + I_{OUT2} = I_{REF} (R_1/R_2) (1+k)$
- $I_{OUT} \sim I_{REF} (R_1/R_2)\ k$

Figure 17-25. FET Pre-Driver Concept for High-Side Driver



17.5.2 Calculation of Bootstrap Capacitor

The size of the bootstrap capacitor C_{BS} depends on the total gate charge Q_G needed to turn on the power FET used in the application. If the bootstrap capacitor is too small there can be a large voltage drop due to charge sharing between bootstrap capacitor C_{BS} and the total gate capacitance of the power FET C_G. The resulting voltage on the gate of the power FET can be calculated as follow:

Eqn. 17-1

$$V_G = \frac{Q_{BS}}{C_{BS} + C_G} = \frac{V_{BS}}{1 + \frac{C_G}{C_{BS}}}$$

For example if $C_{BS} = 20\ C_G$ then the resulting gate voltage is $V_G = 0.95\ V_{BS}$.

Chapter 18
LIN Physical Layer (S12LINPHYV2)

Table 18-1. Revision History Table

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V01.00	10 Dec 2010	All	- Initial Version
V02.00	25 June 2012	All	- Added LIN TxD-dominant timeout feature
V02.06	11 Jan 2013	All	-Added application note to help the ISR development for the Interrupts (timeout and overcurrent)
V02.08	10 Apr 2013	Register and interrupt descriptions, application section	- Added notes regarding the correct handling of clearing LPOCIF and LPDTIF.
V02.09	27 Jun 2013	Feature list	- Added the SAE J2602-2 LIN compliance.
V02.10	21 Aug 2013	Overcurrent and TxD-dominant timeout interrupt descriptions	- Specified the time after which the interrupt flags are set again after having been cleared while the error condition is still present.
V02.11	19 Sep 2013	All	- Removed preliminary note. - Fixed grammar and spelling throughout the document.
V02.12	20 Sep 2013	Standby Mode	- Clarified Standby mode behavior.
V02.13	8 Oct 2013	All	- More grammar, spelling, and formating fixes throughout the document.

18.1 Introduction

The LIN (Local Interconnect Network) bus pin provides a physical layer for single-wire communication in automotive applications. The LIN Physical Layer is designed to meet the LIN Physical Layer 2.2 specification from LIN consortium.

18.1.1 Features

The LIN Physical Layer module includes the following distinctive features:

- Compliant with LIN Physical Layer 2.2 specification.

- Compliant with the SAE J2602-2 LIN standard.
- Standby mode with glitch-filtered wake-up.
- Slew rate selection optimized for the baud rates: 10.4 kbit/s, 20 kbit/s and Fast Mode (up to 250 kbit/s).
- Switchable 34 kΩ/330 kΩ pullup resistors (in shutdown mode, 330 kΩ only)
- Current limitation for LIN Bus pin falling edge.
- Overcurrent protection.
- LIN TxD-dominant timeout feature monitoring the LPTxD signal.
- Automatic transmitter shutdown in case of an overcurrent or TxD-dominant timeout.
- Fulfills the OEM “Hardware Requirements for LIN (CAN and FlexRay) Interfaces in Automotive Applications” v1.3.

The LIN transmitter is a low-side MOSFET with current limitation and overcurrent transmitter shutdown. A selectable internal pullup resistor with a serial diode structure is integrated, so no external pullup components are required for the application in a slave node. To be used as a master node, an external resistor of 1 kΩ must be placed in parallel between VLINSUP and the LIN Bus pin, with a diode between VLINSUP and the resistor. The fall time from recessive to dominant and the rise time from dominant to recessive is selectable and controlled to guarantee communication quality and reduce EMC emissions. The symmetry between both slopes is guaranteed.

18.1.2 Modes of Operation

The LIN Physical Layer can operate in the following four modes:

1. **Shutdown Mode**

   The LIN Physical Layer is fully disabled. No wake-up functionality is available. The internal pullup resistor is replaced by a high ohmic one (330 kΩ) to maintain the LIN Bus pin in the recessive state. All registers are accessible.

2. **Normal Mode**

   The full functionality is available. Both receiver and transmitter are enabled.

3. **Receive Only Mode**

   The transmitter is disabled and the receiver is running in full performance mode.

4. **Standby Mode**

   The transmitter of the LIN Physical Layer is disabled. If the wake-up feature is enabled, the internal pullup resistor can be selected (330 kΩ or 34 kΩ). The receiver enters a low power mode and optionally it can pass wake-up events to the Serial Communication Interface (SCI). If the wake-up feature is enabled and if the LIN Bus pin is driven with a dominant level longer than t_{WUFR} followed by a rising edge, the LIN Physical Layer sends a wake-up pulse to the SCI, which requests a wake-up interrupt. (This feature is only available if the LIN Physical Layer is routed to the SCI).

18.1.3 Block Diagram

Figure 18-1 shows the block diagram of the LIN Physical Layer. The module consists of a receiver with wake-up control, a transmitter with slope and timeout control, a current sensor with overcurrent protection as well as a registers control block.

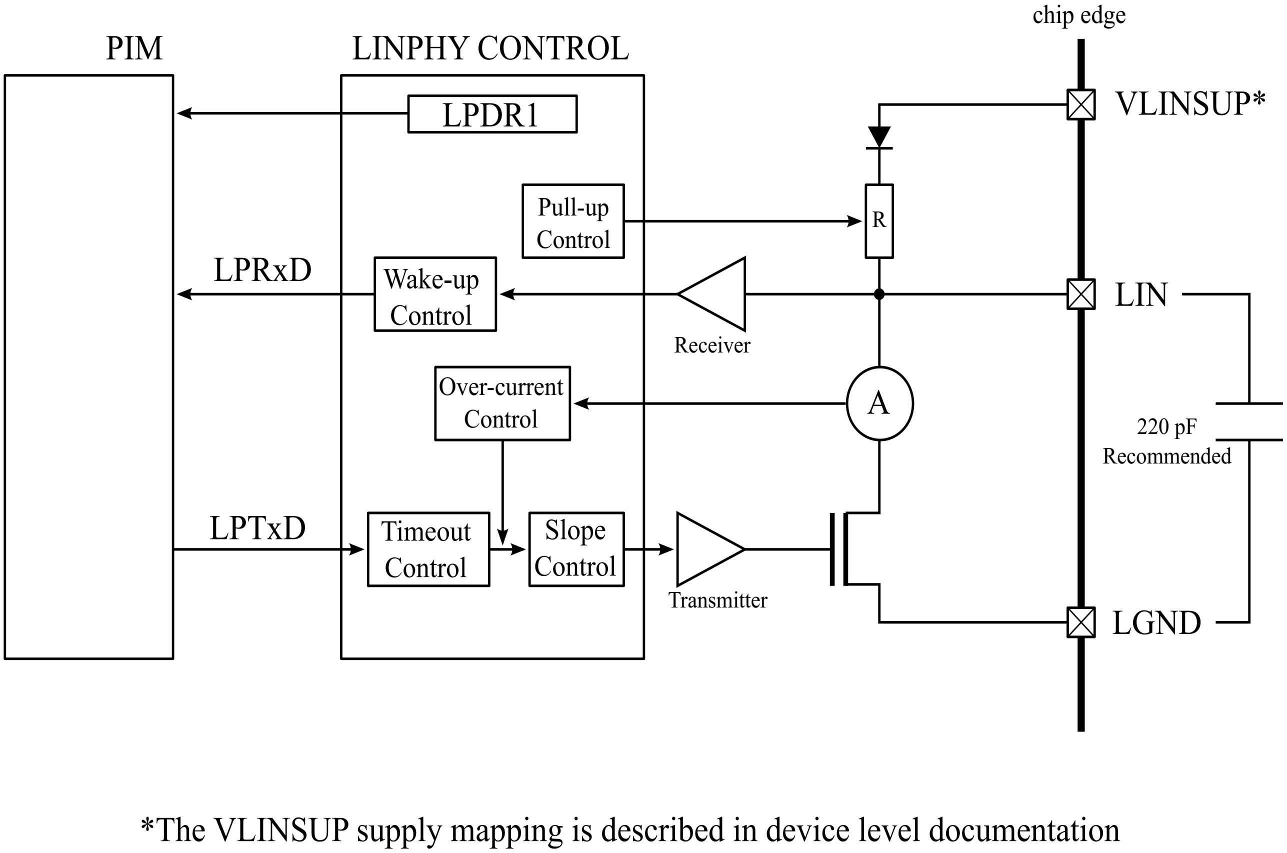


Figure 18-1. LIN Physical Layer Block Diagram

NOTE

The external 220 pF capacitance between LIN and LGND is strongly recommended for correct operation.

18.2 External Signal Description

This section lists and describes the signals that connect off chip as well as internal supply nodes and special signals.

18.2.1 LIN — LIN Bus Pin

This pad is connected to the single-wire LIN data bus.

18.2.2 LGND — LIN Ground Pin

This pin is the device LIN ground connection. It is used to sink currents related to the LIN Bus pin. A decoupling capacitor external to the device (typically 220 pF, X7R ceramic) between LIN and LGND can further improve the quality of this ground and filter noise.

18.2.3 VLINSUP — Positive Power Supply

External power supply to the chip. The VLINSUP supply mapping is described in device level documentation.

18.2.4 LPTxD — LIN Transmit Pin

This pin can be routed to the SCI, LPDR1 register bit, an external pin, or other options. Please refer to the PIM chapter of the device specification for the available routing options.

This input is only used in normal mode; in other modes the value of this pin is ignored.

18.2.5 LPRxD — LIN Receive Pin

This pin can be routed to the SCI, an external pin, or other options. Please refer to the PIM chapter of the device specification for the available routing options.

In standby mode this output is disabled, and sends only a short pulse in case the wake-up functionality is enabled and a valid wake-up pulse was received in the LIN Bus.

18.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the LIN Physical Layer.

18.3.1 Module Memory Map

A summary of the registers associated with the LIN Physical Layer module is shown in Table 18-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

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 LPDR	R	0	0	0	0	0	0	LPDR1	LPDR0
	W								
0x0001 LPCR	R	0	0	0	0	LPE	RXONLY	LPWUE	LPPUE
	W								
0x0002 Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
	W								
0x0003 LPSLRM	R	LPDTDIS	0	0	0	0	0	LPSLR1	LPSLR0
	W								
0x0004 Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
	W								
0x0005 LPSR	R	LPDT	0	0	0	0	0	0	0
	W								
0x0006 LPIE	R	LPDTIE	LPOCIE	0	0	0	0	0	0
	W								
0x0007 LPIF	R	LPDTIF	LPOCIF	0	0	0	0	0	0
	W								

Figure 18-2. Register Summary

18.3.2 Register Descriptions

This section describes all the LIN Physical Layer registers and their individual bits.

18.3.2.1 Port LP Data Register (LPDR)

Module Base + Address 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	LPDR1	LPDR0
W	▨	▨	▨	▨	▨	▨		▨
Reset	0	0	0	0	0	0	1	1

▨ = Unimplemented

Figure 18-3. Port LP Data Register (LPDR)

[1] Read: Anytime

Write: Anytime

Table 18-2. LPDR Field Description

Field	Description
1 LPDR1	**Port LP Data Bit 1** — The LIN Physical Layer LPTxD input (see Figure 18-1) can be directly controlled by this register bit. The routing of the LPTxD input is done in the Port Inetrgation Module (PIM). Please refer to the PIM chapter of the device Reference Manual for more info.
0 LPDR0	**Port LP Data Bit 0** — Read-only bit. The LIN Physical Layer LPRxD output state can be read at any time.

18.3.2.2 LIN Control Register (LPCR)

Module Base + Address 0x0001

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	LPE	RXONLY	LPWUE	LPPUE
W	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-4. LIN Control Register (LPCR)

1. Read: Anytime

Write: Anytime,

Table 18-3. LPCR Field Description

Field	Description
3 LPE	**LIN Enable Bit** — If set, this bit enables the LIN Physical Layer. 0 The LIN Physical Layer is in shutdown mode. None of the LIN Physical Layer functions are available, except that the bus line is held in its recessive state by a high ohmic (330kΩ) resistor. All registers are normally accessible. 1 The LIN Physical Layer is not in shutdown mode.
2 RXONLY	**Receive Only Mode bit** — This bit controls RXONLY mode. 0 The LIN Physical Layer is not in receive only mode. 1 The LIN Physical Layer is in receive only mode.
1 LPWUE	**LIN Wake-Up Enable** — This bit controls the wake-up feature in standby mode. 0 In standby mode the wake-up feature is disabled. 1 In standby mode the wake-up feature is enabled.
0 LPPUE	**LIN Pullup Resistor Enable** — Selects pullup resistor. 0 The pullup resistor is high ohmic (330 kΩ). 1 The 34 kΩ pullup is switched on (except if LPE=0 or when in standby mode with LPWUE=0).

18.3.2.3 Reserved Register

Module Base + Address 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

▨ = Unimplemented

Figure 18-5. LIN Test 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 mode can alter the module's functionality.

Table 18-4. Reserved Register Field Description

Field	Description
7-0 Reserved	These reserved bits are used for test purposes. Writing to these bits can alter the module functionality.

18.3.2.4 LIN Slew Rate Mode Register (LPSLRM)

Module Base + Address 0x0003 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	LPDTDIS	0	0	0	0	0	LPSLR1	LPSLR0
W		▨	▨	▨	▨	▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-6. LIN Slew Rate Mode Register (LPSLRM)

1. Read: Anytime

   Write: Only in shutdown mode (LPE=0)

Table 18-5. LPSLRM Field Description

Field	Description
7 LPDTDIS	**TxD-dominant timeout disable Bit** — This bit disables the TxD-dominant timeout feature. Disabling this feature is only recommended for using the LIN Physical Layer for other applications than LIN protocol. It is only writable in shutdown mode (LPE=0). 0 TxD-dominant timeout feature is enabled. 1 TxD-dominant timeout feature is disabled.
1-0 LPSLR[1:0]	**Slew-Rate Bits** — Please see section 18.4.2 for details on how the slew rate control works. These bits are only writable in shutdown mode (LPE=0). 00 Normal Slew Rate (optimized for 20 kbit/s). 01 Slow Slew Rate (optimized for 10.4 kbit/s). 10 Fast Mode Slew Rate (up to 250 kbit/s). This mode is not compliant with the LIN Protocol (LIN electrical characteristics like duty cycles, reference levels, etc. are not fulfilled). It is only meant to be used for fast data transmission. Please refer to section 18.4.2.2 for more details on fast mode.Please note that an external pullup resistor stronger than 1 kΩ might be necessary for the range 100 kbit/s to 250 kbit/s. 11 Reserved .

18.3.2.5 Reserved Register

Module Base + Address 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

▨ = Unimplemented

Figure 18-7. 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 mode can alter the module's functionality.

Table 18-6. Reserved Register Field Description

Field	Description
7-0 Reserved	These reserved bits are used for test purposes. Writing to these bits can alter the module functionality.

18.3.2.6 LIN Status Register (LPSR)

Module Base + Address 0x0005 Access: User read/write(1)

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

▨ = Unimplemented

Figure 18-8. LIN Status Register (LPSR)

1. Read: Anytime

Write: Never, writes to this register have no effect

Table 18-7. LPSR Field Description

Field	Description
7 LPDT	**LIN Transmitter TxD-dominant timeout Status Bit** — This read-only bit signals that the LPTxD pin is still dominant after a TxD-dominant timeout. As long as the LPTxD is dominant after the timeout the LIN transmitter is shut down and the LPTDIF is set again after attempting to clear it. 0 If there was a TxD-dominant timeout, LPTxD has ceased to be dominant after the timeout. 1 LPTxD is still dominant after a TxD-dominant timeout.

18.3.2.7 LIN Interrupt Enable Register (LPIE)

Module Base + Address 0x0006

Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	LPDTIE	LPOCIE	0	0	0	0	0	0
W			▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-9. LIN Interrupt Enable Register (LPIE)

[1] Read: Anytime

Write: Anytime

Table 18-8. LPIE Field Description

Field	Description
7 LPDTIE	**LIN transmitter TxD-dominant timeout Interrupt Enable** — 0 Interrupt request is disabled. 1 Interrupt is requested if LPDTIF bit is set.
6 LPOCIE	**LIN transmitter Overcurrent Interrupt Enable** — 0 Interrupt request is disabled. 1 Interrupt is requested if LPOCIF bit is set.

18.3.2.8 LIN Interrupt Flags Register (LPIF)

Module Base + Address 0x0007 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	LPDTIF	LPOCIF	0	0	0	0	0	0
W			▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-10. LIN Interrupt Flags Register (LPIF)

1. Read: Anytime

Write: Writing ‘1’ clears the flags, writing a ‘0’ has no effect

Table 18-9. LPIF Field Description

Field	Description
7 LPDTIF	**LIN Transmitter TxD-dominant timeout Interrupt Flag** — LPDTIF is set to 1 when LPTxD is still dominant (0) after t_{TDLIM} of the falling edge of LPTxD. For protection, the transmitter is disabled. This flag can only be cleared by writing a 1. Writing a 0 has no effect. Please make sure that LPDTIF=1 before trying to clear it. Clearing LPDTIF is not allowed if LPDTIF=0 already. If the LPTxD is still dominant after clearing the flag, the transmitter stays disabled and this flag is set again (see 18.4.4.2 TxD-dominant timeout Interrupt). If interrupt requests are enabled (LPDTIE= 1), LPDTIF causes an interrupt request. 0 No TxD-dominant timeout has occurred. 1 A TxD-dominant timeout has occurred.
6 LPOCIF	**LIN Transmitter Overcurrent Interrupt Flag** — LPOCIF is set to 1 when an overcurrent event happens. For protection, the transmitter is disabled. This flag can only be cleared by writing a 1. Writing a 0 has no effect. Please make sure that LPOCIF=1 before trying to clear it. Clearing LPOCIF is not allowed if LPOCIF=0 already. If the overcurrent is still present or LPTxD is dominant after clearing the flag, the transmitter stays disabled and this flag is set again (see18.4.4.1 Overcurrent Interrupt). If interrupt requests are enabled (LPOCIE= 1), LPOCIF causes an interrupt request. 0 No overcurrent event has occurred. 1 Overcurrent event has occurred.

18.4 Functional Description

18.4.1 General

The LIN Physical Layer module implements the physical layer of the LIN interface. This physical layer can be driven by the SCI (Serial Communication Interface) module or directly through the LPDR register.

18.4.2 Slew Rate and LIN Mode Selection

The slew rate can be selected for Electromagnetic Compatibility (EMC) optimized operation at 10.4 kbit/s and 20 kbit/s as well as at fast baud rate (up to 250 kbit/s) for test and programming. The slew rate can be chosen with the bits LPSLR[1:0] in the LIN Slew Rate Mode Register (LPSLRM). The default slew rate corresponds to 20 kbit/s.

The LIN Physical Layer can also be configured to be used for non-LIN applications (for example, to transmit a PWM pulse) by disabling the TxD-dominant timeout (LPDTDIS=1).

Changing the slew rate (LPSLRM Register) during transmission is not allowed in order to avoid unwanted effects. To change the register, the LIN Physical Layer must first be disabled (LPE=0). Once it is updated the LIN Physical Layer can be enabled again.

NOTE

For 20 kbit/s and Fast Mode communication speeds, the corresponding slew rate ***MUST*** be set; otherwise, the communication is not guaranteed (violation of the specified LIN duty cycles). For 10.4 kbit/s, the 20 kbit/s slew rate ***can*** be set but the EMC performance is worse. The up to 250 kbit/s slew rate must be chosen ***ONLY*** for fast mode, not for any of the 10.4 kbit/s or 20 kbit/s LIN compliant communication speeds.

18.4.2.1 10.4 kbit/s and 20 kbit/s

When the slew rate is chosen for 10.4 kbit/s or 20 kbit/s communication, a control loop is activated within the module to make the rise and fall times of the LIN bus independent from VLINSUP and the load on the bus.

18.4.2.2 Fast Mode (not LIN compliant)

Choosing this slew rate allows baud rates up to 250 kbit/s by having much steeper edges (please refer to electricals). As for the 10.4 kbit/s and 20 kbit/s modes, the slope control loop is also engaged. This mode is used for fast communication only, and the LIN electricals are not supported (for example, the LIN duty cycles).

A stronger external pullup resistor might be necessary to sustain communication speeds up to 250 kbit/s. The LIN signal (and therefore the receive LPRxD signal) might not be symmetrical for high baud rates with high loads on the bus.

Please note that if the bit time is smaller than the parameter t_{OCLIM} (please refer to electricals), then no overcurrent is reported nor does an overcurrent shutdown occur. However, the current limitation is always engaged in case of a failure.

18.4.3 Modes

Figure 18-11 shows the possible mode transitions depending on control bits, stop mode, and error conditions.

18.4.3.1 Shutdown Mode

The LIN Physical Layer is fully disabled. No wake-up functionality is available. The internal pullup resistor is high ohmic only (330 kΩ) to maintain the LIN Bus pin in the recessive state. LPTxD is not monitored in this mode for a TxD-dominant timeout. All the registers are accessible.

Setting LPE causes the module to leave the shutdown mode and to enter the normal mode or receive only mode (if RXONLY bit is set).

Clearing LPE causes the module to leave the normal or receive only modes and go back to shutdown mode.

18.4.3.2 Normal Mode

The full functionality is available. Both receiver and transmitter are enabled. The internal pullup resistor can be chosen to be high ohmic (330 kΩ) if LPPUE = 0, or LIN compliant (34 kΩ) if LPPUE = 1.

If RXONLY is set, the module leaves normal mode to enter receive only mode.

If the MCU enters stop mode, the LIN Physical Layer enters standby mode.

18.4.3.3 Receive Only Mode

Entering this mode disables the transmitter and immediately stops any on-going transmission. LPTxD is not monitored in this mode for a TxD-dominant timeout.

The receiver is running in full performance mode in all cases.

To return to normal mode, the RXONLY bit must be cleared.

If the device enters stop mode, the module leaves receive only mode to enter standby mode.

18.4.3.4 Standby Mode with Wake-Up Feature

The transmitter of the LIN Physical Layer is disabled and the receiver enters a low power mode.

NOTE

Before entering standby mode, ensure no transmissions are ongoing.

If LPWUE is not set, no wake up feature is available and the standby mode has the same electrical properties as the shutdown mode. This allows a low-power consumption of the device in stop mode if the wake-up feature is not needed.

If LPWUE is set the receiver is able to pass wake-up events to the SCI (Serial Communication Interface). If the LIN Physical Layer receives a dominant level longer than t_{WUFR} followed by a rising edge, it sends a pulse to the SCI which can generate a wake-up interrupt.

Once the device exits stop mode, the LIN Physical Layer returns to normal or receive only mode depending on the status of the RXONLY bit.

NOTE

Since the wake-up interrupt is requested by the SCI, the wake-up feature is not available if the SCI is not used.

The internal pullup resistor is selectable only if LPWUE = 1 (wake-up enabled). If LPWUE = 0, the internal pullup resistor is not selectable and remains at 330 kΩ regardless of the state of the LPPUE bit.

If LPWUE = 1, selecting the 330 kΩ pullup resistor (LPPUE = 0) reduces the current consumption in standby mode.

NOTE

When using the LIN wake-up feature in combination with other non-LIN device wake-up features (like a periodic time interrupt), some care must be taken.

If the device leaves stop mode while the LIN bus is dominant, the LIN Physical Layer returns to normal or receive only mode and the LIN bus is re-routed to the RXD pin of the SCI and triggers the edge detection interrupt (if the interrupt’s priority of the hardware that awakes the MCU is less than the priority of the SCI interrupt, then the SCI interrupt will execute first). It is up to the software to decide what to do in this case because the LIN Physical Layer can not guarantee it was a valid wake-up pulse.

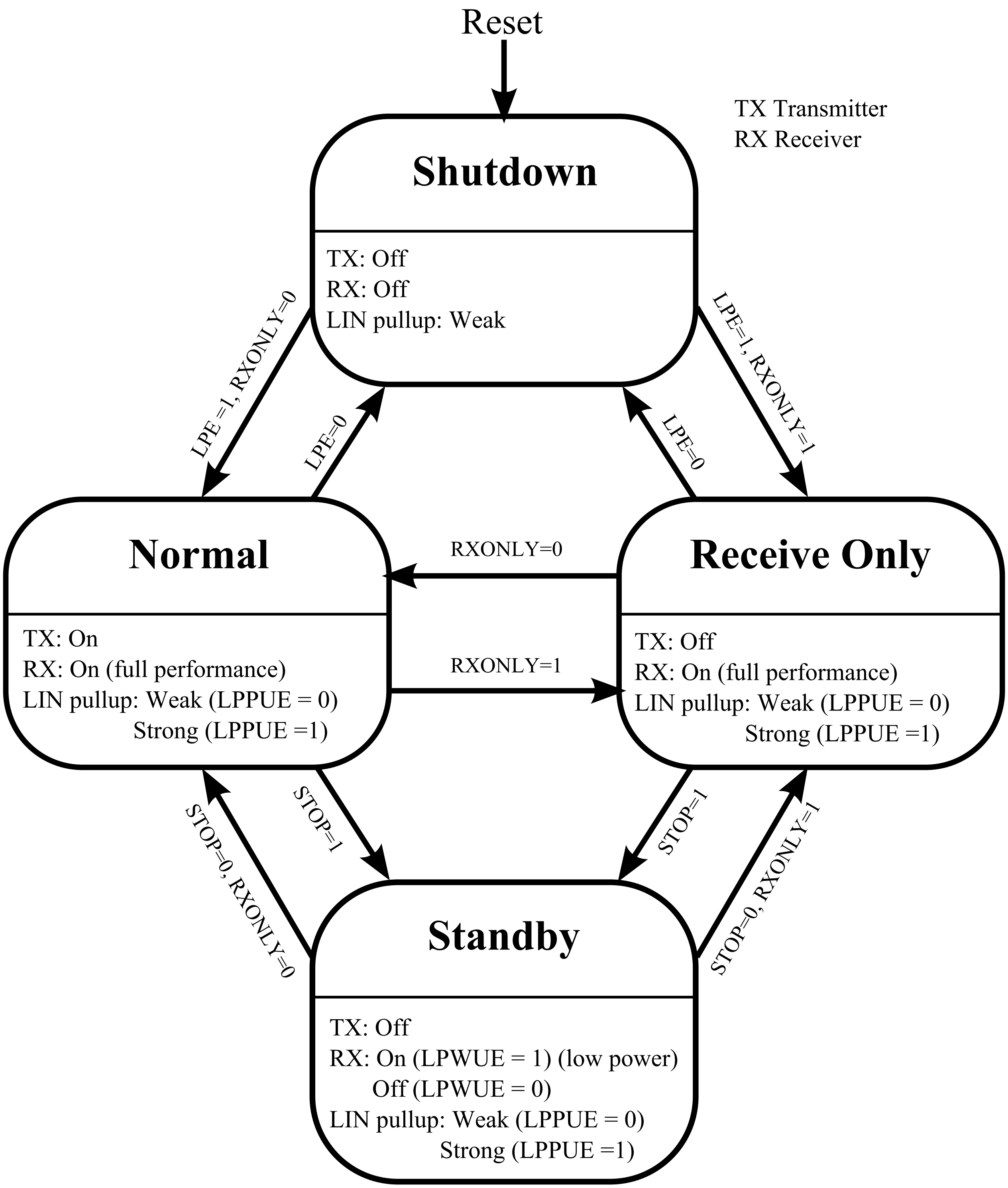


Figure 18-11. LIN Physical Layer Mode Transitions

18.4.4 Interrupts

The interrupt vector requested by the LIN Physical Layer is listed in Table 18-10. Vector address and interrupt priority is defined at the MCU level.

