TMS470PVF24xB/34xB

Technical Reference Manual (TRM)

Literature Number: SPNU239B
April 2011

Texas Instruments

List of Tables

List of Figures

Read This First

About this Manual

This document describes the TMS470PVF24xB/34xB device.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (64 decimal): 40h. Hexadecimal numbers can also be shown as 0xN.
- Registers in this document are shown in figures and described in tables.
- Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
- Sometimes bits are combined. If each bit is used separately and the only value possible is a 0 or 1, the bits are shown as [n1:n2]. If the bits combine to form a bit field, such as an address definition, they are shown as (n1–n2), and the value is either shown as a range, e.g., 0–FFFFh, or as the individual possible values, e.g., 000, 001, etc.

Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documents

The following are documents available for more information. All ARM documents are available on the web at http://www.ARM.com.

- *ARM Technical Reference Manual,* ARM reference number 1020E, ARM LTD
- *ARM Architecture Reference Manual,* ARM reference number DDI0100, latest version, ARM LTD
- *AMBA Specification,* ARM reference number IHI0011, latest version (Note: you must register on the site to be able to access this document)
- *AMBA AXI Protocol,* ARM reference number IHI0022, latest version (Note: you must register on the site to be able to access this document)
- *Embedded Trace Macrocell Architecture Specification,* ARM reference number IHI0014, latest version
- Stahnke, W. *Primitive binary polynomials*, Math. Comp. 27 (1973), pp. 977—980.
- J1850 Class B protocol established by the Society of Automotive Engineers (SAE)
- *ARM Cortex M3 Technical Reference Manual, ARM reference number DDI 0337E*

Trademarks

TMS470Px is a trademark of Texas Instruments.

All other trademarks are the property of their respective owners.

Architecture Specification

This chapter describes the architecture of the Texas Instruments TMS470Px family. It gives the internal device architecture, the memory mapping, and the main system modules.

Topic **Page**

1.1 General Description

The TMS470Px system is built around a bus interconnect. The bus interconnect allows high performance transactions between the ARM7TDMI CPU to the rest of the device.

The CPU features can be found in their respective Technical Reference Manual. For complete information about the ARM Architecture see *ARM Architecture Reference Manual.*

The memory sub-system attached to the CPU is composed of:

- A Bus Matrix module that includes:
  - A decoder per master
  - An arbiter by slave
- eSRAM wrappers that are connected to the bus matrix
- An internal program memory.
- A peripheral bridge that connects the CPU to the peripherals.
- A Cyclic Redundancy Check module (CRC) that provides up to 4 different channels including Parallel signature analysis dependent on device specification.

1.2 Block diagram

1.2.1 Overview

The TMS470Px family is an interconnect platform, which implies the definition of the interface between the CPU and the modules.

The ARM CPU is interconnected with the rest of the device via the Bus Matrix Module (BMM) (see section 2.1). The BMM allows the interconnect also with the rest of the AMBA slave modules which includes the RAM (see section 4.1), the CRC module, and the peripheral bus bridge (see section 3.1).

The TMS470Px family supports internal program memory. The internal program memory wrapper includes its own arbiter to maximize the throughput from the internal program memory.

All peripherals are connected to the peripheral bridge that allows the conversion from the AMBA protocol to the TI Standard VBUS. The bridge also separates the system bus clock domain (AMBA clock) from the peripheral bus clock domain (VBUS clock). The peripheral decoding and muxing are done within the PCR module (see section 3.3) that is attached to the Peripheral bridge.

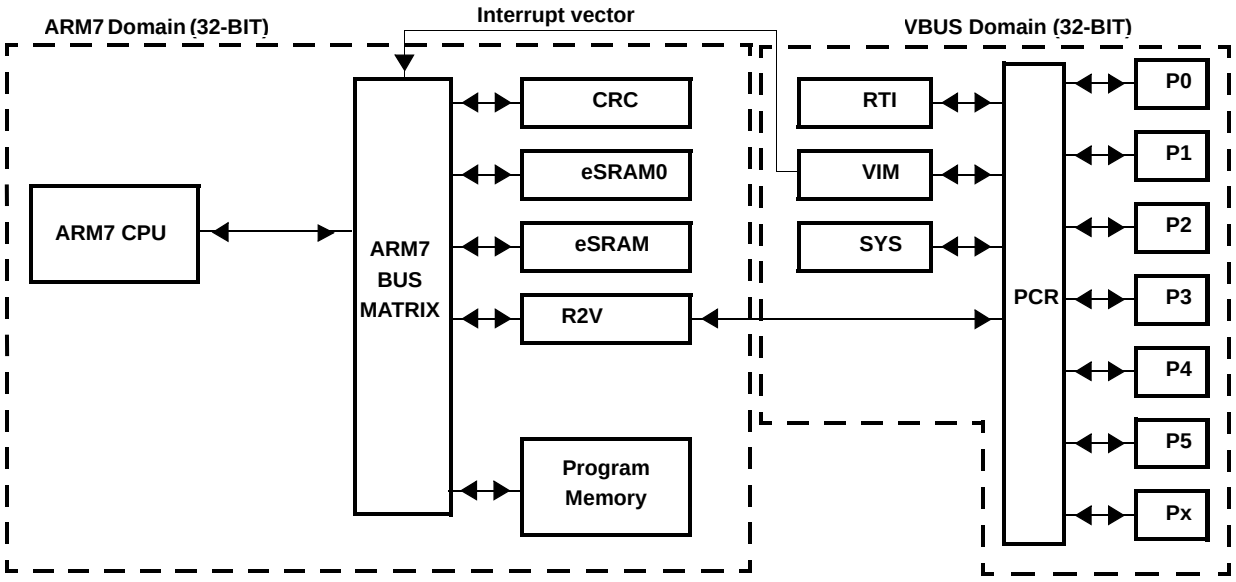
The SYSTEM module is accessible from the peripheral bus and contains all the registers needed for set up of the device (Clock ratios, PLL, etc.) as well as the status flags of the system (Exceptions sources, device ID, etc.) (see section 1.4).

The interrupts generated by the different sources are handled by the Vector Interrupt Module (VIM). The VIM allows backward compatibility with the TMS470Px family and allows new vectored interrupt capability to decrease the interrupt latency.

The TMS470Px family device incorporates a Real Time Interrupt module (RTI) specifically design to support time triggered operating systems and real time operating systems.

1.2.2 TMS470Px with ARM7TDMI

Figure 1-1. Block diagram with an ARM7TDMI



The Figure 1-1 shows a TMS470Px using an ARM7TDMI as the CPU. The CPU is connected to the slaves via a 32-bit bus. The BMM configuration only decodes the incoming address from the CPU. It also shortcuts access to the IRQ interrupt vector and FIQ interrupt vector, in order to improve the interrupt latency.

The R2V (VBUS bridge) in this configuration allows a 1:1 ratio between the CPU clock and the VBUS clock.

In a minimal configuration only 1 eSRAM is connected to the bus, the CRC is reduced to only 1 channel and the RTI allows only 1 time counter with 2 compares, the VIM is also reduced to 32 interrupt requests.

Note: ARM7 system bus
The ARM7TDMI system bus will be referenced, in this specification, as part of the AMBA definition, even tough it is not part of the AMBA specification. This is to simplify further explanation of the TMS470Px architecture.

1.3 Memory

1.3.1 Memory Mapping

When using internal program memory, the Memory mapping is divided into 3 main blocks, which are decoded in the bus matrix decoders:

- Internal program memory frame
  - eSRAM frame
  - CRC Frame
  - Peripheral Frame

The internal memory interface integrates a sub decoder to provide 1 internal chip-select of 16 MB.

The CRC frame also allows a 16 MB memory frame.

The peripheral bridge decodes up to 32 peripheral selects as well as 64 peripheral memory selects. It also decodes the address to access the system registers (Interrupt and RTI) and system memories.

Table 1-1. Memory mapping in normal mode with internal program memory

Frame Name	Start address	End address	Size in Bytes	Data Access
Internal Memory CS				
nCS0	0x0000 0000	0x00FF FFFF	16 MBytes	8/16/32/64
External Memory CS				
nCS1	0x0100 0000	0x01FF FFFF	16 MBytes	8/16/32/64
nCS2	0x0200 0000	0x02FF FFFF	16 MBytes	8/16/32/64
nCS3	0x0300 0000	0x03FF FFFF	16 MBytes	8/16/32/64
nCS4	0x0400 0000	0x04FF FFFF	16 MBytes	8/16/32/64
nCS5	0x0500 0000	0x05FF FFFF	16 MBytes	8/16/32/64
nCS6	0x0600 0000	0x06FF FFFF	16 MBytes	8/16/32/64
nCS7	0x0700 0000	0x07FF FFFF	16 MBytes	8/16/32/64
eSRAM				
CSRAM0	0x0800 0000	0x0BFF FFFF	64 MBytes	8/16/32/64
CSRAM1	0x0C00 0000	0x0FFF FFFF	64 MBytes	8/16/32/64
Reserved CSRAM2	0x1000 0000	0x13FF FFFF	64 MBytes	8/16/32/64
Reserved CSRAM3	0x1400 0000	0x17FF FFFF	64 MBytes	8/16/32/64
Reserved	0x1800 0000	0xFCFF FFFF		
DSP (HPI)				
HPI frame	0xFD00 0000	0xFDFF FFFF	16 MBytes	8/16/32
CRC				
CRC frame	0xFE00 0000	0xFEFF FFFF	16 MBytes	8/16/32/64
Peripheral Memory CS				
PCS0	0xFF00 0000	0xFF01 FFFF	128 KBytes	8/16/32
PCS1	0xFF02 0000	0xFF03 FFFF	128 KBytes	8/16/32
PCS2	0xFF04 0000	0xFF05 FFFF	128 KBytes	8/16/32

Frame Name	Start address	End address	Size in Bytes	Data Access
PCS3	0xFF06 0000	0xFF07 FFFF	128 KBytes	8/16/32
PCS4	0xFF08 0000	0xFF09 FFFF	128 KBytes	8/16/32
PCS5	0xFF0A 0000	0xFF0B FFFF	128 KBytes	8/16/32
PCS6	0xFF0C 0000	0xFF0D FFFF	128 KBytes	8/16/32
PCS7	0xFF0E 0000	0xFF0F FFFF	128 KBytes	8/16/32
PCS8	0xFF10 0000	0xFF11 FFFF	128 KBytes	8/16/32
PCS9	0xFF12 0000	0xFF13 FFFF	128 KBytes	8/16/32
PCS10	0xFF14 0000	0xFF15 FFFF	128 KBytes	8/16/32
PCS11	0xFF16 0000	0xFF17 FFFF	128 KBytes	8/16/32
PCS12	0xFF18 0000	0xFF19 FFFF	128 KBytes	8/16/32
PCS13	0xFF1A 0000	0xFF1B FFFF	128 KBytes	8/16/32
PCS14	0xFF1C 0000	0xFF1D FFFF	128 KBytes	8/16/32
PCS15	0xFF1E 0000	0xFF1F FFFF	128 KBytes	8/16/32
PCS16	0xFF20 0000	0xFF21 FFFF	128 KBytes	8/16/32
PCS17	0xFF22 0000	0xFF23 FFFF	128 KBytes	8/16/32
PCS18	0xFF24 0000	0xFF25 FFFF	128 KBytes	8/16/32
PCS19	0xFF26 0000	0xFF27 FFFF	128 KBytes	8/16/32
PCS20	0xFF28 0000	0xFF29 FFFF	128 KBytes	8/16/32
PCS21	0xFF2A 0000	0xFF2B FFFF	128 KBytes	8/16/32
PCS22	0xFF2C 0000	0xFF2D FFFF	128 KBytes	8/16/32
PCS23	0xFF2E 0000	0xFF2F FFFF	128 KBytes	8/16/32
PCS24	0xFF30 0000	0xFF31 FFFF	128 KBytes	8/16/32
PCS25	0xFF32 0000	0xFF33 FFFF	128 KBytes	8/16/32
PCS26	0xFF34 0000	0xFF35 FFFF	128 KBytes	8/16/32
PCS27	0xFF36 0000	0xFF37 FFFF	128 KBytes	8/16/32
PCS28	0xFF38 0000	0xFF39 FFFF	128 KBytes	8/16/32
PCS29	0xFF3A 0000	0xFF3B FFFF	128 KBytes	8/16/32
PCS30	0xFF3C 0000	0xFF3D FFFF	128 KBytes	8/16/32
PCS31	0xFF3E 0000	0xFF3F FFFF	128 KBytes	8/16/32
PCS32	0xFF40 0000	0xFF41 FFFF	128 KBytes	8/16/32
PCS33	0xFF42 0000	0xFF43 FFFF	128 KBytes	8/16/32
PCS34	0xFF44 0000	0xFF45 FFFF	128 KBytes	8/16/32
PCS35	0xFF46 0000	0xFF47 FFFF	128 KBytes	8/16/32
PCS36	0xFF48 0000	0xFF49 FFFF	128 KBytes	8/16/32
PCS37	0xFF4A 0000	0xFF4B FFFF	128 KBytes	8/16/32
PCS38	0xFF4C 0000	0xFF4D FFFF	128 KBytes	8/16/32
PCS39	0xFF4E 0000	0xFF4F FFFF	128 KBytes	8/16/32
PCS40	0xFF50 0000	0xFF51 FFFF	128 KBytes	8/16/32
PCS41	0xFF52 0000	0xFF53 FFFF	128 KBytes	8/16/32
PCS42	0xFF54 0000	0xFF55 FFFF	128 KBytes	8/16/32
PCS43	0xFF56 0000	0xFF57 FFFF	128 KBytes	8/16/32

Frame Name	Start address	End address	Size in Bytes	Data Access
PCS44	0xFF58 0000	0xFF59 FFFF	128 KBytes	8/16/32
PCS45	0xFF5A 0000	0xFF5B FFFF	128 KBytes	8/16/32
PCS46	0xFF5C 0000	0xFF5D FFFF	128 KBytes	8/16/32
PCS47	0xFF5E 0000	0xFF5F FFFF	128 KBytes	8/16/32
PCS48	0xFF60 0000	0xFF61 FFFF	128 KBytes	8/16/32
PCS49	0xFF62 0000	0xFF63 FFFF	128 KBytes	8/16/32
PCS50	0xFF64 0000	0xFF65 FFFF	128 KBytes	8/16/32
PCS51	0xFF66 0000	0xFF67 FFFF	128 KBytes	8/16/32
PCS52	0xFF68 0000	0xFF69 FFFF	128 KBytes	8/16/32
PCS53	0xFF6A 0000	0xFF6B FFFF	128 KBytes	8/16/32
PCS54	0xFF6C 0000	0xFF6E FFFF	128 KBytes	8/16/32
PCS55	0xFF6E 0000	0xFF6F FFFF	128 KBytes	8/16/32
PCS56	0xFF70 0000	0xFF71 FFFF	128 KBytes	8/16/32
PCS57	0xFF72 0000	0xFF73 FFFF	128 KBytes	8/16/32
PCS58	0xFF74 0000	0xFF75 FFFF	128 KBytes	8/16/32
PCS59	0xFF76 0000	0xFF77 FFFF	128 KBytes	8/16/32
PCS60	0xFF78 0000	0xFF79 FFFF	128 KBytes	8/16/32
PCS61	0xFF7A 0000	0xFF7B FFFF	128 KBytes	8/16/32
PCS62	0xFF7C 0000	0xFF7D FFFF	128 KBytes	8/16/32
PCS63	0xFF7E 0000	0xFF7F FFFF	128 KBytes	8/16/32

Peripheral Select				
Reserved	0xFF80 0000	0xFFF7 7FFF		
PS 31	0xFFF7 8000	0xFFF7 83FF	1 KBytes	8/16/32
PS 30	0xFFF7 8400	0xFFF7 87FF	1 KBytes	8/16/32
PS 29	0xFFF7 8800	0xFFF7 8BFF	1 KBytes	8/16/32
PS 28	0xFFF7 8C00	0xFFF7 8FFF	1 KBytes	8/16/32
PS 27	0xFFF7 9000	0xFFF7 93FF	1 KBytes	8/16/32
PS 26	0xFFF7 9400	0xFFF7 97FF	1 KBytes	8/16/32
PS 25	0xFFF7 9800	0xFFF7 9BFF	1 KBytes	8/16/32
PS 24	0xFFF7 9C00	0xFFF7 9FFF	1 KBytes	8/16/32
PS 23	0xFFF7 A000	0xFFF7 A3FF	1 KBytes	8/16/32
PS 22	0xFFF7 A400	0xFFF7 A7FF	1 KBytes	8/16/32
PS 21	0xFFF7 A800	0xFFF7 ABFF	1 KBytes	8/16/32
PS 20	0xFFF7 AC00	0xFFF7 AFFF	1 KBytes	8/16/32
PS 19	0xFFF7 B000	0xFFF7 B3FF	1 KBytes	8/16/32
PS 18	0xFFF7 B400	0xFFF7 B7FF	1 KBytes	8/16/32
PS 17	0xFFF7 B800	0xFFF7 BBFF	1 KBytes	8/16/32
PS 16	0xFFF7 BC00	0xFFF7 BFFF	1 KBytes	8/16/32
PS 15	0xFFF7 C000	0xFFF7 C3FF	1 KBytes	8/16/32
PS 14	0xFFF7 C400	0xFFF7 C7FF	1 KBytes	8/16/32

Frame Name	Start address	End address	Size in Bytes	Data Access
PS 13	0xFFF7 C800	0xFFF7 CBFF	1 KBytes	8/16/32
PS 12	0xFFF7 CC00	0xFFF7 CFFF	1 KBytes	8/16/32
PS 11	0xFFF7 D000	0xFFF7 D3FF	1 KBytes	8/16/32
PS 10	0xFFF7 D400	0xFFF7 D7FF	1 KBytes	8/16/32
PS 9	0xFFF7 D800	0xFFF7 DBFF	1 KBytes	8/16/32
PS 8	0xFFF7 DC00	0xFFF7 DFFF	1 KBytes	8/16/32
PS 7	0xFFF7 E000	0xFFF7 E3FF	1 KBytes	8/16/32
PS 6	0xFFF7 E400	0xFFF7 E7FF	1 KBytes	8/16/32
PS 5	0xFFF7 E800	0xFFF7 EBFF	1 KBytes	8/16/32
PS 4	0xFFF7 EC00	0xFFF7 EFFF	1 KBytes	8/16/32
PS 3	0xFFF7 F000	0xFFF7 F3FF	1 KBytes	8/16/32
PS 2	0xFFF7 F400	0xFFF7 F7FF	1 KBytes	8/16/32
PS 1	0xFFF7 F800	0xFFF7 FBFF	1 KBytes	8/16/32
PS 0 Quadrant 0 (See Note on Quadrants below)	0xFFF7 FC00	0xFFF7 FCFF	256 Bytes	8/16/32
PS 0 Quadrant 1 (See Note on Quadrants below)	0xFFF7 FD00	0xFFF7 FDFF	256 Bytes	8/16/32
PS 0 Quadrant 2 (See Note on Quadrants below)	0xFFF7 FE00	0xFFF7 FEFF	256 Bytes	8/16/32
PS 0 Quadrant 3 (See Note on Quadrants below)	0xFFF7 FF00	0xFFF7 FFFF	256 Bytes	8/16/32
SYSTEM Peripherals				
DMA RAM (optional)	0xFFF8 0000	0xFFF8 0FFF	4 KBytes	8/16/32
Reserved	0xFFF8 1000	0xFFFF 1FFF	4 KBytes	8/16/32
VIM RAM	0xFFF8 2000	0xFFF8 2FFF	4 KBytes	8/16/32
RTP RAM (optional)	0xFFF8 3000	0xFFF8 3FFF	4 KBytes	8/16/32
Reserved	0xFFF8 4000	0xFFF8 4FFF	4 KBytes	8/16/32
Reserved	0xFFF8 5000	0xFFF8 5FFF	4 KBytes	8/16/32
Reserved	0xFFF8 6000	0xFFF8 6FFF	4 KBytes	8/16/32
ROM wrapper	0xFFF8 7000	0xFFF8 7FFF	4 KBytes	8/16/32
Reserved	0xFFF8 8000	0xFFF8 8FFF	4 KBytes	8/16/32
Reserved	0xFFF8 9000	0xFFF8 9FFF	4 KBytes	8/16/32
Reserved	0xFFF8 A000	0xFFF8 AFFF	4 KBytes	8/16/32
Reserved	0xFFF8 B000	0xFFF8 BFFF	4 KBytes	8/16/32
Reserved	0xFFF8 C000	0xFFF8 CFFF	4 KBytes	8/16/32
Reserved	0xFFF8 D000	0xFFF8 DFFF	4 KBytes	8/16/32
Reserved	0xFFF8 E000	0xFFF8 EFFF	4 KBytes	8/16/32
Reserved	0xFFF8 F000	0xFFF8 FFFF	4 KBytes	8/16/32
Reserved	0xFFF9 0000	0xFFFE FFFF		
Reserved	0xFFFF 0000	0xFFFF 7FFF	32 KBytes	8/16/32
Reserved	0xFFFF 8000	0xFFFF DFFF		
PCR Registers	0xFFFF E000	0xFFFF E0FF	256 Bytes	8/16/32

Frame Name	Start address	End address	Size in Bytes	Data Access
Reserved	0xFFFF E100	0xFFFF E1FF	256 Bytes	8/16/32
Reserved	0xFFFF E200	0xFFFF E2FF	256 Bytes	8/16/32
Reserved	0xFFFF E300	0xFFFF E3FF	256 Bytes	8/16/32
Reserved	0xFFFF E400	0xFFFF E4FF	256 Bytes	8/16/32
Reserved	0xFFFF E500	0xFFFF E5FF	256 Bytes	8/16/32
Reserved	0xFFFF E600	0xFFFF E6FF	256 Bytes	8/16/32
Reserved	0xFFFF E700	0xFFFF E7FF	256 Bytes	8/16/32
Reserved	0xFFFF E800	0xFFFF E8FF	256 Bytes	8/16/32
Reserved	0xFFFF E900	0xFFFF E9FF	256 Bytes	8/16/32
Reserved	0xFFFF EA00	0xFFFF EAFF	256 Bytes	8/16/32
Reserved	0xFFFF EB00	0xFFFF EBFF	256 Bytes	8/16/32
Reserved	0xFFFF EC00	0xFFFF ECFF	256 Bytes	8/16/32
Reserved	0xFFFF ED00	0xFFFF EDFF	256 Bytes	8/16/32
Reserved	0xFFFF EE00	0xFFFF EEFF	256 Bytes	8/16/32
Reserved	0xFFFF EF00	0xFFFF EFFF	256 Bytes	8/16/32
DMA registers (optional)	0xFFFF F000	0xFFFF F3FF	1 KBytes	8/16/32
Reserved	0xFFFF F400	0xFFFF F4FF	256 Bytes	8/16/32
Reserved	0xFFFF F500	0xFFFF F5FF	256 Bytes	8/16/32
Reserved	0xFFFF F600	0xFFFF F6FF	256 Bytes	8/16/32
DMM registers (optional)	0xFFFF F700	0xFFFF F7FF	256 Bytes	8/16/32
EMIF registers (optional)	0xFFFF F800	0xFFFF F8FF	256 Bytes	8/16/32
ECC registers (optional)	0xFFFF F900	0xFFFF F9FF	256 Bytes	8/16/32
RTP registers (optional)	0xFFFF FA00	0xFFFF FAFF	256 Bytes	8/16/32
POM registers (optional)	0xFFFF FB00	0xFFFF FBFF	256 Bytes	8/16/32
RTI	0xFFFF FC00	0xFFFF FCFF	256 Bytes	8/16/32
VIM Parity registers (optional)	0xFFFF FD00	0xFFFF FDFF	256 Bytes	8/16/32
VIM registers	0xFFFF FE00	0xFFFF FEFF	256 Bytes	8/16/32
System module registers	0xFFFF FF00	0xFFFF FFFF	256 Bytes	8/16/32

Note: Quadrants
For Clarity, the previous table does not show the quadrants of all the Peripheral Selects (PS). PS 1 to PS 31 also have 4 quadrants of 256 bytes as documented for PS 0.

1.3.2 Boot memory

1.3.2.1 Device with internal program memory

At boot time, the CPU will fetched starting at 0x0000 0000. By default, The boot memory is the nCS0.

It is also possible to swap the nCS0 with nCSRAM0 by the using the RAM overlay feature by setting the MEMSW bit within the System registers (BMMCR1, see section 5.1.31). In this event, the nCS0 will then be located starting at 0x0800 0000 and the nCSRAM0 will be located in 0x0000 0000.

1.3.3 Endianess

The TMS470Px runs in BIG ENDIAN mode, the following rules apply in BIG endian with an ARM7 CPU.

Table 1-2. Active byte lanes for a 32 bits data bus

Transfer size	Offset Address	DATA 31:24	DATA 23:16	DATA 15:8	DATA 7:0
Word	0	✓	✓	✓	✓
Half-Word	0	✓	✓		
Half-Word	2			✓	✓
Byte	0	✓			
Byte	1		✓		
Byte	2			✓	
Byte	3				✓

1.4 System module (SYS)

The system module controls device operations such as:

- Exception generation
- Clock control functions
- Device identification
- Global configuration registers.
- Software interrupt registers (4 interrupts)
- Reset generation

The system module logs exception events in the exception status registers.

The system module contains all the configuration register for the following modules:

- Global Clock module and PLL
- Bus matrix (priority programming and memory swapping)
- VBUS bridge (bus error logging and clock ratio)
- eSRAM wait-state programming
- External Clock Prescaler (ECP)
- Voltage regulator

1.5 Clock Definition

1.5.1 Definitions

A *primary clock source* is an independent clock source generating a clock independent of other circuitry. Typical examples of primary clock sources are internal oscillators that drive an external crystal/resonator, path for driven external clock, or internal free-running ring oscillator.

A *secondary clock source* is a source whose clocking depends upon the primary clock source. Examples of secondary clock sources are PLL and FLL outputs

An *Initiator* is a module which is able to apply a request onto a bus.

A *target* is the recipient of a request on a bus.

A *central resource* defines a bus interconnect.

A *bridge* defines a bus domain separation.

1.5.2 Clock domain

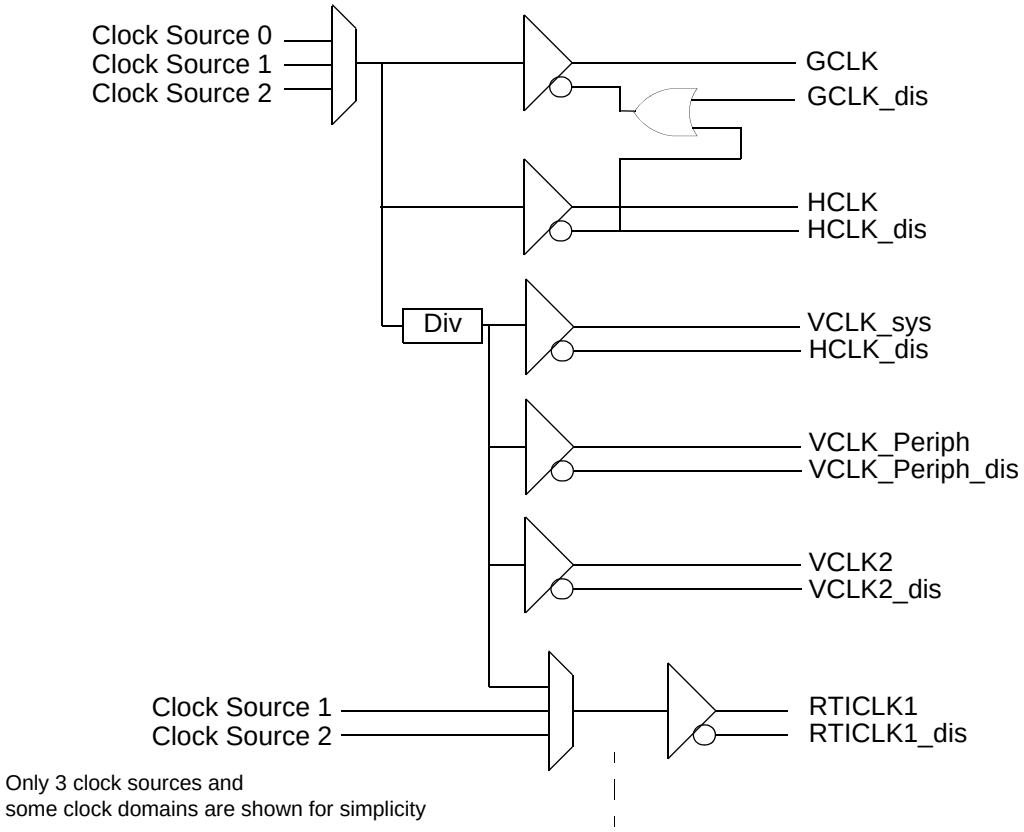
The TMS470Px device is divided into 6 different clock domains:

- CPU clock domain
- System bus clock domain
- System peripheral clock domain
- Peripheral clock domain
- Primary Real Time Interrupt clock domain
- Secondary Real Time Interrupt clock domain

All clocks are generated by the Global Clock module (GCM).

1.5.2.1 GCM Block diagram

Figure 1-2. GCM block diagram



1.5.2.2 CPU clock domain

The CPU clock domain is clocked by GCLK. GCLK controls all the CPU sub-systems including the MPU.

GCLK is generated from the Clock Source 0(e.g. OSCIN).

1.5.2.3 System Bus clock domain

The System Bus clock domain consists of all the modules attached to the CPU bus. It is clocked by HCLK. HCLK controls all of the peripherals attached to the system bus.

HCLK is can be generated from the same clock source as the GCLK. The ratio between GCLK and HCLK is always 1.

1.5.2.4 System peripheral clock domain

The system peripheral clock domain is clocked by VCLK_sys.

VCLK_sys is the primary clock for the VBUS clock domain. It is used for all the VBUS bus transactions as well as all of the system peripherals (System, VIM, etc.) with the exception of the RTI.

VCLK_sys is derived from HCLK, the ratio between HCLK and VCLK_sys is programmable from 1[1] to 16 by the VCLKR field in the Clock Control Register (see section 5.1.34) and the default ratio of the VCLKR is 1:2.

HCLK and VCLK_sys share the same enable bit.

1. This is device specific. Refer to the device datasheet.

1.5.2.5 Peripheral clock domain

The peripheral clock domain is clocked by VCLK_periph and VCLK2.

VCLK_periph is the primary clock for the peripherals. VCLK_periph is synchronous with VCLK_sys (ratio 1:1) but can be shut-down separately.

VCLK2 is a second VBUS clock domain that is used by some peripherals to run faster than VCLK_periph.

VCLK2 can be equal or less than HCLK and greater or equal than VCLK.

$$HCLK \geq VCLK2 \geq VCLK\_sys$$

The ration between HCLK and VCLK2 is programmable from 1 to 16 and the default ratio value is 1:1. The GCM only supports integer ratio between VCLK_sys, VCLK2 and HCLK.

For example:

- VCLK2 ratio is equal to 2 and VCLK_sys ratio is equal to 4 is a valid ratio
- VCLK2 ratio is equal to 2 and VCLK_sys ratio is equal to 3 is not a valid ratio

1.5.2.6 Real Time interrupt clock domain

The RTI module is clocked by RTICLK1 (primary). The RTI time base (RTICLK1) could be either VCLK_sys or OSCIN and is programmed in the system module.

RTICLK1 uses the same ratio as VCLK_sys in normal mode. When the device enters stand-by mode and the PLL is shut-down then the RTICLK1 is switched to OSCIN if the RTI time base is OSCIN.

Note:
When the RTI clock domain is mapped to any clock source other than VCLK, the frequency of the clock input to the RTI module must be at least 3 times slower than the VCLK frequency. This can be managed by configuring the RTI1DIV field of the RCLKSRC register. The RTICLK1 divider only applies when the RTICLKSRC is mapped to a source other than VCLK.

1.5.3 ECP

1.5.3.1 Overview

The ECP allows the device to output a continuous external clock on an I/O pin. The external clock (ECLK) frequency is a user-programmable ratio of the interface clock (VCLK_sys) frequency.

The VBUS clock (VCLK) input is divided by the user-selected value programmed into the ECPCNTL register (see section 5.1.35). The ECLK output is controlled by the ECP functional register bit (SYSPC1, see section 5.1.1), and is output on the external I/O pin of the device.

The ECPDIV provides programmable prescale for generating ECLK from VCLK.

$$ECLK = \frac{VCLK}{(ECPDIV + 1)}$$

1.5.3.2 ECP Prescaler

The ECP prescaler generates the ECLK signal by dividing VCLK to a lower frequency; the prescale is implemented as a digital counter, programmable via the ECPCNTL register (ECPDIV.15:0, see section 5.1.35).

The prescale divider may be changed while the ECP module is enabled. However, in order to assure a smooth transition between output frequencies, the new prescale value does not take effect until the current ECLK period is complete.

The prescaler can be programmed with divider values ranging from a minimum of 1 to a maximum of 65536. The divider values are programmed into the ECPDIV.15:0 bits in the ECPCNTL register (ECPDIV.15:0, see section 5.1.35). When the ECPDIV.15:0 bits are 0x07, the prescaler is set to divide by 8. The default state after reset is divide by 1.

Note:
The duty cycles of the ECP clock is targeted to be 50% by design. However, depending on the I/O used and the load, a deviation might be observed.

1.5.3.3 ECP Enable

The ECLKFUN bit controls the external I/O pin function of the device, as shown in section 1-3.

Table 1-3. Settings for ECLKFUN

ECLKFUN	ECLKDIR	External I/O Pin Function
0	0	GIO input
0	1	GIO output
1	x	ECLK output

The multiplexer output is controlled via the ECLKFUN bit in the SYSPC1 register. If the ECLKFUN bit is set, the multiplexer outputs ECLK on the external I/O pin of the device. If ECLKFUN is cleared, the multiplexer allows control of the external pin by the GIO control registers (SYSPC2 to SYSPC9 register, see section 5-12 to section 5-19)

1.5.3.4 Powerdown

When the device goes into powerdown, the VCLK_sys input is disabled. Therefore, the ECLK output is also disabled. After waking up from low-power, the ECP returns to its previous state, and removing the need to reconfigure the ECPCTRL register after the ECP wakes up from VCLK_sys off.

1.5.3.5 Suspend

When the ECP is in suspend mode, the ECP control register can still be modified. This allows you to modify the control register when a JTAG emulator/debugger tool is being used.

When the ECPCOS bit (ECPCNTL.23) is set, the ECP continues to output its ECLK if the ECLKFUN bit is set when the ECP is in suspend mode.

1.5.4 Clock Domains and Low Power Modes

The TMS470Px clock domains can be disabled independently to provide a better granularity in power consumption.

1.5.4.1 Low Power Mode hierarchy

All modules follow a defined clock suspend/wakeup interface. That is, for each clock used by a target, the target has a handshaking mechanism. When all targets on a specific clock domain have acknowledged a request to stop the clock, then the entire clock domain is disabled.

Note:
An initiator access will not cause the entire clock domain to exit low power mode. If the initiator accesses a target whose clock is stopped, the clock is restarted *for the accessed target* without awakening all other targets in that clock domain. After the access is complete, the clock to the target is stopped and the target re-enters low power mode.
It is recommended that a wakeup should occur by interrupt/wakeup circuitry or by the CPU clearing the clock domain disable bit rather than by an initiator access to a target whose clock is stopped.

1. When all dependent clock domains are disabled, secondary sources may be disabled.
2. When all dependent clock domains and dependent secondary sources are disabled, primary sources may be disabled.

The clock domains and sources are awakened in the reverse order.

1. The primary source is awakened.
2. The dependent secondary sources are awakened.
3. The dependent clock domains are awakened.
4. The entire clock domain is awakened.

1.5.4.2 Clock Tree Disable

Setting the clock domain disable bit for a specific clock domain (see section 5.1.13 to section 5.1.15) activates the clock stop handshaking for all targets on the clock domain. When all target acknowledge, then the clock domain is disabled at the source. Every clock domain has an independent clock domain disable bit with the exception of VBUS_sys which shares a clock domain disable with HCLK.

1.5.4.3 Clock selection

When the CPU enters/exits low power mode, it is possible to automatically change the clock source for HCLK/VCLK/VCLK2 domains. This flexibility is provided so that the user can choose the clocks active while the CPU is inactive and the clock active on CPU wakeup (power vs. performance trade-off) (see section 5.1.16).