The module internal interrupt sources are combined into a single interrupt request at the device level.

Table 18-10. Interrupt Vectors

Module Interrupt Source	Module Internal Interrupt Source	Local Enable
LIN Interrupt (LPI)	LIN Txd-Dominant Timeout Interrupt (LPDTIF)	LPDTIE = 1
	LIN Overcurrent Interrupt (LPOCIF)	LPOCIE = 1

18.4.4.1 Overcurrent Interrupt

The transmitter is protected against overcurrent. In case of an overcurrent condition occurring within a time frame called t_{OCLIM} starting from LPTxD falling edge, the current through the transmitter is limited (the transmitter is not shut down). The masking of an overcurrent event within the time frame t_{OCLIM} is meant to avoid “false” overcurrent conditions that can happen during the discharging of the LIN bus. If an overcurrent event occurs out of this time frame, the transmitter is disabled and the LPOCIF flag is set.

In order to re-enable the transmitter again, the following prerequisites must be met:

1) Overcurrent condition is over

2) LPTxD is recessive or the LIN Physical Layer is in shutdown or receive only mode for a minimum of a transmit bit time.

To re-enable the transmitter then, the LPOCIF flag must be cleared (by writing a 1).

NOTE

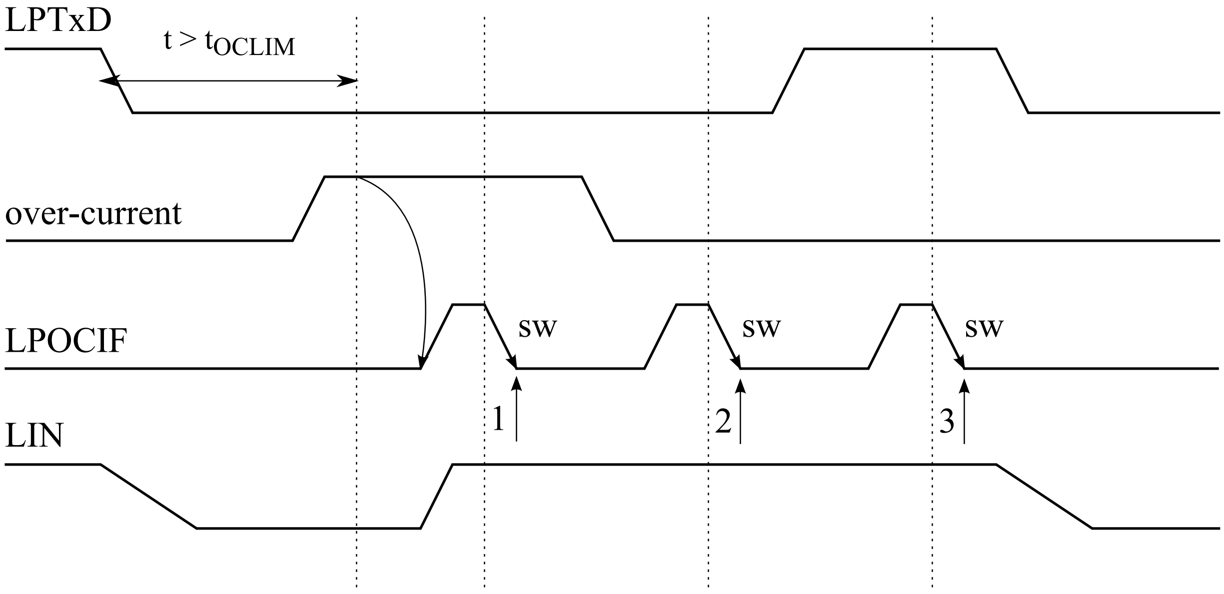
Please make sure that LPOCIF=1 before trying to clear it. It is not allowed to try to clear LPOCIF if LPOCIF=0 already.

After clearing LPOCIF, if the overcurrent condition is still present or the LPTxD pin is dominant while being in normal mode, the transmitter remains disabled and the LPOCIF flag is set again after a time to indicate that the attempt to re-enable has failed. This time is equal to:

- minimum 1 IRC period (1 us) + 2 bus periods
- maximum 2 IRC periods (2 us) + 3 bus periods

If the bit LPOCIE is set in the LPIE register, an interrupt is requested.

Figure 18-12 shows the different scenarios for overcurrent interrupt handling.



1: Flag cleared, transmitter re-enable not successful because over-current is still present
2: Flag cleared, transmitter re-enable not successful because LPTxD is dominant
3: Flag cleared, transmitter re-enable successful

Figure 18-12. Overcurrent interrupt handling

18.4.4.2 TxD-dominant timeout Interrupt

To protect the LIN bus from a network lock-up, the LIN Physical Layer implements a TxD-dominant timeout mechanism. When the LPTxD signal has been dominant for more than t_{DTLIM} the transmitter is disabled and the LPDT status flag and the LPDTIF interrupt flag are set.

In order to re-enable the transmitter again, the following prerequisites must be met:

1) TxD-dominant condition is over (LPDT=0)

2) LPTxD is recessive or the LIN Physical Layer is in shutdown or receive only mode for a minimum of a transmit bit time

To re-enable the transmitter then, the LPDTIF flag must be cleared (by writing a 1).

NOTE

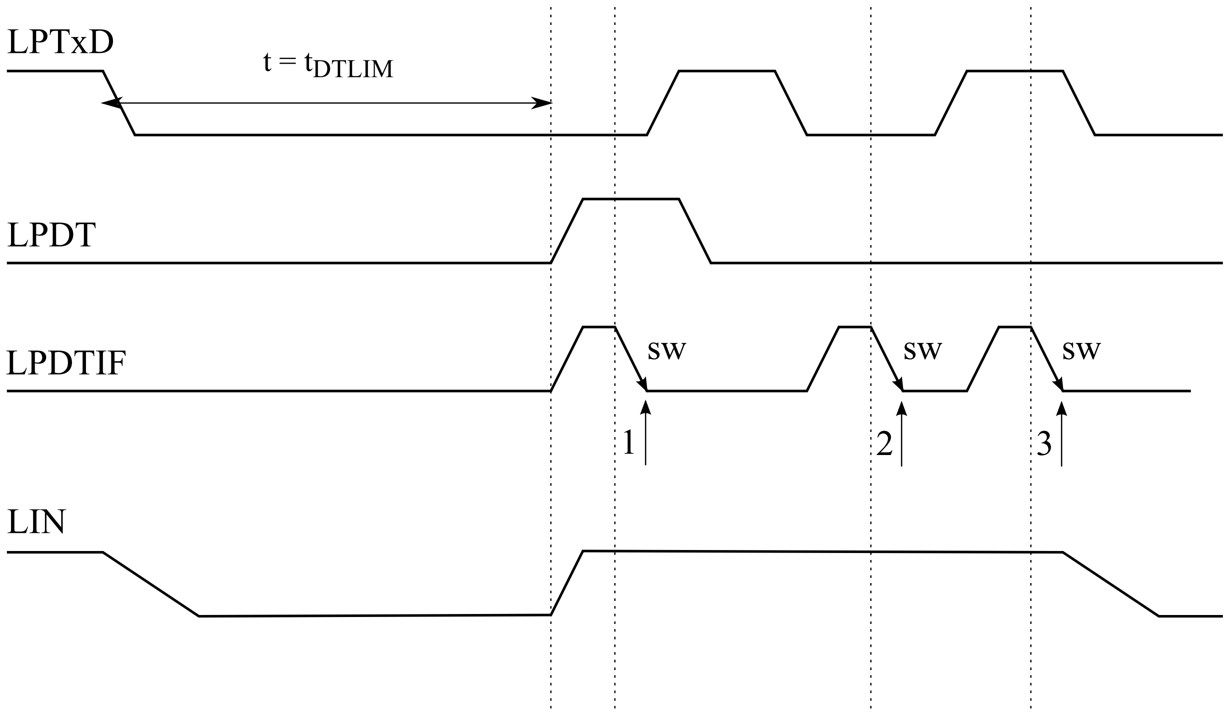
Please make sure that LPDTIF=1 before trying to clear it. It is not allowed to try to clear LPDTIF if LPDTIF=0 already.

After clearing LPDTIF, if the TxD-dominant timeout condition is still present or the LPTxD pin is dominant while being in normal mode, the transmitter remains disabled and the LPDTIF flag is set after a time again to indicate that the attempt to re-enable has failed. This time is equal to:

- minimum 1 IRC period (1 us) + 2 bus periods
- maximum 2 IRC periods (2 us) + 3 bus periods

If the bit LPDTIE is set in the LPIE register, an interrupt is requested.

Figure 18-13 shows the different scenarios of TxD-dominant timeout interrupt handling.



1: Flag cleared, transmitter re-enable not successful because TxD-dominant timeout condition is still present
2: Flag cleared, transmitter re-enable not successful because LPTxD is dominant
3: Flag cleared, transmitter re-enable successful

Figure 18-13. TxD-dominant timeout interrupt handling

18.5 Application Information

18.5.1 Module Initialization

The following steps should be used to configure the module before starting the transmission:

1. Set the slew rate in the LPSLRM register to the desired transmission baud rate.
2. When using the LIN Physical Layer for other purposes than LIN transmission, de-activate the dominant timeout feature in the LPSLRM register if needed.
3. In most cases, the internal pullup should be enabled in the LPCR register.
4. Route the desired source in the PIM module to the LIN Physical Layer.
5. Select the transmit mode (Receive only mode or Normal mode) in the LPCR register.
6. If the SCI is selected as source, activate the wake-up feature in the LPCR register if needed for the application (SCI active edge interrupt must also be enabled).
7. Enable the LIN Physical Layer in the LPCR register.
8. Wait for a minimum of a transmit bit.
9. Begin transmission if needed.

NOTE

It is not allowed to try to clear LPOCIF or LPDTIF if they are already cleared. Before trying to clear an error flag, always make sure that it is already set.

18.5.2 Interrupt handling in Interrupt Service Routine (ISR)

Both interrupts (TxD-dominant timeout and overcurrent) represent a failure in transmission. To avoid more disturbances on the transmission line, the transmitter is de-activated in both cases. The interrupt subroutine must take care of clearing the error condition and starting the routine that re-enables the transmission. For that purpose, the following steps are recommended:

1. First, the cause of the interrupt must be cleared:
   - The overcurrent will be gone after the transmitter has been disabled.
   - The TxD-dominant timeout condition will be gone once the selected source for LPTxD has turned recessive.
2. Clear the corresponding enable bit (LPDTIE or LPOCIE) to avoid entering the ISR again until the flags are cleared.
3. Notify the application of the error condition (LIN Error handler) and leave the ISR.

In the LIN Error handler, the following sequence is recommended:

1. Disable the LIN Physical Layer (LPCR) while re-configuring the transmission.
   - If the receiver must remain enabled, set the LIN Physical Layer into receive only mode instead.
2. Do all required configurations (SCI, etc.) to re-enable the transmission.
3. Wait for a transmit bit (this is needed to successfully re-enable the transmitter).

4. Clear the error flag.
5. Enable the interrupts again (LPDTIE and LPOCIE).
6. Enable the LIN Physical Layer or leave the receive only mode (LPCR register).
7. Wait for a minimum of a transmit bit before beginning transmission again.

If there is a problem re-enabling the transmitter, then the error flag will be set again during step 3 and the ISR will be called again.

Chapter 19
128 KB Flash Module (S12ZFTMRZ128K512V2)

Table 19-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.00	02 Jun 2011		- Initial version
V01.01	18 Aug 2011		- Updated description of Protection Override command, Section 19.4.7.17 - Added description of Erase-All pin feature, Section 19.4.7.7.1 - FCCOB register modified to provide a linear (non-paged) range of addresses on register space, Section 19.3.2 - Updated description of command parameters, Section 19.4.7
V01.02	01 Sep 2011		- Updated description of Erase-All pin feature, Section 19.4.7.7.1
V01.03	17 Oct 2011	19.3.2	- Removed wait state description (WSTAT).
V02.00	31 Jan 2012		- Added description of wait-state control bits, Section 19.3.2.4 and Section 19.3.2.5 - Updated section describing the procedure to enable/disable wait-states, Section 19.4.3
V02.01	06 Mar 2012		- Modified behavior of ECC flags, Section 19.3.2.8 - Attempt to access P-Flash and EEPROM simultaneously when not allowed will report illegal access, Section 19.4.6
V02.02	20 Mar 2012		- Updated feature description, Section 19.4.6
V02.03	12 Apr 2012	19.3	Corrected many typo. Changed caution note
V02.04	17 May 2012	19.3.2.6	- Removed flag DFDIE
V02.05	11 Jul 2012		- Added explanation about when MGSTAT[1:0] bits are cleared, Section 19.3.2.7 - Added note about possibility of reading P-Flash and EEPROM simultaneously, Section 19.4.6
V02.06	18 Mar 2013		- Standardized nomenclature in references to memory sizes
V02.07	24 May 2013		- Revised references to NVM Resource Area to improve readability
V02.8	12 Jun 2013		- Changed MLOADU Section 19.4.7.12 and MLOADF Section 19.4.7.13 **FCCOB1 to FCCOB2**

19.1 Introduction

The FTMRZ128K512 module implements the following:

- 128 KB of P-Flash (Program Flash) memory
- 512 bytes 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 Section 19.4.6.

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.

19.1.1 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.

19.1.2 Features

19.1.2.1 P-Flash Features

- 128 KB of P-Flash composed of one 128 KB Flash block divided into 256 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

19.1.2.2 EEPROM Features

- 512 bytes of EEPROM memory composed of one 512 bytes Flash block divided into 128 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

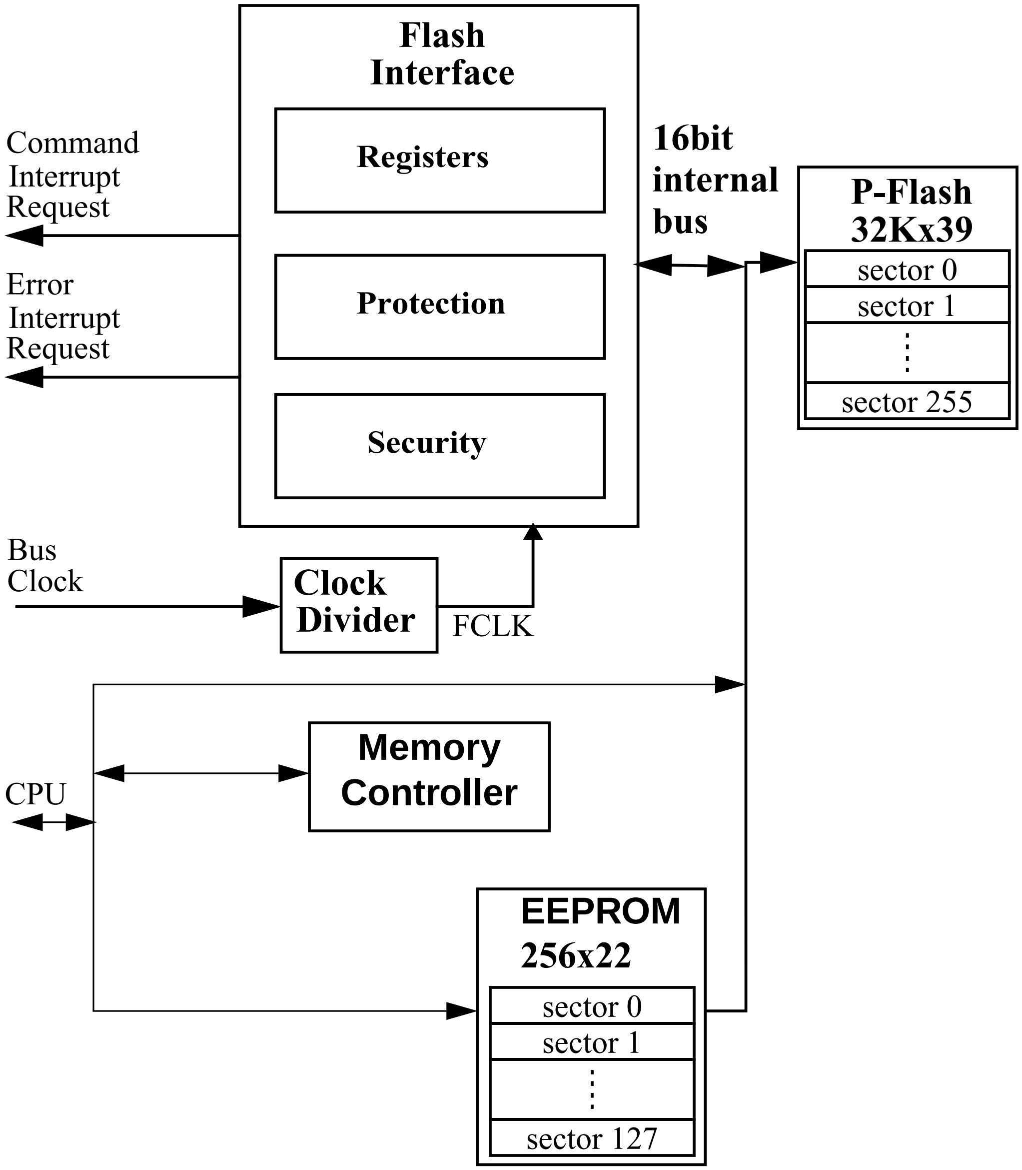
19.1.2.3 Other Flash Module Features

- 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

19.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 19-1.

Figure 19-1. FTMRZ128K512 Block Diagram



19.2 External Signal Description

The Flash module contains no signals that connect off-chip.

19.3 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 Section 19.6 for a complete description of the reset sequence).

.

Table 19-2. FTMRZ Memory Map

Global Address (in Bytes)	Size (Bytes)	Description
0x0_0000 – 0x0_0FFF	4,096	Register Space
0x10_0000 – 0x10_01FF	512	EEPROM memory
0x1F_4000 – 0x1F_FFFF	49,152	NVM Resource Area(1) (see Figure 19-3)
0xFE_0000 – 0xFF_FFFF	131,072	P-Flash Memory

1. See NVM Resource area description in Section 19.4.4

19.3.1 Module Memory Map

The S12Z architecture places the P-Flash memory between global addresses 0xFE_0000 and 0xFF_FFFF as shown in Table 19-3

The P-Flash memory map is shown in Figure 19-2.

Table 19-3. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0xFE_0000 – 0xFF_FFFF	128 K	P-Flash Block Contains Flash Configuration Field (see Table 19-4)

The FPROT register, described in Section 19.3.2.9, 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 19-4.

Table 19-4. Flash Configuration Field

Global Address	Size (Bytes)	Description
0xFF_FE00-0xFF_FE07	8	Backdoor Comparison Key Refer to Section 19.4.7.11, “Verify Backdoor Access Key Command,” and Section 19.5.1, “Unsecuring the MCU using Backdoor Key Access”
0xFF_FE08-0xFF_FE09[1]	2	Protection Override Comparison Key. Refer to Section 19.4.7.17, “Protection Override Command”
0xFF_FE0A-0xFF_FE0B(1)	2	Reserved
0xFF_FE0C[1]	1	P-Flash Protection byte. Refer to Section 19.3.2.9, “P-Flash Protection Register (FPROT)”
0xFF_FE0D[1]	1	EEPROM Protection byte. Refer to Section 19.3.2.10, “EEPROM Protection Register (DFPROT)”
0xFF_FE0E[1]	1	Flash Nonvolatile byte Refer to Section 19.3.2.11, “Flash Option Register (FOPT)”
0xFF_FE0F[1]	1	Flash Security byte Refer to Section 19.3.2.2, “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.

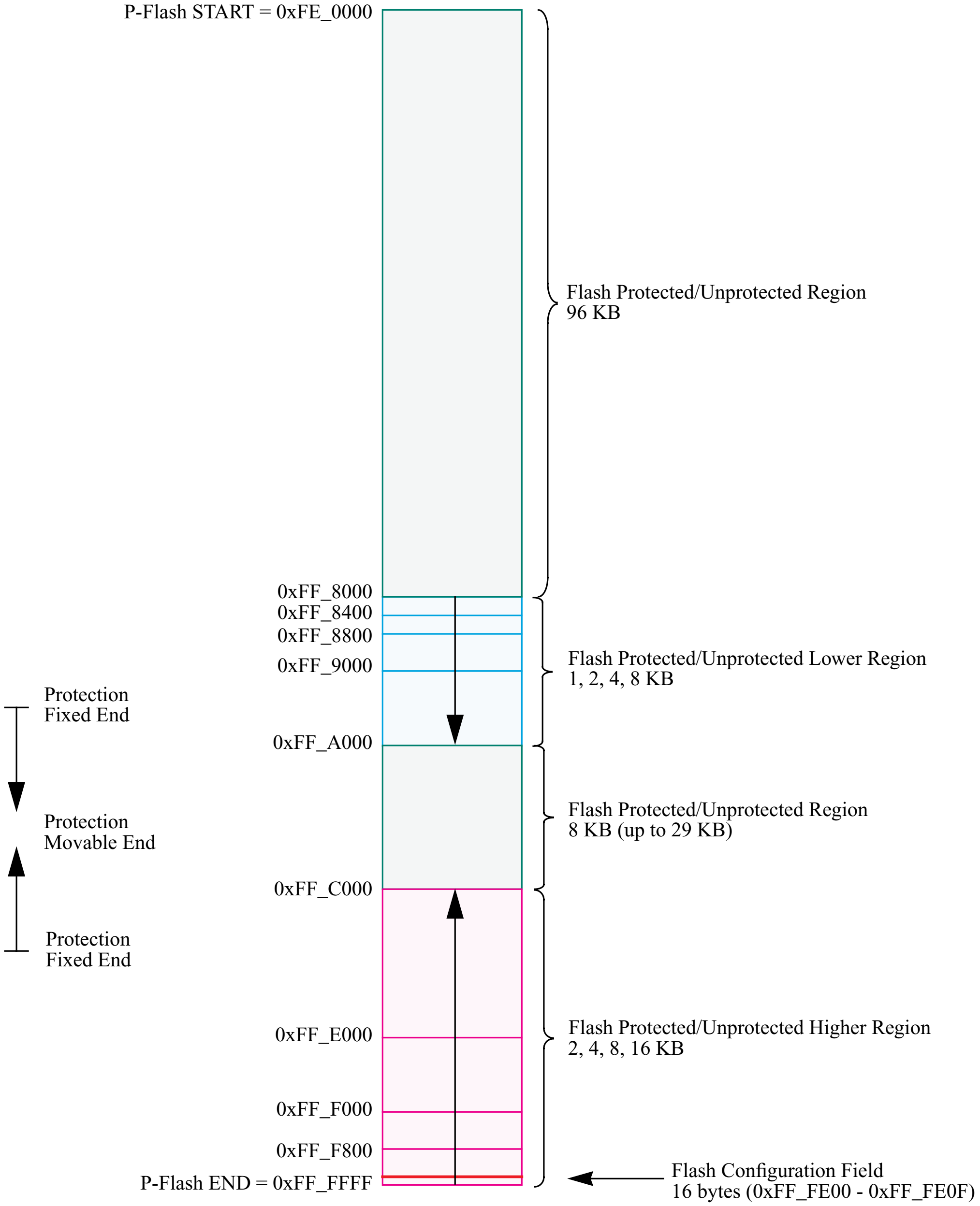


Figure 19-2. P-Flash Memory Map

Table 19-5. 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 19.4.7.6, “Program Once Command”

1. Used to track firmware patch versions, see Section 19.4.2

Table 19-6. 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 19-5)
0x1F_C100 – 0x1F_C1FF	256	Reserved.
0x1F_C200 – 0x1F_FFFF	15,872	Reserved.

1. See Section 19.4.4 for NVM Resources Area description.

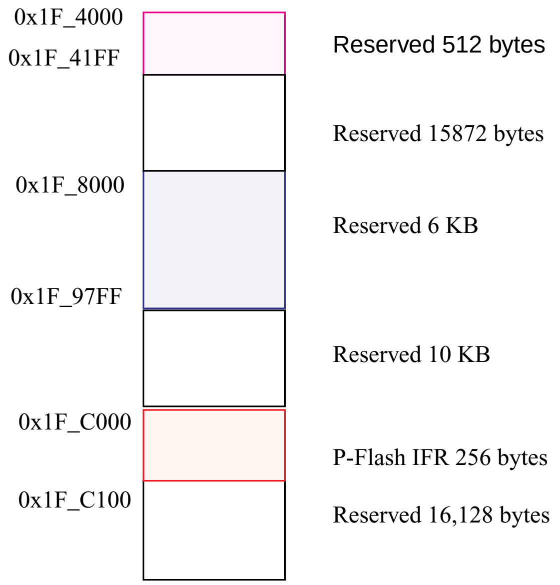


Figure 19-3. Memory Controller Resource Memory Map (NVM Resources Area)

19.3.2 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 Section 19.3).

A summary of the Flash module registers is given in Figure 19-4 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								
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								

Figure 19-4. FTMRZ128K512 Register Summary

Address & Name		7	6	5	4	3	2	1	0
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	0	0	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								
0x0010 FCCOB2HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								

Figure 19-4. FTMRZ128K512 Register Summary (continued)

Address & Name		7	6	5	4	3	2	1	0
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 19-4. FTMRZ128K512 Register Summary (continued)

19.3.2.1 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 19-5. 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 19-7. 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 19-8 shows recommended values for FDIV[5:0] based on the BUSCLK frequency. Please refer to Section 19.4.5, “Flash Command Operations,” for more information.

Table 19-8. FDIV values for various BUSCLK Frequencies

BUSCLK Frequency (MHz)		FDIV[5:0]	BUSCLK Frequency (MHz)		FDIV[5:0]
MIN(1)	MAX(2)		MIN[1]	MAX[2]	
1.0	1.6	0x00	26.6	27.6	0x1A
1.6	2.6	0x01	27.6	28.6	0x1B
2.6	3.6	0x02	28.6	29.6	0x1C
3.6	4.6	0x03	29.6	30.6	0x1D
4.6	5.6	0x04	30.6	31.6	0x1E
5.6	6.6	0x05	31.6	32.6	0x1F
6.6	7.6	0x06	32.6	33.6	0x20
7.6	8.6	0x07	33.6	34.6	0x21
8.6	9.6	0x08	34.6	35.6	0x22
9.6	10.6	0x09	35.6	36.6	0x23
10.6	11.6	0x0A	36.6	37.6	0x24
11.6	12.6	0x0B	37.6	38.6	0x25
12.6	13.6	0x0C	38.6	39.6	0x26
13.6	14.6	0x0D	39.6	40.6	0x27
14.6	15.6	0x0E	40.6	41.6	0x28
15.6	16.6	0x0F	41.6	42.6	0x29
16.6	17.6	0x10	42.6	43.6	0x2A
17.6	18.6	0x11	43.6	44.6	0x2B

Table 19-8. FDIV values for various BUSCLK Frequencies

BUSCLK Frequency (MHz)		FDIV[5:0]		BUSCLK Frequency (MHz)		FDIV[5:0]
MIN(1)	MAX(2)			MIN[1]	MAX[2]	
18.6	19.6	0x12		44.6	45.6	0x2C
19.6	20.6	0x13		45.6	46.6	0x2D
20.6	21.6	0x14		46.6	47.6	0x2E
21.6	22.6	0x15		47.6	48.6	0x2F
22.6	23.6	0x16		48.6	49.6	0x30
23.6	24.6	0x17		49.6	50.6	0x31
24.6	25.6	0x18				
25.6	26.6	0x19				

1. BUSCLK is Greater Than this value.

2. BUSCLK is Less Than or Equal to this value.

19.3.2.2 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 19-6. 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 19-4) as indicated by reset condition F in Figure 19-6. 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 19-9. 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 19-10.
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 19-11. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 19-10. 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 19-11. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED(1)
10	UNSECURED
11	SECURED

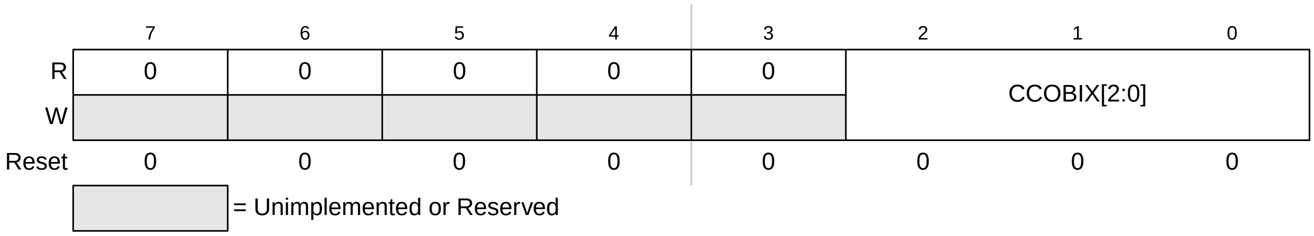
1. Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 19.5.

19.3.2.3 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 19-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 19-12. 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 Section 19.3.2.13, "Flash Common Command Object Registers (FCCOB)"," for more details.

19.3.2.4 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 19-8. Flash Protection Status Register (FPSTAT)

All bits in the FPSTAT register are readable but are not writable.

Table 19-13. 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 19.4.7.17, "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 19.3.2.5, "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 19.4.3, "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]

19.3.2.5 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 19-9. 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 19-14. 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 Section 19.3.2.7)
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 19-9 of Section 19.3.2.2. 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 Section 19.4.7.7.1).
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 19.3.2.8). 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 19-15.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 Section 19.4.3. 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.

Table 19-14. FCNFG Field Descriptions (continued)

Field	Description
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 Section 19.3.2.7)
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 Section 19.3.2.7) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 19.3.2.6)

Table 19-15. 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).

19.3.2.6 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 19-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 19-16. 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 Section 19.3.2.8) 1 An interrupt will be requested whenever the SFDIF flag is set (see Section 19.3.2.8)

19.3.2.7 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 19-11. 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 Section 19.6).

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 19-17. 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 Section 19.4.5.2) 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)

Table 19-17. FSTAT Field Descriptions (continued)

Field	Description
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 19.4.7, “Flash Command Description,” and Section 19.6, “Initialization” for details.

19.3.2.8 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 19-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 19-18. 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 19.4.3, “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.

19.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

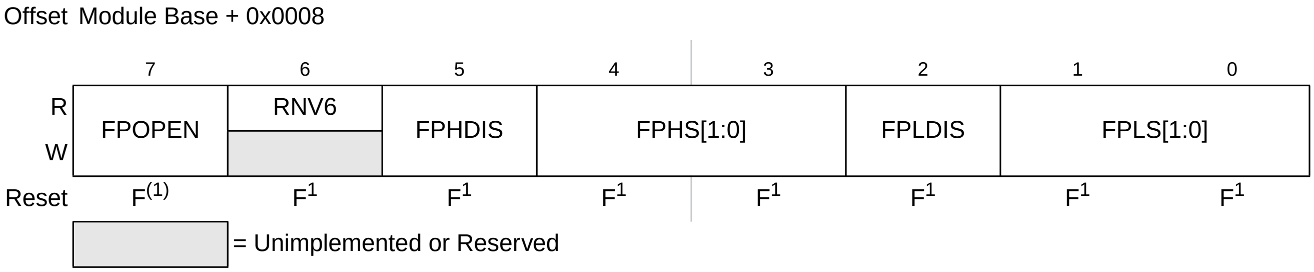


Figure 19-13. Flash Protection Register (FPROT)

1. Loaded from Flash configuration field, during reset sequence.

The (unreserved) bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased(see Section 19.3.2.9.1, “P-Flash Protection Restrictions,” and Table 19-23).

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 19-4) as indicated by reset condition ‘F’ in Figure 19-13. 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 19-19. 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 19-20 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 19-21. The FPHS bits can only be written to while the FPHDIS bit is set.

Table 19-19. FPROT Field Descriptions (continued)

Field	Description
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 19-22. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 19-20. 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 19-21 and Table 19-22.

Table 19-21. 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 19-22. 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 19-14 . 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 single chip mode while providing as much protection as possible if reprogramming is not required.

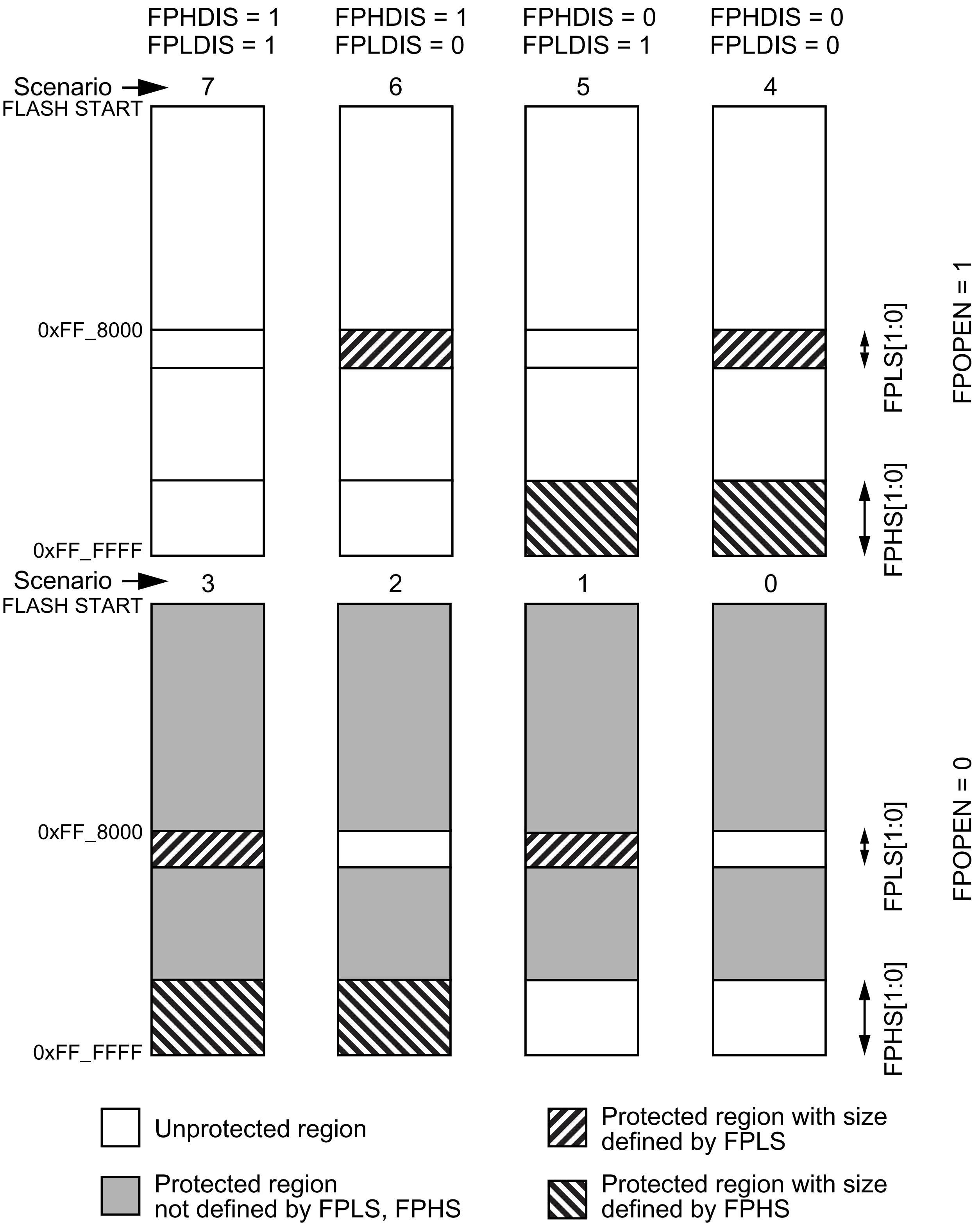


Figure 19-14. P-Flash Protection Scenarios

19.3.2.9.1 P-Flash Protection Restrictions

The general guideline is that P-Flash protection can only be added and not removed. Table 19-23 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 19-23. 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 19-14 for a definition of the scenarios.

19.3.2.10 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	0	0	DPS[3:0]			
W		▨	▨	▨				
Reset	F(1)	0	0	0	F[1]	F[1]	F[1]	F[1]

▨ = Unimplemented or Reserved

Figure 19-15. EEPROM Protection Register (DFPROT)

1. Loaded from IFR Flash configuration field, during reset sequence.

The (unreserved) bits of the DFPROT register are writable with the restriction that protection can be added but not removed. Writes 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.

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 19-4) as indicated by reset condition F in Table 19-25. 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 19-24. 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
3–0 DPS[3:0]	**EEPROM Protection Size** — The DPS[3:0] bits determine the size of the protected area in the EEPROM memory as shown inTable 19-25 .

Table 19-25. EEPROM Protection Address Range

DPS[3:0]	Global Address Range	Protected Size
0000	0x10_0000 – 0x10_001F	32 bytes
0001	0x10_0000 – 0x10_003F	64 bytes
0010	0x10_0000 – 0x10_005F	96 bytes
0011	0x10_0000 – 0x10_007F	128 bytes
0100	0x10_0000 – 0x10_009F	160 bytes
0101	0x10_0000 – 0x10_00BF	192 bytes
The Protection Size goes on enlarging in step of 32 bytes, for each DPS value increasing of one. . . .		
1111	0x10_0000 – 0x10_01FF	512 bytes

19.3.2.11 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 19-16. 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 19-4) as indicated by reset condition F in Figure 19-16. 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 19-26. 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 user guide for proper use of the NV bits.

19.3.2.12 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 19-17. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

19.3.2.13 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 W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 19-18. Flash Common Command Object 0 High Register (FCCOB0HI)

Offset Module Base + 0x000D

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

Figure 19-19. Flash Common Command Object 0 Low Register (FCCOB0LO)

Offset Module Base + 0x000E

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

Figure 19-20. Flash Common Command Object 1 High Register (FCCOB1HI)

Offset Module Base + 0x000F

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

Figure 19-21. Flash Common Command Object 1 Low Register (FCCOB1LO)

Offset Module Base + 0x0010

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

Figure 19-22. 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 19-23. 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 19-24. 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 19-25. 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 19-26. 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 19-27. 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 19-28. 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 19-29. Flash Common Command Object 5 Low Register (FCCOB5LO)

19.3.2.13.1 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 19-27. 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 19-27 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 Section 19.4.7.

Table 19-27. 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]
010	FCCOB2	HI	Data 0 [15:8]
		LO	Data 0 [7:0]

Table 19-27. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Register	Byte	FCCOB Parameter Fields (NVM Command Mode)
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]

19.4 Functional Description

19.4.1 Modes of Operation

The FTMRZ128K512 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 19-29).

19.4.2 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 19-28.

Table 19-28. 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’.

19.4.3 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 Section 19.3.2.5) 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 FTMRZ128K512 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 FTMRZ128K512 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 Section 19.3.2.5) 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 19.3.2.1, "Flash Clock Divider Register (FCLKDIV) and Section 19.4.5.1, "Writing the FCLKDIV Register.

19.4.4 Internal NVM resource

IFR is an internal NVM resource readable by CPU . The IFR fields are shown in Table 19-5.

The NVM Resource Area global address map is shown in Table 19-6.

19.4.5 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.

19.4.5.1 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 19-8 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.

19.4.5.2 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 Section 19.3.2.7) 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.

19.4.5.2.1 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 Section 19.3.2.3).

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 19-30.

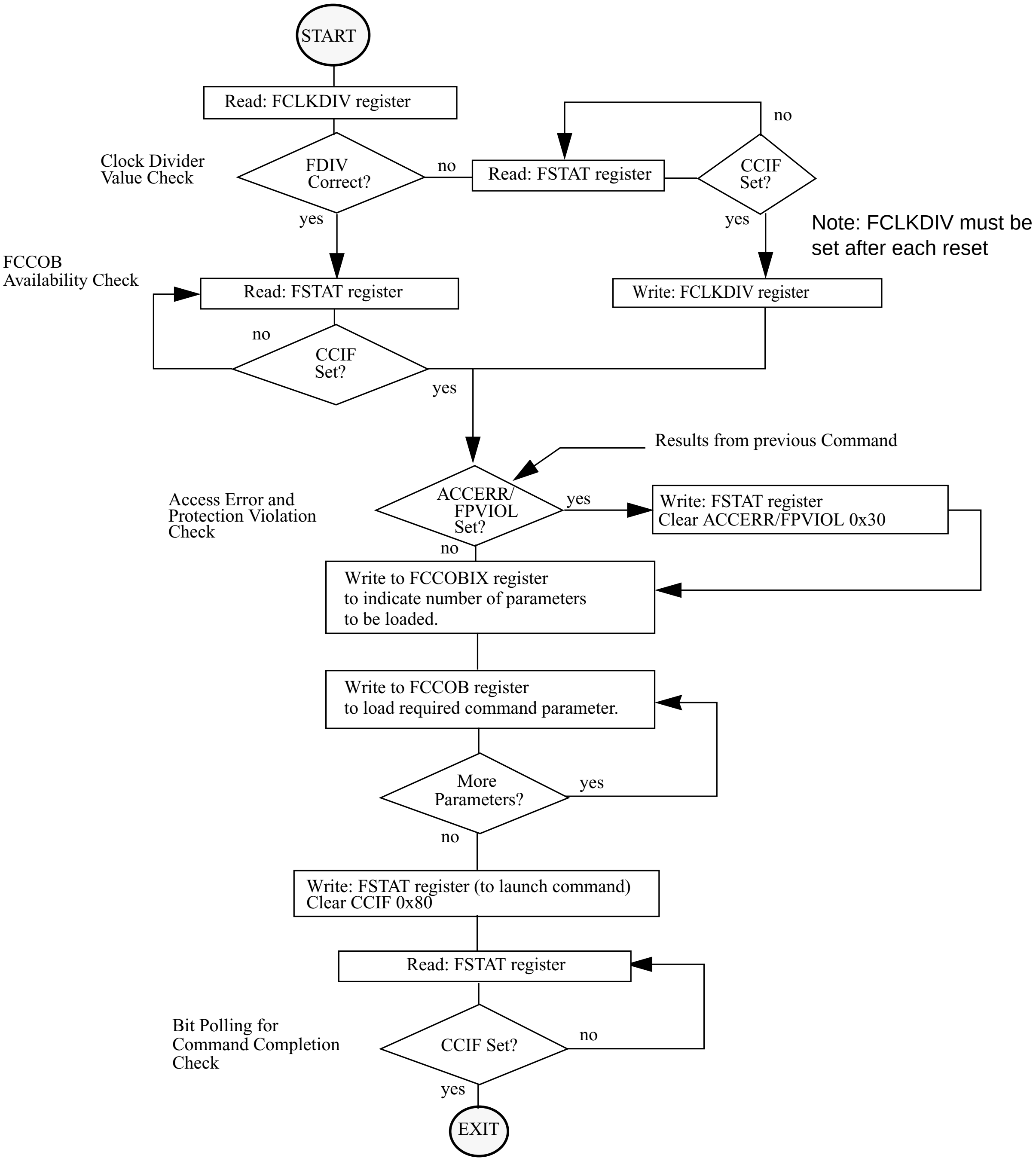


Figure 19-30. Generic Flash Command Write Sequence Flowchart

19.4.5.3 Valid Flash Module Commands

Table 19-29 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 19-29. 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.

19.4.5.4 P-Flash Commands

Table 19-30 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 19-30. 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.

19.4.5.5 EEPROM Commands

Table 19-31 summarizes the valid EEPROM commands along with the effects of the commands on the EEPROM block.

Table 19-31. 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.

19.4.6 Allowed Simultaneous P-Flash and EEPROM Operations

Only the operations marked ‘OK’ in Table 19-32 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 19-32. 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 Section 19.4.7.12 and Section 19.4.7.13.
3. The 'Mass Erase' operations are commands 'Erase All Blocks' and 'Erase Flash Block'

19.4.7 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 Section 19.4.6).

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 Section 19.3.2.7).

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.

19.4.7.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and EEPROM blocks have been erased.

Table 19-33. 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 19-34. 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 reador if blank check failed .
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or if blank check failed.

19.4.7.2 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 19-35. 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 19-36. 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 19-3)
	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.

19.4.7.3 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 19-37. 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 19-38. 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 19-29)
		Set if an invalid global address [23:0] is supplied see Table 19-3)
		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.

19.4.7.4 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 Section 19.4.7.6. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 19-39. 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 19-40. 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 19-29)
		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

19.4.7.5 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 19-41. 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 19-42. 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 19-29)
		Set if an invalid global address [23:0] is supplied see Table 19-3)
		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

19.4.7.6 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 Section 19.4.7.4. 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 19-43. 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 19-43. 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 19-44. 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 19-29)
		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.

19.4.7.7 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and EEPROM memory space.

Table 19-45. 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 19-46. 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 19-29)
	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

19.4.7.7.1 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 Section 19.3.2.1) 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 Section 19.3.2.2). The security byte in the Flash Configuration Field will be programmed to the unsecure state (see Table 19-9). The status of the erase-all request is reflected in the ERSAREQ bit in the FCNFG register (see Section 19.3.2.5). 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 19-47.

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 19-47. Erase All Pin Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if command not available in current mode (see Table 19-29)
	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

19.4.7.8 Erase Flash Block Command

The Erase Flash Block operation will erase all addresses in a P-Flash or EEPROM block.

Table 19-48. 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 19-49. 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 19-29)
		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

19.4.7.9 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 19-50. 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 Section 19.1.2.1 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 19-51. 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 19-29)
		Set if an invalid global address [23:0] is supplied see Table 19-3)
		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

19.4.7.10 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 19-52. 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 19-53. 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 19-29)
	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

19.4.7.11 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 19-10). 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 19-

4). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 19-54. 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 19-55. 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 Section 19.3.2.2)
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

19.4.7.12 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 19-56. 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 19-56. 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 19-57.

Table 19-57. 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 19-58. 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 19-29)
		Set if an invalid global address [23:0] is supplied see Table 19-3)
		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.

19.4.7.13 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 19-59. 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 19-60.

Table 19-60. 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 19-61. 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 19-29)
		Set if an invalid global address [23:0] is supplied see Table 19-3)
		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.

19.4.7.14 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 19-62. 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 19-63. 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 19-29)
		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.

19.4.7.15 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 19-64. 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 19-65. 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 19-29)
		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

19.4.7.16 Erase EEPROM Sector Command

The Erase EEPROM Sector operation will erase all addresses in a sector of the EEPROM block.

Table 19-66. 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 Section 19.1.2.2 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 19-67. 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 19-29)
		Set if an invalid global address [23:0] is suppliedsee Table 19-3
		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

19.4.7.17 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 19-4). 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 19.3.2.4, "Flash Protection Status Register (FPSTAT)).

Table 19-68. Protection Override Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x13	Protection Update Selection [1:0] See Table 19-69
FCCOB1	Comparison Key	
FCCOB2	reserved	New FPROT value
FCCOB3	reserved	New DFPROT value

Table 19-69. 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 19-70. 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 19-29).
		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

19.4.8 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 19-71. 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.

19.4.8.1 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 19.3.2.5, “Flash Configuration Register (FCNFG)”, Section 19.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 19.3.2.7, “Flash Status Register (FSTAT)”, and Section 19.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 19-31.

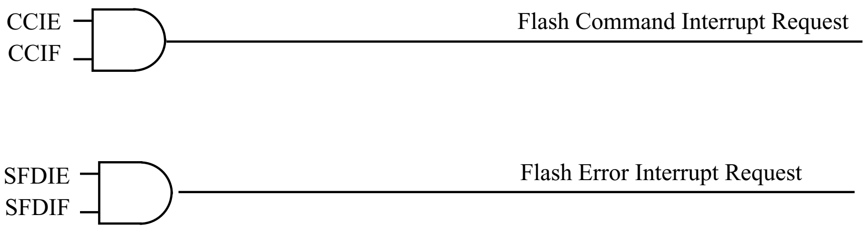


Figure 19-31. Flash Module Interrupts Implementation

19.4.9 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 19.4.8, “Interrupts”).

19.4.10 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.

19.5 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 19-11). 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

19.5.1 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 Section 19.3.2.2), the Verify Backdoor Access Key command (see Section 19.4.7.11) 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 19-11) 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 Section 19.3.2.2), 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 Section 19.4.7.11
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.

19.5.2 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 19.4.7.7.1, “Erase All Pin”, For a complete description about how to activate that procedure please refer to the Reference Manual. Alternatively, a similar (non-automated) procedure to unsecure the MCU in special single chip mode can be done by using the following method to erase the P-Flash and EEPROM memory:

1. Reset the MCU into special single chip mode

2. Delay while the BDM executes the Erase Verify All Blocks command write sequence to check if the P-Flash and EEPROM memories are erased
3. Send BDM commands to disable protection in the P-Flash and EEPROM memory
4. Execute the Erase All Blocks command write sequence to erase the P-Flash and EEPROM memory. Alternatively the Unsecure Flash command can be executed, if so the steps 5 and 6 below are skipped.
5. After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode
6. Delay while the BDM executes the Erase Verify All Blocks command write sequence to verify that the P-Flash and EEPROM memory are erased

If the P-Flash and EEPROM memory are verified as erased, the MCU will be unsecured. All BDM commands will now be enabled and the Flash security byte may be programmed to the unsecure state by continuing with the following steps:

7. Send BDM commands to execute the Program P-Flash command write sequence to program the Flash security byte to the unsecured state
8. Reset the MCU

19.5.3 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 19-29.

19.6 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

A.1 General

This supplement contains the most accurate electrical information for the MC9S12ZVM-Family available at the time of publication.

A.1.1 Parameter Classification

The electrical parameters shown in this supplement are guaranteed by various methods. To give the customer a better understanding the following classification is used and the parameters are tagged accordingly in the tables where appropriate.

NOTE

This classification is shown in the column labeled “C” in the parameter tables where appropriate.