Similarly, other clock selection muxes allow the user to set the source for the respective clock domains (see section 5.1.16 to section 5.1.18).

All muxed clock domains (RTICLK) default to VCLK_sys as the source.

1.5.5 Primary/Secondary clock source control

The ability to change source clocks for clock domains must be accompanied by the ability to disable unused clock sources. A clock source cannot be disabled while a clock domain using the clock source is active (see section 5.1.19); in the case of a primary clock source, the clock must not be used by any clock domain or secondary source. After this condition is met, the clock may be disabled. After being disabled, if the clock is required, then it must be restarted (see section 5.1.10 to section 5.1.12.).

Note:
It is not possible to switch a clock domain to a non-existent clock source or a clock source that is not ready.

1.6 Low Power modes

TMS470Px devices support multiple low power modes. These different modes allow the user to trade-off the amount of current during low power mode versus functionality and wake-up time. Which actual modes are supported on a specific TMS470Px Device is documented within the specific TMS470Px Device datasheet.

The following sections discuss the different low power modes supported by TMS470Px devices. Modes are defined in order from highest current/fastest wakeup time to lowest current/slowest wakeup time.

All TMS470Px Low power modes have the following common characteristics and these conditions must be met in order for the device to be considered to be in a Low Power Mode.

- CPU and System clocks are disabled.
- PLL is disabled.
- Flash banks and pump are in sleep mode.
- All Peripheral modules are in low power modes and clocks are disabled. (Exceptions to this may occur and would be documented in the specific TMS470Px device datasheet).

Because the TMS470Px Architecture is defined with ultimate flexibility of enabling and disabling clocks, there are more possible modes than what are defined below. However, the below modes are the only modes actually characterized or specified for any specific TMS470Px Device. Please consult the specific TMS470Px Device datasheet for the actual modes supported by a specific device.

1.6.1 Active Clocks Modes

Active Clocks Mode is defined as any low power mode which has at least one clock still running. This allows the user to wakeup much quicker than low power modes with no clock running.

All Standby modes support the following wakeup (return to normal operation) methods:

- Interrupt or Wakeup signal from the RTI Module (RTI Clock must still be enabled).
- Any external GIO Interrupt defined as a wakeup interrupt.
- CAN Message
- C2SI Message
- SCI or LIN Message
- I2C Message
- External Wakeup Signal (see section External Wakeup Signal description later in this document).
- Power On or System Reset

1.6.1.1 Doze Mode

Doze mode is defined as follows:

- The Oscillator is enabled.
- The RTI Clock can be either enabled or disabled (depends on user setting).
- If device has an internal Voltage Regulator, it must be put in Halt (LPM1) Mode.
- Since the Oscillator is still enabled in Doze mode, the device will wakeup much faster and start executing instructions in the fewest amount of cycles of any low power mode. Doze mode will therefore also have the highest current of any of the low power modes.

In order to enter Doze mode the following steps are required by the user software:

1. **Software writes to the Flash Control Registers to put all Flash Banks into sleep mode.**
   Please consult the device datasheet to determine which Flash module is on the device. Flash Banks will not be disabled until all accesses to the Flash have stopped.
2. **Software writes to the Flash Control Registers to put all Flash Pumps into sleep mode.**
   Pumps will not be disabled until all Flash Banks have been put into sleep mode.

3. **Software writes to Voltage Regulator Control Register (VRCTL) to put the Voltage regulator in Halt (LPM1) Mode.**
   The voltage regulator will not be put into Halt (LPM1) Mode until all the clock domains assert their clock stop acknowledge signals. This only applies to devices with an internal voltage regulator. Consult the device specific TMS470Px Datasheet.
4. **Software writes to the Clock Source Disable Register (CSDIS) to disable the PLL Clock Source.** Other sources may also be powered off, but the Oscillator must still be enabled to be considered to be in Doze Mode. The source will not actually be disabled until the clock domains which use them are disabled (see section 1.5.2).
5. **Software writes to the Clock Domain Disable Register (CDDIS) to disable the GCLK (CPU Clock), HCLK (System Clock), VCLKP (Peripheral Vbus Clock), VCLK2 (Peripheral Vbus Clock2).**
   All these domains must be disabled in order to be considered in Doze mode. The domain is not actually disabled until all modules using that domain provide their clock stop acknowledgement. The RTICLK1 domain may or may not be disabled. Doze mode can be used in conjunction with an RTI in order to wakeup the device periodically. Some exceptions on which domains must be disabled in Doze mode may exist which would be defined in the device specific TMS470Px Datasheet.
6. **Software Executes an ARM instruction with an IDLE cycle which triggers the CPU clock to stop.**

The device will return to normal operation after a wakeup interrupt from Doze mode from one of the sources mentioned in section 1.6.1. If the PLL is use to return to normal operation, this can cause a longer delay since the PLL needs to re-lock. The PLL can be re-locked after waking up from Doze mode in parallel with doing other operations.

1.6.2 Inactive Clocks Modes

An *Inactive Clocks Mode* is defined as any low power mode in which all clock sources are off. Wakeup will be slower in this mode. Inactive clock mode requires an external wakeup or power-on reset to exit.

Inactive clock modes can wake up on the following sources:

- Any external GIO Interrupt.
- CAN Message
- SCI or LIN Message
- I2C Message
- External Wakeup Signal (see External Wakeup Signal description later in this document).
- Power-On or System Reset

1.6.2.1 Sleep Mode

In Sleep mode all circuits are still powered, but all clock sources are disabled as shown below:

- PLL is disabled.
- OSC is disabled.
- RTI Clock is disabled.
- If device has an internal Voltage Regulator, it must be put in Halt (LPM1) Mode

Since all clocks are disabled in Sleep mode the current consumption of the device is limited to normal leakage currents. Since there are no internal clocks, an external event is required to wakeup from Sleep mode.

In order to enter Sleep mode the following steps are required by the user software. The steps are the same as for Doze mode except ALL clock sources and domains must be disabled. If any clock is still running the device is not considered to be in Sleep mode.

1. **Software writes to the Flash Control Registers to put all Flash Banks into sleep mode.**
   Please consult the device datasheet to determine which Flash module is on the device. Flash Banks will not be disabled until all accesses to the Flash have stopped.
2. **Software writes to the Flash Control Registers to put all Flash Pumps into sleep mode.**
   Pumps will not be disabled until all Flash Banks have been put into sleep mode.
3. **Software writes to Voltage Regulator Control Register (VRCTL) to put the Voltage regulator in Halt (LPM1) mode.**
   The voltage regulator will not be put into Halt (LPM1) mode until all the clock domains provide their clock stop acknowledgment. Consult the device specific TMS470Px Datasheet.
4. **Software writes to the Clock Source Disable Register (CSDIS) to disable all clock sources.** The PLL and Oscillator sources will not actually be disabled until the clock domains which use them are disabled (see section 1.5.4).
5. **Software writes to the Clock Domain Disable Register (CDDIS) to disable all clock domains.**
   All these domains must be disabled in order to be considered in Sleep mode. The domain is not actually disabled until all modules using that domain provide their clock stop acknowledgement.
6. **Software Executes an ARM instruction with an IDLE cycle which triggers the CPU clock to stop.**

The device will return to normal operation after an external wakeup from one of the wakeup sources mentioned in section 1.6.1 above. The clock source used for a given clock domain after wakeup is user programmable (see section 1.5.4). The Oscillator will have to startup before the clocks will be released internally. If the PLL is to be used after a return to normal operation, this can cause a longer delay since the PLL needs to re-lock. The PLL can be re-locked after waking up from Sleep mode in parallel with doing other operations.

1.6.3 External Wake Up Signals

External Wakeup signals are used to wakeup the device from any of the TMS470Px Low Power modes mentioned above.

> **Note:**
> It is possible to configure each wakeup source to also interrupt the CPU. However, it is not advisable to enable both the wakeup and interrupt functionality for the same source. If both functionalities are enabled, care must be taken to handle both the wakeup and the interrupt conditions. If either is left unserviced, the device will not return to low power mode due to one of the conditions being left active.

External Wake Up signals can be routed to different pins on the device. Depending on the device, up to 8 External Wakeup signals can be supported. The actual pins to which the External Wakeup Signals are connected is device specific and is described in the device datasheet.

In order to use an External Wakeup source, the corresponding Wakeup Source bit must be enabled in the External Wakeup Enable register (EXTWAKENR). When a specific Wakeup Signal is asserted a flag will be stored in the External Wakeup Status register (EXTWAKESR).

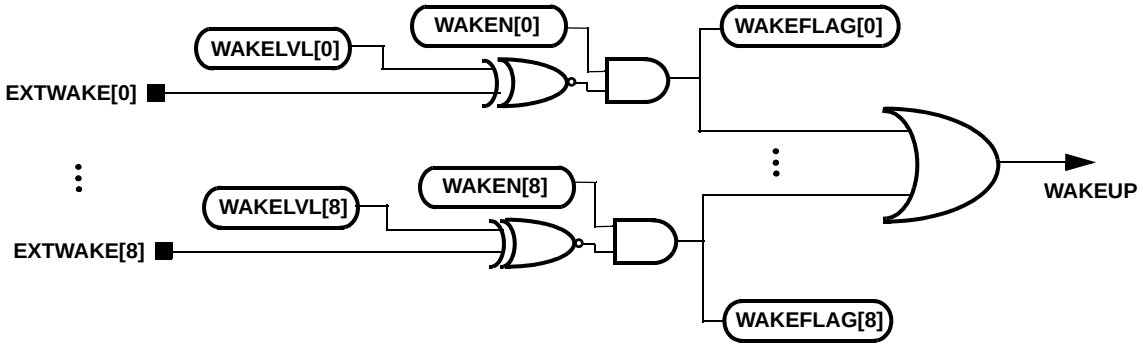
Wakeup levels are programmed in the Wakeup Level register (WAKELVR). Figure 1-3 shows a functional diagram of the External Wakeup Control Logic.The External Wakeup signals can be programmed to be either active high or active low, but default to active low after a Power-On Reset.

External Wakeup Signals do not generate a CPU interrupt like other wakeup sources. An External Wakeup will cause the CPU to restart execution at the next PC (Program Counter) address prior to entering the Low Power mode.

Using External Wakeup signals, it is possible to wakeup from RAM without waking up the Flash. In this case, if the device does not need to wakeup completely, power will be conserved during normal operation since the Flash was never powered up.

The vectored interrupt manager module on TMS470Px devices allows the interrupt to be masked to the CPU, but still generate a wakeup. The user needs to determine if an interrupt is generated or not for other wakeup sources. If not, then the CPU will start execution at the next instruction after the entry point of Low Power Mode. If an interrupt is enabled, the CPU will jump to the Interrupt Service Routine (ISR) directly.

Figure 1-3. External Wakeup Control Logic



1.7 Reset

A TMS470Px device reset can be caused by any of the conditions listed in Table 1-4

Table 1-4. Causes of TMS470Px device resets.

Condition	Description
External cold nPORRST pin	This is an active LOW power up reset signal.
External warm RESET pin	This is an active LOW warm reset generated by TMS470Px system or by another external device
Watchdog reset	A watchdog (WD) reset occurs when either the WD timer times out, or a wrong key is written to the register.
Reset on oscillator fail	If it is determined that the external reference input, either the crystal or the oscillator, is stuck-low, stuck-high, or running at a frequency too slow as determined by an internal reference oscillator, an oscillator fail reset occurs
Software reset	Writing an specific pattern to the software reset bits causes a system to reset.
Illegal transaction reset	An Illegal transaction reset is generated when an illegal transaction is detected by the slave on a masked imprecise transaction in Privileged Mode. An illegal transaction reset is also generated when a system interrupt generated on an illegal masked imprecise transaction is not serviced before a hardware counter expires in User Mode. Please see Table 1-6, *Illegal transaction detection* for the causes of illegal transaction reset
Reset due to wakeup of voltage regulator from sleep mode	For devices with an on chip voltage regulator a reset is generated when the voltage regulator is woken up from sleep mode of if the voltage supplied by the voltage regulator surpasses the voltage thresholds for low or high voltages.

All the different reset conditions are flagged within the system exception register with the exception of a reset due to a wakeup of the voltage regulator from sleep mode or a reset due to an identified oscillator fail condition.

The reset logic holds the device in a reset state for at least eight VCLK cycles. During a reset, while the reset logic holds the device in the reset state, the external RESET signal is driven LOW. This occurs whether the reset is caused by an internal exception or an external exception.

1.8 Illegal Transactions

An illegal transaction is defined as an access to a non-implemented address or to a protected address. Illegal transactions can be classified as follows:

- Illegal Address (non-mapped address)
- Illegal Access (protection violation)

1.8.1 Illegal Addresses

An illegal address is defined as an access to a non-implemented slave address on the System bus or to an undefined peripheral address on the VBUS. In the event of an illegal address detection the slave will generate an error response to the CPU resulting in a system reset, data abort exception or system error interrupt dependent on the type of access and whether the mode is set to user or privileged mode. The definition of a non-implemented address is described in Table 1-5.

Table 1-5. Definition of a non-implemented address

Slave memory frame	Definition of non-implemented address (Illegal address)
eSRAMx	An access to a non-implemented address in the eSRAM frame is defined as an illegal address. The eSRAM wrapper generates an error response.
Reserved RAM frame	If the reserved RAM frame (such as nCSRAM2 & nCSRAM3) is accessed an error response is generated. If the RAM frame is implemented, the eSRAM wrapper generates an error response for any access to the non-implemented RAM banks.
Reserved Memory	For any access to the reserved memory the BMM generates an error response.
CRC slave	For any access to a non-implemented address in the CRC slave frame, the CRC module generates an error response.
Peripheral memory chip select (PCS) frame	Access to the non-implemented peripheral memory selects generate an error response.
Peripheral select (PS) frame	Each peripheral select frame contains 4 quadrants of 256 bytes. An error response is generated for any access to the non-implemented peripheral select quadrants.

There are two ways of detecting an illegal address to these slaves

1. Program the level descriptors in an MPU, if it is available within the device, such that a translation fault is generated when trying to access the non-implemented memory on the system bus. This is taken care by the ARM MPU before generating the physical address to the system bus.
2. When the system/VBUS address bus accesses a non-implemented region in the slave, it generates an error response either through a system reset or through a system interrupt. It is the responsibility of the slaves to generate a response based on the type of transaction.

1.8.2 Illegal Accesses

Accesses to protected memory on the system bus and the peripheral registers or memory frames on the VBUS are defined as illegal accesses.

1. Memory Access Protection:
   Protection for the memory accesses are set with the CPU's MPU unit. Access permissions for the memory on the system bus is configured by the MPU through permission faults. A memory access violation is

logged as a permission fault in the CPU's Fault Status Register and the virtual address of the access is logged into the CPU's Fault Address Register.

2. Peripheral register/Peripheral Memory Access Right Violation:
   The device peripheral accesses can be protected either through the
   MPU by programming the level descriptors to generate permission faults or through peripheral memory protection register and peripheral protection register in the pcr module.

   a. *Peripheral Register Access Violation:*
      The PCR module registers, PPROTSETx/PPROTSETx, contain one protection bit per peripheral register quadrant. The protection bits define the access rights to the peripherals on the VBUS. When the CPU attempts to write to any peripheral register frame for which it does not have the correct permission rights, a protection violation is detected, and an error response is generated. In addition, peripherals on the VBUS define certain register bits which are not writable in certain operating modes (refer to module register specifications, bit marked as RWP - read in all modes, write in privilege modes only). When the CPU attempts to write to these peripheral register bits (for which it does not have the correct permission rights), the write operation is simply ignored and no memory access violation is detected.

   b. *Peripheral Memory Access Violation:*
      The PCR module registers, PMPROTSETx/PMPROTCLRx, have one bit per peripheral memory frame. The protection bits define the access rights to the peripheral memory frames on the VBUS. When the CPU attempts to write to any peripheral memory frame for which it does not have the correct permission rights, a protection right access is detected, and an error response is generated.

1.8.3 Illegal Transaction Detection and Response

Almost all illegal transactions provide an error response to the initiator except in the following cases:

- Writes are buffered in the VBUS bridge. When performing writes to the VBUS peripherals, the VBUS gives an OKAY response to the initiator before the write transaction error is detected. The VBUS bridge is responsible for providing the system module the address, system mode (user/privileged), and the initiator ID of the illegal transaction.

The following is a summary of the TMS470Px system responses on an illegal transaction detection:

Table 1-6. Illegal transaction detection

Access Type	System mode	Illegal transaction response
1)TMS470Px System behavior on Error response to CPU transactions		
CPU accesses	User/Privileged	The CPU generates an external abort for an error response from the slave. The address of the access and the abort status are logged in the MPU Fault Address Register (FAR) and the MPU Fault Status Register (FSR) respectively.
3) Illegal transactions on which the Initiator get an Okay response		
VBUS writes	The address of an illegal transaction on the VBUS is logged in the PILLADDR register of the system module (see section 5.1.26)	
	Privileged	System reset is generated.
	User	System error interrupt is generated.

1.8.3.1 System reset when System interrupt on illegal transaction is not detected

See Section 1.9.1, *"System Error interrupt"* .

1.9 System Interrupts

The following sections describe different system interrupts defined in this architecture.

1.9.1 System Error interrupt

The system module flags any illegal transactions and generates an interrupt to the CPU (see Table 1-6, *Illegal transaction detection*). The error flags are contained in the Interrupt Flag Register (ITIFLAG, see section 5.1.38). An interrupt is serviced by reading from the ITIFLAG register. This is to assure that an illegal transaction generates an exception when the system mode is set to user mode.

A system reset is generated if the interrupt due to an illegal transaction is not serviced within 14000 HCLK cycles. The hardware down counter is triggered (loaded with the initial value) when an interrupt due to an illegal transaction is generated. The counter is disabled when the interrupt due to an illegal transaction is serviced.

Note:
For devices with ECC, a double bit error interrupt will trigger the counter as for an illegal transaction interrupt.

Note:
Multiple errors will not reload the counter as long as the interrupt flag has not been cleared by the software.

1.9.2 System Software interrupt (SSI)

A system software interrupt is generated by writing the correct key value to any of the SSIRx (x=1,2,3,4) registers. While writing the key in the SSIRx register, the user can write an 8-bit value in the register. This value will be then be provided to the ISR when reading the SSIVEC register. The SSIRx registers share the same interrupt request line (see section 5.1.43).

The source of the system software interrupt is available in the System Software Interrupt Vector Register (SSIVEC, see section 5.1.41).

1.10 Emulation and debug

1.10.1 Debug Interface

The TMS470Px provides a JTAG interface in order to connect an external debugger. The TAP interface is based on the IEEE1149.1 and it is used for debugging purposes.

The JTAG port commands only the CPUs TAP port and is defined as follows:

- **TCK** Test Clock
- **TDI** Test Data In
- **TDO** Test Data Out
- **nTRST** Test reset active low
- **TMS1** Test select 1 (ARM CPU)
- **TMS2** Teat select 2 (Second CPU if applicable)
- **RTCK** Return TCK

Note:

1. The TMS470Px does not provide a boundary scan nor a TAP controller for the logic around the CPU.
2. In debug mode, all registers can be updated by the debugger independent of the CPU mode.
3. On ARM7TDMI CPU, the JTAG pins are directly connected to the CPU boundary. In this case, **TCK** is directly looped back to **RTCK**

1.11 Memory Module Hardware Initialization

The TMS470Px system provides the capability to perform an hardware initialization on all modules with memories on the system bus and on the peripheral bus (VBUS). The intent of having the hardware initialization is to program the memory arrays with error detection capability to a known state based on their error detection scheme parity (odd/even) or ECC. An example would be after nPORRST reset, the content of the memory arrays would be unknown and on triggering hardware initialization counter it would program the memory arrays to the corresponding values based on the error checking scheme. Another example would be triggering off the hardware initialization logic after a memory self-test run, because the arrays will be programmed to all ones after the run.

Note: Initialization of Parity or ECC

For initialization of parity/ECC memory, the parity/ECC feature must be enabled in the module first. Refer to the device specific datasheet to determine if ECC or parity is included in the device.

1.11.1 Memory Module Hardware Initialization Features

- Supports up to 32 memory modules on the system bus and the peripheral bus.
- Can run all the memory module initialization in parallel for similar RAMs
- Can run each memory module initialization individually
- Captures memory module initialization completion results in status registers

AMBA Interconnect

This chapter describes the AMBA interconnect of the Texas Instruments TMS470Px family.

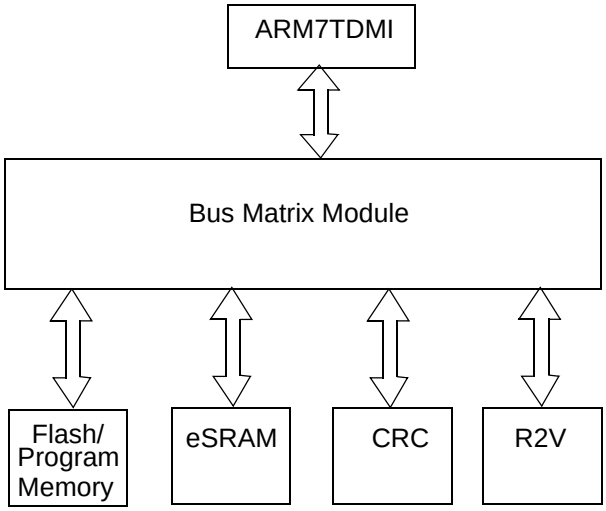
Topic **Page**

2.1 Bus Matrix Module (BMM)

The BMM provides connectivity between different bus slave modules to different bus master modules. Accesses from different master modules are executed in parallel if no resource conflict occurs or if the master modules are kept in series through arbitration.

The bus matrix module (BMM) is illustrated in Figure 2-1.

Figure 2-1. Bus Matrix Block Diagram



The bus matrix provides high bus bandwidth to each slave by providing a dedicated bus to each slave. The functions of the BMM are listed below.

- Allows parallel execution of ARM and other masters when different slaves are accessed
- Decodes accesses to the following slave frames–
  - 128-Mbyte, two eSRAM frames
  - 16-Mbyte peripheral frame
  - 128-Mbyte reserved two expansion RAM frame
- Generates appropriate slave response for accesses to reserved memory map regions
- Arbitrates master accesses to slaves, e.g. Program Memory, RAM0, RAM1, Peripheral Bridge, CRC.
- Supports round robin and fixed priority mastership schemes

2.1.1 Interrupt Vector Support

The BMM provides a read access to the interrupt request (IRQ) and fast interrupt request (FIQ) interrupt vectors maintained in the vector interrupt manager (VIM) module. This feature reduces CPU read latencies to the interrupt vector.

The features of IRQ and FIQ interrupt vector maintained in BMM are–

- The 32-bit IRQ and FIQ interrupt vector are inputs to BMM coming from the VIM module.
- The IRQ and FIQ vector locations are 32-bit, zero-wait state, read-only locations in BMM. Writes to this location are discarded and no abort is issued.
- The BMM IRQ and FIQ vectors are generated in the VBUS domain by the VIM and read by the CPU in SYS domain. No synchronization to SYS domain is done for the IRQ/FIQ vector inside the BMM. The VIM ensures that the IRQ/FIQ vector is stable when CPU reads it.
- All masters can read the IRQ and FIQ vector locations. This feature makes the code generic.

2.2 Arbiter

In a multi-layer architecture, each slave is on a dedicated bus. Each slave has an associated arbiter. The arbiters provide arbitration among all bus masters that accesses each particular slave.

2.2.1 Arbitration Scheme

Arbitration is done at every transfer.

There are two programmable arbitration schemes.

- Fix priority. The order of the fixed priority is–
  - CPU instruction bus
  - CPU data read,
  - CPU data write
- Round robin. The most recent master becomes the lowest priority.

The selection of the priority is done by the BMM control register 2 (BMMCR2) in the system module.

Note: Locked Operations
The ARM instruction set of ARM7TDMI includes a data swap (SWP) instruction that allows the contents of a memory location to be swapped with the contents of a processor register. This instruction is implemented as an uninterrupted pair of accesses– the first access reads the contents of the memory, and the second writes the register data to the memory. These accesses are treated as a contiguous operation by the memory controller to prevent another master from changing the affected memory location before the swap is completed.

2.2.2 Interrupt Vector Support

The features of the IRQ and FIQ interrupt vectors maintained in the BMM are:

- The 32-bit IRQ and FIQ interrupt vectors are input ports to the BMM coming from the VIM module.
- The IRQ and FIQ vector locations are 32-bit zero-wait state read-only locations in BMM. Writes to this location are discarded and no abort is issued.
- The BMM IRQ and FIQ vectors are generated in the VBUS domain by the VIM and read by the CPU in SYS domain. No synchronization to the SYS domain is done for the IRQ/FIQ vector inside the BMM. The VIM takes care that IRQ/FIQ the vector is stable when the CPU reads it.
- All masters can read the IRQ and FIQ vector locations. This is done to make the code generic.

Peripheral Interconnect

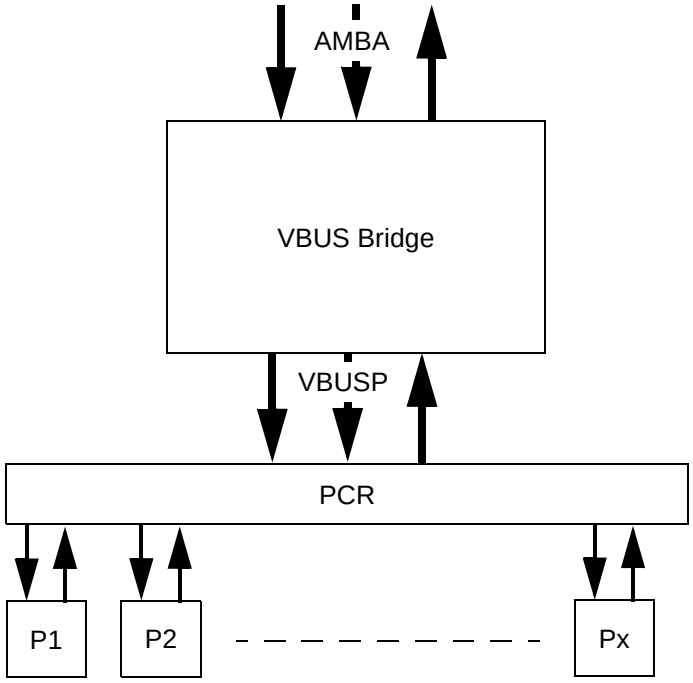
This chapter describes the peripheral interconnect of the Texas Instruments TMS470Px family.

Topic **Page**

3.1 Peripheral Bridge Overview

The peripheral bridge, shown in Example 3-2, allows the CPU to communicate with the peripherals. The peripheral bridge is connected to the bus matrix module (BMM). The peripheral interface supports the VBUSP protocol interface. The bridge and the peripherals are connected together by the Peripheral Central Resource (PCR).

Figure 3-2. Peripheral Bridge Block diagram



The VBUS bridge acts as a BMM bus slave and as a master on VBUSP.

3.2 ARM72VBUS (R2V)

The ARM72VBUS is used in the ARM7TDMI platform. It allows the ARM7TDMI to access the peripherals on the VBUSP bus.

3.3 Peripheral Central Resource (PCR)

The Peripheral Central Resource (PCR) provides decoding and multiplexing between the VBUS bridge, the peripherals, and the memories.

It is possible to restrict writes to each of the non-system peripherals to privileged mode. However, writes are allowed in debug mode regardless of the peripheral protection.

Peripheral protection can be enabled by configuring the PCR_PMPROTx registers or the PCR_PPROTx registers in PCR module. These registers implement one bit per slave. These registers are accessible as set-clear registers. Please refer to section 3.3 for register description.

In debug mode, the PCR does not check for the privilege of VBUSP accesses.

Embedded SRAM (eSRAM)

This chapter describes the embedded SRAM wrappers of the Texas Instruments TMS470Px family.

Topic **Page**

4.1 Overview

The eSRAM wrapper is used to connect the system buses to the RAM array.

The eSRAM wrapper controls the RAM operation (READ, WRITE) as well as decoding operation for the RAM space.

The eSRAM supports memory fault correction/detection via internal SECDED circuit.

The eSRAM wrapper uses a 32-bit wide bus. It allows read and write accesses in 32-bit, 16-bit or 8-bit.

The eSRAM wrapper supports 2 types of wait-states:

– Address pipeline with an introduced wait-state latency (see section 4.4).

– Extension data phase which allows more time for data propagation (see section 4.4).

Both types can be used and are programmable through the system module (see section 5.1.30).

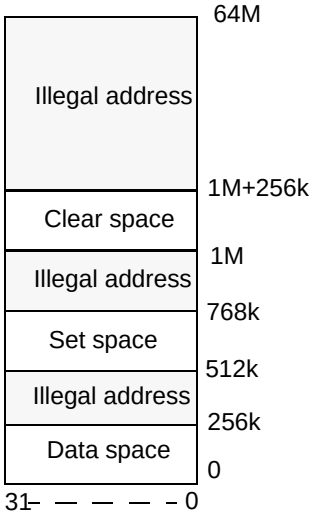
The eSRAM wrapper also allows setting or clearing of one bit thereby reducing the need for read-modify-write (RMW) operations.

4.2 Bit Access Operation

Even though the CPU does not support atomic RMW, the eSRAM wrapper provides a mechanism to set or to clear a single bit in the RAM array.

- When reading/writing from the RAM base address, accesses are realized in bytes, half-words, and words as typical accesses.
  – When accessing the RAM from the base address + 0x80000, the write access is implemented so that bits set to 1 on the data bus will be set to 1 in the RAM array without disturbing the other bits—the same way an RMW from CPU is implemented. Bits set to 0 on the data bus will be untouched.
  – When accessing the RAM from the base address + 0x100000, the write access is implemented so that bits set to 1 on the data bus will be clear to 0 in the RAM array without disturbing the other bits—the same way a RMW from CPU is implemented. Bits set to 0 on the data bus will be untouched.

Figure 4-1. Bit Access Memory Map



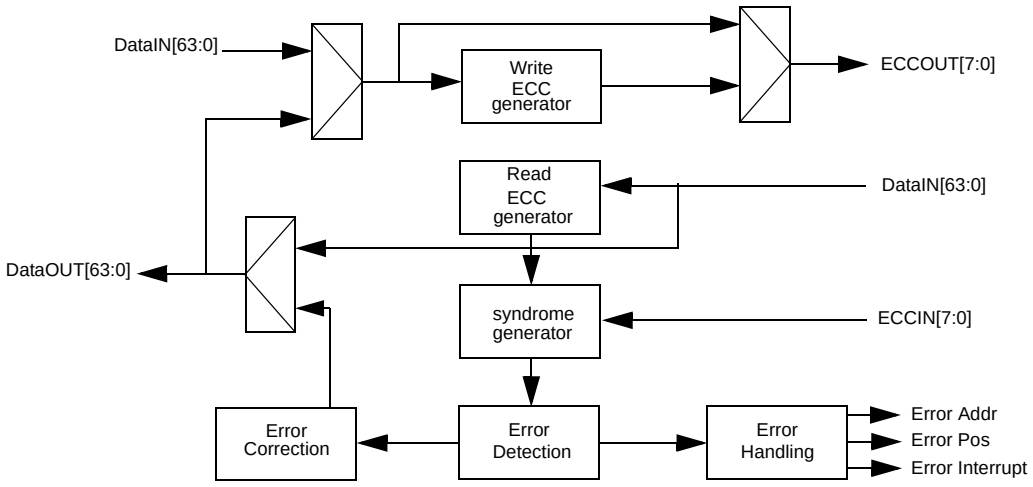
For instance, assuming a word at address location 0x0800 0010, with a value of 0x3459–

1. Writing the value 0x8315 at address 0x0808 0010 (SET address) will result in the value 0xB75D (0x3459 or 0x8315).
2. Writing the value 0x8315 at address 0x0810 0010 (CLEAR address) will result in the value 0x3448 (0x3459 and not(0x8315)).

4.3 Memory Fault Detection

The eSRAM wrapper contains the SECDED (**S**ingle **E**rror **C**orrection and **D**ouble **E**rror **D**etection) logic. After power up reset, the memory fault detection is disabled through a 4-bit ECC_ENABLE key. After power up reset, the 4-bit key has the reset value of 0x5. Any value but 0x5 will enable error detection and correction logic. If error detection and correction is disabled, error correction logic is totally bypassed resulting in optimal data access time. If error detection and correction is enabled then data is returned to the host master with slower access time which can affect the data rate at the frequency demanded by the host unless a proper wait state is added. Figure 4-2 illustrates a high level view of the SECDED.

Figure 4-2. SECDED Block Diagram



When the SECDED logic is enabled, it provides the capability to screen out single and double bit memory faults and correct single bit faults. SECDED requires a total of eight ECC (**E**rror **C**orrection **C**ode) check bits for each 64 bits of data to be stored in memory. During write accesses, SECDED automatically generates the ECC bits and stores them to the ECC memory. During read accesses, SECDED automatically generates eight ECC bits based on the 64 bits data read. These eight ECC bits are then compared with the known good ECC value stored in the ECC memory.

4.3.1 Read-Modify-Write Operation

The error correcting code is computed on 64 bits of data. Therefore, if only a subset of the 64 bits are written, SECDED still requires all 64 bits are available for the calculation. As a result, the SECDED will perform a read modify write operation by first reading the entire 64 bits of the addressed location. If there is a single bit error during the read-modify-write operation, the read data is first corrected. The ECC is then calculated based on the 64 bit data formed by the un-addressed bytes concatenated with the addressed bytes. After the ECC is calculated then, both the 64 bit data and ECC are written into their respective memory space. In short, a write of less than 64 bits will always incur one extra read operation by the eSRAM wrapper. Due to the use of an internal write buffer, the read-modify-write operation is completely hidden from the master as long as there is not another consecutive access.

4.3.2 Consecutive Access

When SECDED is enabled and there is an immediate second access following a write access less than 64 bits, the second access may incur a wait state penalty.

The following table describes the number of wait states incurred to the host master for different types of consecutive accesses.

Table 4-1. Wait state comparison for different access sequences

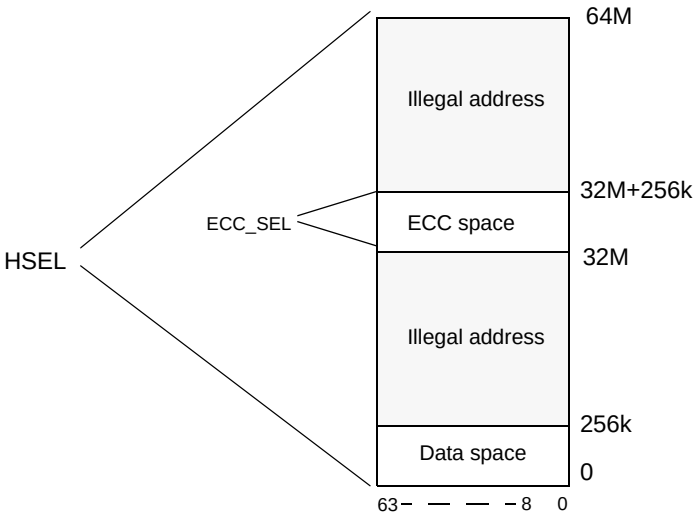
WST_DENA[a]	WST_AENA[b]	First Access	Wait states incurred to host		Second Access	Wait states incurred to host	
			ECC Disabled	ECC Enabled		ECC Disabled	ECC Enabled
0	0	Read	0	0	Read	0	0
		Read	0	0	Write[c]	0	0
		Write	0	0	Read	1	2[d]
		Write	0	0	Write	0	0
0	1	Read	1	1	Read	1	1
		Read	1	1	Write	0	0
		Write	0	0	Read	1	3[d]
		Write	0	0	Write	0	0
1	0	Read	1	1	Read	1	1
		Read	1	1	Write	0	0
		Write	0	0	Read	2	3[d]
		Write	0	0	Write	0	0
1	1	Read	2	2	Read	2	2
		Read	2	2	Write	0	0
		Write	0	0	Read	2	4[d]
		Write	0	0	Write	0	1[d]

Notes: a) Extended data phase wait state
b) Address pipeline wait state
c) All writes in the table are less than 64 bits.
d) Shaded area indicates extra wait states when SECDED is enabled.