- P: These parameters are guaranteed during production testing on each individual device.
- M: These parameters are characterized at 175°C and tested in production over an ambient temperature range of -40°C to 150°C with appropriate guardbanding to guarantee operation at 175°C.
- C: These parameters are achieved in design characterization by measuring a statistically relevant sample size across process variations.
- T: These parameters are achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted. All values shown in the typical column are within this category.
- D: These parameters are derived mainly from simulations.

Table A-1. Power Supplies

Mnemonic	Nominal Voltage	Description
VDD	1.8 V	1.8V core supply voltage generated by on chip voltage regulator
VSS1	0 V	Ground pin for 1.8V core supply voltage generated by on chip voltage regulator
VSS2	0 V	Ground pin for 1.8V core supply voltage generated by on chip voltage regulator
VDDF	2.8 V	2.8V flash supply voltage generated by on chip voltage regulator
$VDDX_1$(1)	5.0 V	5V power supply output for I/O drivers generated by on chip voltage regulator
$VSSX_1$	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
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
LGND	0 V	Ground pin for LIN physical interface
HD	12 V	GDU Highside Drain. Also used as LIN supply, VLINSUP.
VSUP	12 V/18 V	External power supply for voltage regulator
VDDC	5 V	Power supply output for external CANPHY device
VLS_OUT	11 V	GDU voltage regulator output for low side FET-predriver power supply.
VSSB	0 V	Ground pin for boost supply.

1. All VDDX pins are internally connected by metal

NOTE

VDDA is connected to VDDX pins by diodes for ESD protection such that VDDX must not exceed VDDA by more than a diode voltage drop. VSSA and VSSX 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 range of 4.5V to 5.5V. 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 LIN, BST, HD, VCP, CP, VLS_OUT, VLS[2:0], VBS[2:0], HG[2:0], HS[2:0], LG[2:0] 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.

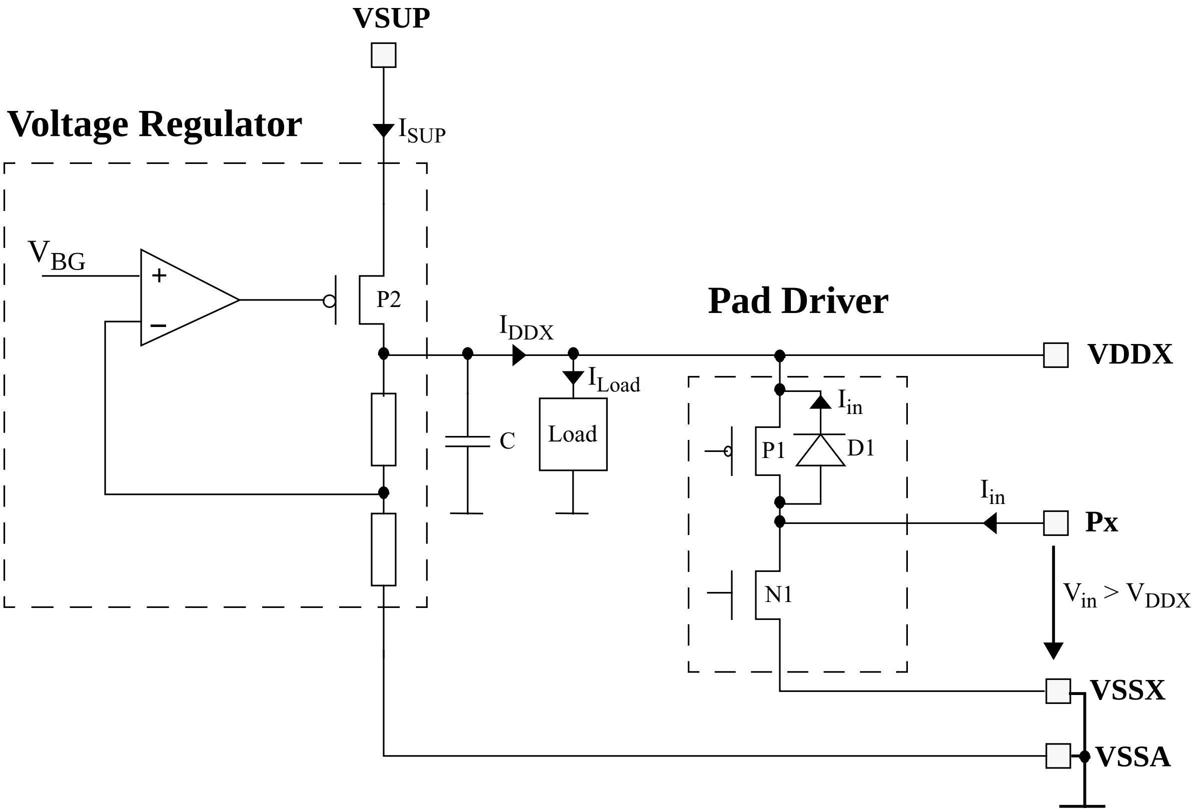
A.1.2.4 TEST

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 and LINPHY supply voltage	V_{SUP}	-0.3	42	V
2	DC voltage on LIN	V_{LIN}	-32	42	V
3	Core logic supply voltage	V_{DD}	-0.3	2.16	V
4	Flash supply voltage	V_{DDF}	-0.3	3.6	V
5	FET-Predriver High-Side Drain	V_{HD}	-0.3	42	V
6	FET-Predriver Bootstrap Capacitor Connection	V_{VBS}	-0.3	42	V
7	FET-Predriver High-Side Gate(1)	V_{HG}	-5	42	V
8	FET-Predriver High-Side Source(1)	V_{HS}	-5	42	V
9	Generated FET-Predriver Low-Side Supply	V_{VLS_OUT}	-0.3	42	V
10	FET-Predriver Low-Side Supply Inputs	V_{VLS}	-0.3	42	V
11	FET-Predriver Low-Side Gate(1)	V_{LG}	-5	42	V
12	FET-Predriver Low-Side Source(1)	V_{LS}	-5	42	V
13	FET-Predriver Charge Pump Output	V_{CP}	-0.3	42	V
14	FET-Predriver Charge Pump Input	V_{VCP}	-0.3	42	V
15	FET-Predriver Boost Converter Connection	V_{BST}	-0.3	42	V
16	FET-Predriver Boost Converter Ground	V_{VSSB}	-0.3	0.3	V
17	Voltage Regulator Ballast Connection	V_{BCTL}	-0.3	42	V
18	Supplies VDDA, VDDC, VDDX	V_{VDDACX}	-0.3	6	V
19	Base connection of bipolar for CANPHY supply	V_{BCTLC}	-0.3	42	V
20	Voltage difference V_{DDX} to V_{DDA}(2)	Δ_{VDDX}	–0.3	0.3	V
21	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.3	0.3	V
22	Digital I/O input voltage	V_{IN}	–0.3	6.0	V
23	EXTAL, XTAL (3)	V_{ILV}	–0.3	2.16	V
24	TEST input	V_{TEST}	–0.3	10.0	V
25	Instantaneous current. Single pin limit for all digital I/O pins(4)	I_D	–25	+25	mA
26	Instantaneous maximum current on EVDD1	I_{EVDD1}	-80	+25	mA
27	Instantaneous maximum current. Single pin limit for EXTAL, XTAL	I_{DL}	–25	+25	mA

Table A-2. Absolute Maximum Ratings

28	Storage temperature range	T .	–65	155	°C

1. Negative limit for pulsed operation only
2. VDDX and VDDA must be shorted
3. EXTAL, XTAL pins configured for external oscillator operation only
4. 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.

For better immunity to ESD events, the external diode and test point for the BST pin should be located at a distance from the device to increase the track length to the BST pin.

Table A-3. ESD and Latch-up Test Conditions

Model	Spec	Description	Symbol	Value	Unit
Human Body	JESD22-A114	Series Resistance	R	1500	W
		Storage Capacitance	C	100	pF
		Number of Pulses per pin positive negative	-	- 1 1	
Charged-Device	JESD22-C101	Series Resistance	R	0	W
		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 HD/VCP/BST/LIN/BCTL/BCTLC		Minimum Input Voltage Limit		-7	V
		Maximum Input Voltage Limit		+27	V
Latch-up for HG,HS		Minimum Input Voltage Limit		-5	V
		Maximum Input Voltage Limit (VBS=10V)		15	V
Latch-up for LG,LS		Minimum Input Voltage Limit		-5	V
		Maximum Input Voltage Limit (VLS=10V)		15	V

Table A-4. ESD Protection and Latch-up Characteristics

Num	C	Rating	Symbol	Min	Max	Unit
1	C	Human Body Model (HBM): -LIN versus LGND -all other pins	V_{HBM} V_{HBM}	+/-6 +/-2	-	KV
2	C	Charged-Device Model (CDM): Corner Pins	V_{CDM}	+/-750	-	V
3	C	Charged-Device Model (CDM): All other pins	V_{CDM}	+/-500	-	V
4	C	Direct Contact Discharge IEC61000-4-2 with and with out 220pF capacitor (R=330, C=150pF): LIN versus LGND	V_{ESDIEC}	+/-6	-	KV
5	C	Latch-up Current of 5V GPIOs at T=125°C positive negative	I_{LAT}	+100 -100	-	mA
6	C	Latch-up Current (VCP, BST, LIN, HD, HS, HG, LG, LS) T=125°C positive negative	I_{LAT}	+100 -100	-	mA
7	C	Latch-up Current of 5V GPIOs at 27°C positive negative	I_{LAT}	+200 -200	-	mA
8	C	Latch-up Current (VCP, BST, LIN, HD, HS, HG, LG, LS) T= 27°C positive negative	I_{LAT}	+200 -200	-	mA

A.1.6 Recommended Capacitor Values

Table A-5. Recommended Capacitor Values (nominal component values)

Num	Characteristic	Symbol	Typical	Unit
1	VDDX decoupling capacitor (1) (2)	$C_{VDDX1,2}$	100-220	nF
2	VDDA decoupling capacitor (1)	C_{VDDA}	100-220	nF
3	VDDX stability capacitor (3) (4)	C_{VDD5}	4.7-10	µF
4	VLS decoupling capacitor (1) (5)	$C_{VLS0,1,2}$	100-220	nF
5	VLS stability capacitor (3) (6)	C_{VLS}	4.7-10	µF
6	VDD decoupling capacitor (1)	C_{VDD}	100-220	nF
7	VDDF decoupling capacitor (1)	C_{VDDF}	100-220	nF
8	LIN decoupling capacitor (1)	C_{LIN}	220	pF

1. X7R ceramic
2. One capacitor per VDDX pin
3. 4.7µF ceramic or 10µF tantalum
4. Can be placed anywhere on the 5V supply node (VDDA, VDDX)
5. One capacitor per each VLS[2:0] pin
6. Can be placed anywhere on the VLS node

A.1.7 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.1.8, "Power Dissipation and Thermal Characteristics".

Table A-6. Operating Conditions

Num	Rating	Symbol	Min	Typ	Max	Unit
1	Voltage regulator and LINPHY 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}	–0.1	—	0.1	V
4	Oscillator	f_{osc}	4	—	16	MHz
5	Bus frequency(2) -40°C < T_j < 150°C 150°C < T_j < 175°C (Temp option W only)	f_{bus}	(4)	— —	50 40	MHz
6	Bus frequency without flash wait states -40°C < T_j < 150°C 150°C < T_j < 175°C (Temp option W only)	f_{WSTAT}		— —	25 20	MHz
7a	Operating junction temperature range Operating ambient temperature range(3) (option V)	T_J T_A	–40 –40	— —	125 105	°C
7b	Operating junction temperature range Operating ambient temperature range(3) (option M)	T_J T_A	–40 –40	— —	150 125	°C
7c	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.
2. The flash program and erase operations must configure f_{NVMOP} as specified in the NVM electrical section.
3. Please refer to Section A.1.8, "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.1.8 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-7 gives an overview of the supply currents.

$$P_D = P_{VSUP} + P_{BCTL} + P_{INT} - P_{GPIO} + P_{LIN} - P_{EVDD1} + P_{GDU}$$

Table A-7. 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
$P_{LIN} = V_{LIN}\ I_{LIN}$	Power dissipation of LINPHY
P_{GDU}(1) $= (-V_{VLS\_OUT}\ I_{VLS\_OUT}) + (V_{VBS}\ I_{VBS}) + (V_{VCP} I_{VCP}) + (V_{VLSn}\ I_{VLSn})$	Power dissipation of FET-Predriver without the outputs switching

1. No switching. GDU power consumption is very load dependent.

Figure A-2. Supply Currents Overview

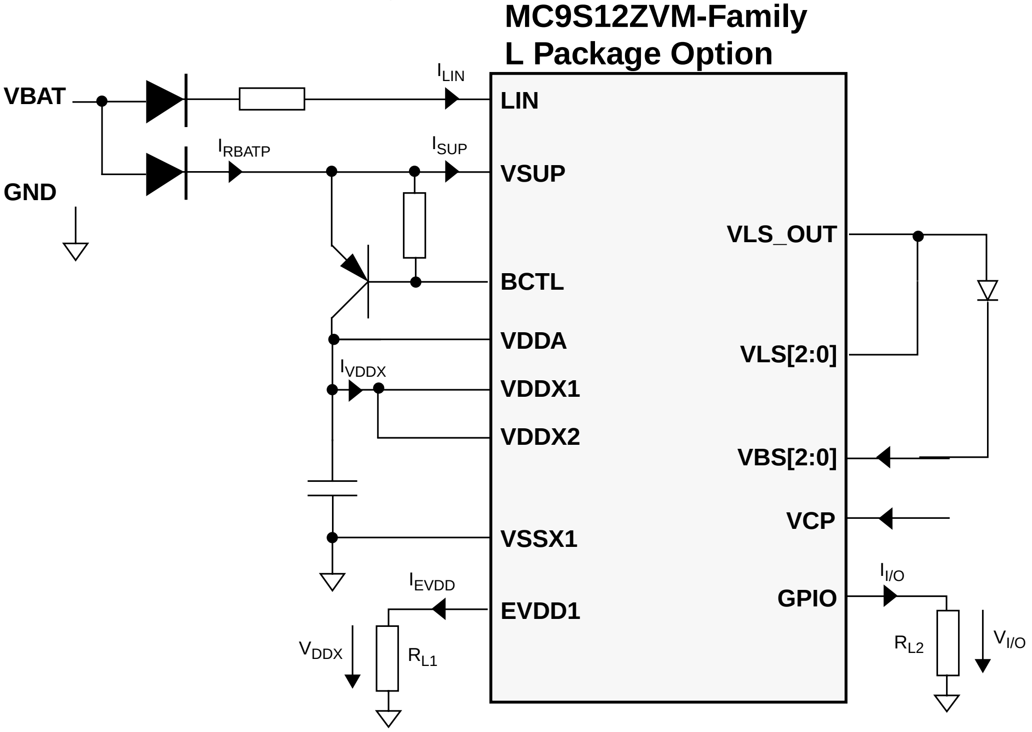


Table A-8. Thermal Package CharacteristicsI/O Characteristics

Num	C (1)	Rating	Symbol	Min	Typ	Max	Unit
64LQFP-EP							
8	D	Thermal resistance 64LQFP-EP, single sided PCB(2) Natural Convection	θ_{JA}	—	64	—	°C/W
9	D	Thermal resistance 64LQFP-EP, double sided PCB(3) with 2 internal planes. Natural Convection.	θ_{JA}	—	30	—	°C/W
10	D	Thermal resistance 64LQFP-EP, single sided PCB(3) (@200 ft./min)	θ_{JA}	—	51	—	°C/W
11	D	Thermal resistance 64LQFP-EP, double sided PCB(3) with 2 internal planes (@200 ft./min).	θ_{JA}	—	24	—	°C/W
12	D	Junction to Board 64LQFP-EP(4)	θ_{JB}	—	13	—	°C/W
13	D	Junction to Case Top 64LQFP-EP(5)	θ_{JCtop}	—	16	—	°C/W
14	D	Junction to Case Bottom 64LQFP-EP(6)	$\theta_{JCbottom}$	—	1.6	—	°C/W
15	D	Junction to Package Top 64LQFP-EP(7)	Ψ_{JT}	—	4	—	°C/W

1. The values for thermal resistance are achieved by package simulations
2. Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to JEDEC JESD51-2 with the single layer board (JESD51-3) horizontal.
3. Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to the JEDEC specification JESD51-6 with the board (JESD51-7) horizontal.

4. 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.
5. Thermal resistance between the die and the case top surface as measured by the cold plate method (MIL SPEC-883 Method 1012.1).
6. Thermal resistance between the die and the solder pad on the bottom of the package based on simulation without any interface resistance
7. Thermal characterization parameter indicating the temperature difference between package top and the junction temperature per JEDEC JESD51-2

A.1.9 I/O Characteristics

Table A-9. 5V I/O Characteristics

Conditions are 4.5 V < V_{DDX}< 5.5 V , -40°C < T_j < 175°C for all GPIO pins (defined in A.1.2.1/A-742) unless otherwise noted.

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	M	Input high voltage	V_{IH}	0.65*V_{DDX}	—	—	V
2	T	Input high voltage	V_{IH}	—	—	V_{DDX}+0.3	V
3	M	Input low voltage	V_{IL}	—	—	0.35*V_{DDX}	V
4	T	Input low voltage	V_{IL}	V_{SSX}–0.3	—	—	V
5	D	Input hysteresis	V_{HYS}	—	250	—	mV
6a	M	Input leakage current. All pins except PAD8 , PP0. (1) V_{in} = V_{DDX} or V_{SSX}	I_{in}	-1	—	1	µA
6b	M	Input leakage current. PAD8, PP0 (1) V_{in} = V_{DDX} or V_{SSX}	I_{in}	-2.5	—	2.5	µA
6c	C	Input leakage current. PAD8, PP0 (1) -40°C < T_j < 150°C, V_{in} = V_{DDX} or V_{SSX}	I_{in}	-1	—	1	µA
7	M	Output high voltage (All GPIO except EVDD1) I_{OH} = –4 mA	V_{OH}	V_{DDX} – 0.8	—	—	V
8a	M	Output high voltage (EVDD1), V_{DDX} > 4.85V Partial Drive I_{OH} = –2 mA Full Drive IOH = –20mA	V_{OH}	V_{DDX} – 0.8	—	—	V
8b	C	Output high voltage (EVDD1), V_{DDX} > 4.85V Full Drive IOH = –10mA	V_{OH}	V_{DDX} – 0.1	—	—	V
9	M	Output low voltage (All GPIO except EVDD1) I_{OL} = +4mA	V_{OL}	—	—	0.8	V
10	M	Output low voltage (EVDD1) Partial drive I_{OL} = +2mA or Full drive I_{OL} = +20mA	V_{OL}	—	—	0.8	V
11	D	Maximum allowed continuous current on EVDD1	I_{EVDD1}	-20	—	10	mA
12	M	Over-current Detect Threshold EVDD1	I_{OCD}	-80	—	-40	mA
13	M	Internal pull up current (All GPIO except RESET) V_{IH} min > input voltage > V_{IL} max	I_{PUL}	-10	—	-130	µA
14	M	Internal pull up resistance (RESET pin)	R_{PUL}	2.5	5	10	KΩ
15	M	Internal pull down current, V_{IH} min > V_{in} > V_{IL} max	I_{PDH}	10	—	130	µA
16	D	Input capacitance	C_{in}	—	7	—	pF
17a	T	Injection current(2) Single pin limit (all GPIO pins) Total device limit, sum of all injected currents	I_{ICS} I_{ICP}	–2.5 –25	—	2.5 25	mA
17b	T	Injection current single pin (HG,HS,LG,LS pins)(3)	I_{ICS}	–2.5	—	2.5	mA

1. Pins in high impedance input mode. 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. For better ADC accuracy, the application should avoid current injection into pin PAD8/VREFH. Refer to Section A.1.3, “Current Injection” for more details

3. For better ADC accuracy, the application should avoid current injection into pin HS0 and HG0 during ADC conversions. This can be achieved by correct synchronization of ADC and FET switching..

This following tables describe the timing characteristics of I/O pins.

Table A-10. Pin Timing 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-742).							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	M	Port P, S, AD interrupt input pulse filtered (STOP mode)	t_{P_MASK}	—	—	3	µs
2	M	Port P, S, AD interrupt input pulse passed (STOP mode)	t_{P_PASS}	10	—	—	µs
3	D	Port P, S, AD interrupt input pulse filtered ($\overline{\text{STOP}}$) in number of bus clock cycles of period $1/f_{bus}$	n_{P_MASK}	—	—	3	
4	D	Port P, S, AD interrupt input pulse passed ($\overline{\text{STOP}}$) in number of bus clock cycles of period $1/f_{bus}$	n_{P_PASS}	4	—	—	
5	D	$\overline{\text{IRQ}}$ pulse width, edge-sensitive mode ($\overline{\text{STOP}}$) in number of bus clock cycles of period $1/f_{bus}$	n_{IRQ}	1	—	—	
6	D	$\overline{\text{RESET}}$ pin input pulse filtered	R_{P_MASK}	—	—	12	ns
7	D	$\overline{\text{RESET}}$ pin input pulse passed	R_{P_PASS}	18	—	—	ns

A.1.10 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

A.1.10.1 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 50MHz. For the temperature range from +150˚C to +175˚C, the bus frequency is 40MHz. Table A-11, Table A-12 and Table A-13 show the configuration of the CPMU module and the peripherals for Run, Wait and Stop current measurement.

Table A-11. 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

Table A-11. CPMU Configuration for Pseudo Stop Current Measurement

CPMU REGISTER	Bit settings/Conditions
CPMUCOP	WCOP=1, CR[2:0]=111

Table A-12. CPMU Configuration for Run/Wait and Full Stop Current Measurement

CPMU REGISTER	Bit settings/Conditions
CPMUSYNR	VCOFRQ[1:0]= 1,SYNDIV[5:0] = 49
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-13. Peripheral Configurations for Run & 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
ADC	The peripheral is configured to operate at its maximum specified frequency and to continuously convert voltages on a single input channel
MSCAN	Configured in loop back mode with a bit rate of 500kbit/s.
DBG	The module is disabled, as in typical final applications
PTU	The module is enabled, bits TG1EN and TG0EN are set. PTUFRE is also set to generate automatic reload events.
PMF	The module is configured with a modulus rate of 10 kHz
TIM	The peripheral is configured to output compare mode,
GDU	LDO enabled. Charge pump enabled. Current sense0 enabled. Boost disabled. No output activity (too load dependent)
COP & RTI	Enabled
BATS	Enabled
LINPHY	connected to SCI and continuously transmit data (0x55) at speed of 19200 baud

Table A-14. Run and Wait Current Characteristics

Conditions see Table A-12 and Table A-13, V_{SUP}=18 V							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Run Current, -40°C < T_J < 150°C, f_{bus}= 50MHz	I_{SUPR}	—	53	66	mA
2	P	Wait Current, -40°C < T_J < 150°C, f_{bus}= 50MHz	I_{SUPW}	—	42	50	mA
3	M	Run Current, T_J =175°C, f_{bus}= 40MHz	I_{SUPR}	—	45	55	mA
4	M	Wait Current, T_J = 175°C, f_{bus}= 40MHz	I_{SUPW}	—	36	45	mA

Table A-15. Stop Current Characteristics

Conditions are: V_{SUP}=12 V							
Num	**C**	**Rating**[1]	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
Stop Current all modules off							
1	P	$T_A = T_J$= -40°C	I_{SUPS}	—	12	25	µA
2	P	$T_A = T_J$= 150°C	I_{SUPS}	—	200	1050	µA
3	C	$T_A = T_J$ = 25°C	I_{SUPS}	—	15	30	µA
4	C	$T_A = T_J$ = 85°C	I_{SUPS}	—	80	—	µA
Stop Current API enabled & LINPHY in standby							
5	T	$T_A = T_J$ = 25°C	I_{SUPS}	—	25	35	µA

1. If MCU is in STOP long enough then TA = TJ . Die self heating due to stop current can be ignored.

Table A-16. Pseudo Stop Current Characteristics

Conditions are: V_{SUP}=12V, API, COP & RTI enabled							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	T_J = 25°C	I_{SUPPS}	—	265	300	µA

A.1.11 ADC Calibration Configuration

The reference voltage V_{BG} is measured under the conditions shown in Table A-17. 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-17. 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
CPMU Electrical Specifications (VREG, OSC, IRC, PLL)

B.1 VREG Electrical Specifications.

Table B-1. Voltage Regulator Electrical Characteristics (Junction Temperature From –40°C To +175°C unless otherwise stated)

Note: VDDA and VDDX must be shorted on the application board.

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit
1	M	Input Voltages	V_{SUP}	3.5	—	40	V
2	M	Output Voltage Core					
		Full Performance Mode	V_{DD}	1.72	1.84	1.98	V
		Reduced Power Mode (stop mode)		—	1.6	—	V
3	M	Output Voltage Flash					
		Full Performance Mode	V_{DDF}	2.6	2.82	2.9	V
		Reduced Power Mode (stop mode)		—	1.6	—	V
4a	C	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) V_{SUP} > =3.5V		2.5	5.5	5.75	V
4b	M	Output Voltage VDDX (without external PNP)(1)					
		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) V_{SUP} > =3.5V		2.5	5.5	5.75	V
5a	P	Load Current VDDX(2)(3) without external PNP					
		Full Performance Mode, V_{SUP} > 6V, -40°C < T_J < 150°C	I_{DDX}	0	—	70	mA
5b	M	Load Current VDDX(2)(3) 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) V_{SUP} > =3.5V		0	—	5	mA
5c	P	Load Current VDDX(2)(3) without external PNP					
		Full Performance Mode 3.5V <= V_{SUP} <=6V -40°C < T_J < 150°C	I_{DDX}	0	—	25	mA
6	C	Output Voltage VDDC with external PNP(4)					
		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) V_{SUP} > =3.5V		2.5	5.5	5.75	V
7	M	Load Current VDDC					
		Reduced Performance Mode (stop mode)	I_{DDC}	0	—	2.5	mA

Table B-1. Voltage Regulator Electrical Characteristics (Junction Temperature From –40°C To +175°C unless otherwise stated)

Note: VDDA and VDDX must be shorted on the application board.