4.3.3 ECC Memory Mapping

ECC bits are memory mapped so they can be accessible to the host master for both read and write. Each ECC word is 8 bits wide and resides in a double-word (64 bits) boundary. Each ECC word is offset by 32Mbyte from the corresponding data word to which it is checking. The ECC bits can be accessed by the master regardless of the state SECDED enable or disable.

Figure 4-3. ECC and Data Memory Map



To avoid accidental over writing of ECC bits, the write to ECC memory is qualified with the ECC_WRT_ENABLE bit. If ECC_WRT_ENABLE is not active then writes to the ECC memory are ignored. ECC bits are always programmed or read using the lower eight bits of the data bus (bits 7 to 0).

4.3.4 ECC Generation

The 64 bit data and 8 bit ECC check bits are sent to SECDED for error detection and correction. ECC bits are generated according to a modified Hamming code. The calculated ECC bits during a read operation are XORed bitwise with the pre-determined known ECC bits to produce the syndrome. If the data contains no error then the syndrome is zero. In the event of an error, the syndrome is decoded to determine whether it is a single bit error or multi-bit error. The syndrome also encodes the bit position in error if it is a single bit error as shown in Table 4-2. .

Table 4-2. ECC encoding

Participating Data bits																																				
63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
								x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x									x	x	x	x	x
x	x	x	x	x	x	x	x																	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x									x	x	x	x	x
x	x							x	x	x	x	x	x			x	x							x	x	x	x	x	x			x	x			
		x	x	x				x	x	x				x	x			x	x	x				x	x	x				x	x			x	x	x
x		x			x	x		x			x	x			x	x		x			x	x		x			x	x			x	x		x		
			x		x		x		x		x		x	x	x				x		x		x		x		x		x	x	x				x	
x		x	x		x			x	x		x				x	x		x	x		x			x	x		x				x		x			x

Participating Data bits																											Parity[2]	Checkbits[1]
26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00		
x	x	x																	x	x	x	x	x	x	x	x	Odd	**ECC[7]**
x	x	x																	x	x	x	x	x	x	x	x	Odd	**ECC[6]**
x	x	x									x	x	x	x	x	x	x	x									Odd	**ECC[5]**
			x	x	x	x	x	x			x	x							x	x	x	x	x	x			Odd	**ECC[4]**
			x	x	x				x	x			x	x	x				x	x	x				x	x	Odd	**ECC[3]**
x	x		x			x	x			x	x		x			x	x		x			x	x			x	Odd	**ECC[2]**
x		x		x		x		x	x	x				x		x		x		x		x		x	x	x	Even	**ECC[1]**
	x	x			x		x	x	x			x			x		x	x			x		x	x	x		Even	**ECC[0]**

Notes:
1) Each ECC [x] bit represents the parity bit for the corresponding data bits marked with x in the same row.
2) The ECC check bit is generated as either an XOR (Even) or an XNOR (Odd) of the data bits marked with x in the same row.
3) All zeros and all ones data will be detected as double error.

Table 4-3. Syndrome Decode to Bit in Error

Data Error Bit																																				
63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0
0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1
1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0
0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0
1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	1	0	1	0	0	1

Data Error Bit																											ECC Error Bit									
26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	07	06	05	04	03	02	01	00		
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	**Bit[7]**	Syndrome[(1)]
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	**Bit[6]**	
1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	**Bit[5]**	
0	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0	**Bit[4]**	
0	0	0	1	1	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	0	0	1	0	0	0	**Bit[3]**	
1	1	0	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0	**Bit[2]**	
1	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	0	0	0	1	0	**Bit1]**	
0	1	1	0	0	1	0	1	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	1	**Bit[0]**	

Notes: 1) The syndrome is an 8 bit value which decodes to the bit in error. The bit in error can be a bit among the 64 data bits or a bit among the 8 ECC check bits. For example, if data bit 30 is in error then the syndrome would be 11110001. If ECC check bit 5 is in error then syndrome would be 00100000. A syndrome value of 00000000 indicates there is no error. Any other syndrome combinations not shown in the table are either a double bit error or un-correctable multi-bit error

4.3.5 Double Error Detection

When there are more than one bit errors detected, the SECDED generates a double bit error signal to the system module, the address location in which the double bit error is detected is saved in an error address register, and The raw data from RAM is returned to the CPU. The error address register is then frozen from being updated until it is read by the CPU. In response to the double bit error signal from the SECDED, The system module will set a double bit error interrupt flag within the system module. In addition, the system module’s programmable error counter will make sure that the double error interrupt is serviced within the programmed cycle count by the CPU. If the error counter times out, then a system reset is issued to the CPU.

4.3.6 Single Error Correction

SECDED is capable of correcting an erroneous bit if it determines the number of error bits is one. Each time a single error is detected and corrected the single error occurrence counter (SEC_OCCUR) is incremented. A single error interrupt can be generated when the single error occurrence counter reaches a threshold value. The threshold value is stored in a 16 bit threshold (SEC_THRESHOLD) register. If the threshold value is programmed to 1 then both the address and error position causing the single bit error are captured and saved into registers. The address and error position are then frozen from being updated until the single error status flag is cleared.

4.3.7 False Double Error Detection

When the system comes out of power up reset the content of memory is in an unknown state. If the SECDED is enabled then either a normal read or the read-modify-write operation forced by a less than 64-bit write access will result in a false double error. The read operation will cause a mismatch between the data and the

ECC memory since both are in an unknown state. Several software and hardware practices can be employed to avoid false double error generation.

- Software Memory Initialization
  - After power up reset, the system software should first initialize the entire data and ECC memory to a known state by resetting all data bits to zero and ECC check bits to 0xFC. This should be done when ECC_ENABLE is disabled.
- Enable Read-Modify-Write CORRECTION BYPASS feature
  - When RMWCBYP (Read-Modify-Write correction bypass) is enabled, the SECDED bypasses the error detection and correction for the read during the read-modify-write operation forced by the less than 64-bit write access. The bypass feature has no effect on a normal read operation. The user can first enable the bypass feature and then write user data such as constants or interrupt vector table to the memory. ECC bits are generated and written along with the user data to the memory. Upon completion the RMWCBYP should be disabled.

Note: Disable RMWCBYP bit after memory initialization is complete.
If RMWCBYP is not disabled after memory initialization and in the event of a true double error then no double error is generated.

Note: Avoid reading from an un-initialized memory location
If SECDED is enabled then the user should avoid reading from any memory location which is un-initialized.

4.3.8 Interrupt and Error Generation

4.3.8.1 Single Error Interrupt

The eSRAM wrapper can generate an interrupt when the number of occurrences of single bit errors is equal to the error threshold value. The interrupt is only generated when ECC_ENABLE is enabled, RMWCBYP is disabled, and SECINTEN is enabled as shown in Table 4-4

Table 4-4. Single Error Interrupt Generation

ECC_ENABLE	RMWCBYP	SECINTEN	SERR_INT	SECINTFLAG
!= 0101	0	1	Yes	Set
!= 0101	0	0	No	Set
All other combinations			No	Not set

4.3.8.2 Double Error Generation

The eSRAM wrapper will notify the System Module when a double error is detected by means of an internal signal. The system module will capture the signal and generate a double error interrupt.

Note:
When a double bit error is detected, the raw data of the addressed location is still passed to CPU.

Table 4-5. Double Error Generation

ECC_ENABLE	RMWCBYP	DERR signal
!= 0101	0	Yes

ECC_ENABLE	RMWCBYP	DERR signal
All other combinations		No

4.3.9 Emulation

If the SUSPEND signal is high during emulation, the data read from memory is still passed to the SECDED for correction if it is enabled. If a single error is detected then it is corrected but a single error interrupt is not generated when the threshold is reached.

The occurrence counter SEC_OCCUR continues to increment if a correctable error is detected and resets when the threshold value is reached.

Note:
During SUSPEND, the interrupt of the single error is not generated.

If a double error address is frozen and is not read by CPU before entering suspend then it will remains frozen during suspend even it is read during suspend. If a double error is detected during suspend then the raw data is returned and the error address is not updated and no double error signal is generated. This ensures that during SUSPEND, no double errors are generated by the debugger reading a non initialize area.

4.4 eSRAM Wrapper ARM7TDMI operation

The eSRAM wrapper supports accesses from 8-bit to 32-bit wide for read and write operations. ARM7 bursts are converted into single access transfer type.

The eSRAM wrapper supports zero wait-state access, address pipeline wait-states and extended data phase wait-states during read operation. Write operations are always treated as zero wait-state.

The number of wait-states are programmed via the RAMGCR register in the system module.

The address pipeline wait-state is used when address and control signals from the ARM7 bus are violating the set-up time of the RAM module. The address and control of the bus are latched at the end of the address phase.

The extended data phase wait-state is used when the timing delay between the RAM and ARM7 master is long enough to violate the set-up of the master. In this case, the data that is output by the RAM is extended by one cycle, which allows the data to propagate to the master within 2 HCLK cycles.

Both types of wait-states can be used simultaneously.

During a WRITE transaction, all the control signals to RAM are pipelined since the DATA will only be valid on the bus during the second cycle. The memory is then selected after a cycle delay.

Adding an address pipeline or data phase extension wait-state will impact the read operation only. Address pipeline and data extended wait-states have no affect during write operations.

4.4.1 eSRAM Wrapper ARM7TDMI timing with ECC

As seen in section 4.3.1, ECC may introduce wait-states when a non-64-bit access in performed.

Less than 64-bits write accesses force a read-modify-write operation. The read-modify-write operation always takes three cycles. Two cycles are for read and ECC calculation and one cycle for the write back. Data Extended waitstate enable has no effect on the number of cycles it takes to perform a read-modify-write since by default the data phase of the read is already extended during a RMW operation. No wait-state is incurred by ARM7 master since the internal address and data buffers capture all information to complete RMW operation.

For consecutive writes without an address pipeline waitstate, the first two writes do not incur any wait states. For the third and subsequent write accesses that are less than 64-bit write accesses, there are two wait states for each access incurred.

When address pipeline waitstate is enabled, the first write access will incur zero wait-state to the host. The second access can be supported with one wait state. The third and subsequent write access less than 64-bits need two wait states.

Data extended waitstate has no effect on the number of cycles it takes to perform a write with ECC since, by default, the data phase of the read is already extended during a RMW operation. Hence the timing is the same as that of a zero-wait RMW.

As explained in Table 4-1. , a write followed by a read with ECC, will create an additional wait-state, resulting in a 2 wait-state access.

The addition of the address pipeline and data extended pipeline waitstates will add additional wait-states.

System and Peripheral Central Resource Registers

This chapter describes the control registers of the Texas Instruments TMS470Px family.

Topic **Page**

5.1 System control register (SYS)

This section describes the SYSTEM registers. The start address of the System module frame is 0xFFFFFF00 and the end address is 0xFFFFFFFF. The registers support 32/16/8-bit writes. The offset is relative to the System module frame start address.

Figure 5-1. SYSTEM Registers

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 SYSPC1 page 104	Reserved															
	Reserved															ECP-CLK-FUN
0x04 SYSPC2 page 105	Reserved															
	Reserved															ECP CLK _DIR
0x08 SYSPC3 page 106	Reserved															
	Reserved															ECP CLK _DIN
0x0C SYSPC4 page 107	Reserved															
	Reserved															ECP CLK_ DOUT
0x10 SYSPC5 page 108	Reserved															
	Reserved															ECP CLK _SET
0x14 SYSPC6 page 109	Reserved															
	Reserved															ECP CLK _CLR
0x18 SYSPC7 page 110	Reserved															
	Reserved															ECP CLK_ ODE
0x1C SYSPC8 page 111	Reserved															
	Reserved															ECP CLK_ PUE

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x30 CSDIS page 113	Reserved															
	Reserved								CLKSR7OFF	CLKSR6OFF	CLKSR5OFF	CLKSR4OFF	CLKSR3OFF	CLKSR2OFF	CLKSR1OFF	CLKSR0OFF
0x34 CSDISSET page 114	Reserved															
	Reserved								CLKSR7OFF-SET	CLKSR6OFF-SET	CLKSR5OFF-SET	CLKSR4OFF-SET	CLKSR3OFF-SET	CLKSR2OFF-SET	CLKSR1OFF-SET	CLKSR0OFF-SET
0x38 CSDISCLR page 115	Reserved															
	Reserved								CLKSR7OFF-CLR	CLKSR6OFF-CLR	CLKSR5OFF-CLR	CLKSR4OFF-CLR	CLKSR3OFF-CLR	CLKSR2OFF-CLR	CLKSR1OFF-CLR	CLKSR0OFF-CLR
0x3C CDDIS page 116	Reserved															
	Reserved								RTICLK2OFF	RTICLK1OFF	VCLKA2OFF	VCLKA1OFF	VCLK2OFF	VCLKPOFF	HCLKOFF	GCLKOFF
0x40 CDDISSET page 118	Reserved															
	Reserved								RTICLK2OFF	RTICLK1OFF	Reserved		VCLK2OFF	VCLKPOFF	HCLKOFF	GCLKOFF
0x44 CDDISCLR page 120	Reserved															
	Reserved								RTICLK2OFF-CLR	RTICLK1OFF-CLR	VCLKA2OFF CLR	VCLKA1OFF-CLR	VCLK2OFF-CLR	VCLK-POFF-CLR	HCLKOFFCLR	GCLK-ENA-CLR
0x48 GHVSRC page 122	Reserved								GHVWAKE(3:0)				Reserved			
	Reserved												GHVLPM(3:0)			
	Reserved												GHVSRC(3:0)			
0x4C VCLKASRC page 124	Reserved															
	Reserved								VCLKA2S(3:0)				Reserved			
	VCLKA1S(3:0)															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x50 RCLKSRC page 126	Reserved															
	Reserved												RTI1SRC(3:0)			
0x54 CSVSTAT page 127	Reserved															
	Reserved								CLK SR8V	CLK SR7V	CLK SR6V	CLK SR5V	CLK SR4V	CLK SR3V	CLK SR2V	CLK SR1V
0x58 Reserved	Reserved															
	Reserved															
0x5C MINITGCR page 128	Reserved															
	Reserved												MINITGENA(3–0)			
0x60 MSIENA page 129	MSIENA[31–16]															
	MSIENA[15–0]															
0x64 MSINIGSTAT page 130	Reserved															
	Reserved							MINI DONE	Reserved							MST DONE
0x68 Reserved	Reserved															
	Reserved															
0x6C MINISTAT page 131	MIDONE[31–16]															
	MIDONE[15–0]															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x70 PLLCTL1 page 132	Reserved								ROF	OSC_DIV	Reserved					
	Reserved			PLL_MUL(4:0)					Reserved		PLL_DIV(4:0)					
0x74 PLLCTL2 page 133	Reserved								FM_ENA	Reserved						
	Reserved		PLL_MODFREQ(4:0)						Reserved		PLL_MDEPTH(4:0)					
0x78 UERFLAG page 134	Reserved															
	Reserved															UCR FLAG0
0x7C Reserved	Reserved															
	Reserved															
0x80 Reserved	Reserved															
	Reserved															
0x84 VRCTL page 135	Reserved															
	Reserved			VSLEEPENA(3:0)					Reserved				VLPMENA(3:0)			
0x88-0xA0 Reserved	Reserved															
	Reserved															
0xAC IAHBIL-LADDR page 136	IAHBILLFADDR(31:16)															
	IAHBILLFADDR(15:0)															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xA8 DAHBIL-LADDR page 137	DAHBILLFADDR(31:16)															
	DAHBILLFADDR(15:0)															
0xAC PILLADDR page 138	PILLWTADDR(31:16)															
	PILLWTADDR(15:0)															
0xB0 Reserved	Reserved															
	Reserved															
0xB4 SSIR2 page 139	Reserved															
	SSKEY2								SSDATA2							
0xB8 SSIR3 page 140	Reserved															
	SSKEY3								SSDATA3							
0xBC SSIR4 page 141	Reserved															
	SSKEY4								SSDATA4							
0xC0 RAMGCR page 142	Reserved															
	Reserved									WST_AENA1	Reserved	WST_DENA1	Reserved	WST_AENA0	Reserved	WST_DENA0
0xC4 BMMCR1 page 143	Reserved															
	Reserved															MEM SW

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xC8 BMMCR2 page 144	Reserved															
	Reserved												PRTY-CRC	PRTYPBRG	PRTYRAM1	PRTYRAM0
0xCC Reserved	Reserved															
	Reserved															
0xD0 CLKCNTL page 145	Reserved								VCLK2R[3:0]				Reserved			
	Reserved								PENA				Reserved			
0xD4 ECPCNTRL page 147	Reserved								ECP-COS				Reserved			
	ECPDIV															
0xD8-0xDC Reserved	Reserved															
	Reserved															
0xE0 SYSECR page 148	Reserved															
	RESET [1]	RESET [0]	Reserved													
0XE4 SYSESR page 149	Reserved															
	PO RST	OSC RST	WD RST	IILT RST	DILT RST	PILT RST	Reserved			UCE RST	MPC RST	SW RST	EXT RST	Reserved		
0xE8 ITIFLAG page 152	Reserved															
	Reserved				IILI FLAG	DILI FLAG	PILI FLAG	DPILI FLAG	Reserved							UCE FLAG0

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xEC GLBSTAT page 154	Reserved															
	Reserved															OSC-FAIL

0xF0 DEVID page 155	CP15	ID														TECH
	TECH			I/O	PPAR	Program parity	RECC	Version						1	0	1

0xF4 SSIVEC page 157	Reserved															
	SSIDATA								SSIVECT[7:0]							

0xF8 SSIF page 159	Reserved															
	Reserved												SSI_FLAG4	SSI_FLAG3	SSI_FLAG2	SSI_FLAG1

0xFC SSIR1 page 160	Reserved															
	SSKEY1								SSDATA1							

5.1.1 SYS Pin Control Register 1 (SYSPC1)

The SYSPC1 register, shown in Figure 5-2 and described in Table 5-1, controls the function of the ECLK pin.

Figure 5-2. SYS Pin Control Register 1 (SYSPC1) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECP CLK FUN
R-0															R/W-0

R = Read in all modes; W = write in all modes; *n* = value after reset

Table 5-1. SYS Pin Control Register 1 (SYSPC1) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKFUN		ECLK function. This bit changes the function of the ECLK pin.
		0	ECLK is in GIO mode.
		1	ECLK is in functional mode as an output.
			Note: **It is recommended not to disable and enable the ECLK in functional mode within the same ECP clock period to guarantee a proper ECP clock duty cycle.**

5.1.2 SYS Pin Control Register 2 (SYSPC2)

The SYSPC2 register, shown in Figure 5-3 and described in Table 5-2, controls the function of the ECLK pin.

Figure 5-3. SYS Pin Control Register 2 (SYSPC2) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECP CLK DIR
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-2. SYS Pin Control Register 2 (SYSPC2) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKDIR		ECLK data direction. This bit controls the direction of the ECLK pin when it is configured to be in GIO mode only (ECPCLKFUN = 0).
		0	The ECLK pin is an input.
		1	The ECLK pin is an output.

5.1.3 SYS Pin Control Register 3 (SYSPC3)

The SYSPC3 register, shown in Figure 5-4 and described in Table 5-3, controls the function of the ECLK pin.

Figure 5-4. SYS Pin Control Register 3 (SYSPC3) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECP CLK DIN
R-0															R-U

R = Read only; -n = value after reset; -U = undefined

Table 5-3. SYS Pin Control Register 3 (SYSPC3) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKDIN		ECLK data in.
		0	The ECLK pin is at logic 0.
		1	The ECLK pin is at logic 1.

5.1.4 SYS Pin Control Register 4 (SYSPC4)

The SYSPC4 register, shown in Figure 5-5 and described in Table 5-4, controls the function of the ECLK pin.

Figure 5-5. SYS Pin Control Register 4 (SYSPC4) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECP CLK DOUT
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-4. SYS Pin Control Register 4 (SYSPC4) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKDOUT		ECLK data out write. This bit is only active when ECLK is configured to be in GIO mode (ECPCLKFUN = 0) and configured to be an output pin (ECPCLKDIR = 1). The value of this bit indicates the value to be output to the ECLK pin.
		0	The ECLK pin is at logic 0.
		1	The ECLK pin is at logic 1.

5.1.5 SYS Pin Control Register 5 (SYSPC5)

The SYSPC5 register, shown in Figure 5-6 and described in Table 5-5, controls the function of the ECLK pin.

Figure 5-6. SYS Pin Control Register 5 (SYSPC5) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECPCLKSET
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-5. SYS Pin Control Register 5 (SYSPC5) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKSET		ECLK data out set. This bit is only active when ECLK is configured to be in GIO mode (ECPCLKFUN = 0).
		0	*Read*– The ECLK pin is at logic 0. *Write*– Writing a 0 has no effect.
		1	*Read*– The ECLK pin is at logic 1. *Write*– The ECLK pin is at logic 1.

5.1.6 SYS Pin Control Register 6 (SYSPC6)

The SYSPC6 register, shown in Figure 5-7 and described in Table 5-6, controls the function of the ECLK pin.

Figure 5-7. SYS Pin Control Register 6 (SYSPC6) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECPCLKCLR
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-6. SYS Pin Control Register 6 (SYSPC6) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKCLR		ECLK data out clear. This bit is only active when ECLK is configured to be in GIO mode (ECPCLKFUN = 0).
		0	*Read*– The ECLK pin is at logic 0. *Write*– The ECLK pin value is unchanged.
		1	*Read*– The ECLK pin is at logic 1. *Write*– The ECLK pin is cleared to logic 0.

5.1.7 SYS Pin Control Register 7 (SYSPC7)

The SYSPC7 register, shown in Figure 5-8 and described in Table 5-7, controls the function of the ECLK pin.

Figure 5-8. SYS Pin Control Register 7 (SYSPC7) [offset = 18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECPCLKODE
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-7. SYS Pin Control Register 7 (SYSPC7) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKODE		ECLK open drain enable. This bit is only active when ECLK is configured to be in GIO mode (ECPCLKFUN = 0).
		0	The ECLK pin is configured in push/pull mode.
		1	The ECLK pin is configured in open drain mode.
			Note– When a pin is configured as an output in GIO mode, the output buffer is tristated when the ECPCLKDOUT register is set to 1. If the ECPCLKDOUT register is set to 0, the output buffer drives a value of 0. If the pin direction is set as an input, the output buffer is tristated no matter what the value of the ECPCLKDOUT register is. In push/pull mode, when a pin is configured as an output in GIO mode, the pin follows whatever value is set in the DOUT register.

5.1.8 SYS Pin Control Register 8 (SYSPC8)

The SYSPC8 register, shown in Figure 5-9 and described in Table 5-8, controls the function of the ECLK pull enable.

Figure 5-9. SYS Pin Control Register 8 (SYSPC8) [offset = 1Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECPCLKPUE
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-8. SYS Pin Control Register 8 (SYSPC8) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKPUE		ECLK pull enable. This bit is only active when ECLK is configured to be an input.
		0	ECLK pull enable is active.
		1	ECLK pull enable is inactive.

5.1.9 SYS Pin Control Register 9 (SYSPC9)

The Values in this register will disable or enable the input buffer when the pull logic is disabled. Details are given in Figure 5-10 and described in Table 5-9.

Figure 5-10. SYS Pin Control Register 9 (SYSPC9) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ECP CLK PS
R-0															R/W-0

R = Read; W = Write; -n = value after reset

Table 5-9. SYS Pin Control Register 9 (SYSPC9) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	ECPCLKPS		ECLK input buffer disable. This bit is only active when ECLK is configured to be an input and the pull logic is disabled. Otherwise, this bit has no affect on the device operation.
		0	When ECLK pull up/pull down logic is disabled, the input buffer is disabled.
		1	When ECLK pull up/pull down logic is disabled, the input buffer is enabled.

5.1.10 Clock Source Disable Register (CSDIS)

The CSDIS register, shown in Figure 5-11 and described in Table 5-10.

Figure 5-11. *Clock Source Disable Register (CSDIS)* [offset = 30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CLKSR7 OFF	CLKSR6 OFF	CLKSR5 OFF	CLKSR4 OFF	CLKSR3 OFF	CLKSR2 OFF	CLKSR1 OFF	CLKSR0 OFF
R-0								RWP-1	RWP-1	RWP-1	RWP-1	RWP-1	RWP-1	RWP-1	RWP-0

RWP- read in all modes and write in privileged mode, -n values after reset

Table 5-10. Clock Source Disable Register (CSDIS) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	CLKSR[7–0] OFF		Clock source[7–0] off.
		0	Clock source[7–0] is enabled.
		1	Clock source[7–0] is disabled.
			Note– On wakeup, only the clock sources that are selected by the device clock domains are awakened (auto wake-up using the wakeup signal).
			Table 5-11 presents the standard mapping for the TMS470Px clock source.

Table 5-11. Standard Mapping for TMS470Px Clock Source

Clock Source	Mapping
Clock Source 0	Oscillator
Clock Source 1	PLL
Clock Source 2 - 7	Reserved

5.1.11 Clock Source Disable Set Register (CSDISSET)

This register is shown in Figure 5-12 and described in Table 5-12.

Figure 5-12. Clock Source Disable Set Register (CSDISSET) [offset = 34h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SETCLKSR7OFF	SETCLKSR6OFF	SETCLKSR5OFF	SETCLKSR4OFF	SETCLKSR3OFF	SETCLKSR2OFF	SETCLKSR1OFF	SETCLKSR0OFF
R-0								R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-1	R/WP-0

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-12. Clock Source Disable Set Register (CSDISSET) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	SETCLKSR[7–0] OFF		Set clock source[7–0] off.
		0	*Read*– Clock source[7–0] is enabled. *Write*– Clock source[7–0] is unchanged.
		1	*Read*– Clock source[7–0] is disabled. *Write*– Clock source[7–0] is disabled.

5.1.12 Clock Source Disable Clear Register (CSDISCLR)

This register is shown in Figure 5-13 and described in Table 5-12.

Figure 5-13. Clock Source Disable Clear Register (CSDISCLR) [offset = 38h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CLRCLKSR7 OFF	CLRCLKSR6 OFF	CLRCLKSR5 OFF	CLRCLKSR4 OFF	CLRCLKSR3 OFF	CLRCLKSR2 OFF	CLRCLKSR1 OFF	CLRCLKSR0 OFF
R-0								R/WP-1	R/WP-1	R/WP-1	R/WP-0	R/WP-1	R/WP-1	R/WP-1	R/WP-0

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-13. Clock Source Disable Clear Register (CSDISCLR) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	CLRCLKSR[7–0] OFF		Clear clock source[7–0] off.
		0	*Read*– Clock source[7–0] is enabled. *Write*– Clock source[7–0] is unchanged.
		1	*Read*– Clock source[7–0] is disabled. *Write*– Clock source[7–0] is enabled.

5.1.13 Clock Domain Disable Register (CDDIS)

This register is shown in Figure 5-14 and described in Table 5-14.

Note: All the clock domains are enabled on wakeup.
The system should guarantee that when HCLK and VCLK_sys are turned off through the HCLKOFF bit, the GCLK domain is also turned off.

Figure 5-14. Clock Domain Disable Register (CDDIS) [offset = 3Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RTI CLK2 OFF	RTI CLK1 OFF	Reserved		VCLK2 OFF	VCLKP OFF	HCLK OFF	GCLK OFF
R-0								R/WP-0	R/WP-0	R-0		R/WP-0	R/WP-0	R/WP-0	R/WP-0

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-14. Clock Domain Disable Register (CDDIS) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–6	RTICLK[2–1]OFF		RTICLK[2–1] domain off.
		0	The RTICLK[2–1] domain is enabled.
		1	The RTICLK[2–1] domain is disabled.
5-4	Reserved		Reads return zero and writes have no effect.
3	VCLK2OFF		VCLK2 domain off.
		0	The VCLK2 domain is enabled.
		1	The VCLK2 domain is disabled.
2	VCLKPOFF		VCLK_periph domain off.
		0	The VCLK_periph domain is enabled.
		1	The VCLK_periph domain is disabled.
1	HCLKOFF		HCLK domain off.
		0	The HCLK domain is enabled.
		1	The HCLK domain is disabled.

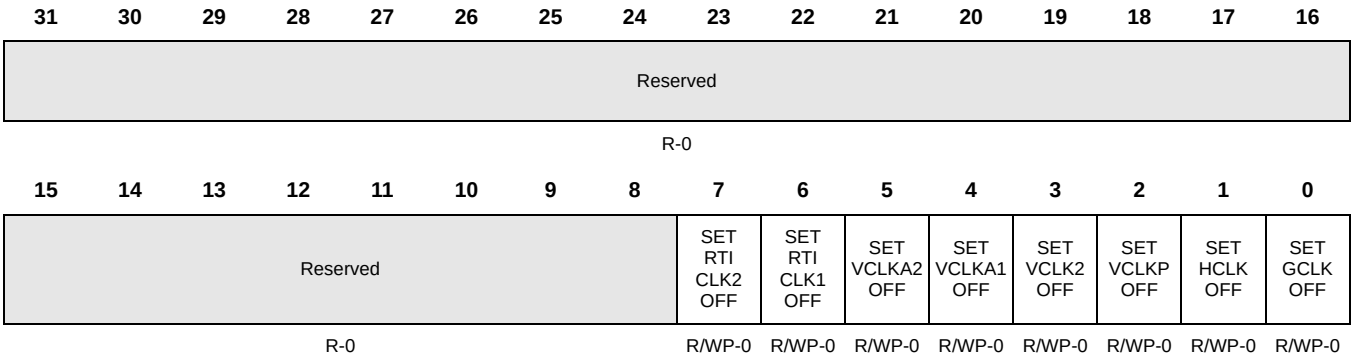
Table 5-14. Clock Domain Disable Register (CDDIS) Field Descriptions (Continued)

Bit	Name	Value	Description
0	GCLKOFF		GCLK domain off.
		0	The GCLK domain is enabled.
		1	The GCLK domain is disabled.

5.1.14 Clock Domain Disable Set Register (CDDISSET)

This register is shown in Figure 5-15 and described in Table 5-15.

Figure 5-15. Clock Domain Disable Set Register (CDDISSET) [offset = 40h]



R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-15. Clock Domain Disable Set Register (CDDISSET) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–6	SETRTICLK[2–1] OFF		Set RTICLK[2–1] domain.
		0	*Read*– The RTICLK[2–1] domain is enabled. *Write*– The RTICLK[2–1] domain is unchanged.
		1	*Read*– The RTICLK[2–1] domain is disabled. *Write*– The RTICLK[2–1] domain is disabled.
5	SETVCLKA2OFF		Set VCLKA2 domain.
		0	*Read*– The VCLKA2 domain is enabled. *Write*– The VCLK2 domain is unchanged.
		1	*Read*– The VCLKA2 domain is disabled. *Write*– The VCLKA2 domain is disabled.
4	SETVCLKA1OFF		Set VCLKA1 domain.
		0	*Read*– The VCLKA1 domain is enabled. *Write*– The VCLKA1 domain is unchanged.
		1	*Read*– The VCLKA1 domain is disabled. *Write*– The VCLKA1 domain is disabled.

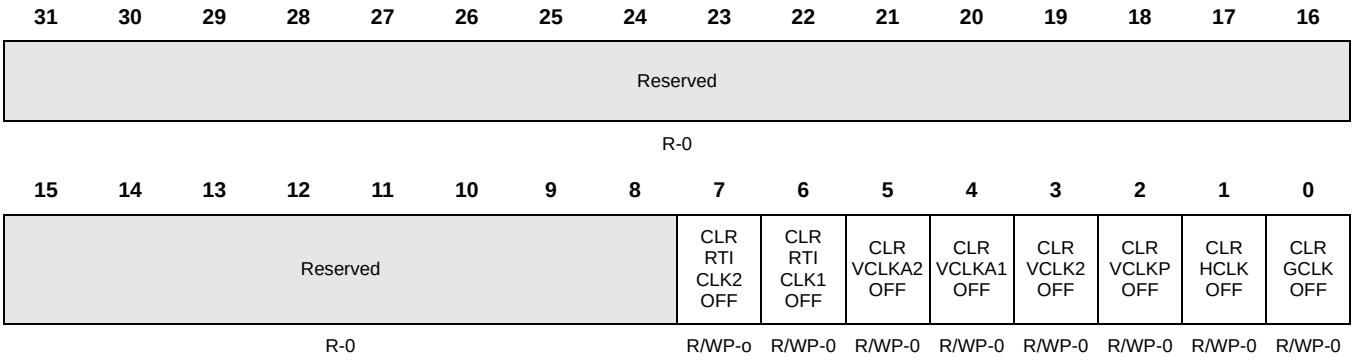
Table 5-15. Clock Domain Disable Set Register (CDDISSET) Field Descriptions (Continued)

Bit	Name	Value	Description
3	SETVCLK2OFF		Set VCLK2 domain.
		0	*Read*– The VCLK2 domain is enabled. *Write*– The VCLK2 domain is unchanged.
		1	*Read* – The VCLKA domain is disabled. *Write*– The VCLK2 domain is disabled.
2	SETVCLKPOFF		Set VCLK_periph domain.
		0	*Read*– The VCLK_periph domain is enabled. *Write*– The VCLK_periph domain is unchanged.
		1	*Read* – The VCLK_periph domain is disabled. *Write*– The VCLK_periph domain is disabled.
1	SETHCLKOFF		Set HCLK domain.
		0	*Read*– The HCLK1 domain is enabled. *Write*– The HCLK1 domain is unchanged.
		1	*Read*– The HCLK1 domain is disabled. *Write*– The HCLK1 domain is disabled.
0	SETGCLKOFF		Set GCLK domain.
		0	*Read*– The GCLK domain is enabled. *Write*– The GCLK domain is unchanged.
		1	*Read*– The GCLK domain is disabled. *Write*– The GCLK domain is disabled.

5.1.15 Clock Domain Disable Clear Register (CDDISCLR)

This register is shown in Figure 5-16 and described in Table 5-16.

Figure 5-16. Clock Domain Disable Clear Register (CDDISCLR) [offset = 44h]



R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-16. Clock Domain Disable Clear Register (CDDISCLR) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–6	CLRRTICLK[2–1] OFF		Clear RTICLK[2–1] domain.
		0	*Read*– The RTICLK[2–1] domain is enabled. *Write*– The RTICLK[2–1] domain is unchanged.
		1	*Read*– The RTICLK[2–1] domain is disabled. *Write*– The RTICLK[2–1] domain is enabled.
5	CLRVCLKA2 OFF		Clear VCLKA2 domain.
		0	*Read*– The VCLKA2 domain is enabled. *Write*– The VCLKA2 domain is unchanged.
		1	*Read* – The VCLKA2 domain is disabled. *Write*– The VLCKA2 domain is enabled.
4	CLRVCLKA1OFF		Clear VCLKA1 domain.
		0	*Read*– The VCLKA1 domain is enabled. *Write*– The VCLKA1 domain is unchanged.
		1	*Read*– The VCLKA1 domain is disabled. *Write*– The VCLKA1 domain is enabled.

Table 5-16. Clock Domain Disable Clear Register (CDDISCLR) Field Descriptions (Continued)

Bit	Name	Value	Description
3	CLRVCLK2OFF		Clear VCLK2 domain.
		0	*Read*– The VCLK2 domain is enabled. *Write*– The VCLK2domain is unchanged.
		1	*Read*– The VCLK2 domain is disabled. *Write*– The VCLK2 domain is enabled.
2	CLRVCLKPOFF		Clear VCLK_periph domain.
		0	*Read*– The VCLK_periph domain is enabled. *Write*– The VCLK_periph is unchanged.
		1	*Read*– The VCLKP domain is disabled. *Write*– The VCLKP domain is enabled.
1	CLRHCLKOFF		Clear HCLK domain.
		0	*Read*– The HCLK domain is enabled. *Write*– The HCLK domain is unchanged.
		1	*Read* – The HCLK domain is disabled. *Write*– The HCLK domain is enabled.
0	CLRGCLKOFF		Clear GCLK enable.
		0	*Read*– The GCLK domain is enabled. *Write*– The GCLK domain is unchanged.
		1	*Read* – The GCLK domain is disabled. *Write*– The GCLK domain is enabled.

5.1.16 GCLK, HCLK, VCLK, VCLK2 Source Register (GHVSRC)

This register is shown in Figure 5-17 and described in Table 5-17.