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit
8	M	Low Voltage Interrupt Assert Level(5) Low Voltage Interrupt Deassert Level	V_{LVIA} V_{LVID}	4.04 4.19	4.23 4.38	4.40 4.49	V V
9a	M	VDDX Low Voltage Reset deassert(6)	V_{LVRXD}	—	3.05	3.13	V
9b	M	VDDX Low Voltage Reset assert (7)	V_{LVRXA}	2.95	3.02	—	V
10	M	Trimmed ACLK output frequency	f_{ACLK}	—	20	—	KHz
11	M	Trimmed ACLK internal clock $\Delta f / f_{nominal}$ (8)	df_{ACLK}	- 6%	—	+ 6%	—
12	D	The first period after enabling the counter by APIFE might be reduced by API start up delay	t_{sdel}	—	—	100	µs
13	T	Temperature Sensor Slope	dV_{HT}	4.8	5.05	5.3	mV/°C
14	T	Temperature Sensor output voltage (150°C) untrimmed	V_{HT}	—	2.25	—	V
15	T	High Temperature Interrupt Assert(9) High Temperature Interrupt Deassert	T_{HTIA} T_{HTID}	120 110	132 122	144 134	°C °C
16	M	Bandgap output voltage	V_{BG}	1.14	1.20	1.28	V
17a	P	Max. Base Current For External PNP (VDDX)(10) $-40°C < T_J < 150°C$	$I_{BCTLMAX}$	2.3	—	—	mA
17b	C	Max. Base Current For External PNP (VDDX)(10) $150°C < T_J < 175°C$	$I_{BCTLMAX}$	1.5	—	—	mA
18a	P	Max. Base Current For External PNP (VDDC)**(10)** $-40°C < T_J < 150°C$	$I_{BCTLCMAX}$	2.3	—	—	mA
18b	C	Max. Base Current For External PNP (VDDC)**(10)** $150°C < T_J < 175°C$	$I_{BCTLCMAX}$	1.5	—	—	mA
19	D	Recovery time from STOP	t_{STP_REC}	—	23	—	µs

1. External PNP regulator has a higher regulation point to ensure that the current flows through the PNP when the application fails to disable the internal regulator byclearing INTXON.

2. Please note that the core current is derived from VDDX

3. Further limitation may apply due to maximum allowable T_J

4. Maximum load current depends on the current gain of the external PNP and available base current

5. LVI is monitored on the VDDA supply domain

6. 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.

7. For the given maximum load currents and V_{SUP} input voltages, the MCU will stay out of reset.

8. The ACLK trimming must be set that the minimum period equals to 0.2ms

9. VREGHTTR=0x88

10. This is the minimum base current that can be guaranteed when the external PNP is delivering maximum current.

B.2 IRC and OSC Electrical Specifications

Table B-2. IRC electrical characteristics

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1a	P	Junction Temperature - 40 to 150 Celsius Internal Reference Frequency, factory trimmed	f_{IRC1M_TRIM}	0.9895	1	1.0145	MHz
1b	M	Junction Temperature 150 to 175 Celsius(1) Internal Reference Frequency, factory trimmed	f_{IRC1M_TRIM}	0.9855	1	1.0145	MHz

1. Full characterization not complete

Table B-3. OSC electrical characteristics (Junction Temperature From –40°C To +175°C)

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	C	Nominal crystal or resonator frequency	f_{OSC}	4.0	—	20	MHz
2	M	Startup Current	i_{OSC}	100	—	—	µA
3a	C	Oscillator start-up time (4MHz)(1)	t_{UPOSC}	—	2	10	ms
3b	C	Oscillator start-up time (8MHz)[1]	t_{UPOSC}	—	1.6	8	ms
3c	C	Oscillator start-up time (16MHz)[1]	t_{UPOSC}	—	1	5	ms
3d	C	Oscillator start-up time (20MHz)[1]	t_{UPOSC}	—	1	4	ms
4	M	Clock Monitor Failure Assert Frequency	f_{CMFA}	200	450	1200	KHz
5	D	Input Capacitance (EXTAL, XTAL pins)	C_{IN}	—	7	—	pF
6	C	EXTAL Pin Input Hysteresis	$V_{HYS,EXTAL}$	—	120	—	mV
7	C	EXTAL Pin oscillation amplitude (loop controlled Pierce)	$V_{PP,EXTAL}$	—	1.0	—	V
8	D	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.

B.3 Phase Locked Loop

B.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 B-1.**.

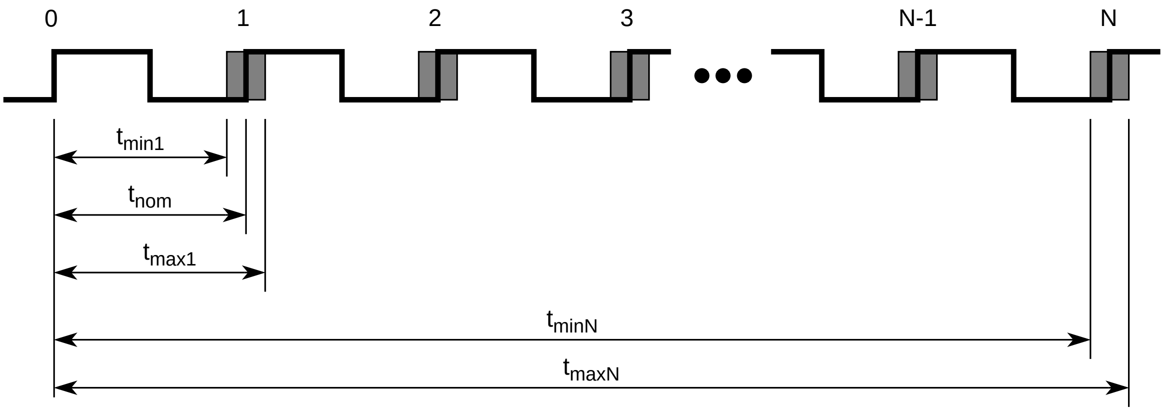


Figure B-1. Jitter Definitions

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:

$$J(N) = \frac{j_1}{\sqrt{N(POSTDIV + 1)}}$$

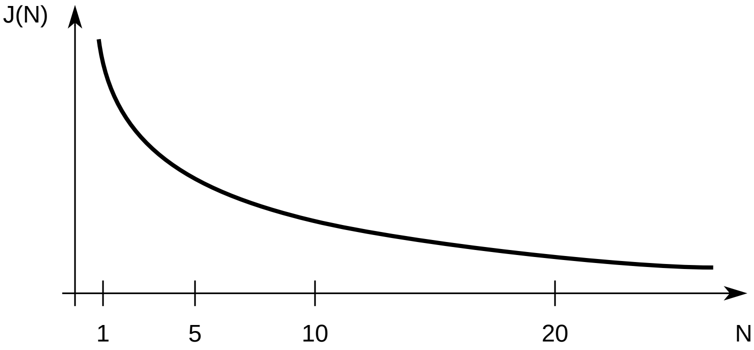


Figure B-2. Maximum Bus Clock Jitter Approximation (N = number of bus cycles)

NOTE

Peripheral module prescalers eliminate the effect of jitter to a large extent.

Table B-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	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	VCO frequency during system reset	f_{VCORST}	8	—	32	MHz
2	C	VCO locking range	f_{VCO}	32	—	64	MHz
3	C	Reference Clock	f_{REF}	1	—	—	MHz
4	D	Lock Detection	$\|\Delta_{Lock}\|$	0	—	1.5	%(1)
5	D	Un-Lock Detection	$\|\Delta_{unl}\|$	0.5	—	2.5	%[1]
7	C	Time to lock	t_{lock}	—	—	150 + 256/f_{REF}	µs
8a	C	Jitter fit parameter 1(2) 40°C < T_J < 150°C	j_1	—	—	2	%
8b	T	Jitter fit parameter 1 150°C < T_J < 175°C	j_1	—	—	2	%
9	M	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} = 50MHz

Appendix C
ADC Electrical Specifications

This section describes the characteristics of the analog-to-digital converter.

C.1 ADC Operating Characteristics

The Table C-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 C-1. ADC Operating Characteristics

Supply voltage 4.5 V < V_{DDA} < 5.5 V, Junction Temperature From –40^{o}C To +175^{o}C							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Reference potential Low High	V_{RL} V_{RH}	V_{SSA} $V_{DDA}/2$	— —	$V_{DDA}/2$ V_{DDA}	V V
2	D	Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	-0.1	0	0.1	V
3	D	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.1	0	0.1	V
4	C	Differential reference voltage(1)	V_{RH}-V_{RL}	3.13	5.0	5.5	V
5	C	ADC Clock Frequency (derived from bus clock via the prescaler).	f_{ATDCLK}	0.25	—	8.33	MHz
6	D	Buffer amplifier turn on time (delay after module start/recovery from Stop mode)	t_{REC}	—	—	1	µs
7	D	ADC disable time	$t_{DISABLE}$	—	—	3	bus clock cycles
8	D	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.

C.1.1 Factors Influencing Accuracy

Source resistance, source capacitance and current injection have an influence on the accuracy of the ADC .**Figure C-1.** A further factor is that PortAD pins that are configured as output drivers switching.

C.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.

C.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.

C.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 and the pin capacitance. For a maximum sampling error of the input voltage $\leq$ 1LSB (10-bit resolution), then the external filter capacitor, $C_f \geq 1024 * (C_{INS} – C_{INN})$.

C.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.

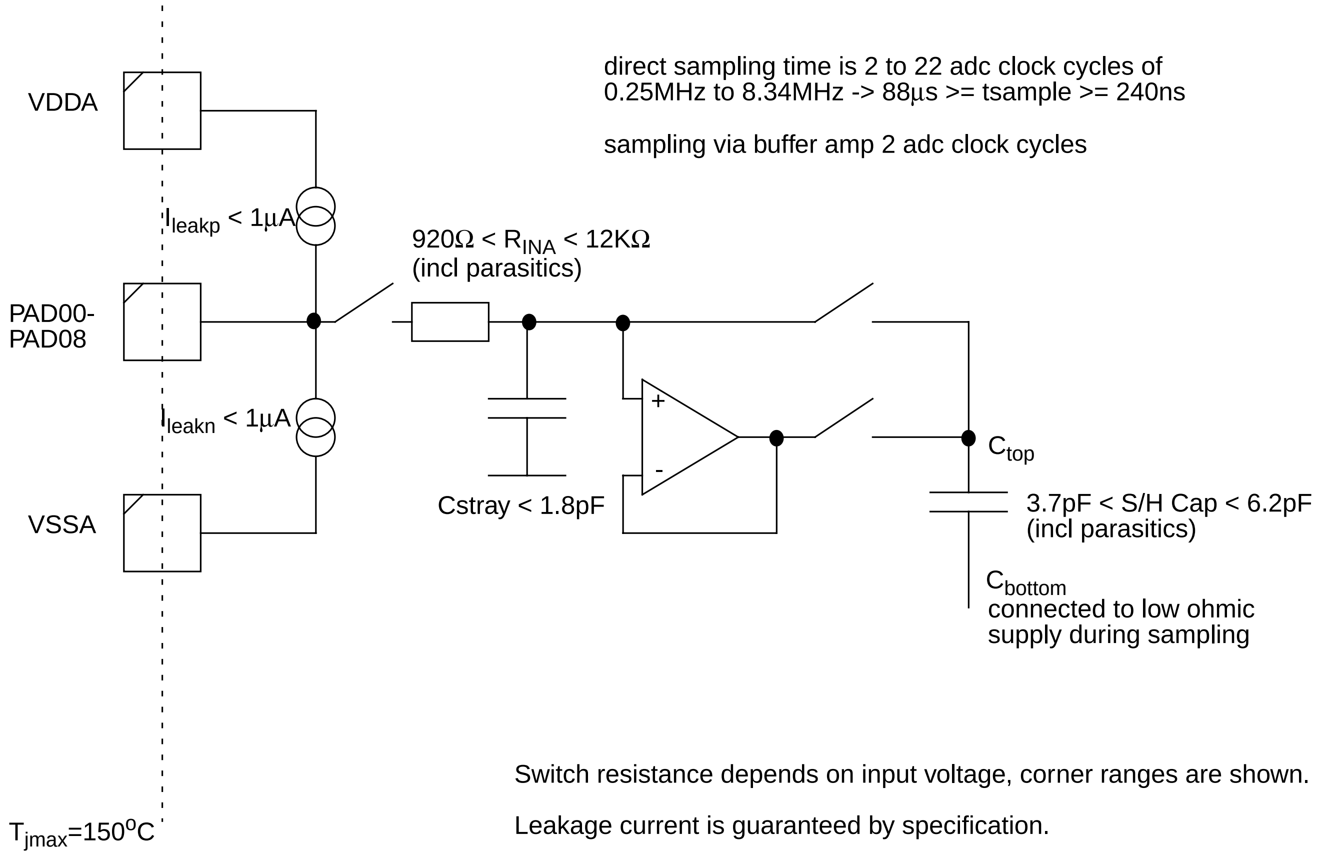
   The additional input voltage error on the converted channel can be calculated as:

   $$V_{ERR} = K * R_S * I_{INJ}$$

   with I_{INJ} being the sum of the currents injected into the two pins adjacent to the converted channel.

Table C-2. ADC Electrical Characteristics (Junction Temperature From –40°C To +175°C)

Supply voltage 3.13 V < V_{DDA} < 5.5 V							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Max input source resistance	R_S	—	—	1	KΩ
2	D	Total input capacitance Non sampling Total input capacitance Sampling	C_{INN} C_{INS}	— —	— —	10 16	pF
3a	D	Input internal Resistance Junction temperature from –40°C to +150°C	R_{INA}	—	5	9.9	kΩ
3b	D	Input internal Resistance Junction temperature from 150°C to +175°C	R_{INA}	—	—	12	kΩ
4	T	Disruptive analog input current	I_{NA}	-2.5	—	2.5	mA
5	T	Coupling ratio positive current injection	K_p	—	—	1E-4	A/A
6	T	Coupling ratio negative current injection	K_n	—	—	5E-3	A/A



direct sampling time is 2 to 22 adc clock cycles of
0.25MHz to 8.34MHz -> 88µs >= tsample >= 240ns

sampling via buffer amp 2 adc clock cycles

Switch resistance depends on input voltage, corner ranges are shown.

Leakage current is guaranteed by specification.

Figure C-1.

C.1.2 ADC Accuracy

Table C-3. specifies the ADC conversion performance excluding any errors due to current injection, input capacitance and source resistance.

C.1.2.1 ADC Accuracy Definitions

For the following definitions see also **Figure C-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:

$$\mathrm{INL}(n) = \sum_{i=1}^{n} \mathrm{DNL}(i) = \frac{V_n - V_0}{1\mathrm{LSB}} - n$$

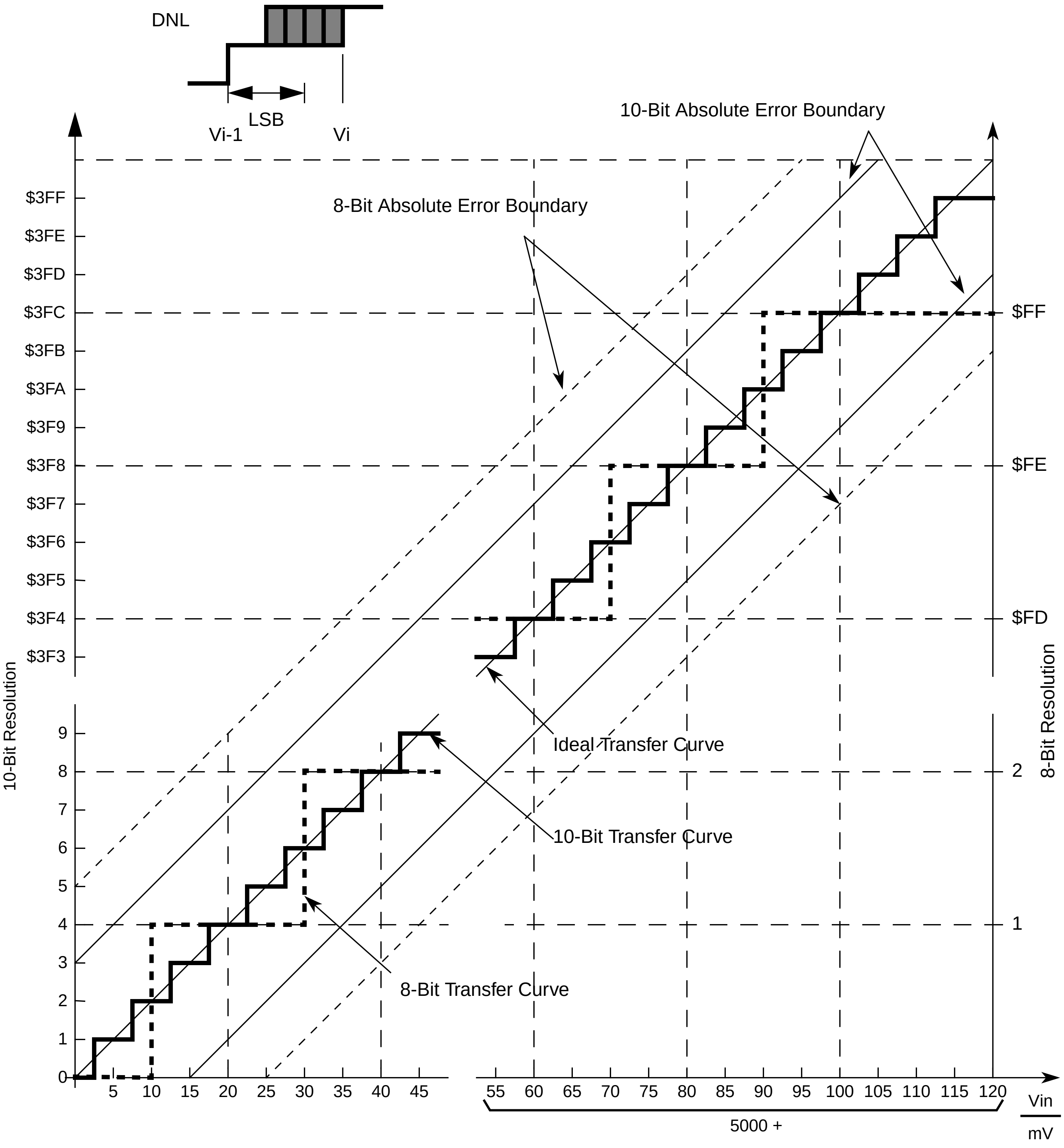


Figure C-2. ADC Accuracy Definitions

Table C-3. ADC Conversion Performance 5 V range (Junction Temperature From –40°C To +150°C)

Supply voltage V_{DDA} = 5.12 V, V_{REF} = V_{RH} - V_{RL} = 5.12 V. f_{ADCCLK} = 8.0 MHz The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.								
Num	**C**	**Rating**[1]		**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Resolution	12-Bit	LSB	—	1.25	—	mV
2	P	Differential Nonlinearity	12-Bit	DNL	-4	±2	4	counts
3	P	Integral Nonlinearity	12-Bit	INL	-5	±2.5	5	counts
4	P	Absolute Error[2]	12-Bit	AE	-7	±4	7	counts
5	C	Resolution	10-Bit	LSB	—	5	—	mV
6	C	Differential Nonlinearity	10-Bit	DNL	-1	±0.5	1	counts
7	C	Integral Nonlinearity	10-Bit	INL	-2	±1	2	counts
8	C	Absolute Error	10-Bit	AE	-3	±2	3	counts
9	C	Resolution	8-Bit	LSB	—	20	—	mV
10	C	Differential Nonlinearity	8-Bit	DNL	-0.5	±0.3	0.5	counts
11	C	Integral Nonlinearity	8-Bit	INL	-1	±0.5	1	counts
12	C	Absolute Error	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.

Table C-4. ADC Conversion Performance 5 V range (Junction Temperature From 150°C To +175°C)

Supply voltage V_{DDA} = 5.12 V, V_{REF} = V_{RH} - V_{RL} = 5.12 V. f_{ADCCLK} = 8.0 MHz The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.								
Num	**C**	**Rating**[1]		**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Resolution	12-Bit	LSB	—	1.25	—	mV
2	C	Differential Nonlinearity	12-Bit	DNL	-4	±2	4	counts
3	C	Integral Nonlinearity	12-Bit	INL	-5	±2.5	5	counts
4	C	Absolute Error[2]	12-Bit	AE	-7	±4	7	counts
5	C	Resolution	10-Bit	LSB	—	5	—	mV
6	C	Differential Nonlinearity	10-Bit	DNL	-1	±0.5	1	counts
7	C	Integral Nonlinearity	10-Bit	INL	-2	±1	2	counts
8	C	Absolute Error	10-Bit	AE	-3	±2	3	counts
9	C	Resolution	8-Bit	LSB	—	20	—	mV
10	C	Differential Nonlinearity	8-Bit	DNL	-0.5	±0.3	0.5	counts
11	C	Integral Nonlinearity	8-Bit	INL	-1	±0.5	1	counts
12	C	Absolute Error	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 D
LINPHY Electrical Specifications

D.1 Static Electrical Characteristics

Table D-1. Static electrical characteristics of the LINPHY (Junction Temperature From -40°C To +175°C)

Characteristics noted under conditions $7V \le V_{LINSUP} \le 18$ V unless otherwise noted(1).(2) Typical values noted reflect the approximate parameter mean at T_A = 25˚C under nominal conditions unless otherwise noted.

Num	C	Ratings	Symbol	Min	Typ	Max	Unit
1	C	V_{LINSUP} range for LIN compliant electrical characteristics	V_{LINSUP}	7[1]	12	18	V
2	M	Current limitation into the LIN pin in dominant state(3) $V_{LIN} = V_{LINSUP\_LIN\_MAX}$	I_{LIN_LIM}	40	—	200	mA
3	M	Input leakage current in dominant state (V_{LIN} = 0V, V_{LINSUP} = 12V)	$I_{LIN\_PAS\_dom}$	-1	—	—	mA
4	M	Input leakage current in recessive state ($8V<V_{LINSUP}<18V$, $8V<V_{LIN}<18V$, $V_{LIN} >= V_{LINSUP}$)	$I_{LIN\_PAS\_rec}$	—	—	20	µA
5	T	Input leakage current when ground disconnected ($GND_{Device} = V_{LINSUP}$, $0V<V_{LIN}<18V$, V_{LINSUP} = 12V)	$I_{LIN\_NO\_GND}$	-1	—	1	mA
6	T	Input leakage current when battery disconnected (V_{LINSUP} = GND, $0<V_{LIN}<18V$)	$I_{LIN\_NO\_BAT}$	—	—	30	µA
7	M	Receiver dominant state	V_{LINdom}	—	—	0.4	V_{LINSUP}
8	M	Receiver recessive state	V_{LINrec}	0.6	—	—	V_{LINSUP}
9	M	$V_{LIN\_CNT} =(V_{th\_dom}+ V_{th\_rec})/2$	V_{LIN_CNT}	0.475	0.5	0.525	V_{LINSUP}
10	M	$V_{HYS} = V_{th\_rec} - V_{th\_dom}$	V_{HYS}	—	—	0.175	V_{LINSUP}
11	—	Maximum capacitance allowed on slave node	C_{slave}	—	220	250	pF
12a	T	Capacitance of LIN pin -40°C < T_J < 150°C, Recessive state	C_{int}	—	20	—	pF
12b	D	Capacitance of LIN pin -40°C < T_J < 150°C, Recessive state	C_{int}	—	—	45	pF
12c	D	Capacitance of LIN pin 150°C < T_J < 175°C, Recessive state	C_{int}	—	—	39	pF
13	M	Internal pull-up (slave)	R_{slave}	27	34	40	kΩ

1. For $3.5V<=V_{LINSUP}<7V$, the LINPHY is still working but with degraded parametrics.
2. The V_{LINSUP} voltage is provided by the VLINSUP supply. This supply mapping is described in device level documentation
3. At temperatures above 25°C the current may be naturally limited by the driver, in this case the limitation circuit is not engaged and the flag is not set.

D.2 Dynamic Electrical Characteristics

Table D-2. Dynamic electrical characteristics of the LINPHY (Junction Temperature From -40°C To +175°C)

Characteristics noted under conditions $7V \le V_{LINSUP} \le 18$ V unless otherwise noted(1). Typical values noted reflect the approximate parameter mean at $T_A = 25^{\circ}C$ under nominal conditions unless otherwise noted.