Figure 5-17. GCLK, HCLK, VCLK, and VCLK2 Source Register (GHVSRC) [offset = 48h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				GHVWAKE(3–0)				Reserved				HVLPM(3–0)			
R-0				R/WP-0000				R-0				R/WP-0000			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												GHVSRC(3–0)			
R-0												R/WP-0000			

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-17. GCLK, HCLK, VCLK, and VCLK2 Source Register (GHVSRC) Field Descriptions

Bit	Name	Value	Description
31–28	Reserved		Reads return zero and writes have no effect.
27–24	GHVWAKE[3–0]		GCLK, HCLK, VCLK, VCLK2 source on wakeup.
		0000	Clock source0 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0001	Clock source1 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0010	Clock source2 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0011	Clock source3 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0100	Clock source4 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0101	Clock source5 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0110	Clock source6 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		0111	Clock source7 is the source for GCLK, HCLK, VCLK, VCLK2 on wakeup.
		1000–1111	Reserved
23–20	Reserved		Reads return zero and writes have no effect.

Table 5-17. GCLK, HCLK, VCLK, and VCLK2 Source Register (GHVSRC) Field Descriptions (Continued)

Bit	Name	Value	Description
19–16	HVLPM[3–0]		HCLK, VCLK, VCLK2 source on wakeup when GCLK is turned off.
		0000	Clock source0 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0001	Clock source1 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0010	Clock source2 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0011	Clock source3 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0100	Clock source4 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0101	Clock source5 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0110	Clock source6 is the source for HCLK, VCLK, VCLK2 on wakeup.
		0111	Clock source7 is the source for HCLK, VCLK, VCLK2 on wakeup.
		1000–1111	Reserved
15–4	Reserved		Reads return zero and writes have no effect.
3–0	GHVSRC[3–0]		GCLK, HCLK, VCLK, VCLK2 current source. **Note: The GHVSRC[3–0] bits are updated with the HVLPM[3–0] when GCLK is turned off, and are updated with GHVWAKE[3–0] on system wakeup.**
		0000	Clock source0 is the source for GCLK, HCLK, VCLK, VCLK2.
		0001	Clock source1 is the source for GCLK, HCLK, VCLK, VCLK2.
		0010	Clock source2 is the source for GCLK, HCLK, VCLK, VCLK2.
		0011	Clock source3 is the source for GCLK, HCLK, VCLK, VCLK2.
		0100	Clock source4 is the source for GCLK, HCLK, VCLK, VCLK2.
		0101	Clock source5 is the source for GCLK, HCLK, VCLK, VCLK2.
		0110	Clock source6 is the source for GCLK, HCLK, VCLK, VCLK2.
		0111	Clock source7 is the source for GCLK, HCLK, VCLK, VCLK2.
		1000–1111	Reserved

5.1.17 Peripheral Asynchronous Clock Source Register (VCLKASRC)

This register is shown in Figure 5-18 and described in Table 5-18.

Figure 5-18. Peripheral Asynchronous Clock Source Register (VCLKASRC) [offset = 4Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				VCLKA2S(3–0)				Reserved				VCLKA1S(3–0)			
R-0				R/WP-1001				R-0				R/WP-1001			

R = Read in all modes; WP = Write in privileged mode only; -n values after reset

Table 5-18. Peripheral Asynchronous Clock Source Register (VCLKASRC) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return zero and writes have no effect.
11–8	VCLKA2S[3–0]		Peripheral asynchronous clock2 source.
		0000	Clock source0 is the source for peripheral asynchronous clock2.
		0001	Clock source1 is the source for peripheral asynchronous clock2.
		0010	Clock source2 is the source for peripheral asynchronous clock2.
		0011	Clock source3 is the source for peripheral asynchronous clock2.
		0100	Clock source4 is the source for peripheral asynchronous clock2.
		0101	Clock source5 is the source for peripheral asynchronous clock2.
		0110	Clock source6 is the source for peripheral asynchronous clock2.
		0111	Clock source7 is the source for peripheral asynchronous clock2.
		1000–1111	VCLK is the source for peripheral asynchronous clock2.
7–4	Reserved		Reads return zero and writes have no effect.

Table 5-18. Peripheral Asynchronous Clock Source Register (VCLKASRC) Field Descriptions (Continued)

Bit	Name	Value	Description
3–0	VCLKA1S[3–0]		Peripheral asynchronous clock1 source.
		0000	Clock source0 is the source for peripheral asynchronous clock1.
		0001	Clock source1 is the source for peripheral asynchronous clock1.
		0010	Clock source2 is the source for peripheral asynchronous clock1.
		0011	Clock source3 is the source for peripheral asynchronous clock1.
		0100	Clock source4 is the source for peripheral asynchronous clock1.
		0101	Clock source5 is the source for peripheral asynchronous clock1.
		0110	Clock source6 is the source for peripheral asynchronous clock1.
		0111	Clock source7 is the source for peripheral asynchronous clock1.
		1000–1111	VCLK is the source for peripheral asynchronous clock1.

5.1.18 RTI Clock Source Register (RCLKSRC)

This register is shown in Figure 5-19 and described in Table 5-19.

Figure 5-19. RTI Clock Source Register (RCLKSRC) [offset = 50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Reserved	Reserved	Reserved	Reserved	Reserved				RTI1SRC(3:0)			
R-0				RWP-1	RWP-0	RWP-0	RWP-1	R-0				RWP-1	RWP-0	RWP-0	RWP-1

RWP- read in all modes and write in privileged mode, -n values after reset

Table 5-19. RTI Clock Source Register (RCLKSRC) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return zero and writes have no effect.
11-8	Reserved		Reads return zero. These bits are writable but have no effect on device operation.
7-4	Reserved		Reads return zero and writes have no effect.
3–0	RTI1SRC[3–0]		RTI clock1 source.
		0000	Clock source0 is the source for RTICLK1.
		0001	Clock source1 is the source for RTICLK1.
		0010	Clock source2 is the source for RTICLK1.
		0011	Clock source3 is the source for RTICLK1.
		0100	Clock source4 is the source for RTICLK1.
		0101	Clock source5 is the source for RTICLK1.
		0110	Clock source6 is the source for RTICLK1.
		0111	Clock source7 is the source for RTICLK1.
		1000–1111	VCLK is the source for RTICLK1.

5.1.19 Clock Source Valid Status Register (CSVSTAT)

This register is shown in Figure 5-20 and described in Table 5-20.

Figure 5-20. Clock Source Valid Status Register (CSVSTAT) [offset = 54h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CLKSR7V	CLKSR6V	CLKSR5V	CLKSR4V	CLKSR3V	CLKSR2V	CLKSR1V	CLKSR0V
R-0								R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R = Read all modes; -n = Value after power-up reset

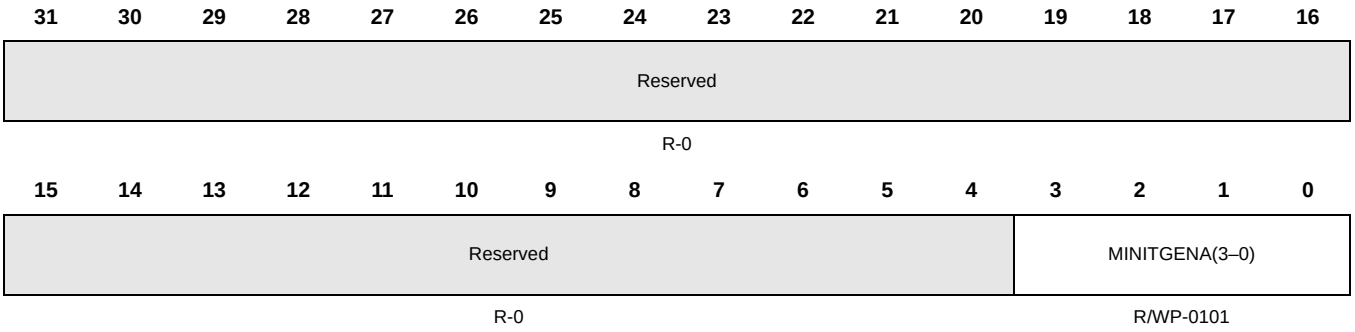
Table 5-20. Clock Source Valid Register (CSVSTAT) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	CLKSR[7–0]V		Clock source[7–0] valid. **Note– If the valid bit of the source of a clock domain is not set (i.e., the clock source is not fully stable), the respective clock domain is disabled by the GCM.**
		0	Clock source[7–0] is not valid.
		1	Clock source[7–0] is valid.

5.1.20 Memory Hardware Initialization Global Control Register (MINITGCR)

This register is shown in Figure 5-21 and described in Table 5-21.

Figure 5-21. Memory Hardware Initialization Global Control Register (MINITGCR) [offset = 5Ch]



R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-21. Memory Hardware Initialization Global Control Register (MINITGCR) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return zero and writes have no effect.
3–0	MINITGENA[3–0]		Memory hardware initialization global enable key. **Note– It is recommended to write a value of 0101 to disable the MINITGENA key to avoid enabling by a single bit flip.**
		1010	Global memory hardware initialization key is enabled
		Others	Global memory hardware initialization key is disabled

5.1.21 Memory Initialization Enable Register (MSINENA)

This register is shown in Figure 5-22 and described in Table 5-22.

Figure 5-22. Memory Initialization Enable Register (MSINENA) [offset = 60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIENA[31–16]															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSIENA[15–0]															
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-22. Memory Initialization Enable Register (MSINENA) Field Descriptions

Bit	Name	Value	Description
31–0	MSIENA[31–0]		Memory initialization enable register. Disabling the MINITGENA key (by writing from a 1010 to any other value) will reset all the MSIENA[31–0] bits to their default values.
		0	Memory module [31–0] hardware initialization is disabled.
		1	Memory module[31–0] hardware initialization is enabled.

5.1.22 MSTC Global Status Register (MSTCGSTAT)

This register is shown in Figure 5-23 and described in Table 5-23.

Figure 5-23. MSTC Global Status Register (MSTCGSTAT) [offset = 64h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							MINI DONE	Reserved							MST DONE
R-0							R/WP-0	R-0							R/WP-0

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

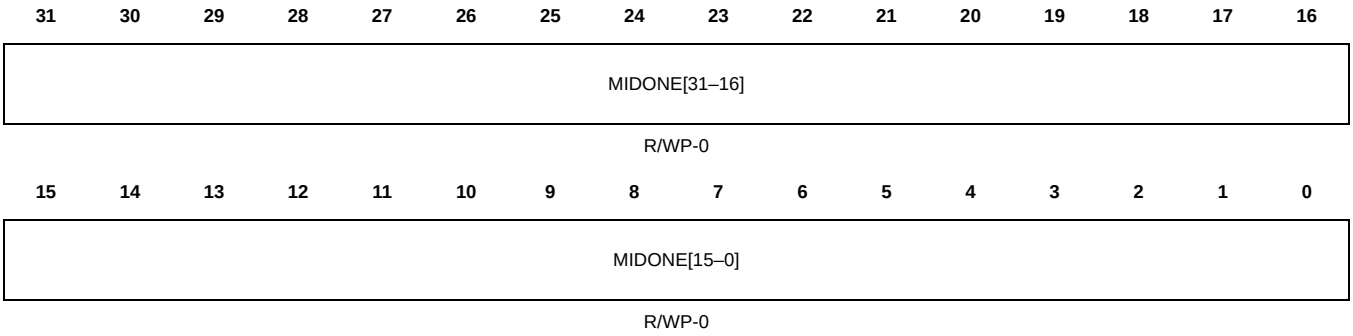
Table 5-23. MSTC Global status register (MSTCGSTAT) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved		Reads return zero and writes have no effect.
8	MINIDONE		Memory hardware initialization test run complete status. **Note– Disabling the MINITGENA key (By writing from a 1010 to any other value) will clear the MINIDONE status bit to 0.**
		0	*Read*– Memory hardware initialization is not completed. *Write*– A write of 0 has no effect.
		1	*Read*– Memory hardware initialization is completed. *Write*– The bit is cleared to 0.
7–1	Reserved		Reads return zero and writes have no effect.
0	MSTDONE		Memory self-test run complete status. **Note– Disabling the MSTGENA key (by writing from a 1010 to any other value) will clear the MSTDONE status bit to 0.**
		0	*Read*– Memory self-test is not completed. *Write*– A write of 0 has no effect.
		1	*Read*– Memory self-test is completed. *Write*– The bit is cleared to 0.

5.1.23 Memory Hardware Initialization Status Register (MINISTAT)

This register is shown in Figure 5-24 and described in Table 5-24.

Figure 5-24. Memory Hardware Initialization Status Register (MINISTAT) [offset = 6Ch]



R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-24. Memory Hardware Initialization Status Register (MINISTAT) Field Descriptions

Bit	Name	Value	Description
31–0	MIDONE[31–0]		Memory hardware initialization status bit. **Note– Disabling the MINITGENA key (by writing from a 1010 to any other value) will reset all the individual fail status bits to 0.**
		0	*Read*– Memory module[31–0] hardware initialization is not completed. *Write*– A write of 0 has no effect.
		1	*Read*– Memory module[31–0] hardware initialization is completed. *Write*– The bit is cleared to 0.

5.1.24 PLL Control Register1 (PLLCTL1)

This register is shown in Figure 5-25 and described in Table 5-25.

Figure 5-25. PLL Control Register 1 (PLLCTL1) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ROF	OSC DIV	Reserved					
R-0								RWP-0	RWP-0	R-0					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PLL_DIV(2:0)		
R-0													RWP-0	RWP-0	RWP-0

RWP: Read in all modes, write in privileged mode only, -n value after reset

Table 5-25. PLL Control Register 1 (PLLCTL1) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return zero and writes have no effect.
23	ROF		Reset on oscillator fail.
		0	The system is not reset when the oscillator is out of range.
		1	The system is reset when the oscillator is out of range.
22	OSCDIV		Enables a divided value of the oscillator as the input to the PLL. **Note: This bit is set when the Oscillator frequency is higher than the maximum allowed input frequency of the PLL.**
		0	Oscillator is the input to the PLL.
		1	Input to the PLL is oscillator divided by 2.
21–3	Reserved		Reads return zero and writes have no effect.
2-0	PLL_DIV		Configures the PLL's division ratio.
		000	PLL output clock divide by 1
		001	PLL output clock divide by 2
		010	PLL output clock divide by 3
		...	
		111	PLL output clock divide by 8

5.1.25 PLL Control Register2 (PLLCTL2)

This register is shown in Figure 5-26 and described in Table 5-26.

Figure 5-26. PLL Control Register 2 (PLLCTL2) [offset = 74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								FM ENA	Reserved						
R-0								RWP-0	R-0						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PLL_MODFREQ(4:0)						Reserved			PLL_MDEPTH(4:0)				
R-0		RWP-0	RWP-0	RWP-0	RWP-0	RWP-0	RWP-0	R-0			RWP-0	RWP-0	RWP-0	RWP-0	RWP-0

RWP: Read in all modes, write in privileged mode only, -n value after reset

Table 5-26. PLL Control Register 2 (PLLCTL2) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return zero and writes have no effect.
23	FM_ENA		Frequency modulation enable. **Note: FM modulation mode is not available on all devices or may be limited in the settings that are allowed. Please refer to the device specific datasheet before enabling this feature to ensure proper use.**
		0	PLL configured in Non-Frequency modulation mode.
		1	PLL configured in Frequency modulation mode.
22–14	Reserved		Reads return zero and writes have no effect.
13-8	PLL_MODFREQ		Configures the PLL's modulation frequency
7-5	Reserved		Reads return zero and writes have no effect.
4-0	PLL_MDEPTH		Configures the PLL's modulation depth.

5.1.26 Uncorrectable Error Reset Status Flag Register (UERFLAG)

This register is shown in Figure 5-27 and described in Table 5-27.

Figure 5-27. Uncorrectable Error Reset Status Flag Register (UERFLAG) [offset = 78h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UCR FLAG0
R-0															R/W-X

R/W- read in all modes and write clear in all modes, -X values unchanged after reset

Table 5-27. Uncorrectable Error Reset Status Flag Register (UERFLAG)

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	UCRFLAG0		Uncorrectable error reset flags. **Note: All the flag bits are cleared to 0 on power up reset (Vcc out of range). The values of all the flags remain unchanged after all other resets. When an Uncorrectable error reset is caused by an interrupt time out on any one of the Uncorrectable error (double bit error) interrupts, the corresponding Uncorrectable interrupt flag values in the ITIFLAG register are copied to the UCERFLAG register.**
		0	*Read*– The Uncorrectable Error Flag is not the source for reset due to an interrupt time out on the Uncorrectable Error interrupts. *Write*– Has no effect.
		1	*Read*– The Uncorrectable Error Flag is the source for reset due to an interrupt time out on the Uncorrectable Error interrupts. *Write*– The corresponding bit is cleared to 0.

5.1.27 Voltage Regulator Control Register (VRCTL)

This register is shown in Figure 5-28. Table 5-28 describes the system configuration to control the voltage regulator's mode.

Figure 5-28. Voltage Regulator Control Register (VRCTL) [offset = 84h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				VSLEEPENA(3:0)				Reserved				VLPMENA(3:0)			
R-0				RWP-0				R-0				RWP-0			

RWP: Read in all modes, write in privileged mode only, -n value after nporst t

Table 5-28. Voltage Regulator Control Register (VRCTL) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return zero and writes have no effect.
7–4	VSLEEPENA[3–0]		Voltage regulator sleep mode enable. **Note: This bit field is documented here for completeness. However, only regulator non-sleep mode is supported by the device. In addition, note that regulator sleep mode should be differentiated from a device level sleep mode. In a device level sleep mode the regulator is placed in Halt mode.**
		1111	Voltage regulator configured to be in sleep (LPM2) mode when GCLK, HCLK/VCLKSYS and VCLKP are gated and the Voltage regulator global low power mode (VLPMENA = 1111) is enabled.
		0000 to 1110	Voltage regulator configured to be in non-sleep mode
7–4	Reserved		Reads return zero and writes have no effect.
3–0	VLPMENA[3–0]		Voltage regulator global low power modes enable.
		1111	Voltage regulator configured to be in Halt (LPM1) mode (VSLEEPENA = 0000 to 1110) or Sleep (LPM2) mode (VSLEEPENA = 1111) when GCLK, HCLK/VCLKSYS and VCLKP are gated. **Note: Please refer to section 1.6 for details on the procedures for properly placing the device in the supported low power modes.**
		0000 to 1110	Voltage regulator configured to be in normal mode

5.1.28 AMBA illegal instruction fetch register (IAHBILLADDR)

This register is shown in Figure 5-29. Table 5-29 contains the field descriptions.

Figure 5-29. AMBA illegal instruction fetch register (IAHBILLADDR) [offset = A4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IAHBILLADDR(31:16)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAHBILLADDR(15:0)															
R-0															

R- read only, -0 Value after nporst

.

Table 5-29. AMBA illegal instruction fetch register (IAHBILLADDR) Field Descriptions

Bit	Name	Value	Description
31–0	IAHBILLADDR(31–0)		Contains the AMBA instruction bus transaction address when an illegal imprecise transaction is detected on the AMBA instruction bus in both user and privileged mode. **Note: A read from this register clears the IILTIFLAG bit in the ITIFLAG register to 0. This register is cleared to zero's only on power-on reset. The value of this register remains unchanged after all other resets.**

5.1.29 AMBA illegal data fetch register (DAHBILLADDR)

This register is shown in Figure 5-29. Table 5-29 contains the field descriptions.

Figure 5-30. AMBA illegal data fetch register (DAHBILLADDR) [offset = A8h]

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

R- read only, -0 Value after nporst

.

Table 5-30. AMBA illegal data fetch register (DAHBILLADDR) Field Descriptions

Bit	Name	Value	Description
31–0	DAHBILLADDR(31–0)		Contains the AMBA data bus transaction address when an illegal imprecise transaction is detected on the AMBA data bus in both user and privileged mode. **Note: A read from this register clears the DILTIFLAG bit in the ITIFLAG register to 0. This register is cleared to zero's only on power-on reset. The value of this register remains unchanged after all other resets.**

5.1.30 Peripheral Bus Illegal Write Transaction Register (PILLADDR)

This register is shown in Figure 5-29. Table 5-29 contains the field descriptions.

Figure 5-31. Peripheral Bus Illegal Write Transaction Register (PILLADDR) [offset = ACh]

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

R- read only, -0 Value after nporst

.

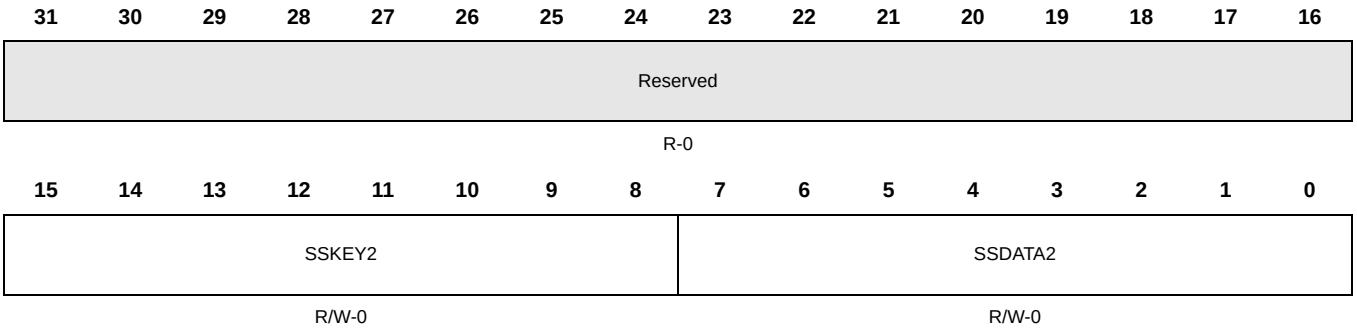
Table 5-31. Peripheral Bus Illegal Write Transaction Register (PILLADDR) Field Descriptions

Bit	Name	Value	Description
31–0	PILLADDR(31–0)		Contains the transaction address when an illegal VBUS write transaction is detected in both user and privileged mode. **Note: A read from this register clears the PILTIFLAG bit in the ITIFLAG register to 0. This register is cleared to zero's only on power-on reset. The value of this register remains unchanged after all other resets.**

5.1.31 System Software Interrupt Request 2 register (SSIR2)

This register is shown in Figure 5-32 and described in Table 5-32.

Figure 5-32. System Software Interrupt Request 2 Register (SSIR2) [offset = B4h]



R = Read; W = Write; -n = Value after reset

.

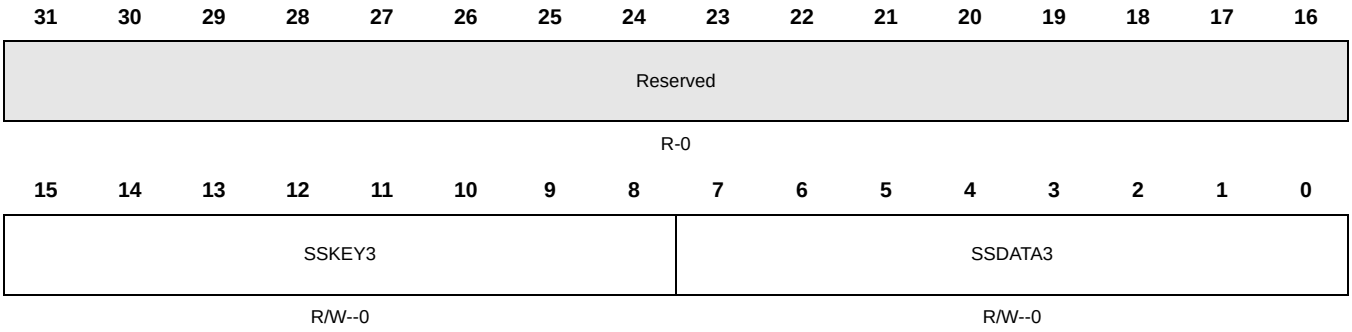
Table 5-32. System Software Interrupt Request 2 Register (SSIR2) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–8	SSKEY2[7–0]	0-FFh	System software interrupt 2 request key. A 84h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY2 field can be written into only if the write data matches the key (84h). The SSDATA2 field can only be written into if the write data into this field, the SSKEY2 field, matches the key (84h).
7–0	SSDATA2[7–0]	0–FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA2 field cannot be written into unless the write data into the SSKEY2 field matches the key (84h); therefore, byte writes cannot be performed on the SSDATA2 field.

5.1.32 System Software Interrupt Request 3 register (SSIR3)

This register is shown in Figure 5-33 and described in Table 5-33.

Figure 5-33. System Software Interrupt Request 3 Register (SSIR3) [offset = B8h]



R = Read; W = Write; U = Undefined; *-n* = Value after reset

.

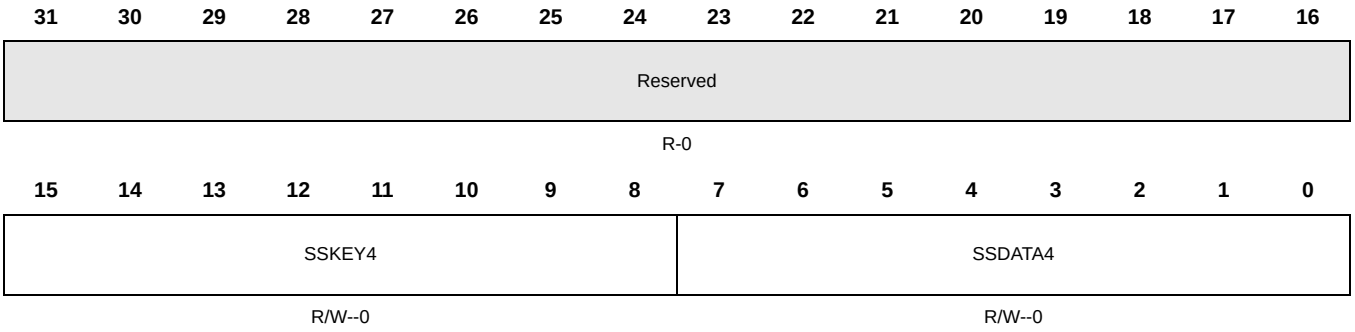
Table 5-33. System Software Interrupt Request 3 Register (SSIR3) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–8	SSKEY3[7–0]	0–FFh	System software interrupt 3 request key. A 93h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY3 field can be written into only if the write data matches the key (93h). The SSDATA3 field can only be written into if the write data into this field, the SSKEY3 field, matches the key (93h).
7–0	SSDATA3[7–0]	0–FF	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA3 field cannot be written into unless the write data into the SSKEY3 field matches the key (93h); therefore, byte writes cannot be performed on the SSDATA3 field.

5.1.33 System Software Interrupt Request 4 register (SSIR4)

This register is shown in Figure 5-34 and described in Table 5-34.

Figure 5-34. System Software Interrupt Request 4 Register (SSIR4) [offset = BCh]



R = Read; W = Write; U = Undefined; *-n* = Value after reset

.

Table 5-34. System Software Interrupt Request 4 Register (SSIR4) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–8	SSKEY4[7–0]	0–FFh	System software interrupt 4 request key. A A2h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY4 field can be written into only if the write data matches the key (A2h). The SSDATA4 field can only be written into if the write data into this field, the SSKEY4 field, matches the key (A2h).
7–0	SSDATA4[7–0]	0–FF	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA4 field cannot be written into unless the write data into the SSKEY4 field matches the key (A2h); therefore, byte writes cannot be performed on the SSDATA4 field.

5.1.34 RAM Control Register (RAMGCR)

This register is shown in Figure 5-35 and described in Table 5-35.

Figure 5-35. RAM Control Register (RAMGCR) [offset = C0h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									WST AENA 1	Reserved	WST DENA1	Reserved	WST AENA 0	Reserved	WST DENA0
R-0									RWP-0	R-0	RWP-0	R-0	RWP-0	R-0	RWP-0

RWP: Read in all modes, write in privileged mode only, -o value after reset

.

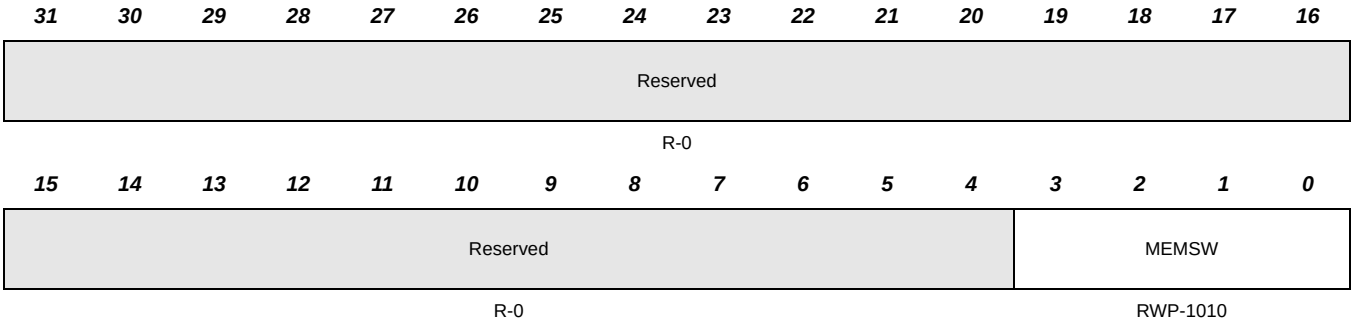
Table 5-35. RAM Control Register (RAMGCR) Field Descriptions

Bit	Name	Value	Description
31–7	Reserved		Reads return zero and writes have no effect.
6	WST_AENA1		eSRAM1 address phase wait state enable bit.
		0	The default address setup time for eSRAM1 is 0.
		1	The eSRAM1 address setup time is increased by one HCLK cycle.
5	Reserved		Reads return zero and writes have no effect.
4	WST_DENA1		eSRAM1 data phase wait state enable bit.
		0	There are no wait states for eSRAM1 during the data phase.
		1	There is one wait state for eSRAM1 during the data phase.
3	Reserved		Reads return zero and writes have no effect.
2	WST_AENA0		eSRAM0 data phase wait state enable bit.
		0	The default address setup time for eSRAM0 is used.
		1	The eSRAM0 address setup time is increased by one HCLK cycle.
1	Reserved		Reads return zero and writes have no effect.
0	WST_DENA0		eSRAM0 data phase wait state enable bit.
		0	There are no wait states for eSRAM0 during the data phase.
		1	There is one wait state for eSRAM0 during the data phase.

5.1.35 Bus Matrix Module Control Register1 (BMMCR1)

This register is shown in Figure 5-36 and described in Table 5-36.

Figure 5-36. Bus Matrix Module Control Register 1 (BMMCR) [offset = C4h]



RWP: Read in all modes, write in privileged mode only, -n value after reset

.

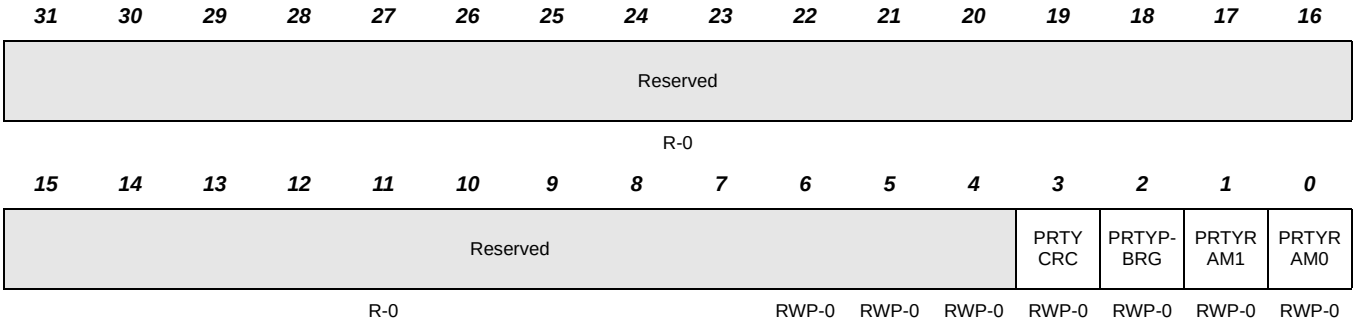
Table 5-36. Bus Matrix Module Control Register 1 (BMMCR) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return zero and writes have no effect.
3–0	MEMSW[3–0]		Memory swap bit key.
		1010	Default memory map– Program memory starts at 0x0000_0000 to 0x07FF_FFFF. eSRAM starts at address 0x0800_0000 to 0x0FFF_FFFF.
		0101	Swapped memory map– eSRAM starts at address 0x0000_0000 to 0x07FF_FFFF. Program memory starts at address 0x0800_0000 to 0x0FFF_FFFF
		Any other value	The device memory map is unchanged.

5.1.36 Bus Matrix Module Control Register2 (BMMCR2)

This register is shown in Figure 5-37 and described in Table 5-37.

Figure 5-37. Bus Matrix Module Control Register2 (BMMCR2) [offset = C8h]



RWP: Read in all modes, write in privileged mode only, -n value after reset

Table 5-37. Bus Matrix Module Control Register2 (BMMCR2) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return zero and writes have no effect.
3	PRTY_CRC		Arbitration priority to CRC.
		0	Fixed priority is used.
		1	Round robin priority is used.
2	PRTY_PRG		Arbitration priority to peripheral bridge.
		0	Fixed priority is used.
		1	Round robin priority is used.
1	PRTY_RAM0		Arbitration priority to eSRAM0.
		0	Fixed priority is used.
		1	Round robin priority is used.
0	PRTY_RAM1		Arbitration priority to eSRAM1.
		0	Fixed priority is used.
		1	Round robin priority is used.

5.1.37 Clock Control Register (CLKCNTL)

This register is shown in Figure 5-38 and described in Table 5-38.

Figure 5-38. Clock Control Register (CLKCNTRL) [offset = D0h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				VCLK2R				Reserved				VCLKR			
R-0				R/WP-0001				R-0				R/WP-0001			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							PENA	Reserved							
R-0							R/WP-0	R-0							

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-38. Clock Control Register (CLKCNTRL) Field Descriptions

Bit	Name	Value	Description
31–28	Reserved		Reads return zero and writes have no effect.
27–24	VCLK2R[3–0]		VBUS clock2 ratio.
		0000	The ratio is HCLK divided by 1.
		. . .	. . .
		1111	The ratio is HCLK divided by 16.
23–20	Reserved		Reads return zero and writes have no effect.
19–16	VCLKR[3–0]		VBUS clock ratio.
		0000	The ratio is HCLK divide by 1.
		. . .	. . .
		1111	The ratio is HCLK divided by 16.
15–9	Reserved		Reads return zero and writes have no effect.
8	PENA		Peripheral enable bit. The application must set this bit before accessing any peripheral.
		0	The global peripheral/peripheral memory frames are in reset.
		1	All peripheral/peripheral memory frames are out of reset.
7–0	Reserved		Reads return zero and writes have no effect.

Note: Restriction in changing the VCLK and VCLK2 clock ratios

VCLK2 must be equal to or greater than VCLK and the VCLK and VCLK2 clock ratios must not be changed simultaneously. The application must configure the VCLK2 ratio, read back the contents of the CLKCNTL register, and then configure the VCLK ratio.

5.1.38 ECP Control Register (ECPCNTL)

This register is shown in Figure 5-39 and described in Table 5-39.

Figure 5-39. ECP Control Register (ECPCNTL) [offset = D4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ECPS-SEL	ECP COS	Reserved						
R-0							R/W-0		R-0						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECPDIV15–0															
R/W-0															

R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-39. ECP Control Register (ECPCNTL) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	ECPSSEL		This bit allows the selection between VCLK and OSCIN as the clock source for ECLK.
		0	VCLK is selected as the ECP clock source.
		1	OSCIN is selected as the ECP clock source.
23	ECPCOS		ECLK continue on suspend.
		0	ECLK output is disabled in suspend mode. ECLK output will be shut off and will not be seen on the I/O pin of the device.
		1	ECLK output is not disabled in suspend mode. ECLK output will not be shut off and will be seen on the I/O pin of the device.
22–16	Reserved		Reads return zero and writes have no effect.
15–0	ECPDIV(15–0)	0–FFFF	ECP divider value. The value of ECPDIV bits determine the external clock (ECP clock) frequency as a ratio of VBUS clock (VCLK) as $ECLK = \frac{VCLK}{(ECPDIV + 1)}$

5.1.39 System exception control Register (SYSECR)

This register is shown in Figure 5-40 and described in Table 5-40.