Num	C	Ratings	Symbol	Min	Typ	Max	Unit
1	M	Minimum duration of wake-up pulse generating a wake-up interrupt $-40^{\circ}C < T_J < 175^{\circ}C$	t_{WUFR}	56	72	120	µs
2	M	TxD-dominant timeout (in IRC clock periods)	t_{DTLIM}	16388	—	16389	t_{IRC}
3	M	Propagation delay of receiver	t_{rx_pd}	—	—	6	µs
4	M	Symmetry of receiver propagation delay rising edge w.r.t. falling edge	t_{rx_sym}	-2	—	2	µs
LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR NOMINAL SLEW RATE - 20.0KBIT/S							
5	T	Rising/falling edge time (min to max / max to min)	t_{rise}	—	6.5	—	µs
6	T	Over-current masking window (IRC trimmed at 1MHz)	t_{OCLIM}	15	—	16	µs
7	M	Duty cycle 1 $T_{HRec(max)} = 0.744 \times V_{LINSUP}$ $T_{HDom(max)} = 0.581 \times V_{LINSUP}$ $V_{LINSUP} = 7.0V...18V$ $t_{Bit} = 50us$ $D1 = t_{Bus\_rec(min)} / (2 \times t_{Bit})$	D1	0.396	—	—	
8	M	Duty cycle 2 $T_{HRec(min)} = 0.422 \times V_{LINSUP}$ $T_{HDom(min)} = 0.284 \times V_{LINSUP}$ $V_{LINSUP} = 7.6V...18V$ $t_{Bit} = 50us$ $D2 = t_{Bus\_rec(max)} / (2 \times t_{Bit})$	D2	—	—	0.581	
LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR SLOW SLEW RATE - 10.4KBIT/S							
9	T	Rising/falling edge time (min to max / max to min)	t_{rise}	—	13	—	µs
10	T	Over-current masking window (IRC trimmed at 1MHz)	t_{OCLIM}	31	—	32	µs
11	M	Duty cycle 3 $T_{HRec(max)} = 0.778 \times V_{LINSUP}$ $T_{HDom(max)} = 0.616 \times V_{LINSUP}$ $V_{LINSUP} = 7.0V...18V$ $t_{Bit} = 96us$ $D3 = t_{Bus\_rec(min)} / (2 \times t_{Bit})$	D3	0.417	—	—	
12	M	Duty cycle 4 $T_{HRec(min)} = 0.389 \times V_{LINSUP}$ $T_{HDom(min)} = 0.251 \times V_{LINSUP}$ $V_{LINSUP} = 7.6V...18V$ $t_{Bit} = 96us$ $D4 = t_{Bus\_rec(max)} / (2 \times t_{Bit})$	D4		—	0.590	
LIN PHYSICAL LAYER: DRIVER CHARACTERISTICS FOR FAST MODE SLEW RATE - 100KBIT/S UP TO 250KBIT/S							

Characteristics noted under conditions 7V ≤ V $_{LINSUP}$ ≤ 18 V unless otherwise noted(1). Typical values noted reflect the approximate parameter mean at T_A = 25˚C under nominal conditions unless otherwise noted.

Num	C	Ratings	Symbol	Min	Typ	Max	Unit
13	T	Rising/falling edge time (min to max / max to min)	t_{rise}	—	0.5	—	µs
14	T	Over-current masking window (IRC trimmed at 1MHz)	t_{OCLIM}	5	—	6	µs

1. For 3.5V<=V_{LINSUP}<7V, the LINPHY is still working but with degraded parametrics.

Appendix E
GDU Electrical Specifications

Table E-1. GDU Electrical Characteristics (Junction Temperature From –40°C To +175°C)

4.85V<=VDDX,VDDA<=5.15V							
Num	**C**	**Characteristic**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	VSUP Supply range	V_{VSUP}	-0.3	—	40	V
2a	M	VSUP, HD Supply range FETs can be turned on(1) (normal range)	V_{VSUP}/V_{HD}	7	14	20	V
2b	M	VSUP, HD Supply range FETs can be turned on(2) (extended range)	V_{VSUP}/V_{HD}	7	14	26	V
3	D	External FET Vgs drive with boost(3) (7V < Vsup < 20V)	V_{VGS}	9	9.6	12	V
4	D	External FET Vgs drive without boost(4)	V_{VGS}	5	9.6	12	V
5	D	External FET total gate charge @ 10V(5)	CG	—	75	—	nC
6	C	Highside desaturation threshold (8 steps of 0.15V)	$V_{desaths}$	HD - 1.4	—	HD - 0.35	V
7	C	Lowside desaturation threshold (8 steps of 0.15V)	$V_{desatls}$	0.35	—	1.4	V
8	M	Pull resistance between HGx and HSx	R_{HSpul}	60	80	120	KΩ
9	M	Pull resistance between LGx and LSx	R_{LSpul}	60	80	120	KΩ
10a	P	VLS output voltage for Vsup >=12.5V, Iout=30mA -40°C < T_j < 150°C	V_{VLS_OUT}	10.5	11	11.5	V
10b	M	VLS output voltage for Vsup >=12.5V, Iout=30mA 150°C < T_j < 175°C	V_{VLS_OUT}	10.0	10.6	11.5	V
11	M	VLS current limit threshold	I_{LIMVLS}	60	77	100	mA
12a	M	VLS low voltage monitor trippoint assert	V_{LVLSA}	6.2	6.5	7	V
12b	M	VLS low voltage monitor trippoint deassert	V_{LVLSD}	6.2	6.58	7	V
13a	M	HD high voltage monitor assert trippoint low	V_{HVHDLA}	20	21	22	V
13b	M	HD high voltage monitor deassert trippoint low	V_{HVHDLD}	19.5	20.5	21.6	V
14a	M	HD high voltage monitor assert trippoint high	V_{HVHDHA}	26.6	28.3	29.4	V
14b	M	HD high voltage monitor deassert trippoint high	V_{HVHDHD}	26.2	27.9	29	V
15	C	HD high voltage monitor filter time constant(6)	τ_{HVHD}	—	2.7	4	µs
16	D	HG/LG turn on time vs 10nF load (fastest slew)(7)	t_{HGON}	120	190	340	ns
17	D	HG/LG turn on time vs 10nF load (slowest slew)(7)	t_{HGON}	350	560	980	ns
18a	D	HG/LG turn off time vs 10nF load(8), -40°C < T_j < 150°C	t_{HGOFF}	55	90	180	ns
18b	D	HG/LG turn off time vs 10nF load(8) , 150°C < T_j < 175°C	t_{HGOFF}	55	90	190	ns
19	D	PMF control to HG/LG start of turn on delay	t_{delon}	0.52	0.75	0.98	µs
20	D	PMF control to HG/LG start of turn off delay	t_{deloff}	0.31	0.45	0.59	µs
21	D	Minimum PMF driver on/off pulse width (fastest slew)	$t_{minpulse}$	2	—	—	µs
22a	P	VBS to HG, VLSx to LGx RDSon (driver on state)(9) -40°C < T_j < 150°C	R_{gduon}	—	6.3	11.6	Ω
22b	M	VBS to HG, VLSx to LGx RDSon (driver on state)(9) 150°C < T_j < 175°C	R_{gduon}	—	8.4	13.6	Ω
23a	P	HGx to HSx, LGx to LSx RDSon (driver off state)(10) nmos part, -40°C < T_j < 150°C	$R_{gduoffn}$	—	4	9	Ω
23b	M	HGx to HSx, LGx to LSx RDSon (driver off state)(10) nmos part, 150°C < T_j < 175°C	$R_{gduoffn}$	—	7	11	Ω

Table E-1. GDU Electrical Characteristics (Junction Temperature From –40°C To +175°C)

4.85V<=VDDX,VDDA<=5.15V							
24a	P	HGx to HSx, LGx to LSx RDSon (driver off state)(11) pmos part, -40°C < T_j < 150°C	$R_{gduoffp}$	—	16	22	Ω
24b	M	HGx to HSx, LGx to LSx RDSon (driver off state)(11) pmos part, 150°C < T_j < 175°C	$R_{gduoffp}$	—	20	26	Ω
25	M	VSUP boost turn on trip point	V_{BSTON}	9.5	10.1	10.6	V
26	M	VSUP boost turn off trip point	V_{BSTOFF}	9.75	10.3	10.8	V
27a	P	Boost coil current limit (GDUBCL=0x0), -40°C < T_j < 150°C	I_{COIL0}	80	190	350	mA
27b	M	Boost coil current limit (GDUBCL=0x0), 150°C < T_j < 175°C	I_{COIL0}	70	160	250	mA
28a	P	Boost coil current limit (GDUBCL=0x8), -40°C < T_j < 150°C	I_{COIL8}	230	380	640	mA
28b	M	Boost coil current limit (GDUBCL=0x8), 150°C < T_j < 175°C	I_{COIL8}	210	330	450	mA
29a	P	Boost coil current limit (GDUBCL=0xF), -40°C < T_j < 150°C	I_{COIL15}	370	530	870	mA
29b	M	Boost coil current limit (GDUBCL=0xF), 150°C < T_j < 175°C	I_{COIL15}	350	485	620	mA
30	M	Phase signal division ratio 3V < V_{HSx} < 20V	A_{HSDIV}	5.7	6	6.3	—
31	M	HD signal division ratio 6V < V_{HD} < 20V	A_{HDDIV}	4.9	5	5.1	—
32a	P	CP driver RDSon highside(12), -40°C < T_j < 150°C	R_{CPHS}	—	44	90	Ω
32b	M	CP driver RDSon highside(12), 150°C < T_j < 175°C	R_{CPHS}	—	71	100	Ω
33a	P	CP driver RDSon lowside(12), -40°C < T_j < 150°C	R_{CPLS}	—	11.5	30	Ω
33b	M	CP driver RDSon lowside(12), 150°C < T_j < 175°C	R_{CPLS}	—	20	35	Ω
34	M	Current Sense Amplifier input voltage range (AMPP/AMPM)	V_{CSAin}	0	—	VDDA - 1.2	V
35	M	Current Sense Amplifier output voltage range	V_{CSAout}	0	—	VDDA	V
36	D	Current Sense Amplifier open loop gain	AV_{CSA}	—	100000	—	—
37	C	Current Sense Amplifier common mode rejection ratio	$CMRR_{CSA}$	—	400	—	—
38	M	Current Sense Amplifier input offset	V_{CSAoff}	-15	—	15	mV
39	C	Max effective Current Sense Amplifier output resistance [0.1V .. VDDA - 0.2V]	R_{CSAout}	—	—	2	Ω
40	C	Min Current Sense Amplifier output current [0.1V .. VDDA - 0.2V](13)	I_{CSAout}	-750	—	750	μA
41	D	Current Sense Amplifier large signal settling time	t_{cslsst}	—	2.9	—	μs
42	C	Over Current Comparator filter time constant(14)	τ_{OCC}	3	5	10	μs
43	C	Over Current Comparator threshold tolerance	V_{OCCtt}	-75	—	75	mV
44	D	HD input current when GDU is enabled	I_{HD}	—	130μ + V_{HD}/63K	—	A
45	D	VLS regulator minimum RDSon (VSUP >= 6V)	R_{VLSmin}	—	—	40	Ω
46	C	VCP to VBSx switch resistance	R_{VCPVBS}	—	600	1000	Ω

1. Without using the boost option. The minimum level can be relaxed when the boost option is used. The lower limit is sensed on VLS, the upper limit is sensed on HD.

2. Without using the boost option. The minimum level can be relaxed when the boost option is used. The lower limit is sensed on VLS, the upper limit is sensed on HD. Operation beyond 20V is limited to 1 hour over lifetime of the device

3. For high side, the performance of external diodes may influence this parameter.

4. If VSUP is lower than 11.2V, external FET gate drive will diminish and roughly follow VSUP - 2* Vbe

5. Total gate charge spec is only a recommendation. FETs with higher gate charge can be used when resulting slew rates are tolerable by the application and resulting power dissipation does not lead to thermal overload.

6. Blanking time for assert (maskset 1N95G)

7. (VBSx - HSx) = 10V respectively VLSx=10V, measured from 1V to 9V HGx/LGx vs HSx/LSx

8. (VBSx - HSx) = 10V respectively VLSx=10V, measured from 9V to 1V HGx/LGx vs HSx/LSx

9. V(VBSx) - V(VLSx) > 9V, resp VLSx > 9V

10. V(VBSx) - V(VLSx) > 9V, resp VLSx > 9V, nmos branch only

11. V(VBSx) - V(VLSx) > 9V, resp VLSx > 9V, pmos branch only

12. VLS > 6V

13. Output current range for which the effective output resistance specification applies

14. Av=10, no frequency compensation in feedback network, 90% output swing

Appendix F NVM Electrical Parameters

F.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 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 F-1.

Table F-1. NVM Timing Characteristics (Junction Temperature From –40°C To +150°C)

Num	C	Command	f_{NVMOP} cycle	f_{NVMBUS} cycle	Symbol	Min(1)	Typ(2)	Max(3)	Worst (4)	Unit
1		Bus frequency	1	—	f_{NVMBUS}	1	50	50		MHz
2		NVM Operating frequency	—	1	f_{NVMOP}	0.8	1.0	1.05		MHz
3	D	Erase Verify All Blocks	0	33760	t_{RD1ALL}	0.68	0.68	1.35	67.52	ms
4	D	Erase Verify Block (Pflash)(5)	0	33320	t_{RD1BLK_P}	0.67	0.67	1.33	66.64	ms
5	D	Erase Verify Block (EEPROM)(6)	0	823	t_{RD1BLK_D}	0.02	0.02	0.03	1.65	ms
6	D	Erase Verify P-Flash Section	0	505	t_{RD1SEC}	0.01	0.01	0.04	2.02	ms
7	D	Read Once	0	481	t_{RDONCE}	9.62	9.62	9.62	481.00	us
8	D	Program P-Flash (4 Word)	164	3077	t_{PGM_4}	0.22	0.23	0.41	12.51	ms
9	D	Program Once	164	3054	$t_{PGMONCE}$	0.22	0.23	0.23	3.26	ms
10	D	Erase All Blocks[5, 6]	100066	34223	t_{ERSALL}	95.99	100.75	101.43	193.53	ms
11	D	Erase Flash Block (Pflash)[5]	100060	33681	t_{ERSBLK_P}	95.97	100.73	101.41	192.44	ms
12	D	Erase Flash Block (EEPROM)[6]	100060	1154	t_{ERSBLK_D}	95.32	100.08	100.11	127.38	ms
13	D	Erase P-Flash Sector	20015	914	t_{ERSPG}	19.08	20.03	20.05	26.85	ms
14	D	Unsecure Flash	100066	34288	t_{UNSECU}	95.99	100.75	101.44	193.66	ms
15	D	Verify Backdoor Access Key	0	493	t_{VFYKEY}	9.86	9.86	9.86	493.00	us
16	D	Set User Margin Level	0	427	t_{MLOADU}	8.54	8.54	8.54	427.00	us
17	D	Set Factory Margin Level	0	436	t_{MLOADF}	8.72	8.72	8.72	436.00	us
18	D	Erase Verify EEPROM Sector	0	583	$t_{DRD1SEC}$	0.01	0.01	0.05	2.33	ms
19	D	Program EEPROM (1 Word)	68	1657	t_{DPGM_1}	0.10	0.10	0.20	6.71	ms
20	D	Program EEPROM (2 Word)	136	2660	t_{DPGM_2}	0.18	0.19	0.35	10.81	ms
21	D	Program EEPROM (3 Word)	204	3663	t_{DPGM_3}	0.27	0.28	0.50	14.91	ms
22	D	Program EEPROM (4 Word)	272	4666	t_{DPGM_4}	0.35	0.37	0.65	19.00	ms
23	D	Erase EEPROM Sector	5015	810	t_{DERSPG}	4.79	5.03	20.34	38.85	ms
24	D	Protection Override	0	475	$t_{PRTOVRD}$	9.50	9.50	9.50	475.00	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. Worst times are based on minimum f_{NVMOP} and minimum f_{NVMBUS} plus aging

5. Affected by Pflash size

6. Affected by EEPROM size

Table F-2. NVM Timing Characteristics (Junction Temperature From 150°C To 175°C)

Num	C	Command	f_{NVMOP} cycle	f_{NVMBUS} cycle	Symbol	Min(1)	Typ(2)	Max(3)	Worst (4)	Unit
1		Bus frequency	1	—	f_{NVMBUS}	1	40	40		MHz
2		NVM Operating frequency	—	1	f_{NVMOP}	0.8	1.0	1.05		MHz
3	D	Erase Verify All Blocks	0	33760	t_{RD1ALL}	0.84	0.84	1.69	67.52	ms
4	D	Erase Verify Block (Pflash)(5)	0	33320	t_{RD1BLK_P}	0.83	0.83	1.67	66.64	ms
5	D	Erase Verify Block (EEPROM)(6)	0	823	t_{RD1BLK_D}	0.02	0.02	0.04	1.65	ms
6	D	Erase Verify P-Flash Section	0	505	t_{RD1SEC}	0.01	0.01	0.03	1.01	ms
7	D	Read Once	0	481	t_{RDONCE}	12.03	12.03	12.03	481.00	us
8	D	Program P-Flash (4 Word)	164	3077	t_{PGM_4}	0.23	0.24	0.47	12.51	ms
9	D	Program Once	164	3054	$t_{PGMONCE}$	0.23	0.24	0.24	3.26	ms
10	D	Erase All Blocks[5, 6]	100066	34223	t_{ERSALL}	96.16	100.92	101.78	193.53	ms
11	D	Erase Flash Block (Pflash)[5]	100060	33681	t_{ERSBLK_P}	96.14	100.90	101.74	192.44	ms
12	D	Erase Flash Block (EEPROM)[6]	100060	1154	t_{ERSBLK_D}	95.32	100.09	100.12	127.38	ms
13	D	Erase P-Flash Sector	20015	914	t_{ERSPG}	19.08	20.04	20.06	26.85	ms
14	D	Unsecure Flash	100066	34288	t_{UNSECU}	96.16	100.92	101.78	193.66	ms
15	D	Verify Backdoor Access Key	0	493	t_{VFYKEY}	12.33	12.33	12.33	493.00	us
16	D	Set User Margin Level	0	427	t_{MLOADU}	10.68	10.68	10.68	427.00	us
17	D	Set Factory Margin Level	0	436	t_{MLOADF}	10.90	10.90	10.90	436.00	us
18	D	Erase Verify EEPROM Sector	0	583	$t_{DRD1SEC}$	0.01	0.01	0.03	1.17	ms
19	D	Program EEPROM (1 Word)	68	1657	t_{DPGM_1}	0.11	0.11	0.23	6.71	ms
20	D	Program EEPROM (2 Word)	136	2660	t_{DPGM_2}	0.20	0.20	0.40	10.81	ms
21	D	Program EEPROM (3 Word)	204	3663	t_{DPGM_3}	0.29	0.30	0.57	14.91	ms
22	D	Program EEPROM (4 Word)	272	4666	t_{DPGM_4}	0.38	0.39	0.74	19.00	ms
23	D	Erase EEPROM Sector	5015	810	t_{DERSPG}	4.80	5.04	20.40	38.85	ms
24	D	Protection Override	0	475	$t_{PRTOVRD}$	11.88	11.88	11.88	475.00	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. Worst times are based on minimum f_{NVMOP} and minimum f_{NVMBUS} plus aging

5. Affected by Pflash size

6. Affected by EEPROM size

F.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.

NOTE

All values shown in Table F-3 are preliminary and subject to further characterization.

Table F-3. NVM Reliability Characteristics

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 ≤ 175°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 ≤ 175°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 Freescale 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 Freescale defines Typical Endurance, please refer to Engineering Bulletin EB619.

F.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 Freescale defines Typical Data Retention, please refer to Engineering Bulletin EB618.

Appendix G BATS Electrical Specifications

G.1 Static Electrical Characteristics

Table G-1. Static Electrical Characteristics - BATS (Junction Temperature From -40°C To +175°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	C	Ratings	Symbol	Min	Typ	Max	Unit
1		Low Voltage Warning (LBI 1)					
	M	Assert (Measured on VSUP pin, falling edge)	V_{LBI1_A}	5	5.5	6	V
	M	Deassert (Measured on VSUP pin, rising edge)	V_{LBI1_D}	–	–	6.5	V
	C	Hysteresis (measured on VSUP pin)	V_{LBI1_H}	–	0.4	–	V
2		Low Voltage Warning (LBI 2)					
	M	Assert (Measured on VSUP pin, falling edge)	V_{LBI2_A}	6	6.75	7.25	V
	M	Deassert (Measured on VSUP pin, rising edge)	V_{LBI2_D}	–	–	7.75	V
	C	Hysteresis (measured on VSUP pin)	V_{LBI2_H}	–	0.4	–	V
3		Low Voltage Warning (LBI 3)					
	M	Assert (Measured on VSUP pin, falling edge)	V_{LBI3_A}	7	7.75	8.5	V
	M	Deassert (Measured on VSUP pin, rising edge)	V_{LBI3_D}	–	–	9	V
	C	Hysteresis (measured on VSUP pin)	V_{LBI3_H}	–	0.4	–	V
4		Low Voltage Warning (LBI 4)					
	M	Assert (Measured on VSUP pin, falling edge)	V_{LBI4_A}	8	9	10	V
	M	Deassert (Measured on VSUP pin, rising edge)	V_{LBI4_D}	–	–	10.5	V
	C	Hysteresis (measured on VSUP pin)	V_{LBI4_H}	–	0.4	–	V
5		High Voltage Warning (HBI 1)					
	M	Assert (Measured on VSUP pin, rising edge)	V_{HBI1_A}	15	16.5	18	V
	M	Deassert (Measured on VSUP pin, falling edge)	V_{HBI1_D}	14.5	–	–	V
	C	Hysteresis (measured on VSUP pin)	V_{HBI1_H}	–	1.0	–	V
6		High Voltage Warning (HBI 2)					
	M	Assert (Measured on VSUP pin, rising edge)	V_{HBI2_A}	25	27.5	**30**	V
	M	Deassert (Measured on VSUP pin, falling edge)	V_{HBI2_D}	24	–	–	V
	C	Hysteresis (measured on VSUP pin)	V_{HBI2_H}	–	1.0	–	V
7	T	Pin Input Divider Ratio[2] $Ratio_{VSUP} = V_{SUP} / V_{ADC}$ 5.5V < VSUP < 29 V	$Ratio_{VSUP}$	–	9	–	–
8	C	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

G.2 Dynamic Electrical Characteristics

Table G-2. 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	**C**	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Enable Uncertainty Time	T_{EN_UNC}	–	1	–	us
2	D	Voltage Warning Low Pass Filter	f_{VWLP_filter}	–	0.5	–	Mhz

[1] T_A: Ambient Temperature

Appendix H
SPI Electrical Specifications

This section provides electrical parametrics and ratings for the SPI.

In **Figure H-1.** the measurement conditions are listed.

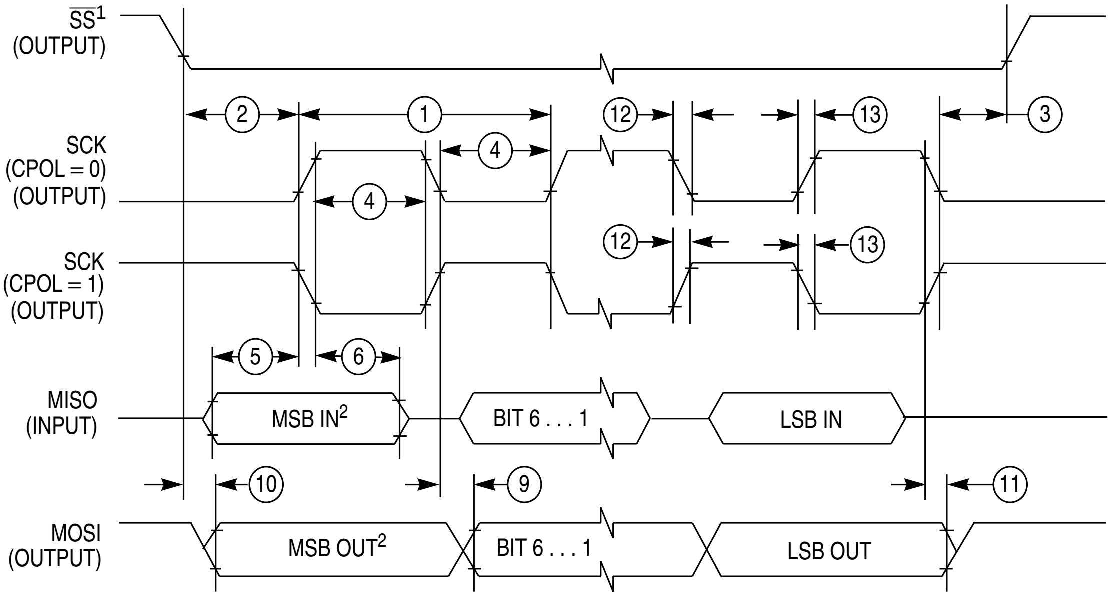
Figure H-1. Measurement Conditions

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.

H.1 Master Mode

In **Figure H-2.** the timing diagram for master mode with transmission format CPHA=0 is depicted.