Figure 5-40. System Exception Control Register (SYSECR) [offset = E0h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET1	RESET0	Reserved													
R/WP-0	R/WP-1	R-0													

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-40. System Exception Control Register (SYSECR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–14	RESET[1–0]		Software reset bits. Setting RESET1 or clearing RESET0 causes a software reset. There are three possible reset cases–
		01	No reset will occur.
		1x, x0	A global system reset will occur.
13–0	Reserved		Reads return zero and writes have no effect.

5.1.40 System Exception Status Register (SYSESR)

This register is shown in Figure 5-41 and described in Table 5-41.

Figure 5-41. System Exception Status Register (SYSESR) [offset = E4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PO RST	OSC RST	WD RST	IILT RST	DILT RST	PILT RST	Reserved			UCE RST	MPC RST	SW RST	EXT RST	Reserved		
R/W-X	R/W-X	R/W-X	R/W-X	R/W-X	R/W-X	R-0			R/W-X	R/W-X	R/W-X	R/W-X	R-0		

R/W- read in all modes and write clear in all modes, -X Value unchanged after reset

Table 5-41. System Exception Status Register (SYSESR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15	PORST		Power-up reset. This bit is set when VCC has been out of regulation, or when nPORRST is asserted. Reset is asserted as long as supply is out of regulation.
		0	*Read*– The V_{CC} has not been out of regulation, and nPORRST has not been asserted. *Write*– The bit is unchanged.
		1	*Read*– The last reset caused by V_{CC} being out of regulation, or nPORRST has been asserted. *Write*– The bit is cleared to 0.
14	OSCRST		Reset due to oscillator failure. Is Set when the last reset is caused by an oscillator failure.
		0	*Read*– No oscillator failure reset. *Write*– Has no effect.
		1	*Read*– Last reset due to an oscillator failure. *Write*– Clears the corresponding bit to zero.
13	WDRST		Watchdog reset flag. This bit is set when the last reset was caused by the analog or digital watchdog (AWD or DWD).
		0	*Read*– No reset has occurred because of the AWD or DWD. *Write*– The bit is unchanged.
		1	Read– The last reset was caused by the AWD or DWD. *Write*– The bit is cleared to 0.

Table 5-41. System Exception Status Register (SYSESR) Field Descriptions (Continued)

Bit	Name	Value	Description
12	IILTRST	IILTRST	AMBA instruction bus illegal transaction reset flag Set when the last reset is caused by an Masked imprecise illegal transaction on the AMBA instruction bus.
		0	*Read*– No AMBA instruction bus illegal address reset. *Write*– Has no effect.
		1	Read– Last reset due to illegal transaction on the AMBA instruction bus. *Write*– Clears the corresponding bit to zero.
11	DILTRST	DILTRST	AMBA data bus illegal transaction reset flag Set when the last reset is caused by an Masked imprecise illegal transaction access on the AMBA data bus.
		0	*Read*– No AMBA instruction bus illegal address reset. *Write*– Has no effect.
		1	Read– Last reset due to illegal transaction on the AMBA data bus. *Write*– Clears the corresponding bit to zero.
10	PILTRST		CPU VBUS write illegal transaction reset flag. Set when the last reset is caused by an CPU illegal write transaction on the VBUS in privileged mode (The address value is available in PILLADDR register).
		1	*Read*– Last reset due to CPU illegal write transaction on the VBUS. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No VBUS illegal address reset. *Write*– Has no effect.
9-7	Reserved		Reads return zero and writes have no effect.
6	UCERST		Uncorrectable error reset flag. Set when the last reset is caused by a interrupt time out on any one of the Uncorrectable error (double error or parity) interrupts.
		1	*Read*– Last reset interrupt time out on any one of the uncorrectable error interrupts. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No Uncorrectable error reset. *Write*– Has no effect.

Table 5-41. System Exception Status Register (SYSESR) Field Descriptions (Continued)

Bit	Name	Value	Description
5	MPCRST		Memory protection mode change reset flag. This bit is set when the last reset to the CPU is caused by switching from MMU to MPU mode or from MPU to MMU mode.
		0	*Read*– No reset due to a Memory protection mode change. *Write*– Has no effect.
		1	Read– Last reset due to a Memory protection mode change. *Write*– Clears the corresponding bit to zero.
4	SWRST		Software reset flag. This bit is set when the last reset is caused by software writing to the RESET1 or RESET0 bits.
		0	*Read*– No software reset has occurred. *Write*– The bit is unchanged.
		1	*Read*– The last reset was a software reset. *Write*– The bit is cleared to 0.
3	EXTRST		External reset flag. This bit is set when the last reset is caused by the external reset pin nRST. Please check the device specification to determine whether the nRST pin is available.
		0	*Read*– No reset caused by the external reset pin has occurred. *Write*– The bit is unchanged.
		1	*Read*– The last reset was caused by the external reset pin. *Write*– The bit is cleared to 0.
2–0	Reserved		Reads return zero and writes have no effect.

5.1.41 Illegal Transaction Interrupt Flag Register (ITIFLAG)

This register is shown in Figure 5-42 and described in Table 5-42.

Figure 5-42. *Illegal Transaction Interrupt Flag Register (ITIFLAG)* [offset = E8h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				IILI FLAG	DILI FLAG	PILI FLAG	DPILI FLAG	Reserved							UCE FLAG0
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R-0							R/W-0

R/W- read in all modes and write clear in all modes, -n after reset

.

Table 5-42. *Illegal Transaction Interrupt Flag Register (ITIFLAG)* Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return zero and writes have no effect.
11	IILIFLAG		AMBA instruction bus illegal transaction Interrupt flag.
		1	*Read*– A FIQ/IRQ interrupt is generated when an Masked imprecise illegal transaction is detected on the AMBA instruction bus (The address value is logged in IAHBILLADDR register) in user mode. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No illegal transaction is detected on the AMBA instruction bus in User mode. *Write*– Has no effect.
10	DILIFLAG		AMBA data bus illegal transaction Interrupt flag.
		1	*Read*– A FIQ/IRQ interrupt is generated when an Masked imprecise illegal transaction is detected on the AMBA data bus (The address value is logged in DAHBILLFADDR register) in user mode. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No illegal transaction is detected on the AMBA data bus in User mode. *Write*– Has no effect.
9	PILTFLAG		CPU Peripheral bus illegal write transaction interrupt flag.
		1	*Read*– A FIQ/IRQ interrupt is generated when an illegal CPU write transaction to a peripheral is detected in User Mode (The address value is logged in VILLWTADDR register). *Write*– Clears the corresponding bit to zero.
		0	*Read*– No illegal CPU write transaction to a peripheral in User mode. *Write*– Has no effect.

Table 5-42. *Illegal Transaction Interrupt Flag Register (ITIFLAG)* Field Descriptions (Continued)

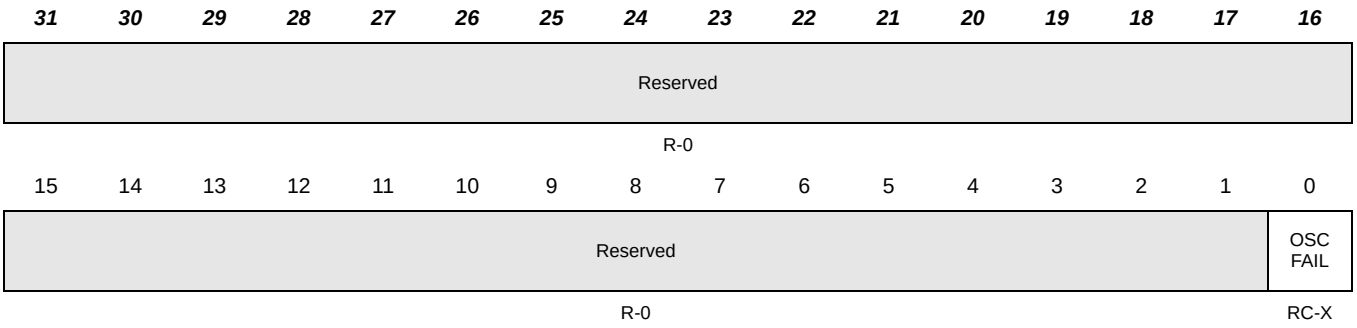
Bit	Name	Value	Description
8	DPILIFLAG		CPU Peripheral bus illegal write transaction interrupt flag.
		1	*Read*– A FIQ/IRQ interrupt is generated when an illegal DMA write transaction to a peripheral is detected in User mode (The address value is logged in VILLWTADDR register). *Write*– Clears the corresponding bit to zero.
		0	*Read*– No illegal DMA write transaction to a peripheral in User mode. *Write*– Has no effect.
7-1	Reserved		Reads return zero and writes have no effect.
0	UCEFLAG0		Uncorrectable error interrupt flags
		1	*Read*– A FIQ/IRQ is generated when an Uncorrectable (ECC double error or Parity Error) occurs. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No Uncorrectable error has occurred. *Write*– Has no effect.

Note:
Reads from the registers PILLADDR clears the corresponding PILTFLAGbit in the ITIFLAG register

5.1.42 Global Status Register (GLBSTAT)

This register is shown in Figure 5-43 and described in Table 5-43.

Figure 5-43. *Global Status Register (GLBSTAT)* [offset = ECh]



RC- Readclear, -X Value unchanged after reset

.

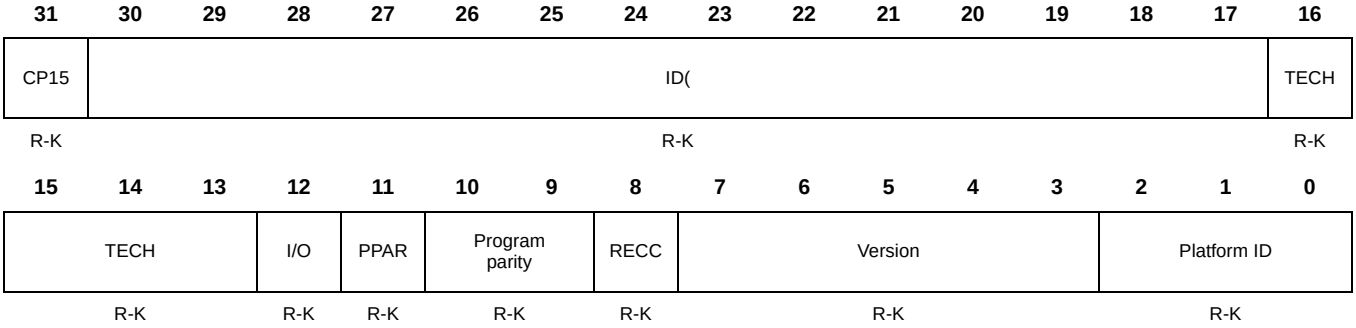
Table 5-43. *Global Status Register (GLBSTAT)* Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	OSCFAIL		Oscillator fail flag bit.
		1	*Read*– Oscillator failure has been detected. *Write*– Clears the corresponding bit to zero.
		0	*Read*– No Oscillator failure has been detected. *Write*– Has no effect.

5.1.43 Device Identification Register (DEV)

The DEV is a read-only register. It contains device-specific information that is hard-coded during device manufacture.This register is shown in Figure 5-44 and described in Table 5-44.

Figure 5-44. Device Identification Register (DEV) [offset = F0h]



R = Read only; *-K* = Constant value

.

Table 5-44. Device Identification Register (DEV) Field Descriptions

Bit	Name	Value	Description
31	CP15		CP15 CPU. This bit indicates whether the CPU has a coprocessor 15 (CP15)
		0	The CPU has no CP15 present.
		1	The CPU has a CP15 present. The CPU ID can be read using the CP15 C0,C0,0 register.
30–17	ID	0–3FFFh	Device ID. The device ID is unique by device configuration and is defined in the device datasheet.
16–13	TECH		These bits define the process technology by which the device was manufactured.
		0000	The C05 process technology was used.
		0001	The F05 process technology was used.
		0010	The C035 process technology was used.
		0011	The F035 process technology was used.
12	I/O		Input/output voltage. This bit defines the I/O voltage of the device.
		0	The I/O voltage is 3.3 V.
		1	The I/O voltage is 5 V.

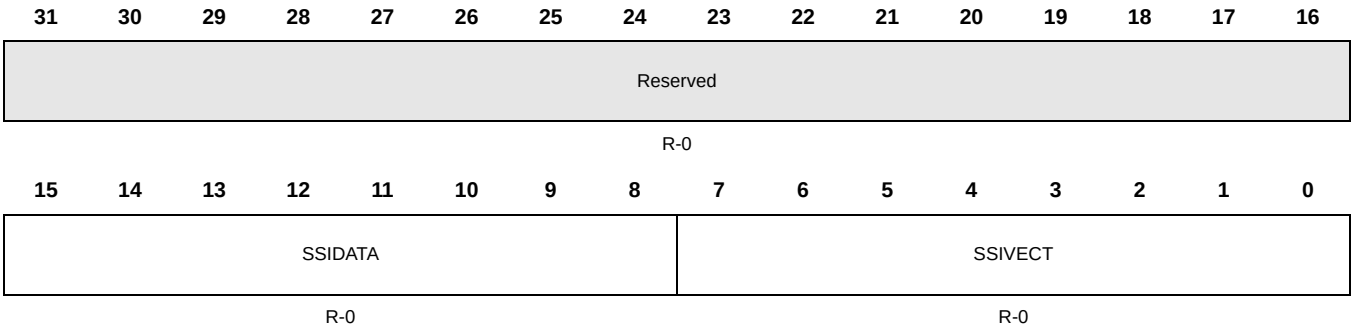
Table 5-44. Device Identification Register (DEV) Field Descriptions (Continued)

Bit	Name	Value	Description
11	PPAR		Peripheral parity. This bit indicates whether or not peripheral memory parity is present.
		0	The peripheral memories have no parity.
		1	The peripheral memories have parity.
10–9	Program parity		These bits indicate which parity is present for the program memory.
		00	No memory protection is present.
		01	The program memory has single bit parity.
		10	The program memory has ECC parity.
		11	This combination is reserved.
8	RECC		RAM ECC. This bit indicates whether or not RAM memory ECC is present.
		0	The RAM memories do not have ECC.
		1	The RAM memories have ECC.
7–3	VERSION	0–1Fh	Version. These bits provide the revision of the device.
2–0	Platform ID	101	The device is part of the TMS470Px family. The TMS470Px ID is always 101b

5.1.44 Software interrupt Vector register (SSIVEC)

This register is shown in Figure 5-45 and described in Table 5-45.

Figure 5-45. Software Interrupt Vector Register (SSIVEC) [offset = F4h]



R = Read only; -*n* = Value after reset

.

Table 5-45. Software Interrupt Vector Register (SSIVEC) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–8	SSIDATA[7–0]	0–FF	System software interrupt data key. These bits contain the data key value of the source for the system software interrupt, which is indicated by the vector in the SSIVEC[7–0] field.

Table 5-45. Software Interrupt Vector Register (SSIVEC) Field Descriptions (Continued)

Bit	Name	Value	Description
7–0	SSIVECT[7–0]		These bits contain the source for the system software interrupt. **Note– A read from the SSIVECT bits clears the corresponding SSI_FLAG[4–1] bit in the SSIF register, corresponding to the source vector of the system software interrupt.** **Note– The SSIR[4–1] interrupt has the following priority scheme–** **SSIR1 has the highest priority.** **SSIR4 has the lowest priority.**
		00000000	No software interrupt is pending.
		00000001	A software interrupt has been generated by writing the correct key value to The SSIR1 register.
		00000010	A software interrupt has been generated by writing the correct key value to the SSIR2 register.
		00000011	A software interrupt has been generated by writing the correct key value to the SSIR3 register.
		00000100	A software interrupt has been generated by writing the correct key value to the SSIR4 register.
		00000101 to 11111111	Reserved

5.1.45 System Software Interrupt Flag Register (SSIF)

This register is shown in Figure 5-46 and described in Table 5-46.

Figure 5-46. System Software Interrupt Flag Register (SSIF) [offset = F8h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SSI_ FLAG4	SSI_ FLAG3	SSI_ FLAG2	SSI_ FLAG1
R-0												R/C-0	R/C-0	R/C-0	R/C-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

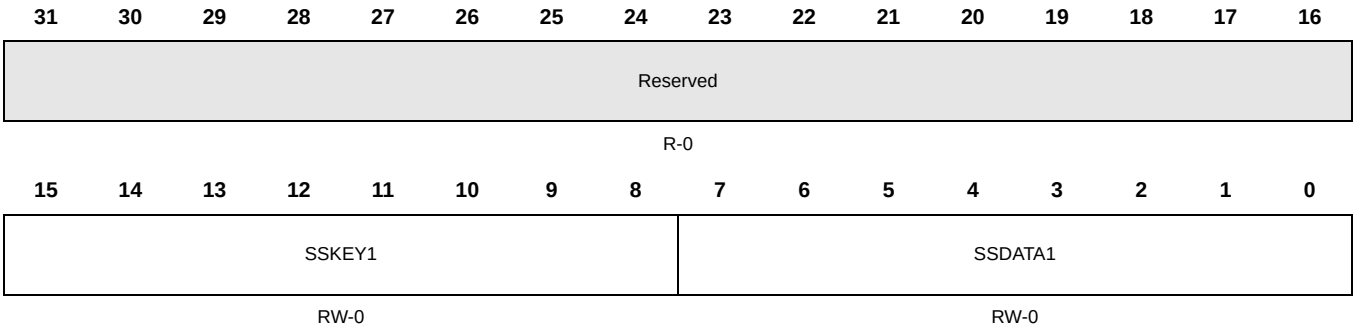
Table 5-46. System Software Interrupt Flag Register (SSIF) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return zero and writes have no effect.
3–0	SSI_FLAG[4–1]		System software interrupt flag[4–1]. This flag is set when the correct SSKEY is written to the SSIR register[4–1]. **Note– A read from the SSIVEC register clears the corresponding SSI_FLAG[4–1] bit in the SSIF, corresponding to the source vector of the system software interrupt.**
		0	*Read*– No IRQ/FIQ interrupt was generated since the bit was last cleared. *Write*– The bit is unchanged.
		1	*Read*– An IRQ/FIQ interrupt was generated. *Write*– The bit is cleared to 0.

5.1.46 System Software Interrupt Request1 register (SSIR1)

This register is shown in Figure 5-47 and described in Table 5-47.

Figure 5-47. System Software Interrupt Request 1 Register (SSIR1) [offset = FCh]



R = Read; W = Write in all modes; *-n* = Value after reset

.

Table 5-47. System Software Interrupt Request 1 Register (SSIR1) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–8	SSKEY1[7–0]	0–FFh	System software interrupt request key. A 75h written to these bits initiates IRQ/FIQ interrupts. Data in this field is always read as 0. The SSKEY1 field can be written into only if the write data matches the key (75h). The SSDATA1 field can only be written into if the write data into this field, the SSKEY1 field, matches the key (75h).
7–0	SSDATA1[7–0]	0–FFh	System software interrupt data. These bits contain user read/write register bits. They may be used by the application software as different entry points for the interrupt routine. The SSDATA1 field cannot be written into unless the write data into the SSKEY1 field matches the key (75h); therefore, byte writes cannot be performed on the SSDATA1 field.

Note:
This register is mirrored at offset FCh for compatibility reasons.

5.2 PCR control register

This section describes the peripheral central resource (PCR) registers. The start address of the PCR register frame is 0xFFFF E000. Table 5-48. illustrates the mapping of the PCR registers which are used to configure protection to the peripherals in the PCS and PS regions.

Table 5-48. PCR Control register map

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 PMPROTSET0 page 168	PCS 31 PROT SET	PCS 30 PROT SET	PCS 29 PROT SET	PCS 28 PROT SET	PCS 27 PROT SET	PCS 26 PROT SET	PCS 25 PROT SET	PCS 24 PROT SET	PCS 23 PROT SET	PCS 22 PROT SET	PCS 21 PROT SET	PCS 20 PROT SET	PCS 19 PROT SET	PCS 18 PROT SET	PCS 17 PROT SET	PCS 16 PROT SET
	PCS 15 PROT SET	PCS 14 PROT SET	PCS 13 PROT SET	PCS 12 PROT SET	PCS 11 PROT SET	PCS 10 PROT SET	PCS 9 PROT SET	PCS 8 PROT SET	PCS 7 PROT SET	PCS 6 PROT SET	PCS 5 PROT SET	PCS 4 PROT SET	PCS 3 PROT SET	PCS 2 PROT SET	PCS 1 PROT SET	PCS 0 PROT SET
0x04 PMPROTSET1 page 169	PCS 63 PROT SET	PCS 62 PROT SET	PCS 61 PROT SET	PCS 60 PROT SET	PCS 59 PROT SET	PCS 58 PROT SET	PCS 57 PROT SET	PCS 56 PROT SET	PCS 55 PROT SET	PCS 54 PROT SET	PCS 53 PROT SET	PCS 52 PROT SET	PCS 51 PROT SET	PCS 50 PROT SET	PCS 49 PROT SET	PCS 48 PROT SET
	PCS 47 PROT SET	PCS 46 PROT SET	PCS 45 PROT SET	PCS 44 PROT SET	PCS 43 PROT SET	PCS 42 PROT SET	PCS 41 PROT SET	PCS 40 PROT SET	PCS 39 PROT SET	PCS 38 PROT SET	PCS 37 PROT SET	PCS 36 PROT SET	PCS 35 PROT SET	PCS 34 PROT SET	PCS 33 PROT SET	PCS 32 PROT SET
0x08	Reserved															
	Reserved															
0x0C	Reserved															
	Reserved															
0x10 PMPROTCLR0 page 170	PCS 31 PROT CLR	PCS 30 PROT CLR	PCS 29 PROT CLR	PCS 28 PROT CLR	PCS 27 PROT CLR	PCS 26 PROT CLR	PCS 25 PROT CLR	PCS 24 PROT CLR	PCS 23 PROT CLR	PCS 22 PROT CLR	PCS 21 PROT CLR	PCS 20 PROT CLR	PCS 19 PROT CLR	PCS 18 PROT CLR	PCS 17 PROT CLR	PCS 16 PROT CLR
	PCS 15 PROT CLR	PCS 14 PROT CLR	PCS 13 PROT CLR	PCS 12 PROT CLR	PCS 11 PROT CLR	PCS 10 PROT CLR	PCS 9 PROT CLR	PCS 8 PROT CLR	PCS 7 PROT CLR	PCS 6 PROT CLR	PCS 5 PROT CLR	PCS 4 PROT CLR	PCS 3 PROT CLR	PCS 2 PROT CLR	PCS 1 PROT CLR	PCS 0 PROT CLR
0x14 PMPROTCLR1 page 171	PCS 63 PROT CLR	PCS 62 PROT CLR	PCS61 PROT CLR	PCS 60 PROT CLR	PCS 59 PROT CLR	PCS 58 PROT CLR	PCS 57 PROT CLR	PCS 56 PROT CLR	PCS 55 PROT CLR	PCS 54 PROT CLR	PCS 53 PROT CLR	PCS 52 PROT CLR	PCS 51 PROT CLR	PCS 50 PROT CLR	PCS 49 PROT CLR	PCS 48 PROT CLR
	PCS47 PROT CLR	PCS46 PROT CLR	PCS45 PROT CLR	PCS44 PROT CLR	PCS43 PROT CLR	PCS42 PROT CLR	PCS41 PROT CLR	PCS40 PROT CLR	PCS39 PROT CLR	PCS38 PROT CLR	PCS37 PROT CLR	PCS36 PROT CLR	PCS35 PROT CLR	PCS34 PROT CLR	PCS33 PROT CLR	PCS32 PROT CLR
0x18	Reserved															
	Reserved															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x1C	Reserved															
	Reserved															
0x20 PPROTSET0 page 172	PS7 QUAD 3 PROT SET	PS7 QUAD 2 PROT SET	PS7 QUAD 1 PROT SET	PS7 QUAD 0 PROT SET	PS6 QUAD 3 PROT SET	PS6 QUAD 2 PROT SET	PS6 QUAD 1 PROT SET	PS6 QUAD 0 PROT SET	PS5 QUAD 3 PROT SET	PS5 QUAD 2 PROT SET	PS5 QUAD 1 PROT SET	PS5 QUAD 0 PROT SET	PS4 QUAD 3 PROT SET	PS4 QUAD 2 PROT SET	PS4 QUAD 1 PROT SET	PS4 QUAD 0 PROT SET
	PS3 QUAD 3 PROT SET	PS3 QUAD 2 PROT SET	PS3 QUAD 1 PROT SET	PS3 QUAD 0 PROT SET	PS2 QUAD 3 PROT SET	PS2 QUAD 2 PROT SET	PS2 QUAD 1 PROT SET	PS2 QUAD 0 PROT SET	PS1 QUAD 3 PROT SET	PS1 QUAD 2 PROT SET	PS1 QUAD 1 PROT SET	PS1 QUAD 0 PROT SET	PS0 QUAD 3 PROT SET	PS0 QUAD 2 PROT SET	PS0 QUAD 1 PROT SET	PS0 QUAD 0 PROT SET
0x24 PPROTSET1 page 174	PS15 QUAD 3 PROT SET	PS15 QUAD 2 PROT SET	PS15 QUAD 1 PROT SET	PS15 QUAD 0 PROT SET	PS14 QUAD 3 PROT SET	PS14 QUAD 2 PROT SET	PS14 QUAD 1 PROT SET	PS14 QUAD 0 PROT SET	PS13 QUAD 3 PROT SET	PS13 QUAD 2 PROT SET	PS13 QUAD 1 PROT SET	PS13 QUAD 0 PROT SET	PS12 QUAD 3 PROT SET	PS12 QUAD 2 PROT SET	PS12 QUAD 1 PROT SET	PS12 QUAD 0 PROT SET
	PS11 QUAD 3 PROT SET	PS11 QUAD 2 PROT SET	PS11 QUAD 1 PROT SET	PS11 QUAD 0 PROT SET	PS10 QUAD 3 PROT SET	PS10 QUAD 2 PROT SET	PS10 QUAD 1 PROT SET	PS10 QUAD 0 PROT SET	PS9 QUAD 3 PROT SET	PS9 QUAD 2 PROT SET	PS9 QUAD 1 PROT SET	PS9 QUAD 0 PROT SET	PS8 QUAD 3 PROT SET	PS8 QUAD 2 PROT SET	PS8 QUAD 1 PROT SET	PS8 QUAD 0 PROT SET
0x28 PPROTSET2 page 175	PS23 QUAD 3 PROT SET	PS23 QUAD 2 PROT SET	PS23 QUAD 1 PROT SET	PS23 QUAD 0 PROT SET	PS22 QUAD 3 PROT SET	PS22 QUAD 2 PROT SET	PS22 QUAD 1 PROT SET	PS22 QUAD 0 PROT SET	PS21 QUAD 3 PROT SET	PS21 QUAD 2 PROT SET	PS21 QUAD 1 PROT SET	PS21 QUAD 0 PROT SET	PS20 QUAD 3 PROT SET	PS20 QUAD 2 PROT SET	PS20 QUAD 1 PROT SET	PS20 QUAD 0 PROT SET
	PS19 QUAD 3 PROT SET	PS19 QUAD 2 PROT SET	PS19 QUAD 1 PROT SET	PS19 QUAD 0 PROT SET	PS18 QUAD 3 PROT SET	PS18 QUAD 2 PROT SET	PS18 QUAD 1 PROT SET	PS18 QUAD 0 PROT SET	PS17 QUAD 3 PROT SET	PS17 QUAD 2 PROT SET	PS17 QUAD 1 PROT SET	PS17 QUAD 0 PROT SET	PS16 QUAD 3 PROT SET	PS16 QUAD 2 PROT SET	PS16 QUAD 1 PROT SET	PS16 QUAD 0 PROT SET
0x2C PPROTSET3 page 176	PS31 QUAD 3 PROT SET	PS31 QUAD 2 PROT SET	PS31 QUAD 1 PROT SET	PS31 QUAD 0 PROT SET	PS30 QUAD 3 PROT SET	PS30 QUAD 2 PROT SET	PS30 QUAD 1 PROT SET	PS30 QUAD 0 PROT SET	PS29 QUAD 3 PROT SET	PS29 QUAD 2 PROT SET	PS29 QUAD 1 PROT SET	PS29 QUAD 0 PROT SET	PS28 QUAD 3 PROT SET	PS28 QUAD 2 PROT SET	PS28 QUAD 1 PROT SET	PS28 QUAD 0 PROT SET
	PS27 QUAD 3 PROT SET	PS27 QUAD 2 PROT SET	PS27 QUAD 1 PROT SET	PS27 QUAD 0 PROT SET	PS26 QUAD 3 PROT SET	PS26 QUAD 2 PROT SET	PS26 QUAD 1 PROT SET	PS26 QUAD 0 PROT SET	PS25 QUAD 3 PROT SET	PS25 QUAD 2 PROT SET	PS25 QUAD 1 PROT SET	PS25 QUAD 0 PROT SET	PS24 QUAD 3 PROT SET	PS24 QUAD 2 PROT SET	PS24 QUAD 1 PROT SET	PS24 QUAD 0 PROT SET
0x30	Reserved															
	Reserved															
0x34	Reserved															
	Reserved															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x38	Reserved															
	Reserved															
0x3C	Reserved															
	Reserved															
0x40 PPROTCLR0 page 177	PS7 QUAD 3 PROT CLR	PS7 QUAD 2 PROT CLR	PS7 QUAD 1 PROT CLR	PS7 QUAD 0 PROT CLR	PS6 QUAD 3 PROT CLR	PS6 QUAD 2 PROT CLR	PS6 QUAD 1 PROT CLR	PS6 QUAD 0 PROT CLR	PS5 QUAD 3 PROT CLR	PS5 QUAD 2 PROT CLR	PS5 QUAD 1 PROT CLR	PS5 QUAD 0 PROT CLR	PS4 QUAD 3 PROT CLR	PS4 QUAD 2 PROT CLR	PS4 QUAD 1 PROT CLR	PS4 QUAD 0 PROT CLR
	PS3 QUAD 3 PROT CLR	PS3 QUAD 2 PROT CLR	PS3 QUAD 1 PROT CLR	PS3 QUAD 0 PROT CLR	PS2 QUAD 3 PROT CLR	PS2 QUAD 2 PROT CLR	PS2 QUAD 1 PROT CLR	PS2 QUAD 0 PROT CLR	PS1 QUAD 3 PROT CLR	PS1 QUAD 2 PROT CLR	PS1 QUAD 1 PROT CLR	PS1 QUAD 0 PROT CLR	PS0 QUAD 3 PROT CLR	PS0 QUAD 2 PROT CLR	PS0 QUAD 1 PROT CLR	PS0 QUAD 0 PROT CLR
0x44 PPROTCLR1 page 178	PS15 QUAD 3 PROT CLR	PS15 QUAD 2 PROT CLR	PS15 QUAD 1 PROT CLR	PS15 QUAD 0 PROT CLR	PS14 QUAD 3 PROT CLR	PS14 QUAD 2 PROT CLR	PS14 QUAD 1 PROT CLR	PS14 QUAD 0 PROT CLR	PS13 QUAD 3 PROT CLR	PS13 QUAD 2 PROT CLR	PS13 QUAD 1 PROT CLR	PS13 QUAD 0 PROT CLR	PS12 QUAD 3 PROT CLR	PS12 QUAD 2 PROT CLR	PS12 QUAD 1 PROT CLR	PS12 QUAD 0 PROT CLR
	PS11 QUAD 3 PROT CLR	PS11 QUAD 2 PROT CLR	PS11 QUAD 1 PROT CLR	PS11 QUAD 0 PROT CLR	PS10 QUAD 3 PROT CLR	PS10 QUAD 2 PROT CLR	PS10 QUAD 1 PROT CLR	PS10 QUAD 0 PROT CLR	PS9 QUAD 3 PROT CLR	PS9 QUAD 2 PROT CLR	PS9 QUAD 1 PROT CLR	PS9 QUAD 0 PROT CLR	PS8 QUAD 3 PROT CLR	PS8 QUAD 2 PROT CLR	PS8 QUAD 1 PROT CLR	PS8 QUAD 0 PROT CLR
0x48 PPROTCLR2 page 179	PS23 QUAD 3 PROT CLR	PS23 QUAD 2 PROT CLR	PS23 QUAD 1 PROT CLR	PS23 QUAD 0 PROT CLR	PS22 QUAD 3 PROT CLR	PS22 QUAD 2 PROT CLR	PS22 QUAD 1 PROT CLR	PS22 QUAD 0 PROT CLR	PS21 QUAD 3 PROT CLR	PS21 QUAD 2 PROT CLR	PS21 QUAD 1 PROT CLR	PS21 QUAD 0 PROT CLR	PS20 QUAD 3 PROT CLR	PS20 QUAD 2 PROT CLR	PS20 QUAD 1 PROT CLR	PS20 QUAD 0 PROT CLR
	PS19 QUAD 3 PROT CLR	PS19 QUAD 2 PROT CLR	PS19 QUAD 1 PROT CLR	PS19 QUAD 0 PROT CLR	PS18 QUAD 3 PROT CLR	PS18 QUAD 2 PROT CLR	PS18 QUAD 1 PROT CLR	PS18 QUAD 0 PROT CLR	PS17 QUAD 3 PROT CLR	PS17 QUAD 2 PROT CLR	PS17 QUAD 1 PROT CLR	PS17 QUAD 0 PROT CLR	PS16 QUAD 3 PROT CLR	PS16 QUAD 2 PROT CLR	PS16 QUAD 1 PROT CLR	PS16 QUAD 0 PROT CLR
0x4C PPROTCLR3 page 180	PS31 QUAD 3 PROT CLR	PS31 QUAD 2 PROT CLR	PS31 QUAD 1 PROT CLR	PS31 QUAD 0 PROT CLR	PS30 QUAD 3 PROT CLR	PS30 QUAD 2 PROT CLR	PS30 QUAD 1 PROT CLR	PS30 QUAD 0 PROT CLR	PS29 QUAD 3 PROT CLR	PS29 QUAD 2 PROT CLR	PS29 QUAD 1 PROT CLR	PS29 QUAD 0 PROT CLR	PS28 QUAD 3 PROT CLR	PS28 QUAD 2 PROT CLR	PS28 QUAD 1 PROT CLR	PS28 QUAD 0 PROT CLR
	PS27 QUAD 3 PROT CLR	PS27 QUAD 2 PROT CLR	PS27 QUAD 1 PROT CLR	PS27 QUAD 0 PROT CLR	PS26 QUAD 3 PROT CLR	PS26 QUAD 2 PROT CLR	PS26 QUAD 1 PROT CLR	PS26 QUAD 0 PROT CLR	PS25 QUAD 3 PROT CLR	PS25 QUAD 2 PROT CLR	PS25 QUAD 1 PROT CLR	PS25 QUAD 0 PROT CLR	PS24 QUAD 3 PROT CLR	PS24 QUAD 2 PROT CLR	PS24 QUAD 1 PROT CLR	PS24 QUAD 0 PROT CLR
0x50	Reserved															
	Reserved															

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x54	Reserved															
	Reserved															
0x58	Reserved															
	Reserved															
0x5C	Reserved															
	Reserved															
0x60 PCSPWRDWNSET0 page 181	PCS31 PWRDWN SET	PCS30 PWRDWN SET	PCS29 PWRDWN SET	PCS28 PWRDWN SET	PCS27 PWRDWN SET	PCS26 PWRDWN SET	PCS25 PWRDWN SET	PCS24 PWRDWN SET	PCS23 PWRDWN SET	PCS22 PWRDWN SET	PCS21 PWRDWN SET	PCS20 PWRDWN SET	PCS19 PWRDWN SET	PCS18 PWRDWN SET	PCS17 PWRDWN SET	PCS16 PWRDWN SET
	PCS15 PWRDWN SET	PCS14 PWRDWN SET	PCS13 PWRDWN SET	PCS12 PWRDWN SET	PCS11 PWRDWN SET	PCS10 PWRDWN SET	PCS9 PWRDWN SET	PCS8 PWRDWN SET	PCS7 PWRDWN SET	PCS6 PWRDWN SET	PCS5 PWRDWN SET	PCS4 PWRDWN SET	PCS3 PWRDWN SET	PCS2 PWRDWN SET	PCS1 PWRDWN SET	PCS0 PWRDWN SET
0x64 PCSPWRDWNSET1 page 182	PCS63 PWRDWN SET	PCS62 PWRDWN SET	PCS61 PWRDWN SET	PCS60 PWRDWN SET	PCS59 PWRDWN SET	PCS58 PWRDWN SET	PCS57 PWRDWN SET	PCS56 PWRDWN SET	PCS55 PWRDWN SET	PCS54 PWRDWN SET	PCS53 PWRDWN SET	PCS52 PWRDWN SET	PCS51 PWRDWN SET	PCS50 PWRDWN SET	PCS49 PWRDWN SET	PCS48 PWRDWN SET
	PCS47 PWRDWN SET	PCS46 PWRDWN SET	PCS45 PWRDWN SET	PCS44 PWRDWN SET	PCS43 PWRDWN SET	PCS42 PWRDWN SET	PCS41 PWRDWN SET	PCS40 PWRDWN SET	PCS39 PWRDWN SET	PCS38 PWRDWN SET	PCS37 PWRDWN SET	PCS36 PWRDWN SET	PCS35 PWRDWN SET	PCS34 PWRDWN SET	PCS33 PWRDWN SET	PCS32 PWRDWN SET
0x68	Reserved															
	Reserved															
0x6C	Reserved															
	Reserved															
0x70 PCSPWRDWNCLR0 page 183	PCS31 PWRDWN CLR	PCS30 PWRDWN CLR	PCS29 PWRDWN CLR	PCS28 PWRDWN CLR	PCS27 PWRDWN CLR	PCS26 PWRDWN CLR	PCS25 PWRDWN CLR	PCS24 PWRDWN CLR	PCS23 PWRDWN CLR	PCS22 PWRDWN CLR	PCS21 PWRDWN CLR	PCS20 PWRDWN CLR	PCS19 PWRDWN CLR	PCS18 PWRDWN CLR	PCS17 PWRDWN CLR	PCS16 PWRDWN CLR
	PCS15 PWRDWN CLR	PCS14 PWRDWN CLR	PCS13 PWRDWN CLR	PCS12 PWRDWN CLR	PCS11 PWRDWN CLR	PCS10 PWRDWN CLR	PCS9 PWRDWN CLR	PCS8 PWRDWN CLR	PCS7 PWRDWN CLR	PCS6 PWRDWN CLR	PCS5 PWRDWN CLR	PCS4 PWRDWN CLR	PCS3 PWRDWN CLR	PCS2 PWRDWN CLR	PCS1 PWRDWN CLR	PCS0 PWRDWN CLR

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x74 PCSPWRDWNCLR1 page 184	PCS63 PWRDWN CLR	PCS62 PWRDWN CLR	PCS61 PWRDWN CLR	PCS60 PWRDWN CLR	PCS59 PWRDWN CLR	PCS58 PWRDWN CLR	PCS57 PWRDWN CLR	PCS56 PWRDWN CLR	PCS55 PWRDWN CLR	PCS54 PWRDWN CLR	PCS53 PWRDWN CLR	PCS52 PWRDWN CLR	PCS51 PWRDWN CLR	PCS50 PWRDWN CLR	PCS49 PWRDWN CLR	PCS48 PWRDWN CLR
	PCS47 PWRDWN CLR	PCS46 PWRDWN CLR	PCS45 PWRDWN CLR	PCS44 PWRDWN CLR	PCS43 PWRDWN CLR	PCS42 PWRDWN CLR	PCS41 PWRDWN CLR	PCS40 PWRDWN CLR	PCS39 PWRDWN CLR	PCS38 PWRDWN CLR	PCS37 PWRDWN CLR	PCS36 PWRDWN CLR	PCS35 PWRDWN CLR	PCS34 PWRDWN CLR	PCS33 PWRDWN CLR	PCS32 PWRDWN CLR
0x78	Reserved															
	Reserved															
0x7C	Reserved															
	Reserved															
0x80 PSPWRDWNSET0 page 185	PS7 QUAD3 PWRDWN SET	PS7 QUAD2 PWRDWN SET	PS7 QUAD1 PWRDWN SET	PS7 QUAD0 PWRDWN SET	PS6 QUAD3 PWRDWN SET	PS6 QUAD2 PWRDWN SET	PS6 QUAD1 PWRDWN SET	PS6 QUAD0 PWRDWN SET	PS5 QUAD3 PWRDWN SET	PS5 QUAD2 PWRDWN SET	PS5 QUAD1 PWRDWN SET	PS5 QUAD0 PWRDWN SET	PS4 QUAD3 PWRDWN SET	PS4 QUAD2 PWRDWN SET	PS4 QUAD1 PWRDWN SET	PS4 QUAD0 PWRDWN SET
	PS3 QUAD3 PWRDWN SET	PS3 QUAD2 PWRDWN SET	PS3 QUAD1 PWRDWN SET	PS3 QUAD0 PWRDWN SET	PS2 QUAD3 PWRDWN SET	PS2 QUAD2 PWRDWN SET	PS2 QUAD1 PWRDWN SET	PS2 QUAD0 PWRDWN SET	PS1 QUAD3 PWRDWN SET	PS1 QUAD2 PWRDWN SET	PS1 QUAD1 PWRDWN SET	PS1 QUAD0 PWRDWN SET	PS0 QUAD3 PWRDWN SET	PS0 QUAD2 PWRDWN SET	PS0 QUAD1 PWRDWN SET	PS0 QUAD0 PWRDWN SET
0x84 PSPWRDWNSET1 page 186	PS15 QUAD3 PWRDWN SET	PS15 QUAD2 PWRDWN SET	PS15 QUAD1 PWRDWN SET	PS15 QUAD0 PWRDWN SET	PS14 QUAD3 PWRDWN SET	PS14 QUAD2 PWRDWN SET	PS14 QUAD1 PWRDWN SET	PS14 QUAD0 PWRDWN SET	PS13 QUAD3 PWRDWN SET	PS13 QUAD2 PWRDWN SET	PS13 QUAD1 PWRDWN SET	PS13 QUAD0 PWRDWN SET	PS12 QUAD3 PWRDWN SET	PS12 QUAD2 PWRDWN SET	PS12 QUAD1 PWRDWN SET	PS12 QUAD0 PWRDWN SET
	PS11 QUAD3 PWRDWN SET	PS11 QUAD2 PWRDWN SET	PS11 QUAD1 PWRDWN SET	PS11 QUAD0 PWRDWN SET	PS10 QUAD3 PWRDWN SET	PS10 QUAD2 PWRDWN SET	PS10 QUAD1 PWRDWN SET	PS10 QUAD0 PWRDWN SET	PS9 QUAD3 PWRDWN SET	PS9 QUAD2 PWRDWN SET	PS9 QUAD1 PWRDWN SET	PS9 QUAD0 PWRDWN SET	PS8 QUAD3 PWRDWN SET	PS8 QUAD2 PWRDWN SET	PS8 QUAD1 PWRDWN SET	PS8 QUAD0 PWRDWN SET
0x88 PSPWRDWNSET2 page 187	PS23 QUAD3 PWRDWN SET	PS23 QUAD2 PWRDWN SET	PS23 QUAD1 PWRDWN SET	PS23 QUAD0 PWRDWN SET	PS22 QUAD3 PWRDWN SET	PS22 QUAD2 PWRDWN SET	PS22 QUAD1 PWRDWN SET	PS22 QUAD0 PWRDWN SET	PS21 QUAD3 PWRDWN SET	PS21 QUAD2 PWRDWN SET	PS21 QUAD1 PWRDWN SET	PS21 QUAD0 PWRDWN SET	PS20 QUAD3 PWRDWN SET	PS20 QUAD2 PWRDWN SET	PS20 QUAD1 PWRDWN SET	PS20 QUAD0 PWRDWN SET
	PS19 QUAD3 PWRDWN SET	PS19 QUAD2 PWRDWN SET	PS19 QUAD1 PWRDWN SET	PS19 QUAD0 PWRDWN SET	PS18 QUAD3 PWRDWN SET	PS18 QUAD2 PWRDWN SET	PS18 QUAD1 PWRDWN SET	PS18 QUAD0 PWRDWN SET	PS17 QUAD3 PWRDWN SET	PS17 QUAD2 PWRDWN SET	PS17 QUAD1 PWRDWN SET	PS17 QUAD0 PWRDWN SET	PS16 QUAD3 PWRDWN SET	PS16 QUAD2 PWRDWN SET	PS16 QUAD1 PWRDWN SET	PS16 QUAD0 PWRDWN SET

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x8C PSPWRDWNSET3 page 188	PS31 QUAD 3 PWRDWN SET	PS31 QUAD 2 PWRDWN SET	PS31 QUAD 1 PWRDWN SET	PS31 QUAD 0 PWRDWN SET	PS30 QUAD 3 PWRDWN SET	PS30 QUAD 2 PWRDWN SET	PS30 QUAD 1 PWRDWN SET	PS30 QUAD 0 PWRDWN SET	PS29 QUAD 3 PWRDWN SET	PS29 QUAD 2 PWRDWN SET	PS29 QUAD 1 PWRDWN SET	PS29 QUAD 0 PWRDWN SET	PS28 QUAD 3 PWRDWN SET	PS28 QUAD 2 PWRDWN SET	PS28 QUAD 1 PWRDWN SET	PS28 QUAD 0 PWRDWN SET
	PS27 QUAD 3 PWRDWN SET	PS27 QUAD 2 PWRDWN SET	PS27 QUAD 1 PWRDWN SET	PS27 QUAD 0 PWRDWN SET	PS26 QUAD 3 PWRDWN SET	PS26 QUAD 2 PWRDWN SET	PS26 QUAD 1 PWRDWN SET	PS26 QUAD 0 PWRDWN SET	PS25 QUAD 3 PWRDWN SET	PS25 QUAD 2 PWRDWN SET	PS25 QUAD 1 PWRDWN SET	PS25 QUAD 0 PWRDWN SET	PS24 QUAD 3 PWRDWN SET	PS24 QUAD 2 PWRDWN SET	PS24 QUAD 1 PWRDWN SET	PS24 QUAD 0 PWRDWN SET
0x90	Reserved															
	Reserved															
0x94	Reserved															
	Reserved															
0x98	Reserved															
	Reserved															
0x9C	Reserved															
	Reserved															
0xA0 PSPWRDWNCLR0 page 189	PS7 QUAD 3 PWRDWN CLR	PS7 QUAD 2 PWRDWN CLR	PS7 QUAD 1 PWRDWN CLR	PS7 QUAD 0 PWRDWN CLR	PS6 QUAD 3 PWRDWN CLR	PS6 QUAD 2 PWRDWN CLR	PS6 QUAD 1 PWRDWN CLR	PS6 QUAD 0 PWRDWN CLR	PS5 QUAD 3 PWRDWN CLR	PS5 QUAD 2 PWRDWN CLR	PS5 QUAD 1 PWRDWN CLR	PS5 QUAD 0 PWRDWN CLR	PS4 QUAD 3 PWRDWN CLR	PS4 QUAD 2 PWRDWN CLR	PS4 QUAD 1 PWRDWN CLR	PS4 QUAD 0 PWRDWN CLR
	PS3 QUAD 3 PWRDWN CLR	PS3 QUAD 2 PWRDWN CLR	PS3 QUAD 1 PWRDWN CLR	PS3 QUAD 0 PWRDWN CLR	PS2 QUAD 3 PWRDWN CLR	PS2 QUAD 2 PWRDWN CLR	PS2 QUAD 1 PWRDWN CLR	PS2 QUAD 0 PWRDWN CLR	PS1 QUAD 3 PWRDWN CLR	PS1 QUAD 2 PWRDWN CLR	PS1 QUAD 1 PWRDWN CLR	PS1 QUAD 0 PWRDWN CLR	PS0 QUAD 3 PWRDWN CLR	PS0 QUAD 2 PWRDWN CLR	PS0 QUAD 1 PWRDWN CLR	PS0 QUAD 0 PWRDWN CLR
0xA4 PSPWRDWNCLR1 page 190	PS15 QUAD 3 PWRDWN CLR	PS15 QUAD 2 PWRDWN CLR	PS15 QUAD 1 PWRDWN CLR	PS15 QUAD 0 PWRDWN CLR	PS14 QUAD 3 PWRDWN CLR	PS14 QUAD 2 PWRDWN CLR	PS14 QUAD 1 PWRDWN CLR	PS14 QUAD 0 PWRDWN CLR	PS13 QUAD 3 PWRDWN CLR	PS13 QUAD 2 PWRDWN CLR	PS13 QUAD 1 PWRDWN CLR	PS13 QUAD 0 PWRDWN CLR	PS12 QUAD 3 PWRDWN CLR	PS12 QUAD 2 PWRDWN CLR	PS12 QUAD 1 PWRDWN CLR	PS12 QUAD 0 PWRDWN CLR
	PS11 QUAD 3 PWRDWN CLR	PS11 QUAD 2 PWRDWN CLR	PS11 QUAD 1 PWRDWN CLR	PS11 QUAD 0 PWRDWN CLR	PS10 QUAD 3 PWRDWN CLR	PS10 QUAD 2 PWRDWN CLR	PS10 QUAD 1 PWRDWN CLR	PS10 QUAD 0 PWRDWN CLR	PS9 QUAD 3 PWRDWN CLR	PS9 QUAD 2 PWRDWN CLR	PS9 QUAD 1 PWRDWN CLR	PS9 QUAD 0 PWRDWN CLR	PS8 QUAD 3 PWRDWN CLR	PS8 QUAD 2 PWRDWN CLR	PS8 QUAD 1 PWRDWN CLR	PS8 QUAD 0 PWRDWN CLR

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xA8 PSPWRDWNCLR2 page 191	PS23 QUAD 3 PWRDWN CLR	PS23 QUAD 2 PWRDWN CLR	PS23 QUAD 1 PWRDWN CLR	PS23 QUAD 0 PWRDWN CLR	PS22 QUAD 3 PWRDWN CLR	PS22 QUAD 2 PWRDWN CLR	PS22 QUAD 1 PWRDWN CLR	PS22 QUAD 0 PWRDWN CLR	PS21 QUAD 3 PWRDWN CLR	PS21 QUAD 2 PWRDWN CLR	PS21 QUAD 1 PWRDWN CLR	PS21 QUAD 0 PWRDWN CLR	PS20 QUAD 3 PWRDWN CLR	PS20 QUAD 2 PWRDWN CLR	PS20 QUAD 1 PWRDWN CLR	PS20 QUAD 0 PWRDWN CLR
	PS19 QUAD 3 PWRDWN CLR	PS19 QUAD 2 PWRDWN CLR	PS19 QUAD 1 PWRDWN CLR	PS19 QUAD 0 PWRDWN CLR	PS18 QUAD 3 PWRDWN CLR	PS18 QUAD 2 PWRDWN CLR	PS18 QUAD 1 PWRDWN CLR	PS18 QUAD 0 PWRDWN CLR	PS17 QUAD 3 PWRDWN CLR	PS17 QUAD 2 PWRDWN CLR	PS17 QUAD 1 PWRDWN CLR	PS17 QUAD 0 PWRDWN CLR	PS16 QUAD 3 PWRDWN CLR	PS16 QUAD 2 PWRDWN CLR	PS16 QUAD 1 PWRDWN CLR	PS16 QUAD 0 PWRDWN CLR
0xAC PSPWRDWNCLR3 page 192	PS31 QUAD 3 PWRDWN CLR	PS31 QUAD 2 PWRDWN CLR	PS31 QUAD 1 PWRDWN CLR	PS31 QUAD 0 PWRDWN CLR	PS30 QUAD 3 PWRDWN CLR	PS30 QUAD 2 PWRDWN CLR	PS30 QUAD 1 PWRDWN CLR	PS30 QUAD 0 PWRDWN CLR	PS29 QUAD 3 PWRDWN CLR	PS29 QUAD 2 PWRDWN CLR	PS29 QUAD 1 PWRDWN CLR	PS29 QUAD 0 PWRDWN CLR	PS28 QUAD 3 PWRDWN CLR	PS28 QUAD 2 PWRDWN CLR	PS28 QUAD 1 PWRDWN CLR	PS28 QUAD 0 PWRDWN CLR
	PS27 QUAD 3 PWRDWN CLR	PS27 QUAD 2 PWRDWN CLR	PS27 QUAD 1 PWRDWN CLR	PS27 QUAD 0 PWRDWN CLR	PS26 QUAD 3 PWRDWN CLR	PS26 QUAD 2 PWRDWN CLR	PS26 QUAD 1 PWRDWN CLR	PS26 QUAD 0 PWRDWN CLR	PS25 QUAD 3 PWRDWN CLR	PS25 QUAD 2 PWRDWN CLR	PS25 QUAD 1 PWRDWN CLR	PS25 QUAD 0 PWRDWN CLR	PS24 QUAD 3 PWRDWN CLR	PS24 QUAD 2 PWRDWN CLR	PS24 QUAD 1 PWRDWN CLR	PS24 QUAD 0 PWRDWN CLR

5.2.1 Peripheral Memory Protection Set Register 0 (PMPROTSET0)

This register is shown in Figure 5-48 and described in Table 5-49.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to non-implemented bits have no effect and reads are 0.

Figure 5-48. Peripheral Memory Protection Set Register 0 (PMPROTSET0) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS31 PROT SET	PCS30 PROT SET	PCS29 PROT SET	PCS28 PROT SET	PCS27 PROT SET	PCS26 PROT SET	PCS25 PROT SET	PCS24 PROT SET	PCS23 PROT SET	PCS22 PROT SET	PCS21 PROT SET	PCS20 PROT SET	PCS19 PROT SET	PCS18 PROT SET	PCS17 PROT SET	PCS16 PROT SET
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS15 PROT SET	PCS14 PROT SET	PCS13 PROT SET	PCS12 PROT SET	PCS11 PROT SET	PCS10 PROT SET	PCS9 PROT SET	PCS8 PROT SET	PCS7 PROT SET	PCS6 PROT SET	PCS5 PROT SET	PCS4 PROT SET	PCS3 PROT SET	PCS2 PROT SET	PCS1 PROT SET	PCS0 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

Table 5-49. Peripheral Memory Protection Set Register 0 (PMPROTSET0) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0]PROTSET		Peripheral memory frame protection set.
		0	*Read*– The peripheral memory frame *n* can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory frame *n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write*– The corresponding bit in PMPROTSET0 and PMPROTCLR0 registers is set to 1.

5.2.2 Peripheral Memory Protection Set Register 1 (PMPROTSET1)

This register is shown in Figure 5-49 and described in Table 5-50.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-49. Peripheral Memory Protection Set Register 1 (PMPROTSET1) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS63 PROT SET	PCS62 PROT SET	PCS61 PROT SET	PCS60 PROT SET	PCS59 PROT SET	PCS58 PROT SET	PCS57 PROT SET	PCS56 PROT SET	PCS55 PROT SET	PCS54 PROT SET	PCS53 PROT SET	PCS52 PROT SET	PCS51 PROT SET	PCS50 PROT SET	PCS49 PROT SET	PCS48 PROT SET
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PCS47 PROT SET	PCS46 PROT SET	PCS45 PROT SET	PCS44 PROT SET	PCS43 PROT SET	PCS42 PROT SET	PCS41 PROT SET	PCS40 PROT SET	PCS39 PROT SET	PCS38 PROT SET	PCS37 PROT SET	PCS36 PROT SET	PCS35 PROT SET	PCS34 PROT SET	PCS33 PROT SET	PCS32 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 5-50. Peripheral Memory Protection Set Register 1 (PMPROTSET1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[63–32]PROTSET		Peripheral memory frame protection set.
		0	*Read*– The peripheral memory frame *n* can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory frame *n* can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write*– The corresponding bit in PMPROTSET1 and PMPROTCLR1 registers is set to 1.

5.2.3 Peripheral Memory Protection Clear Register 0 (PMPROTCLR0)

This register is shown in Figure 5-50 and described in Table 5-51.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-50. Peripheral Memory Protection Clear Register 0 (PMPROTCLR0) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS31 PROT CLR	PCS30 PROT CLR	PCS29 PROT CLR	PCS28 PROT CLR	PCS27 PROT CLR	PCS26 PROT CLR	PCS25 PROT CLR	PCS24 PROT CLR	PCS23 PROT CLR	PCS22 PROT CLR	PCS21 PROT CLR	PCS20 PROT CLR	PCS19 PROT CLR	PCS18 PROT CLR	PCS17 PROT CLR	PCS16 PROT CLR
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS15 PROT CLR	PCS14 PROT CLR	PCS13 PROT CLR	PCS12 PROT CLR	PCS11 PROT CLR	PCS10 PROT CLR	PCS9 PROT CLR	PCS8 PROT CLR	PCS7 PROT CLR	PCS6 PROT CLR	PCS5 PROT CLR	PCS4 PROT CLR	PCS3 PROT CLR	PCS2 PROT CLR	PCS1 PROT CLR	PCS0 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-51. Peripheral Memory Protection Clear Register 0 (PMPROTCLR0) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0]PROTCLR		Peripheral memory frame protection set.
		0	*Read*– The peripheral memory frame[31–0] can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory frame[31–0] can be written to only in privileged mode, but it can be read in both user and privileged modes. Write– The corresponding bit in PMPROTSET0 and PMPROTCLR0 registers is cleared to 0.

5.2.4 Peripheral Memory Protection Clear Register 1 (PMPROTCLR1)

This register is shown in Figure 5-51 and described in Table 5-52.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-51. Peripheral Memory Protection Clear Register 1 (PMPROTCLR1) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS63 PROT CLR	PCS62 PROT CLR	PCS61 PROT CLR	PCS60 PROT CLR	PCS59 PROT CLR	PCS58 PROT CLR	PCS57 PROT CLR	PCS56 PROT CLR	PCS55 PROT CLR	PCS54 PROT CLR	PCS53 PROT CLR	PCS52 PROT CLR	PCS51 PROT CLR	PCS50 PROT CLR	PCS49 PROT CLR	PCS48 PROT CLR
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS47 PROT CLR	PCS46 PROT CLR	PCS45 PROT CLR	PCS44 PROT CLR	PCS43 PROT CLR	PCS42 PROT CLR	PCS41 PROT CLR	PCS40 PROT CLR	PCS39 PROT CLR	PCS38 PROT CLR	PCS37 PROT CLR	PCS36 PROT CLR	PCS35 PROT CLR	PCS34 PROT CLR	PCS33 PROT CLR	PCS32 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-52. Peripheral Memory Protection Clear Register 1 (PMPROTCLR1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[63–32]PROTCLR		Peripheral memory frame protection clear.
		0	*Read*– The peripheral memory frame[63–32] can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory frame[63–32] can be written to only in privileged mode, but it can be read in both user and privileged modes. *Write*– The corresponding bit in PMPROTSET1 and PMPROTCLR1 registers is cleared to 0.

5.2.5 Peripheral Protection Set Register 0 (PPROTSET0)

There is one bit for each quadrant for PS0 to PS7. (See section 1.3.1 for quadrant definition.)

The following are the ways in which quadrants are used within a PS frame–

a. The slave uses all the four quadrants
   Only the bit corresponding to the quadrant 0 of PSn is implemented. It protects the whole 1K-byte frame. The remaining three bits are not implemented.

b. The slave uses two quadrants
   Each quadrant has to be in one of these groups– (Quad 0 and Quad 1), or (Quad 2 and Quad 3).
   For the group Quad0/Quad1, the bit quadrant 0 protects both quadrants 0 and 1. The bit quadrant 1 is not implemented.
   For the group Quad2/Quad3, the bit quadrant 2 protects both quadrants 2 and 3. The bit quadrant 3 is not implemented.

c. The slave uses only one quadrant.
   In this case, the bit as specified in Table 5-53 protects the slave.

The above arrangement is true for all the peripheral select (PS0 to PS31), presented in section 5.2.6—section 5.2.12. This register holds bits for PS0 to PS7 and is shown in Figure 5-52 and described in Table 5-53.

Note:
Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-52. Peripheral Protection Set Register 0 (PPROTSET0) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS7 QUAD3 PROT SET	PS7 QUAD2 PROT SET	PS7 QUAD1 PROT SET	PS7 QUAD0 PROT SET	PS6 QUAD3 PROT SET	PS6 QUAD2 PROT SET	PS6 QUAD1 PROT SET	PS6 QUAD0 PROT SET	PS5 QUAD3 PROT SET	PS5 QUAD2 PROT SET	PS5 QUAD1 PROT SET	PS5 QUAD0 PROT SET	PS4 QUAD3 PROT SET	PS4 QUAD2 PROT SET	PS4 QUAD1 PROT SET	PS4 QUAD0 PROT SET
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS3 QUAD3 PROT SET	PS3 QUAD2 PROT SET	PS3 QUAD1 PROT SET	PS3 QUAD0 PROT SET	PS2 QUAD3 PROT SET	PS2 QUAD2 PROT SET	PS2 QUAD1 PROT SET	PS2 QUAD0 PROT SET	PS1 QUAD3 PROT SET	PS1 QUAD2 PROT SET	PS1 QUAD1 PROT SET	PS1 QUAD0 PROT SET	PS0 QUAD3 PROT SET	PS0 QUAD2 PROT SET	PS0 QUAD1 PROT SET	PS0 QUAD0 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-53. Peripheral Protection Set Register 0 (PPROTSET0)) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[7–0]QUAD[3–0] PROTSET		Peripheral select quadrant protection set,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PMPROTSET0 and PMPROTCLR0 registers is set to 1.

5.2.6 Peripheral Protection Set Register 1 (PPROTSET1)

There is one bit for each quadrant for PS8 to PS15 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-53 and described in Table 5-54.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-53. Peripheral Protection Set Register 1 (PPROTSET1) [offset = 24h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS15 QUAD3 PROT SET	PS15 QUAD2 PROT SET	PS15 QUAD1 PROT SET	PS15 QUAD0 PROT SET	PS14 QUAD3 PROT SET	PS14 QUAD2 PROT SET	PS14 QUAD1 PROT SET	PS14 QUAD0 PROT SET	PS13 QUAD3 PROT SET	PS13 QUAD2 PROT SET	PS13 QUAD1 PROT SET	PS13 QUAD0 PROT SET	PS12 QUAD3 PROT SET	PS12 QUAD2 PROT SET	PS12 QUAD1 PROT SET	PS12 QUAD0 PROT SET
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS11 QUAD3 PROT SET	PS11 QUAD2 PROT SET	PS11 QUAD1 PROT SET	PS11 QUAD0 PROT SET	PS10 QUAD3 PROT SET	PS10 QUAD2 PROT SET	PS10 QUAD1 PROT SET	PS10 QUAD0 PROT SET	PS9 QUAD3 PROT SET	PS9 QUAD2 PROT SET	PS9 QUAD1 PROT SET	PS9 QUAD0 PROT SET	PS8 QUAD3 PROT SET	PS8 QUAD2 PROT SET	PS8 QUAD1 PROT SET	PS8 QUAD0 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-54. Peripheral Protection Set Register 1 (PPROTSET1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[15–8]QUAD[3–0] PROTSET		Peripheral select quadrant protection set,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PMPROTSET1 and PMPROTCLR1 registers is set to 1.

5.2.7 Peripheral Protection Set Register 2 (PPROTSET2)

There is one bit for each quadrant for PS16 to PS23 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-54 and described in Table 5-55.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-54. Peripheral Protection Set Register 2 (PPROTSET2) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS23 QUAD3 PROT SET	PS23 QUAD2 PROT SET	PS23 QUAD1 PROT SET	PS23 QUAD0 PROT SET	PS22 QUAD3 PROT SET	PS22 QUAD2 PROT SET	PS22 QUAD1 PROT SET	PS22 QUAD0 PROT SET	PS21 QUAD3 PROT SET	PS21 QUAD2 PROT SET	PS21 QUAD1 PROT SET	PS21 QUAD0 PROT SET	PS20 QUAD3 PROT SET	PS20 QUAD2 PROT SET	PS20 QUAD1 PROT SET	PS20 QUAD0 PROT SET
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS19 QUAD3 PROT SET	PS19 QUAD2 PROT SET	PS19 QUAD1 PROT SET	PS19 QUAD0 PROT SET	PS18 QUAD3 PROT SET	PS18 QUAD2 PROT SET	PS18 QUAD1 PROT SET	PS18 QUAD0 PROT SET	PS17 QUAD3 PROT SET	PS17 QUAD2 PROT SET	PS17 QUAD1 PROT SET	PS17 QUAD0 PROT SET	PS16 QUAD3 PROT SET	PS16 QUAD2 PROT SET	PS16 QUAD1 PROT SET	PS16 QUAD0 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-55. Peripheral Protection Set Register 2 (PPROTSET2) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[23–16]QUAD[3–0] PROTSET		Peripheral select quadrant protection set,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PMPROTSET2 and PMPROTCLR2 registers is set to 1.

5.2.8 Peripheral Protection Set Register 3 (PPROTSET3)

There is one bit for each quadrant for PS24 to PS31 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-55 and described in Table 5-56.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-55. Peripheral Protection Set Register 3 (PPROTSET3) [offset = 2Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS31 QUAD3 PROT SET	PS31 QUAD2 PROT SET	PS31 QUAD1 PROT SET	PS31 QUAD0 PROT SET	PS30 QUAD3 PROT SET	PS30 QUAD2 PROT SET	PS30 QUAD1 PROT SET	PS30 QUAD0 PROT SET	PS29 QUAD3 PROT SET	PS29 QUAD2 PROT SET	PS29 QUAD1 PROT SET	PS29 QUAD0 PROT SET	PS28 QUAD3 PROT SET	PS28 QUAD2 PROT SET	PS28 QUAD1 PROT SET	PS28 QUAD0 PROT SET
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS27 QUAD3 PROT SET	PS27 QUAD2 PROT SET	PS27 QUAD1 PROT SET	PS27 QUAD0 PROT SET	PS26 QUAD3 PROT SET	PS26 QUAD2 PROT SET	PS26 QUAD1 PROT SET	PS26 QUAD0 PROT SET	PS25 QUAD3 PROT SET	PS25 QUAD2 PROT SET	PS25 QUAD1 PROT SET	PS25 QUAD0 PROT SET	PS24 QUAD3 PROT SET	PS24 QUAD2 PROT SET	PS24 QUAD1 PROT SET	PS24 QUAD0 PROT SET
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-56. Peripheral Protection Set Register 3 (PPROTSET3) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0]QUAD[3–0] PROTSET		Peripheral select quadrant protection set,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PMPROTSET3 and PMPROTCLR3 registers is set to 1.

5.2.9 Peripheral Protection Clear Register 0 (PPROTCLR0)

There is one bit for each quadrant for PS0 to PS7 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-56 and described in Table 5-57.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-56. Peripheral Protection Clear Register 0 (PPROTCLR0) [offset = 40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS7 QUAD3 PROT CLR	PS7 QUAD2 PROT CLR	PS7 QUAD1 PROT CLR	PS7 QUAD0 PROT CLR	PS6 QUAD3 PROT CLR	PS6 QUAD2 PROT CLR	PS6 QUAD1 PROT CLR	PS6 QUAD0 PROT CLR	PS5 QUAD3 PROT CLR	PS5 QUAD2 PROT CLR	PS5 QUAD1 PROT CLR	PS5 QUAD0 PROT CLR	PS4 QUAD3 PROT CLR	PS4 QUAD2 PROT CLR	PS4 QUAD1 PROT CLR	PS4 QUAD0 PROT CLR
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS3 QUAD3 PROT CLR	PS3 QUAD2 PROT CLR	PS3 QUAD1 PROT CLR	PS3 QUAD0 PROT CLR	PS2 QUAD3 PROT CLR	PS2 QUAD2 PROT CLR	PS2 QUAD1 PROT CLR	PS2 QUAD0 PROT CLR	PS1 QUAD3 PROT CLR	PS1 QUAD2 PROT CLR	PS1 QUAD1 PROT CLR	PS1 QUAD0 PROT CLR	PS0 QUAD3 PROT CLR	PS0 QUAD2 PROT CLR	PS0 QUAD1 PROT CLR	PS0 QUAD0 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-57. Peripheral Protection Clear Register 0 (PPROTCLR0) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[7–0]QUAD[3–0] PROTCLR		Peripheral select quadrant protection clear,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PPROTSET0 and PPROTCLR0 registers is cleared to 0.

5.2.10 Peripheral Protection Clear Register 1 (PPROTCLR1)

There is one bit for each quadrant for PS8 to PS15 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-57 and described in Table 5-58.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-57. Peripheral Protection Clear Register 1 (PPROTCLR1) [offset = 44h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS15 QUAD3 PROT CLR	PS15 QUAD2 PROT CLR	PS15 QUAD1 PROT CLR	PS15 QUAD0 PROT CLR	PS14 QUAD3 PROT CLR	PS14 QUAD2 PROT CLR	PS14 QUAD1 PROT CLR	PS14 QUAD0 PROT CLR	PS13 QUAD3 PROT CLR	PS13 QUAD2 PROT CLR	PS13 QUAD1 PROT CLR	PS13 QUAD0 PROT CLR	PS12 QUAD3 PROT CLR	PS12 QUAD2 PROT CLR	PS12 QUAD1 PROT CLR	PS12 QUAD0 PROT CLR
R/WP-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS11 QUAD3 PROT CLR	PS11 QUAD2 PROT CLR	PS11 QUAD1 PROT CLR	PS11 QUAD0 PROT CLR	PS10 QUAD3 PROT CLR	PS10 QUAD2 PROT CLR	PS10 QUAD1 PROT CLR	PS10 QUAD0 PROT CLR	PS9 QUAD3 PROT CLR	PS9 QUAD2 PROT CLR	PS9 QUAD1 PROT CLR	PS9 QUAD0 PROT CLR	PS8 QUAD3 PROT CLR	PS8 QUAD2 PROT CLR	PS8 QUAD1 PROT CLR	PS8 QUAD0 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-58. Peripheral Protection Clear Register 1 (PPROTCLR1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[15–8]QUAD[3–0] PROTCLR		Peripheral select quadrant protection clear,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PPROTSET1 and PPROTCLR1 registers is cleared to 0.

5.2.11 Peripheral Protection Clear Register 2 (PPROTCLR2)

There is one bit for each quadrant for PS16 to PS23 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-58 and described in Table 5-59.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-58. Peripheral Protection Clear Register 2 (PPROTCLR2) [offset = 48h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS23 QUAD3 PROT CLR	PS23 QUAD2 PROT CLR	PS23 QUAD1 PROT CLR	PS23 QUAD0 PROT CLR	PS22 QUAD3 PROT CLR	PS22 QUAD2 PROT CLR	PS22 QUAD1 PROT CLR	PS22 QUAD0 PROT CLR	PS21 QUAD3 PROT CLR	PS21 QUAD2 PROT CLR	PS21 QUAD1 PROT CLR	PS21 QUAD0 PROT CLR	PS20 QUAD3 PROT CLR	PS20 QUAD2 PROT CLR	PS20 QUAD1 PROT CLR	PS20 QUAD0 PROT CLR
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS19 QUAD3 PROT CLR	PS19 QUAD2 PROT CLR	PS19 QUAD1 PROT CLR	PS19 QUAD0 PROT CLR	PS18 QUAD3 PROT CLR	PS18 QUAD2 PROT CLR	PS18 QUAD1 PROT CLR	PS18 QUAD0 PROT CLR	PS17 QUAD3 PROT CLR	PS17 QUAD2 PROT CLR	PS17 QUAD1 PROT CLR	PS17 QUAD0 PROT CLR	PS16 QUAD3 PROT CLR	PS16 QUAD2 PROT CLR	PS16 QUAD1 PROT CLR	PS16 QUAD0 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-59. Peripheral Protection Clear Register 2 (PPROTCLR2) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[23–16]QUAD[3–0] PROTCLR		Peripheral select quadrant protection clear,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PPROTSET2 and PPROTCLR2 registers is cleared to 0.