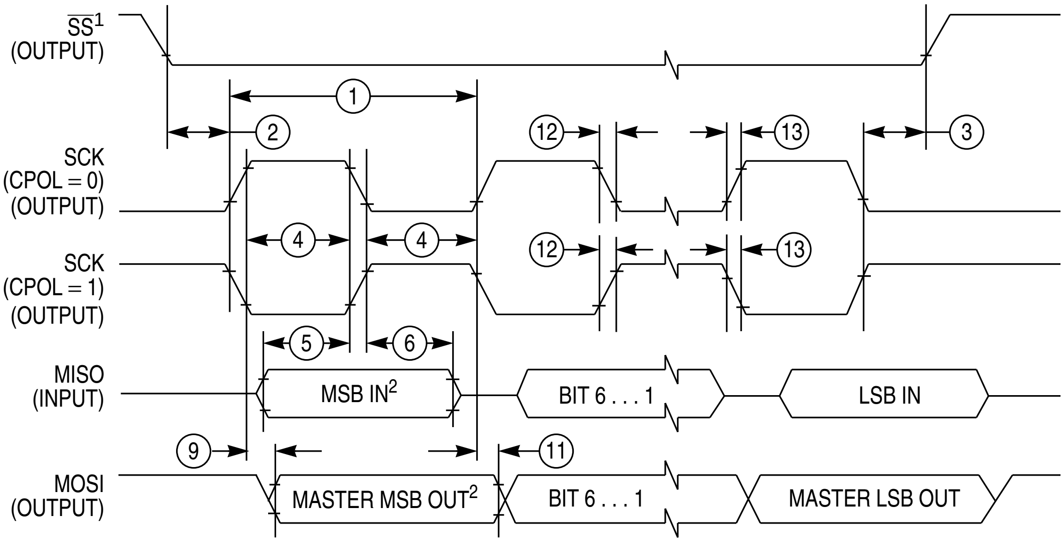


1. If enabled.

2. LSBFE = 0. For LSBFE = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure H-2. SPI Master Timing (CPHA=0)

In **Figure H-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 H-3. SPI Master Timing (CPHA=1)

Table H-1. SPI Master Mode Timing Characteristics (Junction Temperature From -40°C To +175°C)

Num	C	Characteristic	Symbol				Unit
				Min	Typ	Max	
1	D	SCK Frequency	f_{sck}	1/2048	—	1/2(1)(2)	f_{bus}
1	D	SCK Period	t_{sck}	2	—	2048	t_{bus}
2	D	Enable Lead Time	t_{lead}	—	1/2	—	t_{sck}
3	D	Enable Lag Time	t_{lag}	—	1/2	—	t_{sck}
4	D	Clock (SCK) High or Low Time	t_{wsck}	—	1/2	—	t_{sck}
5	D	Data Setup Time (Inputs)	t_{su}	4	—	—	ns
6	D	Data Hold Time (Inputs)	t_{hi}	5	—	—	ns
9	D	Data Valid after SCK Edge	t_{vsck}	—	—	10	ns
10	D	Data Valid after $\overline{SS}$ fall (CPHA=0)	t_{vss}	—	—	9	ns
11	D	Data Hold Time (Outputs)	t_{ho}	-1.2	—	—	ns
12	D	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	D	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

1. See **Figure H-4.**

2. f_{bus} max is 40MHz at temperatures above 150°C

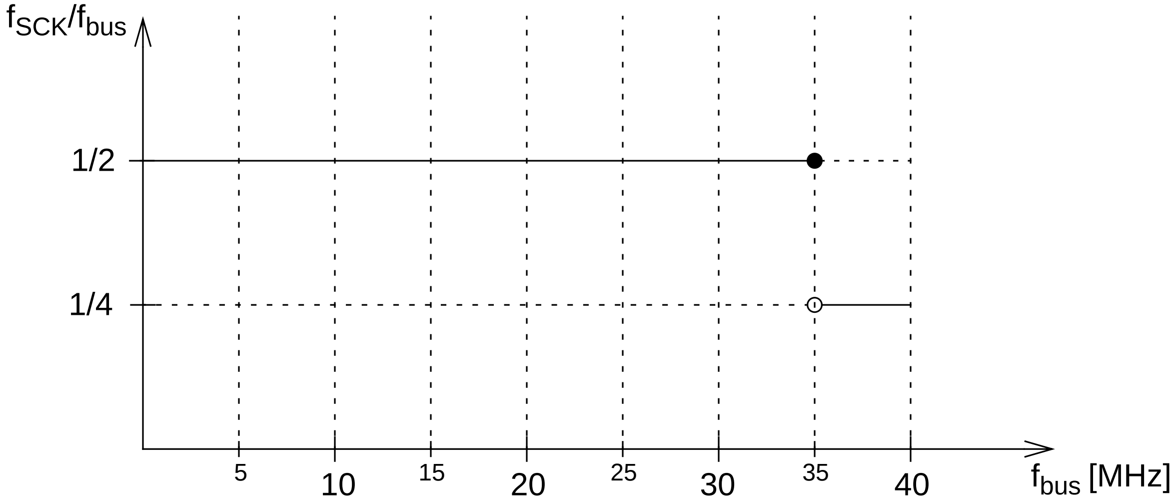


Figure H-4. Derating of maximum f_{SCK} to f_{bus} ratio in Master Mode

In Master Mode the allowed maximum f_{SCK} to f_{bus} ratio (= minimum Baud Rate Divisor, pls. see SPI Block Guide) derates with increasing f_{bus}, please see **Figure H-4.**.

H.1.1 Slave Mode

In **Figure H-1.** the timing diagram for slave mode with transmission format CPHA=0 is depicted.

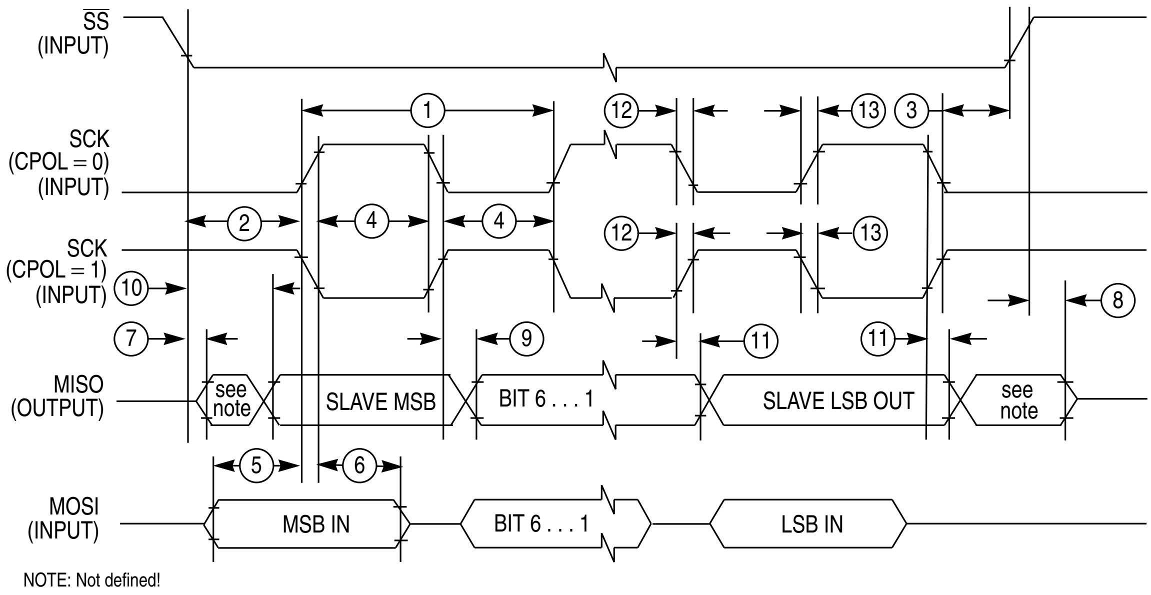


Figure H-5. SPI Slave Timing (CPHA=0)

In **Figure H-6.** the timing diagram for slave mode with transmission format CPHA=1 is depicted.

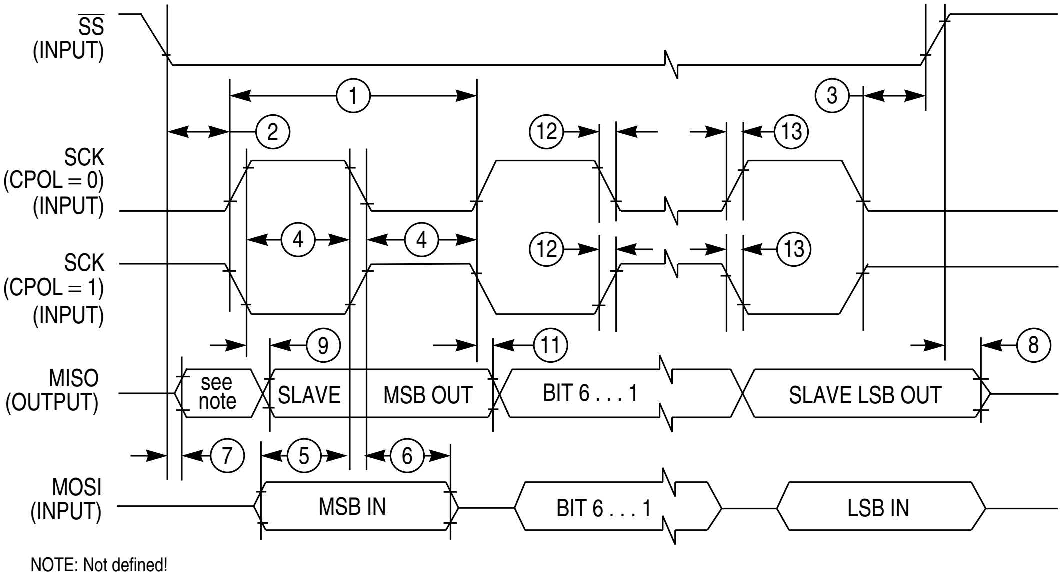


Figure H-6. SPI Slave Timing (CPHA=1)

Table H-2. SPI Slave Mode Timing Characteristics -40°C to 175°C

Num	C	Characteristic	Symbol				Unit
				Min	Typ	Max	
1	D	SCK Frequency	f_{sck}	DC	—	1/4(1)	f_{bus}
1	D	SCK Period	t_{sck}	4	—	∞	t_{bus}
2	D	Enable Lead Time	t_{lead}	4	—	—	t_{bus}
3	D	Enable Lag Time	t_{lag}	4	—	—	t_{bus}
4	D	Clock (SCK) High or Low Time	t_{wsck}	$2t_{bus}$ - $(t_{rfi} + t_{rfo})$	—	—	ns
5	D	Data Setup Time (Inputs)	t_{su}	3	—	—	ns
6	D	Data Hold Time (Inputs)	t_{hi}	2	—	—	ns
7	D	Slave Access Time (time to data active)	t_a	—	—	28	ns
8	D	Slave MISO Disable Time	t_{dis}	—	—	26	ns
9a	D	Data Valid after SCK Edge (-40°C < T_j < 150°C)	t_{vsck}	—	—	$23 + 0.5 \cdot t_{bus}$ (2)	ns
9b	D	Data Valid after SCK Edge (150°C <T_j < 175°C)(1)	t_{vsck}	—	—	$25 + 0.5 \cdot t_{bus}$ (2)	ns
10a	D	Data Valid after $\overline{SS}$ fall (-40°C < T_j < 150°C)	t_{vss}	—	—	$23 + 0.5 \cdot t_{bus}$ (2)	ns
10b	D	Data Valid after $\overline{SS}$ fall (150°C < T_j < 175°C)(1)	t_{vss}	—	—	$25 + 0.5 \cdot t_{bus}$ (2)	ns
11	D	Data Hold Time (Outputs)	t_{ho}	22	—	—	ns
12	D	Rise and Fall Time Inputs	t_{rfi}	—	—	8	ns
13	D	Rise and Fall Time Outputs	t_{rfo}	—	—	8	ns

1. f_{bus} max is 40MHz at temperatures above 150°C

2. $0.5t_{bus}$ added due to internal synchronization delay

Appendix I
MSCAN Electrical Specifications

Table I-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	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	M	MSCAN wake-up dominant pulse filtered	t_{WUP}	—	—	1.5	µs
2	M	MSCAN wake-up dominant pulse pass	t_{WUP}	5	—	—	µs

Appendix J Package Information

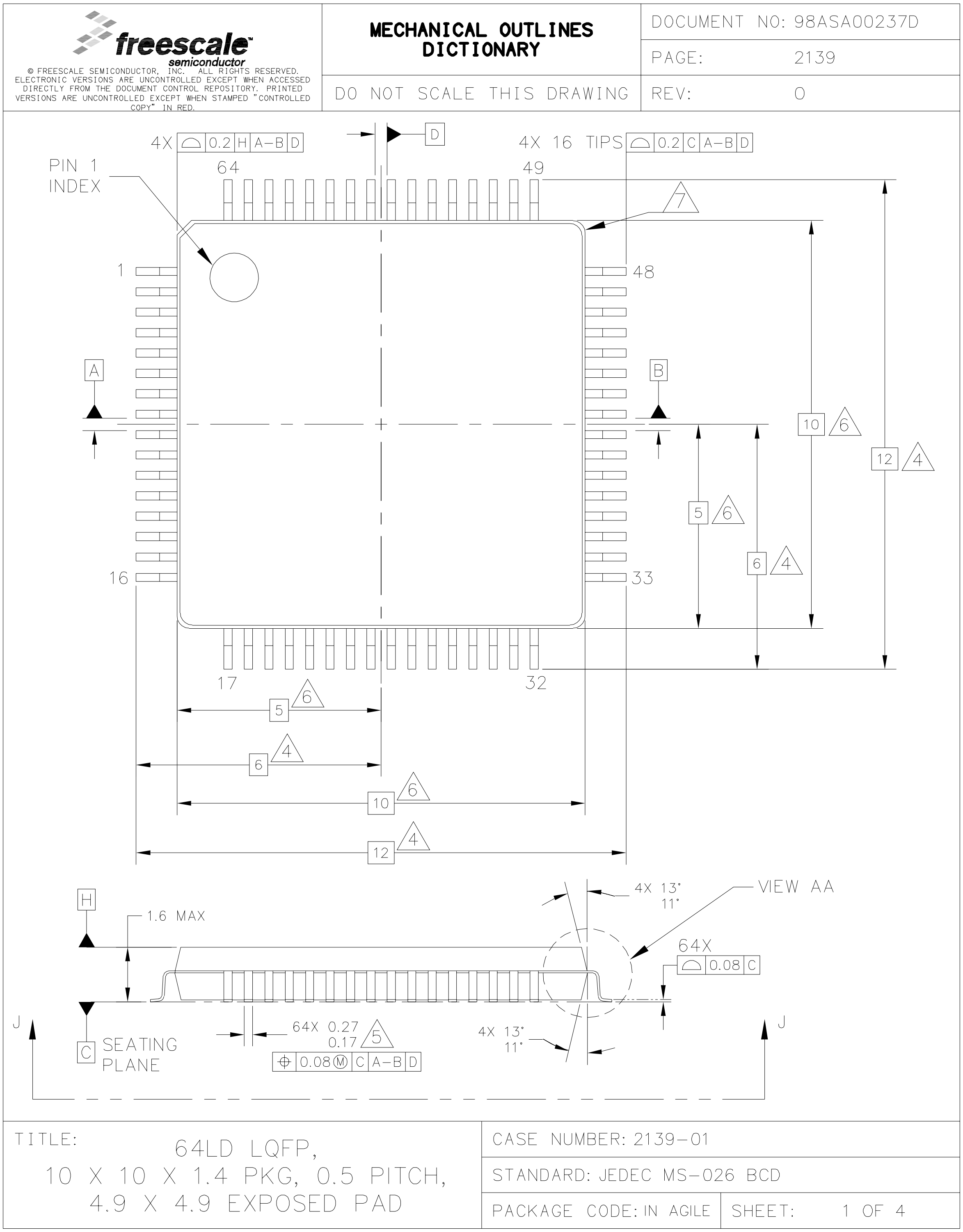
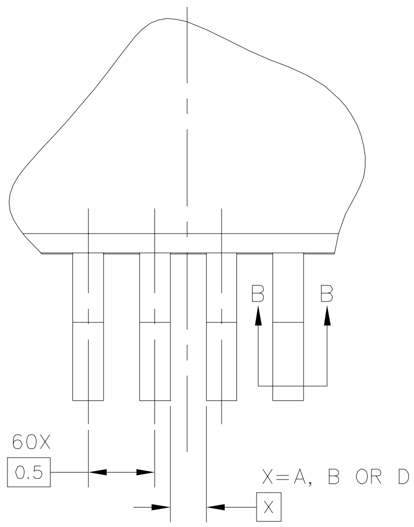
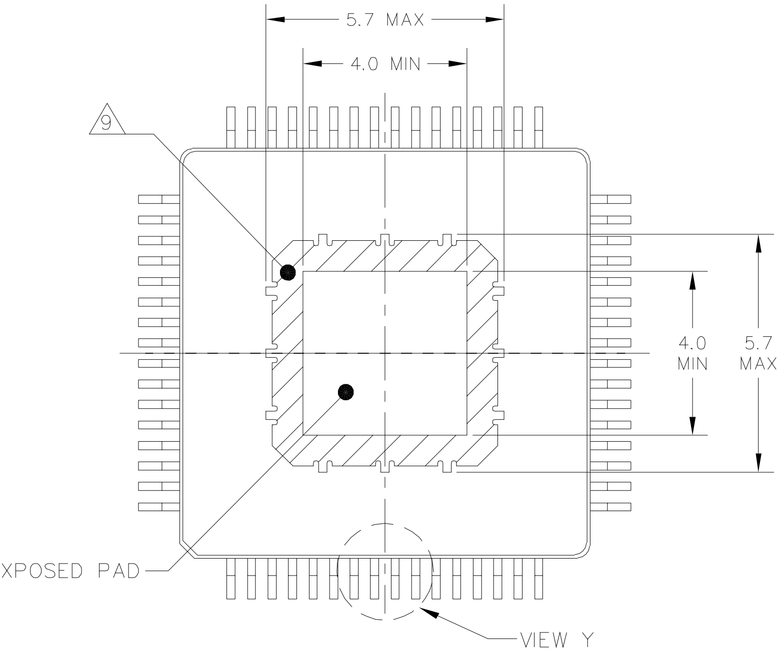


Figure J-1. 64LQFP-EP Mechanical Information

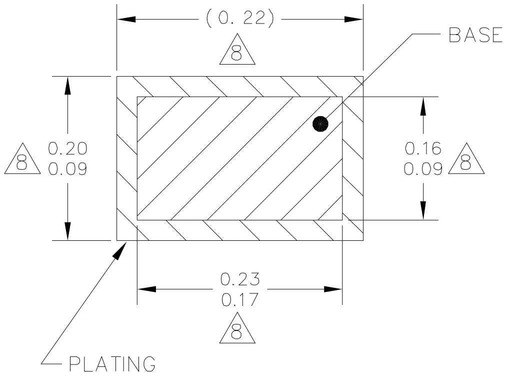
freescale semiconductor © FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED. ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.	**MECHANICAL OUTLINES DICTIONARY**	DOCUMENT NO: 98ASA00237D
		PAGE: 2139
	DO NOT SCALE THIS DRAWING	REV: 0



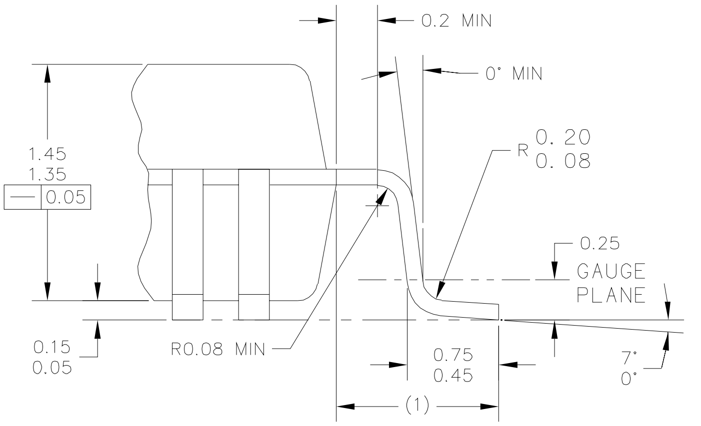
VIEW Y



VIEW J-J



SECTION B-B



VIEW AA

TITLE: 64LD LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, 4.9 X 4.9 EXPOSED PAD	CASE NUMBER: 2139-01	
	STANDARD: JEDEC MS-026 BCD	
	PACKAGE CODE: IN AGILE	SHEET: 2 OF 4

freescale semiconductor 	**MECHANICAL OUTLINES DICTIONARY**	DOCUMENT NO: 98ASA00237D PAGE: 2139
	DO NOT SCALE THIS DRAWING	REV: 0

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994.

3. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE H.

4. DIMENSION TO BE DETERMINED AT SEATING PLANE C.

5. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE UPPER LIMIT BY MORE THAN 0.08 MM AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD SHALL NOT BE LESS THAN 0.07 MM.

6. THIS DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 MM PER SIDE. THIS DIMENSION IS MAXIMUM PLASTIC BODY SIZE DIMENSION INCLUDING MOLD MISMATCH.

7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.1 MM AND 0.25 MM FROM THE LEAD TIP.

9. HATCHED AREA TO BE KEEP OUT ZONE FOR PCB ROUTING.

TITLE: 64LD LQFP, 10 X 10 X 1.4 PKG, 0.5 PITCH, 4.9 X 4.9 EXPOSED PAD	CASE NUMBER: 2139–01	
	STANDARD: JEDEC MS–026 BCD	
	PACKAGE CODE: IN AGILE	SHEET: 3 OF 4

Freescale's Package Reflow capability meets Pb-free requirements for JEDEC standard J-STD-020C. For Peak Package Reflow Temperature and Moisture Sensitivity Levels (MSL), Go to www.freescale.com, search by part number and review parametrics.

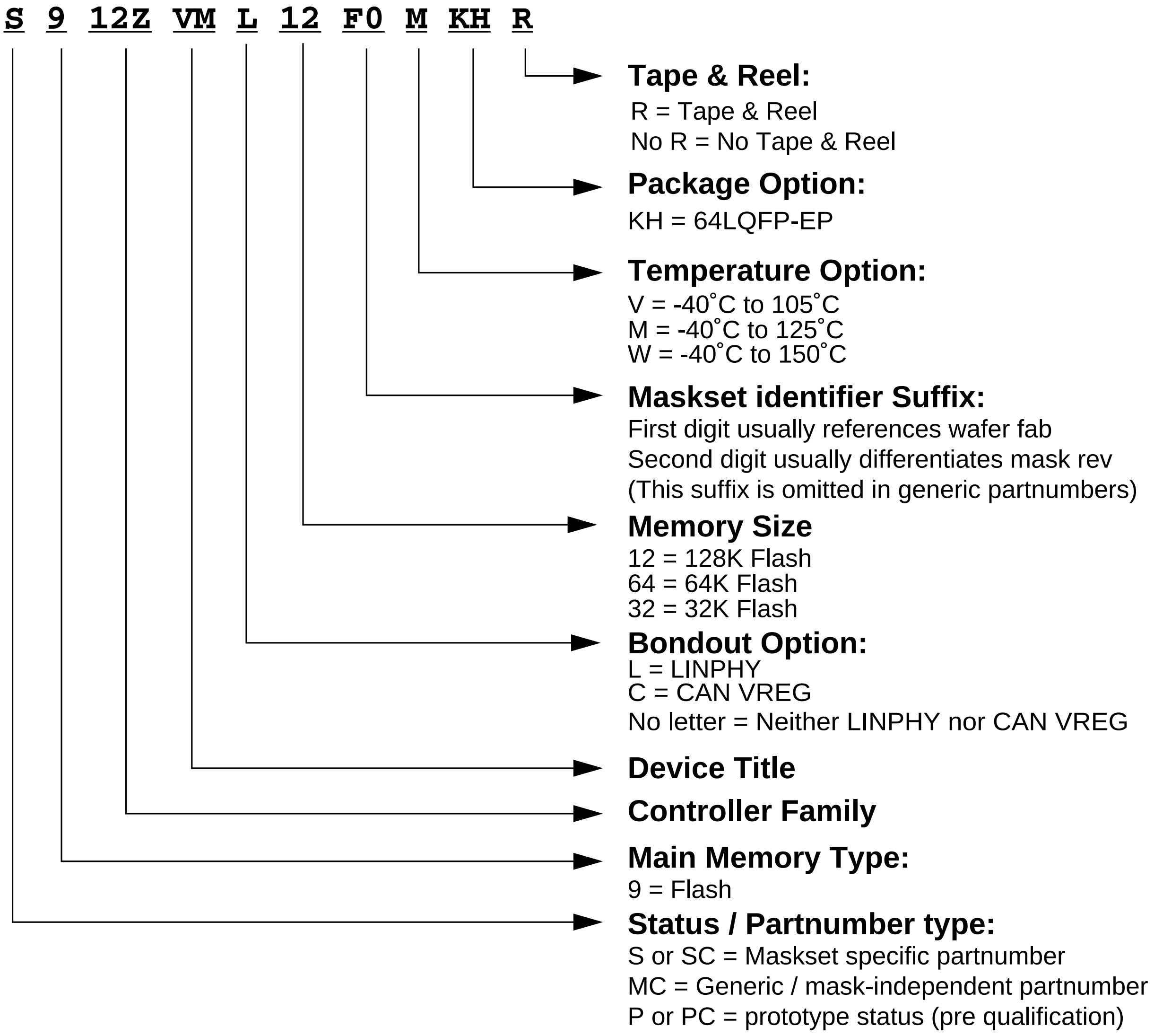
Appendix K
Ordering Information

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 1.2.1 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 12Z VM L 12 F0 M KH R

Tape & Reel:
R = Tape & Reel
No R = No Tape & Reel

Package Option:
KH = 64LQFP-EP

Temperature Option:
V = -40˚C to 105˚C
M = -40˚C to 125˚C
W = -40˚C to 150˚C

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
32 = 32K Flash

Bondout Option:
L = LINPHY
C = CAN VREG
No letter = Neither LINPHY nor CAN VREG

Device Title

Controller Family

Main Memory Type:
9 = Flash

Status / Partnumber type:
S or SC = Maskset specific partnumber
MC = Generic / mask-independent partnumber
P or PC = prototype status (pre qualification)

Appendix L
Detailed Register Address Map

The following tables show the detailed register map of the MC9S12ZVM-Family.

NOTE

Smaller derivatives within the MC9S12ZVM-Family feature a subset of the listed modules.