5.2.12 Peripheral Protection Clear Register 3 (PPROTCLR3)

There is one bit for each quadrant for PS24 to PS31 (See section 1.3.1 for quadrant definition). The protection scheme is described in PPROTSET0, in section 5.2.5. This register is shown in Figure 5-59 and described in Table 5-60.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-59. Peripheral Protection Clear Register 3 (PPROTCLR3) [offset = 4Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS31 QUAD3 PROT CLR	PS31 QUAD2 PROT CLR	PS31 QUAD1 PROT CLR	PS31 QUAD0 PROT CLR	PS30 QUAD3 PROT CLR	PS30 QUAD2 PROT CLR	PS30 QUAD1 PROT CLR	PS30 QUAD0 PROT CLR	PS29 QUAD3 PROT CLR	PS29 QUAD2 PROT CLR	PS29 QUAD1 PROT CLR	PS29 QUAD0 PROT CLR	PS28 QUAD3 PROT CLR	PS28 QUAD2 PROT CLR	PS28 QUAD1 PROT CLR	PS28 QUAD0 PROT CLR
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS27 QUAD3 PROT CLR	PS27 QUAD2 PROT CLR	PS27 QUAD1 PROT CLR	PS27 QUAD0 PROT CLR	PS26 QUAD3 PROT CLR	PS26 QUAD2 PROT CLR	PS26 QUAD1 PROT CLR	PS26 QUAD0 PROT CLR	PS25 QUAD3 PROT CLR	PS25 QUAD2 PROT CLR	PS25 QUAD1 PROT CLR	PS25 QUAD0 PROT CLR	PS24 QUAD3 PROT CLR	PS24 QUAD2 PROT CLR	PS24 QUAD1 PROT CLR	PS24 QUAD0 PROT CLR
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-60. Peripheral Protection Clear Register 3 (PPROTCLR3) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0]QUAD[3–0] PROTCLR		Peripheral select quadrant protection clear,
		0	*Read*– The peripheral select quadrant can be written to and read from in both user and privileged modes. *Write*– The bit is unchanged.
		1	*Read*– The peripheral select quadrant can be written to only in privileged mode but can be read in both user and privileged modes. *Write*– The bit in PPROTSET3 and PPROTCLR3 registers is cleared to 0.

5.2.13 Peripheral Memory Powerdown Set Register 0 (PCSPWRDWNSET0)

Each bit corresponds to a bit at the same index in the PMPROT register in that they both relate to the same peripheral.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

This register is shown in Figure 5-60 and described in Table 5-61.

Figure 5-60. Peripheral Memory Power-Down Set Register 0 (PCSPWRDWNSET0) [offset = 60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS31 PWR DWN SET	PCS30 PWR DWN SET	PCS29 PWR DWN SET	PCS28 PWR DWN SET	PCS27 PWR DWN SET	PCS26 PWR DWN SET	PCS25 PWR DWN SET	PCS24 PWR DWN SET	PCS23 PWR DWN SET	PCS22 PWR DWN SET	PCS21 PWR DWN SET	PCS20 PWR DWN SET	PCS19 PWR DWN SET	PCS18 PWR DWN SET	PCS17 PWR DWN SET	PCS16 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS15 PWR DWN SET	PCS14 PWR DWN SET	PCS13 PWR DWN SET	PCS12 PWR DWN SET	PCS11 PWR DWN SET	PCS10 PWR DWN SET	PCS9 PWR DWN SET	PCS8 PWR DWN SET	PCS7 PWR DWN SET	PCS6 PWR DWN SET	PCS5 PWR DWN SET	PCS4 PWR DWN SET	PCS3 PWR DWN SET	PCS2 PWR DWN SET	PCS1 PWR DWN SET	PCS0 PWR DWN SET
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-61. Peripheral Memory Power-Down Set Register 0 (PCSPWRDWNSET0) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0] PWRDNSET		Peripheral memory clock power-down set.
		0	*Read*– The peripheral memory clock[31–0] is active. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory clock[31–0] is inactive. *Write*– The corresponding bit in the PCSPWRDWNSET0 and PCSPWRDWNCLR0 registers is set to 1.

5.2.14 Peripheral Memory Powerdown Set Register 1 (PCSPWRDWNSET1)

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

This register is shown in Figure 5-61 and described in Table 5-64.

Figure 5-61. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNSET1) [offset = 64h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS63 PWR DWN SET	PCS62 PWR DWN SET	PCS61 PWR DWN SET	PCS60 PWR DWN SET	PCS59 PWR DWN SET	PCS58 PWR DWN SET	PCS57 PWR DWN SET	PCS56 PWR DWN SET	PCS55 PWR DWN SET	PCS54 PWR DWN SET	PCS53 PWR DWN SET	PCS52 PWR DWN SET	PCS51 PWR DWN SET	PCS50 PWR DWN SET	PCS49 PWR DWN SET	PCS48 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS47 PWR DWN SET	PCS46 PWR DWN SET	PCS45 PWR DWN SET	PCS44 PWR DWN SET	PCS43 PWR DWN SET	PCS42 PWR DWN SET	PCS41 PWR DWN SET	PCS40 PWR DWN SET	PCS39 PWR DWN SET	PCS38 PWR DWN SET	PCS37 PWR DWN SET	PCS36 PWR DWN SET	PCS35 PWR DWN SET	PCS34 PWR DWN SET	PCS33 PWR DWN SET	PCS32 PWR DWN SET
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-62. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNSET1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[63–32] PWRDNSET		Peripheral memory clock power-down set.
		0	*Read*– The peripheral memory clock[63–32] is active. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory clock[63–32] is inactive. *Write*– The corresponding bit in the PCSPWRDWNSET1 and PCSPWRDWNCLR1 registers is set to 1.

5.2.15 Peripheral Memory Powerdown Clear Register 0 (PCSPWRDWNCLR0)

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

This register is shown in Figure 5-62 and described in Table 5-63.

Figure 5-62. Peripheral Memory Power-Down Clear Register 0 (PCSPWRDWNCLR0) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS31 PWR DWN CLR	PCS30 PWR DWN CLR	PCS29 PWR DWN CLR	PCS28 PWR DWN CLR	PCS27 PWR DWN CLR	PCS26 PWR DWN CLR	PCS25 PWR DWN CLR	PCS24 PWR DWN CLR	PCS23 PWR DWN CLR	PCS22 PWR DWN CLR	PCS21 PWR DWN CLR	PCS20 PWR DWN CLR	PCS19 PWR DWN CLR	PCS18 PWR DWN CLR	PCS17 PWR DWN CLR	PCS16 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS15 PWR DWN CLR	PCS14 PWR DWN CLR	PCS13 PWR DWN CLR	PCS12 PWR DWN CLR	PCS11 PWR DWN CLR	PCS10 PWR DWN CLR	PCS9 PWR DWN CLR	PCS8 PWR DWN CLR	PCS7 PWR DWN CLR	PCS6 PWR DWN CLR	PCS5 PWR DWN CLR	PCS4 PWR DWN CLR	PCS3 PWR DWN CLR	PCS2 PWR DWN CLR	PCS1 PWR DWN CLR	PCS0 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 5-63. Peripheral Memory Power-Down Clear Register 0 (PCSPWRDWNCLR0) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[31–0] PWRDNCLR0		Peripheral memory clock power-down clear.
		0	*Read*– The peripheral memory clock[31–0] is active. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory clock[31–0] is inactive. *Write*– The corresponding bit in the PCSPWRDWNSET0 and PCSPWRDWNCLR0 registers is cleared to 0.

5.2.16 Peripheral Memory Powerdown Clear Register 1 (PCSPWRDWNCLR1)

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Therefore, the size of this register is device-dependent. Writes to nonimplemented bits have no effect and reads are 0.

This register is shown in Figure 5-63 and described in Table 5-64.

Figure 5-63. Peripheral Memory Power-Down Clear Register 1 (PCSPWRDWNCLR1) [offset = 74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCS63 PWR DWN CLR	PCS62 PWR DWN CLR	PCS61 PWR DWN CLR	PCS60 PWR DWN CLR	PCS59 PWR DWN CLR	PCS58 PWR DWN CLR	PCS57 PWR DWN CLR	PCS56 PWR DWN CLR	PCS55 PWR DWN CLR	PCS54 PWR DWN CLR	PCS53 PWR DWN CLR	PCS52 PWR DWN CLR	PCS51 PWR DWN CLR	PCS50 PWR DWN CLR	PCS49 PWR DWN CLR	PCS48 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCS47 PWR DWN CLR	PCS46 PWR DWN CLR	PCS45 PWR DWN CLR	PCS44 PWR DWN CLR	PCS43 PWR DWN CLR	PCS42 PWR DWN CLR	PCS41 PWR DWN CLR	PCS40 PWR DWN CLR	PCS39 PWR DWN CLR	PCS38 PWR DWN CLR	PCS37 PWR DWN CLR	PCS36 PWR DWN CLR	PCS35 PWR DWN CLR	PCS34 PWR DWN CLR	PCS33 PWR DWN CLR	PCS32 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-64. Peripheral Memory Power-Down Set Register 1 (PCSPWRDWNCLR1) Field Descriptions

Bit	Name	Value	Description
31–0	PCS[63–32] PWRDNSET		Peripheral memory clock power-down clear.
		0	*Read*– The peripheral memory clock[63–32] is active. *Write*– The bit is unchanged.
		1	*Read*– The peripheral memory clock[63–32] is inactive. *Write*– The corresponding bit in the PCSPWRDWNSET1 and PCSPWRDWNCLR1 registers is cleared to 0.

5.2.17 Peripheral Powerdown Set Register 0 (PSPWRDWNSET0)

There is one bit for each quadrant for PS0 to PS7 (See section 1.3.1 for quadrant definition). Each bit of this register corresponds to the bit at the same index in the corresponding PPROT register in that they relate to the same peripheral.These bits are used to power down/power up the clock to the corresponding peripheral.

For every bit implemented in the PPROT register, there is one bit in the PSnPWRDWN register, except when two peripherals (both in PS area) share buses. In that case, only one Power-Down bit is implemented, at the position corresponding to that peripheral whose quadrant comes first (the lower numbered).

The ways in which quadrants can be used within a frame are identical to what is described under PPROTSET0, section 5.2.5.

This arrangement is the same for bits of PS8 to PS31, presented in section 5.2.18—section 5.2.24. This register holds bits for PS0 to PS7. This register is shown in Figure 5-64 and described in Table 5-65.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-64. Peripheral Power-Down Set Register 0 (PSPWRDWNSET0) [offset = 80h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS7 QUAD3 PWR DWN SET	PS7 QUAD2 PWR DWN SET	PS7 QUAD1 PWR DWN SET	PS7 QUAD0 PWR DWN SET	PS6 QUAD3 PWR DWN SET	PS6 QUAD2 PWR DWN SET	PS6 QUAD1 PWR DWN SET	PS6 QUAD0 PWR DWN SET	PS5 QUAD3 PWR DWN SET	PS5 QUAD2 PWR DWN SET	PS5 QUAD1 PWR DWN SET	PS5 QUAD0 PWR DWN SET	PS4 QUAD3 PWR DWN SET	PS4 QUAD2 PWR DWN SET	PS4 QUAD1 PWR DWN SET	PS4 QUAD0 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS3 QUAD3 PWR DWN SET	PS3 QUAD2 PWR DWN SET	PS3 QUAD1 PWR DWN SET	PS3 QUAD0 PWR DWN SET	PS2 QUAD3 PWR DWN SET	PS2 QUAD2 PWR DWN SET	PS2 QUAD1 PWR DWN SET	PS2 QUAD0 PWR DWN SET	PS1 QUAD3 PWR DWN SET	PS1 QUAD2 PWR DWN SET	PS1 QUAD1 PWR DWN SET	PS1 QUAD0 PWR DWN SET	PS0 QUAD3 PWR DWN SET	PS0 QUAD2 PWR DWN SET	PS0 QUAD1 PWR DWN SET	PS0 QUAD0 PWR DWN SET
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-65. Peripheral Power-Down Set Register 0 (PSPWRDWNSET0) Field Descriptions

Bit	Name	Value	Description
31–0	PS[7–0]QUAD[3–0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET0 and PSPWRDWNCLR0 registers is set to 1

5.2.18 Peripheral Powerdown Set Register 1 (PSPWRDWNSET1)

There is one bit for each quadrant for PS8 to PS15 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-65 and described in Table 5-66.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-65. Peripheral Power-Down Set Register 1 (PSPWRDWNSET1) [offset = 84h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS15 QUAD3 PWR DWN SET	PS15 QUAD2 PWR DWN SET	PS15 QUAD1 PWR DWN SET	PS15 QUAD0 PWR DWN SET	PS14 QUAD3 PWR DWN SET	PS14 QUAD2 PWR DWN SET	PS14 QUAD1 PWR DWN SET	PS14 QUAD0 PWR DWN SET	PS13 QUAD3 PWR DWN SET	PS13 QUAD2 PWR DWN SET	PS13 QUAD1 PWR DWN SET	PS13 QUAD0 PWR DWN SET	PS12 QUAD3 PWR DWN SET	PS12 QUAD2 PWR DWN SET	PS12 QUAD1 PWR DWN SET	PS12 QUAD0 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS11 QUAD3 PWR DWN SET	PS11 QUAD2 PWR DWN SET	PS11 QUAD1 PWR DWN SET	PS11 QUAD0 PWR DWN SET	PS10 QUAD3 PWR DWN SET	PS10 QUAD2 PWR DWN SET	PS10 QUAD1 PWR DWN SET	PS10 QUAD0 PWR DWN SET	PS9 QUAD3 PWR DWN SET	PS9 QUAD2 PWR DWN SET	PS9 QUAD1 PWR DWN SET	PS9 QUAD0 PWR DWN SET	PS8 QUAD3 PWR DWN SET	PS8 QUAD2 PWR DWN SET	PS8 QUAD1 PWR DWN SET	PS8 QUAD0 PWR DWN SET
R/WP-1															

R/WP = Read in all modes, any time, write in privileged mode only, -*n* = Value after reset

.

Table 5-66. Peripheral Power-Down Set Register 1 (PSPWRDWNSET1) Field Descriptions

Bit	Name	Value	Description
31–0	PS[15–8]QUAD[3–0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET1 and PSPWRDWNCLR1 registers is set to 1

5.2.19 Peripheral Powerdown Set Register 2 (PSPWRDWNSET2)

There is one bit for each quadrant for PS16 to PS23 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-59 and described in Table 5-60.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-66. Peripheral Power-Down Set Register 2 (PSPWRDWNSET2) [offset = 88h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS23 QUAD3 PWR DWN SET	PS23 QUAD2 PWR DWN SET	PS23 QUAD1 PWR DWN SET	PS23 QUAD0 PWR DWN SET	PS22 QUAD3 PWR DWN SET	PS22 QUAD2 PWR DWN SET	PS22 QUAD1 PWR DWN SET	PS22 QUAD0 PWR DWN SET	PS21 QUAD3 PWR DWN SET	PS21 QUAD2 PWR DWN SET	PS21 QUAD1 PWR DWN SET	PS21 QUAD0 PWR DWN SET	PS20 QUAD3 PWR DWN SET	PS20 QUAD2 PWR DWN SET	PS20 QUAD1 PWR DWN SET	PS20 QUAD0 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS19 QUAD3 PWR DWN SET	PS19 QUAD2 PWR DWN SET	PS19 QUAD1 PWR DWN SET	PS19 QUAD0 PWR DWN SET	PS18 QUAD3 PWR DWN SET	PS18 QUAD2 PWR DWN SET	PS18 QUAD1 PWR DWN SET	PS18 QUAD0 PWR DWN SET	PS17 QUAD3 PWR DWN SET	PS17 QUAD2 PWR DWN SET	PS17 QUAD1 PWR DWN SET	PS17 QUAD0 PWR DWN SET	PS16 QUAD3 PWR DWN SET	PS16 QUAD2 PWR DWN SET	PS16 QUAD1 PWR DWN SET	PS16 QUAD0 PWR DWN SET
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-67. Peripheral Power-Down Set Register 2 (PSPWRDWNSET2) Field Descriptions

Bit	Name	Value	Description
31–0	PS[23–16]QUAD[3–0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET2 and PSPWRDWNCLR2 registers is set to 1

5.2.20 Peripheral Powerdown Set Register 3 (PSPWRDWNSET3)

There is one bit for each quadrant for PS24 to PS31 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-67 and described in Table 5-68.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-67. Peripheral Power-Down Set Register 3 (PSPWRDWNSET3) [offset = 8Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS31 QUAD3 PWR DWN SET	PS31 QUAD2 PWR DWN SET	PS31 QUAD1 PWR DWN SET	PS31 QUAD0 PWR DWN SET	PS30 QUAD3 PWR DWN SET	PS30 QUAD2 PWR DWN SET	PS30 QUAD1 PWR DWN SET	PS30 QUAD0 PWR DWN SET	PS29 QUAD3 PWR DWN SET	PS29 QUAD2 PWR DWN SET	PS29 QUAD1 PWR DWN SET	PS29 QUAD0 PWR DWN SET	PS28 QUAD3 PWR DWN SET	PS28 QUAD2 PWR DWN SET	PS28 QUAD1 PWR DWN SET	PS28 QUAD0 PWR DWN SET
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS27 QUAD3 PWR DWN SET	PS27 QUAD2 PWR DWN SET	PS27 QUAD1 PWR DWN SET	PS27 QUAD0 PWR DWN SET	PS26 QUAD3 PWR DWN SET	PS26 QUAD2 PWR DWN SET	PS26 QUAD1 PWR DWN SET	PS26 QUAD0 PWR DWN SET	PS25 QUAD3 PWR DWN SET	PS25 QUAD2 PWR DWN SET	PS25 QUAD1 PWR DWN SET	PS25 QUAD0 PWR DWN SET	PS24 QUAD3 PWR DWN SET	PS24 QUAD2 PWR DWN SET	PS24 QUAD1 PWR DWN SET	PS24 QUAD0 PWR DWN SET
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-68. Peripheral Power-Down Set Register 3 (PSPWRDWNSET3) Field Descriptions

Bit	Name	Value	Description
31–0	PS[31–24]QUAD[3–0] PWRDWNSET		Peripheral select quadrant clock power-down set.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET3 and PSPWRDWNCLR3 registers is set to 1

5.2.21 Peripheral Powerdown Clear Register 0 (PSPWRDWNCLR0)

There is one bit for each quadrant for PS0 to PS7 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-68 and described in Table 5-69.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-68. Peripheral Power-Down Clear Register 0 (PSPWRDWNCLR0) [offset = A0h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS7 QUAD3 PWR DWN CLR	PS7 QUAD2 PWR DWN CLR	PS7 QUAD1 PWR DWN CLR	PS7 QUAD0 PWR DWN CLR	PS6 QUAD3 PWR DWN CLR	PS6 QUAD2 PWR DWN CLR	PS6 QUAD1 PWR DWN CLR	PS6 QUAD0 PWR DWN CLR	PS5 QUAD3 PWR DWN CLR	PS5 QUAD2 PWR DWN CLR	PS5 QUAD1 PWR DWN CLR	PS5 QUAD0 PWR DWN CLR	PS4 QUAD3 PWR DWN CLR	PS4 QUAD2 PWR DWN CLR	PS4 QUAD1 PWR DWN CLR	PS4 QUAD0 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS3 QUAD3 PWR DWN CLR	PS3 QUAD2 PWR DWN CLR	PS3 QUAD1 PWR DWN CLR	PS3 QUAD0 PWR DWN CLR	PS2 QUAD3 PWR DWN CLR	PS2 QUAD2 PWR DWN CLR	PS2 QUAD1 PWR DWN CLR	PS2 QUAD0 PWR DWN CLR	PS1 QUAD3 PWR DWN CLR	PS1 QUAD2 PWR DWN CLR	PS1 QUAD1 PWR DWN CLR	PS1 QUAD0 PWR DWN CLR	PS0 QUAD3 PWR DWN CLR	PS0 QUAD2 PWR DWN CLR	PS0 QUAD1 PWR DWN CLR	PS0 QUAD0 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 5-69. Peripheral Power-Down Clear Register 0 (PSPWRDWNCLR0) Field Descriptions

Bit	Name	Value	Description
31–0	PS[7–0]QUAD[3–0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET0 and PSPWRDWNCLR0 registers is cleared to 0.

5.2.22 Peripheral Powerdown Clear Register 1 (PSPWRDWNCLR1)

There is one bit for each quadrant for PS8 to PS15 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-69 and described in Table 5-70.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-69. Peripheral Power-Down Clear Register 1 (PSPWRDWNCLR1) [offset = A4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS15 QUAD3 PWR DWN CLR	PS15 QUAD2 PWR DWN CLR	PS15 QUAD1 PWR DWN CLR	PS15 QUAD0 PWR DWN CLR	PS14 QUAD3 PWR DWN CLR	PS14 QUAD2 PWR DWN CLR	PS14 QUAD1 PWR DWN CLR	PS14 QUAD0 PWR DWN CLR	PS13 QUAD3 PWR DWN CLR	PS13 QUAD2 PWR DWN CLR	PS13 QUAD1 PWR DWN CLR	PS13 QUAD0 PWR DWN CLR	PS12 QUAD3 PWR DWN CLR	PS12 QUAD2 PWR DWN CLR	PS12 QUAD1 PWR DWN CLR	PS12 QUAD0 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS11 QUAD3 PWR DWN CLR	PS11 QUAD2 PWR DWN CLR	PS11 QUAD1 PWR DWN CLR	PS11 QUAD0 PWR DWN CLR	PS10 QUAD3 PWR DWN CLR	PS10 QUAD2 PWR DWN CLR	PS10 QUAD1 PWR DWN CLR	PS10 QUAD0 PWR DWN CLR	PS9 QUAD3 PWR DWN CLR	PS9 QUAD2 PWR DWN CLR	PS9 QUAD1 PWR DWN CLR	PS9 QUAD0 PWR DWN CLR	PS8 QUAD3 PWR DWN CLR	PS8 QUAD2 PWR DWN CLR	PS8 QUAD1 PWR DWN CLR	PS8 QUAD0 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 5-70. Peripheral Power-Down Clear Register 1 (PSPWRDWNCLR1) Field Descriptions

Bit	Name	Value	Description
31–0	PS[15–8]QUAD[3–0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET1 and PSPWRDWNCLR1 registers is cleared to 0.

5.2.23 Peripheral Powerdown Clear Register 2 (PSPWRDWNCLR2)

There is one bit for each quadrant for PS16 to PS23 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-70 and described in Table 5-71.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-70. Peripheral Power-Down Clear Register 2 (PSPWRDWNCLR2) [offset = A8h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS23 QUAD3 PWR DWN CLR	PS23 QUAD2 PWR DWN CLR	PS23 QUAD1 PWR DWN CLR	PS23 QUAD0 PWR DWN CLR	PS22 QUAD3 PWR DWN CLR	PS22 QUAD2 PWR DWN CLR	PS22 QUAD1 PWR DWN CLR	PS22 QUAD0 PWR DWN CLR	PS21 QUAD3 PWR DWN CLR	PS21 QUAD2 PWR DWN CLR	PS21 QUAD1 PWR DWN CLR	PS21 QUAD0 PWR DWN CLR	PS20 QUAD3 PWR DWN CLR	PS20 QUAD2 PWR DWN CLR	PS20 QUAD1 PWR DWN CLR	PS20 QUAD0 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS19 QUAD3 PWR DWN CLR	PS19 QUAD2 PWR DWN CLR	PS19 QUAD1 PWR DWN CLR	PS19 QUAD0 PWR DWN CLR	PS18 QUAD3 PWR DWN CLR	PS18 QUAD2 PWR DWN CLR	PS18 QUAD1 PWR DWN CLR	PS18 QUAD0 PWR DWN CLR	PS17 QUAD3 PWR DWN CLR	PS17 QUAD2 PWR DWN CLR	PS17 QUAD1 PWR DWN CLR	PS17 QUAD0 PWR DWN CLR	PS16 QUAD3 PWR DWN CLR	PS16 QUAD2 PWR DWN CLR	PS16 QUAD1 PWR DWN CLR	PS16 QUAD0 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 5-71. Peripheral Power-Down Clear Register 2 (PSPWRDWNCLR2) Field Descriptions

Bit	Name	Value	Description
31–0	PS[23–16]QUAD[3–0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET2 and PSPWRDWNCLR2 registers is cleared to 0.

5.2.24 Peripheral Powerdown Clear Register 3 (PSPWRDWNCLR3)

There is one bit for each quadrant for PS24 to PS31 (See section 1.3.1 for quadrant definition). The protection scheme is described in section 5.2.17. This register is shown in Figure 5-71 and described in Table 5-72.

Note:
Only those bits that have a slave at the corresponding bit position are implemented. Writes to nonimplemented bits have no effect and reads are 0.

Figure 5-71. Peripheral Power-Down Clear Register 3 (PSPWRDWNCLR) [offset = ACh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PS31 QUAD3 PWR DWN CLR	PS31 QUAD2 PWR DWN CLR	PS31 QUAD1 PWR DWN CLR	PS31 QUAD0 PWR DWN CLR	PS30 QUAD3 PWR DWN CLR	PS30 QUAD2 PWR DWN CLR	PS30 QUAD1 PWR DWN CLR	PS30 QUAD0 PWR DWN CLR	PS29 QUAD3 PWR DWN CLR	PS29 QUAD2 PWR DWN CLR	PS29 QUAD1 PWR DWN CLR	PS29 QUAD0 PWR DWN CLR	PS28 QUAD3 PWR DWN CLR	PS28 QUAD2 PWR DWN CLR	PS28 QUAD1 PWR DWN CLR	PS28 QUAD0 PWR DWN CLR
R/WP-1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS27 QUAD3 PWR DWN CLR	PS27 QUAD2 PWR DWN CLR	PS27 QUAD1 PWR DWN CLR	PS27 QUAD0 PWR DWN CLR	PS26 QUAD3 PWR DWN CLR	PS26 QUAD2 PWR DWN CLR	PS26 QUAD1 PWR DWN CLR	PS26 QUAD0 PWR DWN CLR	PS25 QUAD3 PWR DWN CLR	PS25 QUAD2 PWR DWN CLR	PS25 QUAD1 PWR DWN CLR	PS25 QUAD0 PWR DWN CLR	PS24 QUAD3 PWR DWN CLR	PS24 QUAD2 PWR DWN CLR	PS24 QUAD1 PWR DWN CLR	PS24 QUAD0 PWR DWN CLR
R/WP-1															

R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 5-72. Peripheral Power-Down Clear Register 3 (PSPWRDWNCLR3) Field Descriptions

Bit	Name	Value	Description
31–0	PS[31–24]QUAD[3–0] PWRDWNCLR		Peripheral select quadrant clock power-down clear.
		0	*Read*– The clock to the peripheral select quadrant is active. *Write*– The bit is unchanged.
		1	*Read*– The clock to the peripheral select quadrant is inactive. *Write*– The corresponding bit in PSPWRDWNSET3 and PSPWRDWNCLR3 registers is cleared to 0.

5.3 eSRAM Control Register

This section describes the RAM ECC control registers. The start address of the RAM ECC register frame is 0xFFFF F9FF. Read and write access in 8,16 and 32 bits are supported.

Write to the RAM ECC registers in non-privileged mode is blocked except in suspend mode.

Note:
Due to the platform architecture, there is a 2 VCLK cycle latency for all writes to the SRAM control registers. When a control register is updated, accesses to RAM should be delayed until the write has been completed in order to ensure validity of the RAM data that is written and/or read. For example, if the VCLK to HCLK ratio is VCLK=HCLK/2 there should be a minimum of 4 HCLK cycles delay between the register write and the first RAM access.

Table 5-73. eSRAM Control register map

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 RAMCTRL page 195	Reserved															
	Reserved				RMWCBYP			ECC WRT ENA	Reserved				ECC_ENABLE			
0x04 RAMTHRESH-OLD page 197	Reserved															
	SEC_THRESHOLD															
0x08 RAMOCCUR page 198	Reserved															
	SEC_OCCUR															
0x0C RAMINTCTRL page 199	Reserved															
	Reserved															SEC INT EN
0x10 RAMERR STATUS page 200	Reserved															
																SEC INT FLAG
0x14 RAMSERR ADDR page 201	Reserved															
	Reserved	SERRADDR[14:0]														

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x18 RAMSERR POSITION page 202	Reserved															
	Reserved								ERR TYPE	SBZ	SBZ	SERRPOSITION[5:0]				
0x1C RAMDERR ADDR page 203	Reserved															
	Reserved	DERRADDR[14:0]														

5.3.1 Ram Control Register (RAMCTRL)

This register is shown in Figure 5-72 and described in Table 5-74.

Figure 5-72. RAM Control Register (RAMCTRL) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			RMW CBYP				ECC WRT ENA	Reserved				ECC_ENABLE[3:0]			
R-0			RWP-1010				RWP-0	R-0				RWP-0101			

U = Undefined; R = Read, WP = Write in privileged mode,-*n* = Value after reset

.

Table 5-74. RAM Control Register (RAMCTRL) Field Descriptions

Bit	Name	Value	Description
31–13	Reserved		Reads return zeros and writes have no effect.
12-9	RMWCBYP		Read-modify-write correction bypass. When RMWCBYP is disabled (any value other than 0101), during a read-modify-write operation forced by a less-than-64-bits write access, the read operation goes through error detection and correction as if a normal read had occurred. When RMWCBYP is enabled (0101), the read operation does not go through error correction and detection. When the device comes out of a power-up reset, the data and ECC memory are in an unknown state. A less-than-64-bits write access to the memory will force a read-modify-write operation and will result in false double error if ECC_ENABLE is enabled (1010). To avoid a false double error, you can first enable the RMWCBYP bit. User data can be loaded to the memory with ECC calculated by SECDED logic. RMWCBYP is then disabled upon completion of the user data transfer. After reset, RMWCBYP is disabled (set to 0b1010).
		0101	ECC correction bypass is enabled.
		All other values	ECC correction bypass is disabled.
8	ECC_WRT_ENA		ECC write enable. To avoid accidental write to ECC memory bits the ECC_WRT_ENA must first be enabled before the ECC memory bits can be written.
		0	Write to ECC memory bits is disabled.
		1	Write to ECC memory bits is enabled.
7–4	Reserved		Reads return zeros and writes have no effect.

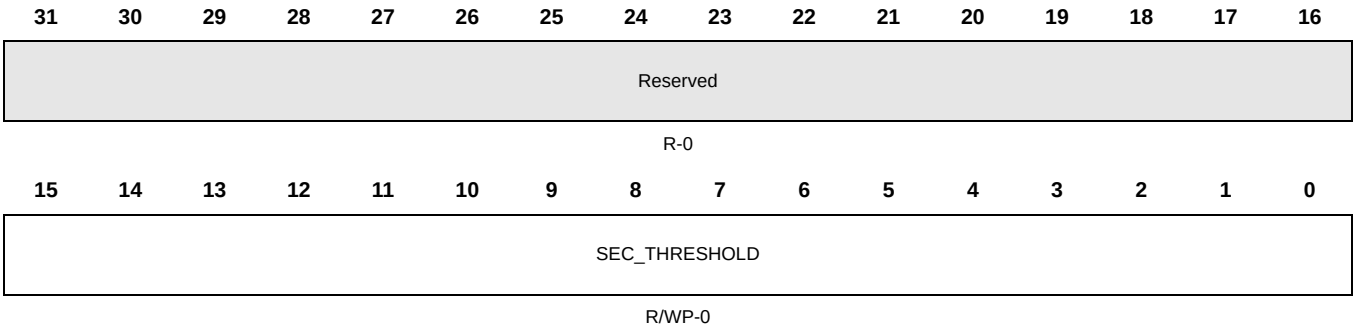
Table 5-74. RAM Control Register (RAMCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
3–0	ECC_ENABLE[3–0]		ECC enable. When enabled, ECC is automatically generated by SECDED for both read and write access. During write access, ECC is generated and written to the memory at the same time as the data is generated and written. During read access, ECC is generated based on the read data from memory, and compares it with the known good ECC value stored in the memory. A 1-bit error is corrected before the data is returned to the host. A 2-bit error is signaled to the system module via interrupt.
		0101	The ECC is disabled.
		all other values	The ECC is enabled.

5.3.2 Threshold Register (RAMTHRESHOLD)

This register is shown in Figure 5-71 and described in Table 5-72.

Figure 5-73. Threshold Register (RAMTHRESHOLD) [offset = 08h]



R = Read in all modes; WP = Write in privileged mode; U = Undefined; -*n* = Value after reset

.

Table 5-75. Threshold Register (RAMTHRESHOLD) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zeros and writes have no effect.
15–0	SEC_THRESHOLD	0–FFFFh	Single error correction threshold. This register contains the threshold value for the SEC (single error correction) occurrences before SERR_INT interrupt is generated.

5.3.3 Occurrence Register (RAMOCCUR)

This register is shown in Figure 5-74 and described in Table 5-76.

Figure 5-74. Occurrence Register (RAMOCCUR) [offset = 08h]

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

R = Read in all modes; WP = Write in privileged mode only; U = Undefined; -*n* = Value after reset

.

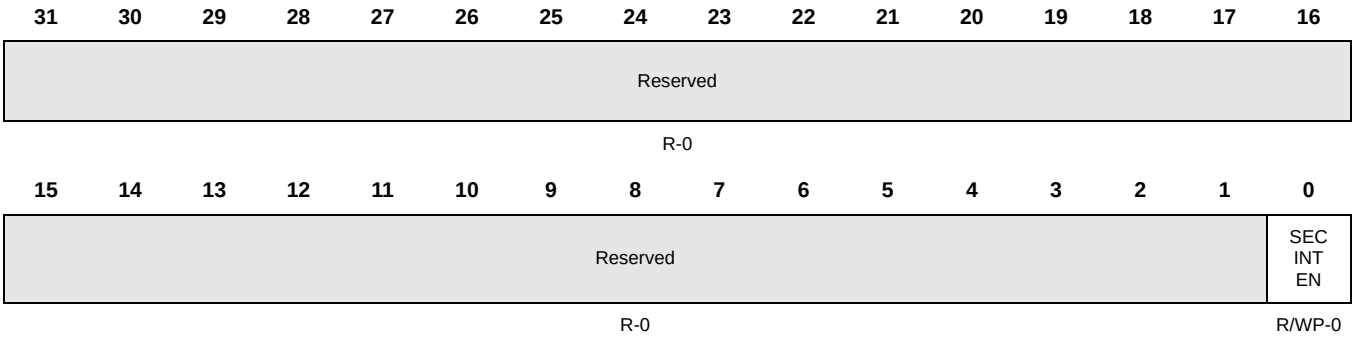
Table 5-76. Occurrence Register (RAMOCCUR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zeros and writes have no effect.
15–0	SEC_OCCUR	0–FFFFh	Single error occurrence counter. This 16-bit counter contains the number of single bit error occurrences. SEC_OCCUR is reset to 0 when it is equal to the SEC_THRESHOLD value.

5.3.4 Interrupt Control Register (RAMINTCTRL)

This register is shown in Figure 5-75 and described in Table 5-77.

Figure 5-75. Interrupt Control Register (RAMINTCTRL) [offset = 0Ch]



R = Read in all modes; WP = Write in privileged mode only; U = Undefined; -*n* = Value after reset

.

Table 5-77. Occurrence Register (RAMOCCUR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zeros and writes have no effect.
0	SECINTEN		Single error correct interrupt enable.
		0	Single error interrupt generation is disabled.
		1	Single error interrupt generation is enabled.

5.3.5 Memory Fault Detect Status Register (RAMERRSTATUS)

This register is shown in Figure 5-76 and described in Table 5-78.

Figure 5-76. Memory Fault Detect Status Register (RAMERRSTATUS) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SEC INT FLAG
R-0															R/WP-0

R = Read in all modes; WP = Write in privileged mode only; U = Undefined; -*n* = Value after reset

.

Table 5-78. Memory Fault Detect Status Register (RAMERRSTATUS) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zeros and writes have no effect.
0	SECINTFLAG		Single error correct interrupt flag. This flag is set when the number of correctable single error reaches the threshold value, even if SECINTEN is disabled. The host clears the flag by writing a 1 to it. If the CPU has not cleared the status bit and another correctable error is detected, then the RAMOCCUR counter will still be incremented.
		0	The number of correctable single errors has not reached the threshold value.
		1	The number of correctable single errors has reached the threshold value.

5.3.6 Single Error Address Register (RAMSERRADDR)

This register is shown in Figure 5-77 and described in Table 5-79.