L.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								

L.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								
0x0012- 0x0016	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0017	INT_CFADDR	R	0	INT_CFADDR[6:3]				0	0	0
		W								
0x0018	INT_CFDATA0	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x0019	INT_CFDATA1	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001A	INT_CFDATA2	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001B	INT_CFDATA3	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001C	INT_CFDATA4	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001D	INT_CFDATA5	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001E	INT_CFDATA6	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								
0x001F	INT_CFDATA7	R	0	0	0	0	0	PRIOLVL[2:0]		
		W								

L.3 0x0070-0x00FF 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								

Address	**Name**		**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
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								

L.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	EEVE	
		W		TRIG						
0x0101	DBGC2	R	0	0	0	0	CDCM		ABCM	
		W								
0x0102	DBGTCRH	R	reserved	TSOURCE	TRANGE		TRCMOD		TALIGN	
		W								
0x0103	DBGTCRL	R	0	0	0	0	DSTAMP	PDOE	PROFILE	STAMP
		W								
0x0104	DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0105	DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0106	DBGCNT	R	0	CNT						
		W								
0x0107	DBGSCR1	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								

L.4 0x0100-0x017F S12ZDBG

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0108	DBGSCR2	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0109	DBGSCR3	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x010A	DBGEFR	R	PTBOVF	TRIGF	0	EEVF	ME3	ME2	ME1	ME0
		W								
0x010B	DBGSR	R	TBF	0	0	PTACT	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								
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								

L.4 0x0100-0x017F S12ZDBG

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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	DBGCCTL	R	0	NDB	INST	0	RW	RWE	reserved	COMPE
		W								
0x0131- 0x0134	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0135	DBGCAH	R	DBGCA[23:16]							
		W								
0x0136	DBGCAM	R	DBGCA[15:8]							
		W								
0x0137	DBGCAL	R	DBGCA[7:0]							
		W								
0x0138	DBGCD0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x0139	DBGCD1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x013A	DBGCD2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x013B	DBGCD3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x013C	DBGCDM0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x013D	DBGCDM1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x013E	DBGCDM2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x013F	DBGCDM3	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0140	DBGDCTL	R	0	0	INST	0	RW	RWE	reserved	COMPE
		W								

L.4 0x0100-0x017F S12ZDBG

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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								

L.5 0x0200-0x02FF PIM

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0200	MODRR0	R	0	0	SPI0SSRR	SPI0RR	SCI1RR	S0L0RR2-0		
		W								
0x0201	MODRR1	R	0	0	0	0	PWMPRR	PWM54RR	PWM32RR	PWM10RR
		W								
0x0202	MODRR2	R	0	0	0	0	T0IC3RR1-0		T0IC1RR	0
		W								
0x0203– 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	PIMMISC	R	0	0	0	0	0	0	OCPE1	0
		W								
0x020B– 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	0	0	0	0	0	0	0	PTADH0
		W								
0x0281	PTADL	R	PTADL7	PTADL6	PTADL5	PTADL4	PTADL3	PTADL2	PTADL1	PTADL0
		W								
0x0282	PTIADH	R	0	0	0	0	0	0	0	PTIADH0
		W								
0x0283	PTIADL	R	PTIADL7	PTIADL6	PTIADL5	PTIADL4	PTIADL3	PTIADL2	PTIADL1	PTIADL0
		W								
0x0284	DDRADH	R	0	0	0	0	0	0	0	DDRADH0
		W								
0x0285	DDRADL	R	DDRADL7	DDRADL6	DDRADL5	DDRADL4	DDRADL3	DDRADL2	DDRADL1	DDRADL0
		W								
0x0286	PERADH	R	0	0	0	0	0	0	0	PERADH0
		W								
0x0287	PERADL	R	PERADL7	PERADL6	PERADL5	PERADL4	PERADL3	PERADL2	PERADL1	PERADL0
		W								
0x0288	PPSADH	R	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0	PIEADH0
		W								
0x028D	PIEADL	R	PIEADL7	PIEADL6	PIEADL5	PIEADL4	PIEADL3	PIEADL2	PIEADL1	PIEADL0
		W								
0x028E	PIFADH	R	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0	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	0	0	0	0	PTT3	PTT2	PTT1	PTT0
		W								
0x02C1	PTIT	R	0	0	0	0	PTIT3	PTIT2	PTIT1	PTIT0
		W								
0x02C2	DDRT	R	0	0	0	0	DDRT3	DDRT2	DDRT1	DDRT0
		W								
0x02C3	PERT	R	0	0	0	0	PERT3	PERT2	PERT1	PERT0
		W								
0x02C4	PPST	R	0	0	0	0	PPST3	PPST2	PPST1	PPST0
		W								
0x02C5–0x02CF	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	0	0	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		W								
0x02D1	PTIS	R	0	0	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		W								
0x02D2	DDRS	R	0	0	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		W								
0x02D3	PERS	R	0	0	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		W								
0x02D4	PPSS	R	0	0	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		W								
0x02D5	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02D6	PIES	R	0	0	PIES5	PIES4	PIES3	PIES2	PIES1	PIES0
		W								
0x02D7	PIFS	R	0	0	PIFS5	PIFS4	PIFS3	PIFS2	PIFS1	PIFS0
		W								
0x02D8– 0x02DE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02DF	WOMS	R	0	0	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		W								
0x02E0– 0x02EF	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F0	PTP	R	0	0	0	0	0	PTP2	PTP1	PTP0
		W								
0x02F1	PTIP	R	0	0	0	0	0	PTIP2	PTIP1	PTIP0
		W								
0x02F2	DDRP	R	0	0	0	0	0	DDRP2	DDRP1	DDRP0
		W								
0x02F3	PERP	R	0	0	0	0	0	PERP2	PERP1	PERP0
		W								
0x02F4	PPSP	R	0	0	0	0	0	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	OCIE1	0	0	0	0	PIEP2	PIEP1	PIEP0
		W								
0x02F7	PIFP	R	OCIF1	0	0	0	0	PIFP2	PIFP1	PIFP0
		W								
0x02F8–0x02FC	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02FD	RDRP	R	0	0	0	0	0	0	0	RDRP0
		W								
0x02FE–0x02FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

L.6 0x0380-0x039F FTMRZ128K512

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	0	SFDIE
		W								
0x0386	FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
		W								
0x0387	FERSTAT	R	0	0	0	0	0	0	DFDF	SFDIF
		W								
0x0388	FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
		W								

L.6 0x0380-0x039F FTMRZ128K512 (continued)

Address	Name		7	6	5	4	3	2	1	0
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								
0x038C	FCCOB0HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x038D	FCCOB0LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x038E	FCCOB1HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x038F	FCCOB1LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x0390	FCCOB2HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x0391	FCCOB2LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x0392	FCCOB3HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x0393	FCCOB3LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x0394	FCCOB4HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x0395	FCCOB4LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x0396	FCCOB5HI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x0397	FCCOB5LO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								

L.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								

L.7 0x03C0-0x03CF SRAM_ECC_32D7P

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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								
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								

L.8 0x0500-x053F PMF15B6C

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0500	PMFCFG0	R	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
		W								
0x0501	PMFCFG1	R	0	ENCE	BOTNEGC	TOPNEGC	BOTNEGB	TOPNEGB	BOTNEGA	TOPNEGA
		W								
0x0502	PMFCFG2	R	REV1	REV0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		W								
0x0503	PMFCFG3	R	PMFWAI	PMFFRZ	0	VLMODE		PINVC	PINVB	PINVA
		W								
0x0504	PMFFEN	R	0	FEN5	0	FEN4	FEN3	FEN2	FEN1	FEN0
		W								

L.8 0x0500-x053F PMF15B6C

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0505	PMFFMOD	R	0	FMOD5	0	FMOD4	FMOD3	FMOD2	FMOD1	FMOD0
		W								
0x0506	PMFFIE	R	0	FIE5	0	FIE4	FIE3	FIE2	FIE1	FIE0
		W								
0x0507	PMFFIF	R	0	FIF5	0	FIF4	FIF3	FIF2	FIF1	FIF0
		W								
0x0508	PMFQSMP0	R	0	0	0	0	QSMP5		QSMP4	
		W								
0x0509	PMFQSMP1	R	QSMP3		QSMP2		QSMP1		QSMP0	
		W								
0x050A- 0x050B	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x050C	PMFOUTC	R	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
		W								
0x050D	PMFOUTB	R	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
		W								
0x050E	PMFDTMS	R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
		W								
0x050F	PMFCCTL	R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
		W								
0x0510	PMFVAL0	R	PMFVAL0							
		W								
0x0511	PMFVAL0	R	PMFVAL0							
		W								
0x0512	PMFVAL1	R	PMFVAL1							
		W								
0x0513	PMFVAL1	R	PMFVAL1							
		W								
0x0514	PMFVAL2	R	PMFVAL2							
		W								
0x0515	PMFVAL2	R	PMFVAL2							
		W								
0x0516	PMFVAL3	R	PMFVAL3							
		W								

L.8 0x0500-x053F PMF15B6C

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0517	PMFVAL3	R	PMFVAL3							
		W								
0x0518	PMFVAL4	R	PMFVAL4							
		W								
0x0519	PMFVAL4	R	PMFVAL4							
		W								
0x051A	PMFVAL5	R	PMFVAL5							
		W								
0x051B	PMFVAL5	R	PMFVAL5							
		W								
0x051C	PMFROIE	R	0	0	0	0	0	PMFROIEC	PMFROIEB	PMFROIEA
		W								
0x051D	PMFROIF	R	0	0	0	0	0	PMFROIFC	PMFROIFB	PMFROIFA
		W								
0x051E	PMFICCTL	R	0	0	PECC	PECB	PECA	ICCC	ICCB	ICCA
		W								
0x051F	PMFCINV	R	0	0	CINV5	CINV4	CINV3	CINV2	CINV1	CINV0
		W								
0x0520	PMFENCA	R	PWMENA	GLDOKA	0	0	0	RSTRTA	LDOKA	PWMRIEA
		W								
0x0521	PMFFQCA	R	LDFQA				HALFA	PRSCA		PWMRFA
		W								
0x0522	PMFCNTA	R	0	PMFCNTA						
		W								
0x0523	PMFCNTA	R	PMFCNTA							
		W								
0x0524	PMFMODA	R	0	PMFMODA						
		W								
0x0525	PMFMODA	R	PMFMODA							
		W								
0x0526	PMFDTMA	R	0	0	0	0	PMFDTMA			
		W								
0x0527	PMFDTMA	R	PMFDTMA							
		W								

L.8 0x0500-x053F PMF15B6C

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0528	PMFENCB	R	PWMENB	GLDOKB	0	0	0	RSTRTB	LDOKB	PWMRIEB
		W								
0x0529	PMFFQCB	R	LDFQB				HALFB	PRSCB		PWMRFB
		W								
0x052A	PMFCNTB	R	0	PMFCNTB						
		W								
0x052B	PMFCNTB	R	PMFCNTB							
		W								
0x052C	PMFMODB	R	0	PMFMODB						
		W								
0x052D	PMFMODB	R	PMFMODB							
		W								
0x052E	PMFDTMB	R	0	0	0	0	PMFDTMB			
		W								
0x052F	PMFDTMB	R	PMFDTMB							
		W								
0x0530	PMFENCC	R	PWMENC	GLDOKC	0	0	0	RSTRTC	LDOKC	PWMRIEC
		W								
0x0531	PMFFQCC	R	LDFQC				HALFC	PRSCC		PWMRFC
		W								

L.8 0x0500-x053F PMF15B6C

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0532	PMFCNTC	R	0	PMFCNTC						
		W								
0x0533	PMFCNTC	R	PMFCNTC							
		W								
0x0534	PMFMODC	R	0	PMFMODC						
		W								
0x0535	PMFMODC	R	PMFMODC							
		W								
0x0536	PMFDTMC	R	0	0	0	0	PMFDTMC			
		W								
0x0537	PMFDTMC	R	PMFDTMC							
		W								
0x0538	PMFDMP0	R	DMP05		DMP04		DMP03	DMP02	DMP01	DMP00
		W								
0x0539	PMFDMP1	R	DMP15		DMP14		DMP13	DMP12	DMP11	DMP10
		W								
0x053A	PMFDMP2	R	DMP25		DMP24		DMP23	DMP22	DMP21	DMP20
		W								
0x053B	PMFDMP3	R	DMP35		DMP34		DMP33	DMP32	DMP31	DMP30
		W								
0x053C	PMFDMP4	R	DMP45		DMP44		DMP43	DMP42	DMP41	DMP40
		W								
0x053D	PMFDMP5	R	DMP55		DMP54		DMP53	DMP52	DMP51	DMP50
		W								
0x053E	PMFOUTF	R	0	0	OUTF5	OUTF4	OUTF3	OUTF2	OUTF1	OUTF0
		W								
0x053F	Reserved	R	0	0	0	0	0	0	0	0
		W								

L.9 0x0580-0x059F PTU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0580	PTUE	R	0	PTUFRZ	0	0	0	0	TG1EN	TG0EN
		W								

L.9 0x0580-0x059F PTU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0581	PTUC	R	0	0	0	0	0	0	0	PTULDOK
		W								
0x0582	PTUIEH	R	0	0	0	0	0	0	0	PTUROIE
		W								
0x0583	PTUIEL	R	TG1AEIE	TG1REIE	TG1TEIE	TG1DIE	TG0AEIE	TG0REIE	TG0TEIE	TG0DIE
		W								
0x0584	PTUIFH	R	0	0	0	0	0	0	PTUDEEF	PTUROIF
		W								
0x0585	PTUIFL	R	TG1AEIF	TG1REIF	TG1TEIF	TG1DIF	TG0AEIF	TG0REIF	TG0TEIF	TG0DIF
		W								
0x0586	TG0LIST	R	0	0	0	0	0	0	0	TG0LIST
		W								
0x0587	TG0TNUM	R	0	0	0	TG0TNUM[4:0]				
		W								
0x0588	TG0TVH	R	TG0TV[15:8]							
		W								
0x0589	TG0TVL	R	TG0TV[7:0]							
		W								
0x058A	TG1LIST	R	0	0	0	0	0	0	0	TG1LIST
		W								
0x058B	TG1TNUM	R	0	0	0	TG1TNUM4:0]				
		W								
0x058C	TG1TVH	R	TG1TV[15:8]							
		W								
0x058D	TG1TVL	R	TG1TV[7:0]							
		W								
0x058E	PTUCNTH	R	PTUCNT[15:8]							
		W								
0x058F	PTUCNTL	R	PTUCNT[7:0]							
		W								
0x0590	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0591	PTUPTRH	R	PTUPTR[23:16]							
		W								

L.9 0x0580-0x059F PTU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0592	PTUPTRM	R	PTUPTR[15:8]							
		W								
0x0593	PTUPTRL	R	PTUPTR[7:1]							0
		W								
0x0594	TG0L0IDX	R	0	0	0	0	0	0	0	0
		W								
0x0595	TG0L1IDX	R	0	TG0L1IDX[6:0]						
		W								
0x0596	TG1L0IDX	R	0	TG1L0IDX[6:0]						
		W								
0x0597	TG1L1IDX	R	0	TG1L1IDX[6:0]						
		W								
0x0598 - 0x059E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x059F	PTUDEBUG	R	0	PTUREPE	PTUT1PE	PTUT0PE	0	0	0	0
		W						PTUFRE	TG1FTE	TG0FTE

L.10 0x05C0-0x05FF TIM0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05C0	TIM0TIOS	R					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	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
		W								
0x05C5	TIM0TCNTL	R	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
		W								
0x05C6	TIM0TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
		W								
0x05C7	TIM0TTOV	R					TOV3	TOV2	TOV1	TOV0
		W								

L.10 0x05C0-0x05FF TIM0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05C8	TIM0TCTL1	R								
		W								
0x05C9	TIM0TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		W								
0x05CA	TIM0TCTL3	R								
		W								
0x05CB	TIM0TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		W								
0x05CC	TIM0TIE	R					C3I	C2I	C1I	C0I
		W								
0x05CD	TIM0TSCR2	R	TOI	0	0	0		PR2	PR1	PR0
		W								
0x05CE	TIM0TFLG1	R					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								
0x05E0	Reserved	R								
		W								

L.10 0x05C0-0x05FF TIM0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x05E1	Reserved	R								
		W								
0x05E2	Reserved	R								
		W								
0x05E3	Reserved	R								
		W								
0x05E4–0x05EB	Reserved	R								
		W								
0x05EC	TIM0OCPD	R					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								

L.11 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								
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								

L.11 0x0600-0x063F ADC0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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	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								
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								

L.11 0x0600-0x063F ADC0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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								

L.12 0x0640-0x067F ADC1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0640	ADC1CTL_0	R	ADC_EN	ADC_SR	FRZ_MOD	SWAI	ACC_CFG[1:0]		STR_SEQA	MOD_CFG
		W								
0x0641	ADC1CTL_1	R	CSL_BMOD	RVL_BMOD	SMOD_ACC	AUT_RSTA	0	0	0	0
		W								
0x0642	ADC1STS	R	CSL_SEL	RVL_SEL	DBECC_ERR	Reserved	READY	0	0	0
		W								
0x0643	ADC1TIM	R	0	PRS[6:0]						
		W								
0x0644	ADC1FMT	R	DJM	0	0	0	0	SRES[2:0]		
		W								
0x0645	ADC1FLWCTL	R	SEQA	TRIG	RSTA	LDOK	0	0	0	0
		W								
0x0646	ADC1EIE	R	IA_EIE	CMD_EIE	EOL_EIE	Reserved	TRIG_EIE	RSTAR_EIE	LDOK_EIE	0
		W								
0x0647	ADC1IE	R	SEQAD_IE	CONIF_OIE	Reserved	0	0	0	0	0
		W								
0x0648	ADC1EiF	R	IA_EIF	CMD_EIF	EOL_EIF	Reserved	TRIG_EIF	RSTAR_EIF	LDOK_EIF	0
		W								
0x0649	ADC1IF	R	SEQAD_IF	CONIF_OIF	Reserved	0	0	0	0	0
		W								
0x064A	ADC1CONIE_0	R	CON_IE[15:8]							
		W								
0x064B	ADC1CONIE_1	R	CON_IE[7:1]							EOL_IE
		W								
0x064C	ADC1CONIF_0	R	CON_IF[15:8]							
		W								

L.12 0x0640-0x067F ADC1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x064D	ADC1CONIF_1	R	CON_IF[7:1]							EOL_IF
		W								
0x064E	ADC1IMDRI_0	R	CSL_IMD	RVL_IMD	0	0	0	0	0	0
0x064F	ADC1IMDRI_1	R	0	0	RIDX_IMD					
		W								
0x0650	ADC1EOLRI	R	CSL_EOL	RVL_EOL	0	0	0	0	0	0
		W								
0x0651	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0652	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0653	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0654	ADC1CMD_0	R	CMD_SEL		0	0	INTFLG_SEL[3:0]			
		W								
0x0655	ADC1CMD_1	R	VRH_SEL	VRL_SEL	CH_SEL[5:0]					
		W								
0x0656	ADC1CMD_2	R	SMP[4:0]					0	0	Reserved
		W								
0x0657	ADC1CMD_3	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0658	Reserved	R	Reserved							
		W								
0x0659	Reserved	R	Reserved							
		W								
0x065A	Reserved	R	Reserved							
		W								
0x065B	Reserved	R	Reserved							
		W								
0x065C	ADC1CIDX	R	0	0	CMD_IDX[5:0]					
		W								
0x065D	ADC1CBP_0	R	CMD_PTR[23:16]							
		W								
0x065E	ADC1CBP_1	R	CMD_PTR[15:8]							
		W								
0x065F	ADC1CBP_2	R	CMD_PTR[7:2]						0	0
		W								
0x0660	ADC1RIDX	R	0	0	RES_IDX[5:0]					
		W								
0x0661	ADC1RBP_0	R	0	0	0	0	RES_PTR[19:16]			
		W								
0x0662	ADC1RBP_1	R	RES_PTR[15:8]							
		W								
0x0663	ADC1RBP_2	R	RES_PTR[7:2]						0	0
		W								

L.12 0x0640-0x067F ADC1

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0664	ADC1CROFF0	R	0	CMDRES_OFF0[6:0]						
		W								
0x0665	ADC1CROFF1	R	0	CMDRES_OFF1[6:0]						
		W								
0x0666	Reserved	R	0	0	0	0	Reserved			
		W								
0x0667	Reserved	R	Reserved							
		W								
0x0668	Reserved	R	Reserved						0	0
		W								
0x0669	Reserved	R	Reserved	0	Reserved					
		W								
0x066A- 0x067F	Reserved	R	0	0	0	0	0	0	0	0
		W								

L.13 0x06A0-0x06BF GDU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06A0	GDUE	R	GWP	0	EPRES	GCS1E	GBOE	GCS0E	GCPE	GFDE
		W								
0x06A1	GDUCTR	R	GHHDLVL	0	GBKTIM2[3:0]				GBKTIM1[1:0]	
		W								
0x06A2	GDUIE	R	0	0	0	GOCIE[1:0]		GDSEIE	GHHDIE	GLVLSIE
		W								
0x06A3	GDUDSE	R	0	GDHSIF[2:0]			0	GDLSIF[2:0]		
		W								
0x06A4	GDUSTAT	R	GPHS[2:0]			GOCS[1:0]			GHHDS	GLVLSS
		W								
0x06A5	GDUSRC	R	0	GSRCHS[2:0]			0	GSRCLS[2:0]		
		W								
0x06A6	GDUF	R	GSUF	GHHDF	GLVLSF	GOCIF[1:0]		0	GHHDIF	GLVLSIF
		W								
0x06A7	GDUCLK1	R	0	GBOCD[4:0]					GBODC[1:0]	
		W								
0x06A8	GDUBCL	R	0	0	0	0	GBCL[3:0]			
		W								
0x06A9	GDUPHMUX	R	0	0	0	0	0	0	GPHMX[1:0]	
		W								

L.13 0x06A0-0x06BF GDU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06AA	GDUCSO	R	0	GCSO1[2:0]			0	GCSO0[2:0]		
		W								
0x06AB	GDUDSLVL	R	0	GDSLHS[2:0]			0	GDSLLS[2:0]		
		W								
0x06AC	GDUPHL	R	0	0	0	0	0	GPHL[2:0]		
		W								
0x06AD	GDUCLK2	R	0	0	0	0	GCPCD[3:0]			
		W								
0x06AE	GDUOC0	R	GOCA0	GOCE0	0	0	GOCT0[3:0]			
		W								
0x06AF	GDUOC1	R	GOCA1	GOCE1	0	0	GOCT1[3:0]			
		W								
0x06B0- 0x06BF	Reserved	R								
		W								

L.14 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								

L.14 0x06C0-0x06DF CPMU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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								
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								

L.14 0x06C0-0x06DF CPMU

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x06DD	CPMU VREGCTL	R	0	0	0	0	0	EXTCON	EXTXON	INTXON
		W								
0x06DE	CPMUOSC2	R	0	0	0	0	0	0	OMRE	OSCMOD
		W								
0x06DF	CPMU RESERVED1F	R	0	0	0	0	0	0	0	0
		W								

L.15 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								

L.16 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								

L.16 0x0700-0x0707 SCI0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
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								
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.

L.17 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.

L.18 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								

L.18 0x0780-0x0787 SPI0

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
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								

L.19 0x0800–0x083F CAN0

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0800	CAN0CTL0	R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
		W								
0x0801	CAN0CTL1	R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
		W								
0x0802	CAN0BTR0	R	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		W								
0x0803	CAN0BTR1	R	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
		W								
0x0804	CAN0RFLG	R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
		W								
0x0805	CAN0RIER	R	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
		W								
0x0806	CAN0TFLG	R	0	0	0	0	0	TXE2	TXE1	TXE0
		W								
0x0807	CAN0TIER	R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
		W								
0x0808	CAN0TARQ	R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
		W								
0x0809	CAN0TAAK	R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
		W								
0x080A	CAN0TBSEL	R	0	0	0	0	0	TX2	TX1	TX0
		W								
0x080B	CAN0IDAC	R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
		W								
0x080C	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x080D	CAN0MISC	R	0	0	0	0	0	0	0	BOHOLD
		W								
0x080E	CAN0RXERR	R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
		W								

L.19 0x0800–0x083F CAN0

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x080F	CAN0TXERR	R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
		W								
0x0810–0x0813	CAN0IDAR0–CAN0IDAR3	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		W								
0x0814–0x0817	CAN0IDMR0–CAN0IDMR3	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		W								
0x0818–0x081B	CAN0IDAR4–CAN0IDAR7	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		W								
0x081C–0x081F	CAN0IDMR4–CAN0IDMR7	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		W								
0x0820–0x082F	CAN0RXFG	R	FOREGROUND RECEIVE BUFFER							
		W								
0x0830–0x083F	CAN0TXFG	R	FOREGROUND TRANSMIT BUFFER							
		W								

L.20 0x0980-0x0987 LINPHY0

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0980	LP0DR	R	0	0	0	0	0	0	LPDR1	LPDR0
		W								
0x0981	LP0CR	R	0	0	0	0	LPE	RXONLY	LPWUE	LPPUE
		W								
0x0982	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0983	LP0SLRM	R	LPDTDIS	0	0	0	0	0	LPSLR1	LPSLR0
		W								
0x0984	Reserved	R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
		W								
0x0985	LP0SR	R	LPDT	0	0	0	0	0	0	0
		W								
0x0986	LP0IE	R	LPDTIE	LPOCIE	0	0	0	0	0	0
		W								
0x0987	LP0IF	R	LPDTIF	LPOCIF	0	0	0	0	0	0
		W								

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. Freescale reserves the right to make changes without further notice to any products herein.

Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. MagniV is a trademark of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2014 Freescale Semiconductor, Inc.

Document Number MC9S12ZVMRMV1
Revision 1.3

LifeElectronics

Живое партнерство

ООО "ЛайфЭлектроникс" **"LifeElectronics" LLC**

ИНН 7805602321 КПП 780501001 Р/С 40702810122510004610 ФАКБ "АБСОЛЮТ БАНК" (ЗАО) в г.Санкт-Петербурге К/С 30101810900000000703 БИК 044030703

Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.

С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические), за счёт заключения дистрибьюторских договоров

Мы предлагаем:

- *Конкурентоспособные цены и скидки постоянным клиентам.*
- *Специальные условия для постоянных клиентов.*
- *Подбор аналогов.*
- *Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.*

- *Приемлемые сроки поставки, возможна ускоренная поставка.*
- *Доставку товара в любую точку России и стран СНГ.*
- *Комплексную поставку.*
- *Работу по проектам и поставку образцов.*
- *Формирование склада под заказчика.*

- *Сертификаты соответствия на поставляемую продукцию (по желанию клиента).*
- *Тестирование поставляемой продукции.*
- *Поставку компонентов, требующих военную и космическую приемку.*
- *Входной контроль качества.*
- *Наличие сертификата ISO.*

В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.

Конструкторский отдел помогает осуществить:

- *Регистрацию проекта у производителя компонентов.*
- *Техническую поддержку проекта.*
- *Защиту от снятия компонента с производства.*
- *Оценку стоимости проекта по компонентам.*
- *Изготовление тестовой платы монтаж и пусконаладочные работы.*

Тел: +7 (812) 336 43 04 (многоканальный)
Email: org@lifeelectronics.ru

www.lifeelectronics.ru