Figure 5-77. Single Error Address Register (RAMSERRADD) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SERRADDR[15–0]														
R-0	R-0														

R = Read in all modes; WP = Write in privileged mode only; U = Undefined; -*n* = Value after power-on reset

Table 5-79. Single Error Address Register (RAMSERRADD) Field Descriptions

Bit	Name	Value	Description
31–15	Reserved		Reads return zeros and writes have no effect.
14–0	SERRADDR[14–0]	0–7FFFh	Single error address. SERRADDR records the double word address at which an memory fault error is detected. SERRADDR captures the error address when detecting a single bit error if the error threshold value stored in SEC_TRESHOLD register is equal to 1. SERRADDR is frozen from being updated until SECINTFLAG is cleared by the CPU. SERRADDR contains the double word address, which means only HADDR[17–3] are saved. To convert to the byte address, simply left-shift SERRADDR by 3 bits. **Note: This register is only reset during power-up reset.**

5.3.7 RAM Error Position Register (RAMERRPOSITION)

This register is shown in Figure 5-78 and described in Table 5-80.

Figure 5-78. RAM Error Position Register (RAMERRPOSITION) [offset = 18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ERR TYPE	SBZ	SBZ	SERRPOSITION[5–0]					
R-0							R-0	R-0	R-0	R-0					

R = Read in all modes; WP = Write in privileged mode only; U = Undefined; -*n* = Value after reset

Table 5-80. RAM Error Position Register (RAMERRPOSITION) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved		Reads return zeros and writes have no effect.
8	ERRTYPE		Error type. This bit indicates whether the single error detected is a data bit error or check bit error (ECC bit).
		0	The error is a data bit error.
		1	The error is a check bit error.
7–6	SBZ	0	These bits always read 0.
5–0	SERRPOSITION[5–0]	0–3Fh	Single error position. SERRPOSITION records the binary encoded error position at which a single error is detected. The error position is captured into SERRPOSITION when a single bit error is detected and the error threshold value stored in SEC_TRESHOLD register is equal to 1.

The following examples illustrate the error position being captured in RAMERRPOSITION.

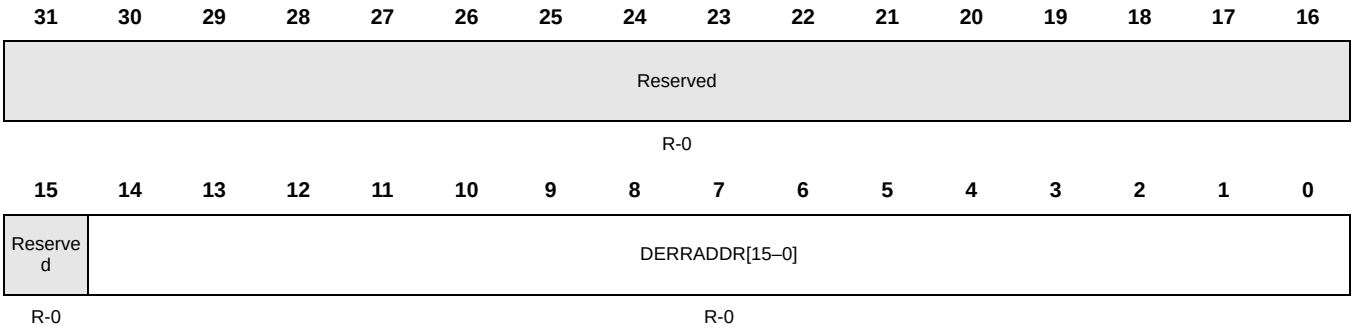
0_0000_0000	Data bit[0] is in error.
0_0001_1111	Data bit[31] is in error.
0_0011_1111	Data bit[63] is in error.
1_0000_0000	ECC bit[0] is in error.
1_0000_0111	ECC bit[7] is in error.

Note:
This register is only reset during power-up reset.

5.3.8 Double Error Address Register (RAMDERRADDR)

This register is shown in Figure 5-79 and described in Table 5-81.

Figure 5-79. Double Error Address Register (RAMDERRADDR) [offset = 1Ch]



R = Read in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 5-81. Double Error Address Register (RAMDERRADD) Field Descriptions

Bit	Name	Value	Description
31–15	Reserved		Reads return zeros and writes have no effect.
14–0	DERRADDR[14–0]	0–7FFFh	Double error address. DERRADDR records the double word address at which a memory fault error is detected. DERRADDR captures the error address when detecting a double bit error. DERRADDR is frozen from being updated until DERRADDR is read by the CPU. DERRADDR contains the double word address, which means only HADDR[17–3] are saved. To convert to the byte address, simply left shift DERRADDR by 3 bits.

Note:
This register is only reset during power-up reset.

F05 Flash Module

The flash electrically erasable programmable read-only memory (Flash EEPROM, or Flash) module is a type of nonvolatile memory which has fast read access times but slower write and erase times. Flash EEPROM performs erasing by sector, rather than by word unlike regular EEPROM.

Topic **Page**

6.1 Overview

The TMS470Px F05 Flash is generally used to provide permanent program/data storage or factory calibration data; and can be programmed and electrically erased many times to allow for faster code development.

Flash EEPROM differs from standard EEPROM in that all bits in a flash sector are erased in bulk, whereas standard EEPROM is erased a word at a time.

6.1.1 Features

- Supports multiple flash banks for program and data storage.
- Supports up to 8 banks of up to 16 Megabits each
- Supports automating flash erase and programming through an integrated state machine
- Supports simultaneous read access on a bank while performing a write or erase operation on any one of the remaining banks
- Supports erase and program suspend, which allows software to fetch data from the flash bank currently being erased or programmed.
- Supports up to 32 sectors per flash bank
- Provides four 32-bit write protection keys.
- Provides built-in power mode control logic.

6.1.2 Definition of Terms

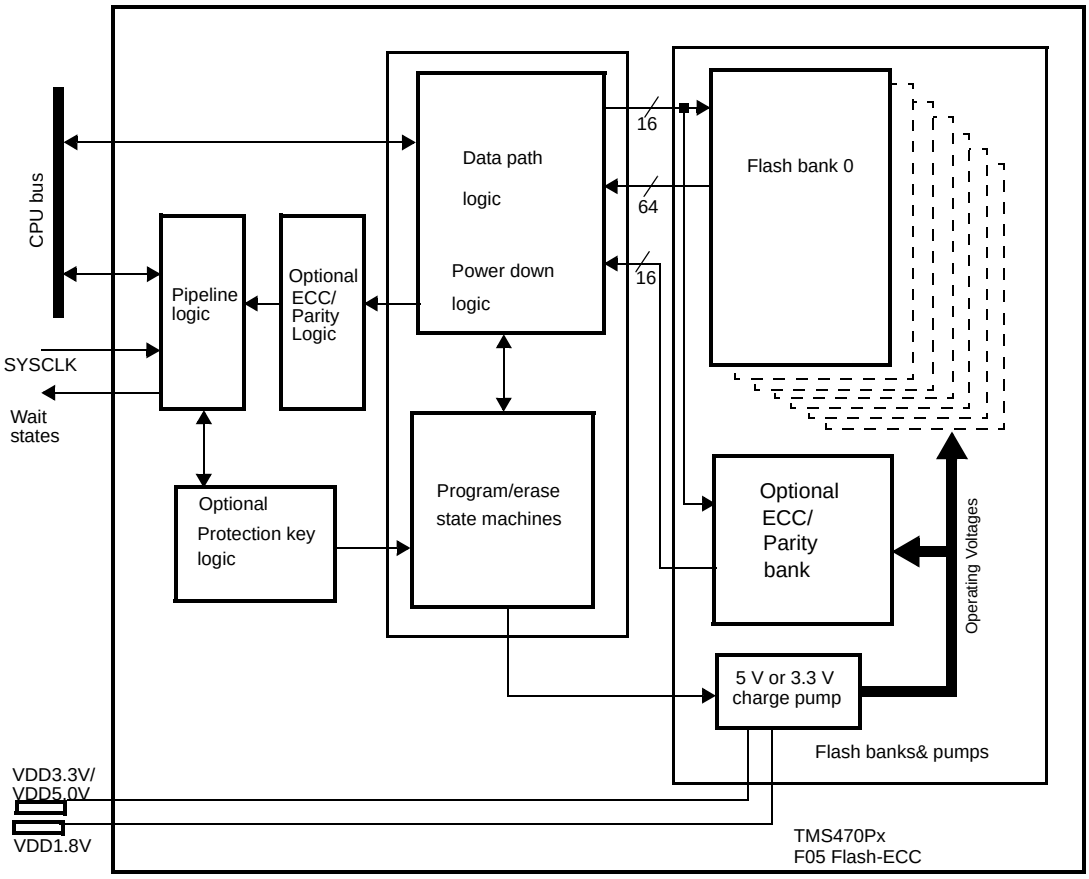
Terms used in this document have the following meanings:

- BAGP (bank active grace period): Time (in SYSCLK cycles) from the most recent flash access of a particular bank until that bank enters fallback power mode. This reduces power consumption by the flash. However, it can also increase access time.
- Charge pump: Voltage generators and associated control (logic, oscillator, and bandgap, for example).
- CSM (command state machine): One of the three state machines present in the flash module.
- Fallback power mode: The power mode (active, standby or sleep, depending on which mode is selected) into which a bank or the charge pump falls back each time the active grace period expires.
- Flash bank: A group of flash sectors which share input/output buffers, data paths, sense amplifiers, and control logic. Flash bank can store both program instructions and data.
- Flash module: Flash banks, charge pump, power and mode control logic, data path, wait logic, and write/erase state machines.
- TMS470Px F05 Flash: Flash module and CPU interface which includes pipeline logic and protection logic.
- OTP (one-time programmable): A program-only-once flash sector (cannot be erased)
- PAGP (pump active grace period): Time (in SYSCLK cycles) from when the last of the banks have entered fallback power mode until the pump enters a fallback power mode. This can reduce power consumption by the flash; however, it can also increase access time.
- Pipeline mode: The mode in which flash is read 64 bits at a time, giving faster apparent access times.
- Sector: A contiguous region of flash memory which must be erased simultaneously due to physical construction constraints.
- Standard read mode: The mode assumed when the pipeline mode is not enabled and the flash is read no more than 32 bits at a time.
- Read Margin 1 mode: More stringent read mode designed for early detection of marginally erased bits.
- Read Margin 0 mode: More stringent read mode designed for early detection of marginally programmed bits.
- Read Vt mode: Special read mode (for debug/validation purposes only) in which the read level for Flash bits is provided through an external device pin.

6.2 Functional Block Diagram

The TMS470Px F05 Flash consists of several major functional elements as shown in Figure 6-1.

Figure 6-1. TMS470Px F05 Flash Functional Block Diagram



This document describes an implementation:

- With a 3.3V charge pump
- With Protection key logic

All other optional features are not described in this document

Up to eight banks of flash can be present in a single flash module. The current design supports up to 32 sectors per bank.

State machines in the flash module perform program, erase, and verify operations. The state machines permit the use of simpler program/erase/verify software algorithms.

A single charge pump can supply all of the required voltages for up to eight flash banks. The charge pump requires 1.8V and 3.3V power supplies for normal operation. Program and erase operations do not require any additional power supply.

The pipeline logic speeds up the apparent read access time of the flash module.

Programming the power-down logic correctly can produce significant power savings. The power-down registers control the power mode for each bank individually as well as for the charge pump.

The entire module is protected from unauthorized erasures or writes when the optional four 32-bit protection key words are used.

6.3 Operation

The following sections discuss various issues related to reset, reading, erasing, programming, power mode, protection, and wait state generation.

6.3.1 Reset State

The reset state exhibits the following properties:

1. Wait states are set to 1
2. Pipeline mode default is device specific
3. The OTP sector is turned off
4. All levels of protection are enabled
5. Power modes are set to *Active* (no power savings)

The boot code must initialize the wait states and the desired pipeline mode to achieve the optimum system performance.

6.3.2 Flash Read Modes

In addition to *standard read mode*, the flash module also has *pipeline mode*, which affects the technique used to fetch the next memory word. Using this mode correctly increases clock speeds and CPU throughput.

6.3.2.1 Standard Read Mode

Standard read mode is defined as the mode in effect when pipeline mode is inactive. The number of wait states used in standard read mode is user programmable within the flash control register FMBAC2.7:0.

6.3.2.2 Pipeline Mode

In *pipeline mode*, two words are read in parallel from the flash core. Storing these two words in pipeline data buffers increases the bandwidth for the data coming out of the flash core, which provides effectively zero wait states on as many accesses as possible.

In pipeline mode, the flash data is always latched into the pipeline buffer first, then read from the pipeline buffer to the CPU. Pipeline mode removes the flash memory access time from the critical timing path, which allows the clock frequency to be higher.

Pipeline mode is enabled by setting the ENPIPE bit in the flash control register FMREGOPT.0. The pipeline mode default is device specific.

6.3.3 Flash Commands

The TMS470Px F05 Flash-ECC should be programmed, erased and verified only by using the TMS470Px F05 Flash API Library functions. These functions are written, compiled and validated by Texas Instruments. The information provided in this document is intended to help explain the requirements for functionality not covered by the Flash API. Refer to the TMS470Px F05 Flash API documentation (pf05a_api.pdf) for more information.

The flash module contains a Command State Machine (CSM) to perform program, erase, and validate operations. The CSM supports the commands shown in Table 6-1 At the end of a system reset, the CSM is in Idle mode. The completion of any command also returns the CSM to Idle mode.

Table 6-1. Flash Command Summary

Command Operand	Description	Write Cycles Required
0x0010	Program	2
0x0020	Erase Sector	2

Command Operand	Description	Write Cycles Required
0x0040	Clear Status Register	1
0x0080	Suspend (program or erase)	1
0x0200	Resume Programming	1
0x0400	Resume Erasing	1
0x0800	Program OTP	2
0x1000	Validate Sector	2

Command writes (as well as OTP reads) can be performed only when pipeline mode is off. This is necessary to prevent data stored in the pipeline buffers from becoming inconsistent with the data stored in memory.

The program, erase, and validate sector commands require two write cycles, whereas the remaining commands require only one write cycle.

After a program, erase, or validate sector command has been written to the flash module, the module waits indefinitely for the second write cycle to latch the address and/or data. As a result, the valid data write cycle must have valid data, whether it immediately follows the command or not.

The control port can be used to read or write registers, or bank reads can be performed between the command and the data for the program or erase command.

The BUSY status bit becomes active upon receipt of any command except for the clear status command.

After an erase or program command has been issued, the operation cannot be aborted except by resetting the flash module. The operation can, however, be allowed to complete or be suspended. A suspended operation, likewise, can only be aborted by resetting the flash module.

6.3.3.1 CPU Operations Required for Executing Commands

Before the TMS470Px F05 Flash-ECC can execute a command, the host CPU must do the following:

1. Select the desired bank to be programmed by writing BANK[2:0] of FMMAC2
2. Select one or more bits in FMBSEA or FMBSEB to disable Level 1 protection for the particular sector to be erased/written
3. Disable Level 2 sector protection by matching the correct four 32-bit FMPKEY words.
4. Clear the READOTP bit in the FMREGOPT register
5. Write an operand, as shown in Table 6-1, to any location in flash memory which initiates the operation

For operations that apply to specific address locations (program, erase, and validate) the host CPU must also:

6. Write the desired data half-word to the appropriate address in the selected flash bank

After these operations have been completed in sequence, the command is performed. These flash commands are described in more detail in the following sections.

6.3.3.2 Program Sector Command

If not already set up, the host CPU must perform the procedures outlined in Section 6.3.3.1. Writing the program operand, as shown in Table 6-1, to any location in flash memory initiates the program operation. This must be followed by writing the desired data half-word to the appropriate address in the selected flash bank.

Note: Program Data to the Flash Module in Half-Words

All data programmed into the TMS470Px F05 Flash must be done one half-word (16 bits) at a time.

The flash module checks the sector protection status and, if allowed, proceeds to program and verify the data. Once the command is received, the busy flag is held active.

While the busy flag is active, the flash module responds only to the suspend command. If an attempt is made to read a bank being programmed, invalid data is returned. Reads may be performed correctly on other banks. If the program operation is suspended and the suspended bank is read, then only data other than the address that was being programmed should be assumed valid.

If the program operation is suspended, the busy flag becomes inactive within 2.0ms of receiving the suspend command.

During the program operation, 3.3/5.0 V V_{DD} must remain within the appropriate range. Otherwise, the error flag 3_5VSTAT is set in the status register FMMSTAT.3.

When the program command is complete, the user should parse the Status register to verify that the command was successful. Upon completion, the flash module resets the CSM state to Idle mode. The program command must be reissued before another half-word is written to flash memory.

6.3.3.3 Erase Sector Command

Note: Erase Sector is implemented by the following Flash API functions:

- Flash_Erase_B
- Flash_Erase_Sector_B
- Flash_Start_Erase_B

If it is not already set up, the host CPU must perform the procedures outlined in Section 6.3.3.1. Writing the erase operand, as shown in Table 6-1, to any location in flash memory, initiates the erase operation. This is followed by writing any data to the selected sector address (an address anywhere within the sector to be erased) in the flash memory.

The flash module checks the sector enable status and, if allowed, proceeds to erase and verify the data. Once the command is received, the busy flag is held active.

While the busy flag is active, the flash module responds only to the suspend command. If an attempt is made to read a bank being erased, invalid data is returned. Reads can be performed on other banks. If the erase operation is suspended and the suspended bank is read, then the entire sector's data is assumed invalid. Read data from other sectors and banks is invalid due to the partial erasure of this sector.

If the erase operation is suspended, the busy flag becomes inactive within 2 ms of receiving the suspend command.

Upon completion of the erase command, the user should parse the Status register to verify that the command was successful.

During the erase operation, 3.3/5 V V_{DD} must remain within the appropriate range. Otherwise, the error flag 3_5VSTAT is set in the status register FMMSTAT.3.

When the erase operation is complete, the flash module resets the CSM state to Idle mode. The erase command must be reissued before another sector can be erased.

6.3.3.4 Suspend/Resume Commands

The suspend command allows erase or program operations, whichever is active, to be suspended so other operations can be performed on the affected bank. It is also possible to suspend a program operation so that a sector in a different bank can be erased, or to suspend an erase operation so that information may be programmed in a different bank. The suspend command is acted upon only if the BUSY bit of FMMSTAT is high and the flash module is performing an erase or program operation. Issuing a suspend command when BUSY is low has no effect on the flash module.

Note: Flash Module Ignores All But Suspend Command When BUSY Bit is High
If a command other than suspend is issued while the BUSY bit is high, it is ignored.

It is possible to suspend an operation at certain points in the operation of the state machine so that the operation actually finishes and the suspend flags are never set. Issuing a resume command when suspend is not active has no effect on the flash module.

Only one operation can be active at a time. Also, a program operation cannot be initiated while a previous program operation is suspended, and an erase operation cannot be initiated while a previous erase operation is suspended. The flash module ignores any command that violates these conditions.

6.3.3.5 Clear Status Command

The Status register allows the user to determine whether an erase/program operation was successfully completed, in progress, suspended, or failed.

If not already set up, the host must perform the procedures outlined in Section 6.3.3.1. Writing the clear status operand, as shown in Table 6-1, to any location in flash memory, executes the clear status operation.

If the BUSY bit is low, this operation clears the following status bits:

- SLOCK (sector locked)
- CSTAT (command status)
- VSTAT (3.3V V_{DD} status)
- INVDAT (invalid program data) in the FMMSTAT register

6.3.3.6 Program OTP Command

The OTP Program command is implemented by the following Flash API function:

- OTP_Prog_B

The flash module may have one or more OTP sectors, up to one OTP sector per flash bank. A minimum of one is required for production and test purposes. Each OTP sector contains 2K bytes. The bits can be programmed using the program OTP command. Once the command is issued, the bank identifier, address within the OTP sector, and the data must be specified.

The program OTP flow is the same as the program half-word flow. It can be suspended and resumed.

6.3.3.7 Validate Sector Command

The Validate Sector command is implemented by the following Flash API function:

- Flash_Compact_B

A validate sector command (which performs compaction on the target sector as needed) has been provided to enable the host CPU to determine if a sector contains depleted bits and to correct the problem. A depleted bit may occur only in the unlikely event that an erase operation has been started, but not completed. The following conditions could leave depleted bits in a sector:

- An erase is in progress when power is removed
- An erase suspend is active when power is removed
- A flash module reset interrupts an erase operation
- A flash module reset occurs while an erase suspend is active

After a flash module reset, including power up, the CPU should perform a validation check on sectors which may have been incompletely erased to correct any depleted bits. It is not required to check sectors which are not programmed in the field, as they cannot have depleted bits. It is, however, possible for depleted bits in one sector to cause bits in another sector to read incorrectly; therefore, it is vital that depleted bits be corrected.

Note: There Is No Suspend for the Validate Sector Command
The *validate sector* command cannot be suspended.

6.3.4 Data Security

Data security against either accidental or deliberate access by unauthorized agents is built into the flash module with two levels of data security. Level 1 security allows each sector to be individually protected from any access other than read. Level 2 security protects the entire module from non-read access using four 32-bit protection keys.

6.3.4.1 Sector Enable Command

The Sector Enable command is implemented by the following Flash API function:

- Flash_Sector_Enable_B

A sector enable command (registers BSEA and BSEB) is a means of preventing data from being modified within a sector. Because the flash module may store permanent and/or semi-permanent program code and/or data, this capability is provided. A sector is protected if data within that sector is prevented from being modified (the sector-enable bit for that sector is cleared). Currently the TMS470Px F05 Flash-ECC supports a maximum of 32 sectors per bank.

Flash memory can be programmed or erased only if the specified sector is enabled. If the sector is protected, then the state machine of the flash module halts and sets the status bit SLOCK in FMMSTAT.0.

At power up, all of the sector-enable bits are initialized so that the flash memory location cannot be modified. The sector enable command is a feature used only by the flash module when erasing or programming flash memory.

When an erase or program operation resumes from a suspended state, the sector-enable bit is checked again as though it were a new erase or program command.

6.3.4.2 Four-Word Protection Keys

Unlocking of protection keys is implemented by the following Flash API function:

- Flash_Match_Key_B

If this option is present, the CPU reads the four stored protection keys out of the flash bank one at a time and into a register in the flash module. After the CPU loads each key from the bank to the control logic, the CPU must load an identical user key into the FMPKEY control register. The CPU must load and match all four keys before any program or erase command can be sent to the flash module.

To enable the module for programming, the CPU must load each stored key value from the bank to the control logic by performing a normal read access to one of the four protection key addresses in the flash module. The CPU must then load the matching user key value into the FMPKEY register. This process is repeated until all four keys are loaded and matched. The control logic monitors which keys have been matched, so the CPU cannot gain write access until it loads and matches all four keys at least once without any intervening mismatch.

If the CPU writes a mismatching key at any time (that is, if the user key does not match the key that was most recently loaded from the bank to the control logic), then all key match states are cleared and the CPU must *reload and rematch all four keys again* to gain write access to the module. This feature can be used to disable write access after programming is completed.

After the CPU has successfully loaded and matched all four keys, flash write access is enabled and the PROTL2DIS flag (FMBBUSY.15) is set until either a device reset occurs or until the CPU writes a mismatching key to the FMPKEY register.

To store the key values, the CPU programs the key data into the bank by performing normal program and erase operations on these four protection key addresses. The key values are stored in the bank as ordinary data, so the CPU must provide the correct keys before it can perform any program or erasure of the key values.

When a new device is delivered to the customer, the keys will be all ones, so keys of all ones should be used to enable flash writes for the first time. Once the keys are changed in the Flash bank, the CPU must deliberately write a mismatching key value to FMPKEY in order to disable further programming until the new key values have been loaded and matched. In other words, the flash module remains enabled for the remainder of this programming session even though the keys have been modified in flash.

The difference between flash protection key read accesses and other reads is that the key data does not propagate to the CPU data bus *until the correct keys are written* and the user has taken all required steps to gain programming access. This is intended to prevent unauthorized discovery of the stored keys by reading them out through the CPU. Only a user who already knows the keys can gain access to them.

The location of the four protection keys depend on the specific device being used (as specified in the specific device data sheet).

6.3.5 Automatic Power-down of Flash Banks

The flash module provides a mechanism to automatically power down flash banks after they have not been accessed for some user programmable time. Special timers automatically sequence the power up and power down of each bank independently of each other. The charge pump module has its own independent power up/down timers.

During global low power mode when SYS_LPM=1, the flash module which includes the wrapper logic, banks, and pump module are put into sleep mode. If there is an operation such as a write then the flash module waits until the operation to finish before it enters sleep mode. The flash wrapper uses a nLP_RDY handshake signal to indicate it is ready.

When the SYS_LPM signal goes low again, the banks and pump module will go through the normal power up sequence from sleep to active power state. CPU access to the flash module during the power up sequence will be held in wait state.

6.3.5.1 Active Grace Period

Active grace period (AGP) is a feature which the host can use to optimize the flash module power consumption/access time trade-off. Faster access times are associated with higher power modes of operation. At one extreme, the power control logic could attempt to reduce power consumption by putting the banks and charge pump into a low-power mode immediately at the end of every flash access. However, if accesses are a few cycles apart, this can actually increase power consumption over leaving the flash powered, because the banks and charge pump consume more power during flash startup and access.

Active grace periods (supported for each bank independently in addition to the charge pump module) allow the banks and/or charge pump to be maintained in active mode for a specified period following an access. This is done in anticipation of another read within the AGP time, to allow the subsequent read to have a faster access and spend less time dissipating power than if the bank went into one of the low power modes immediately. On the other hand, if the next access does not occur within the AGP time, the power control logic can automatically put the bank and/or charge pump into a low-power mode to reduce power consumption during long periods of inactivity.

AGP is realized by a set of host-programmable counters which keep the flash bank or charge pump in active mode until the counter expires, at which time the bank or charge pump reverts to its fallback power mode. See register descriptions for FMBAC1, FMBAC2, FMMAC1, FMMAC2, and FMPAGP for details.

The bank and/or charge pump are kept active in anticipation of another read. Assuming AGP is being utilized, the bank AGP counter is set, keeping the bank active while an access is in progress. The counter begins counting when no more accesses are in progress. If the AGP timers have not timed out and another access occurs then there is a substantial performance gain compared to the access when the bank is not active (either the bank is in standby or Sleep mode). If the AGP counter times out, the bank or charge pump is put into a host programmable fallback power mode. The host CPU can program the fallback power mode to be standby or sleep mode to reduce power consumption, or program it to be active mode to keep the bank active regardless of counter settings (default).

The charge pump AGP counter remains in its initialized state when any one of the banks is active, including the AGP counter of the bank. The charge pump AGP counter begins counting when all banks have become inactive.

The bank and charge pump AGP counters count at SYSCLK frequency.

6.3.5.2 Power Mode Control

The primary contributors to flash module power consumption are the flash banks, and the charge pump. This section details how the flash wrapper controls the different power modes of the flash banks and charge pump.

Several components of the module power reduction have been discussed. These are the bank fallback power bits in the FMBAC1 register, the charge pump fallback power bits in the FMMAC2 register, and the BAGP and

PAGP operation. The fallback power control bits contain the bank and charge pump modes, which become active upon time-out of the AGP counters described in Section 6.3.5.2. Any access to a flash bank causes the bank and charge pump to go into active mode, regardless of their current state. Also, any erase, program, or validate sector command causes the charge pump to become active.

If the charge pump is in sleep mode when the flash access begins, the power mode control logic automatically sequences the charge pump to standby mode, then to active mode. Also, if any bank is active or in standby mode, the charge pump is active, independent of the charge pump fallback power mode.

The host can override the power control functions of the flash module by setting all of the AGP counters to zero. In this case, the power mode control logic still sequences the pump through standby mode automatically if needed, and it activates the pump automatically if any bank is put into any power mode other than sleep mode.

6.3.6 Wait States

The number of wait states can vary depending on the type of read access performed. The different types of read accesses are defined below:

- *Initial Access* - First address read after initialization of the flash. Typically after reset.
- *Sequential Access* - Instruction accesses the very next flash address from the previous instruction's flash address.
- *Random Access* - Instruction accesses a non-sequential flash address from the previous instruction's address.

Wait states are added to a read access when either the flash bank or charge pump are not active, or when the flash bank cannot have data valid at the frequency demanded by the host CPU. The flash module generates an internal ready signal for each bank based on the values in the wait state registers which depend on the status of the bank, the charge pump, and the data rate.

The bank has three wait state generators and the pump has two wait state counters. Both the bank and the charge pump have standby and sleep counters which are initialized when a transition is made from non-active mode to active mode. The bank also has a wait state counter to allow for a faster clock than the flash data rate. All wait state generators are clocked by the flash system clock.

The outputs of the charge pump counters and bank counters are combined. The resulting signal generates the wait states and ready signal of each bank when an access is in progress.

6.3.7 VDD Out of Range Check

The VDD out of range status bit informs the host if there was a low supply during an erase or program operation. A comparator in the charge pump module sets the VSTAT bit if the 3.3V supply is less than 2.4V for more than three SYSCLK cycles; otherwise it is cleared.

6.4 Control Registers

This section covers the TMS470Px F05 Flash control register memory map and a basic description of each register and its bits.

6.4.1 Memory Map

The TMS470Px F05 Flash uses several registers to control the various modes of operation, numbers of wait states, and bank selection. All user flash registers are shown in Table 6-2 and Table 6-3. The addresses shown are relative offsets from the base address, which depends upon the device being used. See the specific TMS470Px device data sheet for the register base address. The start address of the Flash module is 0xFFF8 7000.

Table 6-2. 32-bit Flash Memory Registers

Offset Address†	Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x0100	FMRE-GOPT	Reserved															
		Reserved											SPOTP	RD MRGN1	RD MRGN0	READ OTP	ENPIPE
0x0108	FMBBUSY	PROTL 2DIS	Reserved							BUSY[7:0]							
0x010C	FMPKEY	PKEY[31:16]															
		PKEY[15:0]															
0x0128	FMUER-RADDR	Reserved								UERRADDR[23:16]							
		UERRADDR[15:0]															
0x0138	FMSEC-DIS	bankID1_inverse[2:0]			SectorID1_inverse[4:0]					bankID1[2:0]			SectorID1[4:0]				
		bankID0_inverse[2:0]			SectorID0_inverse[4:0]					bankID0[2:0]			SectorID0[4:0]				

Table 6-3. 16-bit Flash Memory Registers

Offset Address	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0000	FMBAC1	BAGP								BSTDBY						BNKPWR	
0x0004	FMBAC2	OTP PROT DIS	BSLEEP							WAIT				WAIT			
0x0008	FMBSEA	BSEA															
0x000C	FMBSEB	BSEB															
0x0010	FMBRDY	Reserved										BANK RDY	Reserved				
0x0014	FMPRDY	Reserved						PUMP RDY	Reserved								
0x0018	FMMAC1	PROTL 1 DIS	PSLEEP														
0x001C	FMMAC2	PSTDBY											PMPPWR	BANK			
0x0020	FMPAGP	PAGP															
0x0024	FMMSTAT	Reserved								BUSY	ERS	PGM	INV DAT	CSTAT	3V STAT	ESUSP	PSUSP	SLOCK

6.4.2 Register Access

The flash module control registers can only be read and/or written by the CPU while in privileged mode.

Note: All reserved fields will read as zeros. Byte, half-word and word writes are supported to all registers.

6.4.2.1 Option Control Register (FMREGOPT)

FMREGOPT is a word-access only register. It supports OTP sector access, margin testing, and pipeline mode. There is only one FMREGOPT register for the entire TMS470Px F05 Flash.

Figure 6-2. Option Control Register (FMREGOPT) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Reserved	RD MRGN1	RD MRGN0	READ OTP	EN PIPE
R-0											R-0	RWP-0	RWP-0	RWP-0	RWP-d

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

.

Table 6-4. Option Control Register (FMREGOPT) Field Descriptions

Bit	Name	Value	Description
31–5	Reserved		Reads return zeros and writes have no effect.
4	Reserved		This bit is reserved for TI's use. **Do not set this bit** as it will cause the user OTP sector to be disabled even when otherwise enabled by READOTP.
3	RDMRGN1		Read Margin 1. When set, enables Read Margin 1 mode (RDMRGN1 is overridden by RDMRGN0).
2	RDMRGN0		Read Margin 0. When set, enables Read Margin 0 mode (RDMRGN0 mode overrides RDMRGN1 mode).
1	READOTP		Read OTP Sector. When set, this bit enables reading from the OTP sector. The starting address of the OTP sector is located at the relative flash module address 0x0000 in the first bank of the flash module. Clear this bit after the OTP sector is read to ensure that flash programming functions normally.

Table 6-4. Option Control Register (FMREGOPT) Field Descriptions (Continued)

Bit	Name	Value	Description
0	ENPIPE		Enable Pipe Mode. Pipeline mode is active when ENPIPE is set and the TMS470Px F05 Flash-ECC is not in halt mode. Pipeline mode is overridden in halt mode. The default value of ENPIPE is device specific. See the device-specific datasheet for the reset state of ENPIPE.

Note: Command Writes and OTP Reads Performed Only in Pipeline Mode
Command writes and OTP reads can be performed only if pipeline mode is off. This is necessary to prevent data stored in the pipeline buffers from becoming inconsistent with the data stored in memory.

6.4.2.2 Bank Busy Register (FMBBUSY)

FMBBUSY is a word-access read-only register. It supports checking the busy status of all banks in parallel. The TMS470Px F05 Flash-ECC has only one FMBBUSY register.

Figure 6-3. Bank Busy Register (FMBBUSY) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROTL2DISReserved	Reserved							BUSY[7:0]							
R-0	R-0							R-0							

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

Table 6-5. Bank Busy Register (FMBBUSY) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zeros and writes have no effect.
15	PROTL2DISReserved		Protection Key Level 2 Disabled Flag - This bit is read-only and when set, it indicates that all four protection keys have been written correctly. If any of the 128 protection key bits are incorrect then PROTL2DIS is zero. Writes to this bit have no effect.Reads return zeros and writes have no effect.
14-8	Reserved		Reads return zeros and writes have no effect.
7-0	BUSY[7:0]		Bank Busy - This read-only location allows the CPU to determine if any flash bank is busy performing an OTP read, a command operation, or in the process of being reset. It displays the states of the BUSY signals from each bank simultaneously. Each bit corresponds to one flash bank. These bits are read-only. Writes have no effect.

6.4.2.3 Protection Key Register (FMPKEY)

FMPKEY is a word-access only register. It controls access protection for the entire TMS470Px F05 Flash-ECC. The TMS470Px F05 Flash-ECC has only one FMPKEY register.

Figure 6-4. Protection Key Register (FMPKEY) [offset = 0Ch]

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

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

Table 6-6. Protection Key Register (FMPKEY) Field Descriptions

Bit	Name	Value	Description
31–0	PKEY[31:0]		Protection Key - These bits receive and store the four 32-bit protection keys from the CPU for the four 32-bit protection key feature. See Section 6.3.4.2, *Four-Word Protection Keys*.

.

6.4.2.4 Bank Access Control Register 1 (FMBAC1)

FMBAC1 is a half-word register. It controls bank standby mode wait state generation, bank fallback power mode, and bank active grace period (AGP) delay. Each bank in the flash module has one FMBAC1 register. The bank is selected via BANK[2:0] of the FMMAC2 register. As only one bank at a time can be selected by FMMAC2, only the register of the selected bank appears at this address.

Figure 6-5. Bank Access Control Register 1 (FMBAC1) [offset = 00h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAGP[7:0]								BSTDBY[5:0]						BNKPWR[1:0]	
RWP-00000000								RWP-111111						RWP-11	

R = Read, W = Write, S = Set, U = Undefined, -*n* = Value after reset

.

Table 6-8. Bank Access Control Register 1 (FMBAC1) Field Descriptions

Bit	Name	Value	Description
15–8	BAGP[7:0]		Bank Active Grace Period - These bits contain the starting count value for the BAGP down counter. Any access to a given bank causes its BAGP counter to reload the BAGP value for that bank. After the last access to this flash bank, the down counter delays from 0 to 255 SYSCLK cycles before putting the bank into one of the fallback power modes as determined by BNKPWR[1:0] in this register. See Section 6.3.5.2, *Power Mode Control* on page 213 for bank activation logic.
7-2	BSTBY[5:0]		Bank Standby - These bits contain the starting count value for the bank standby down counter. While the bank is in standby mode, the power mode management logic holds the bank standby counter at this value. When the bank exits standby power mode, the down counter delays (counts down to zero) from 0 to 63 SYSCLK cycles before putting the bank into bank active mode.

Table 6-8. Bank Access Control Register 1 (FMBAC1) Field Descriptions (Continued)

Bit	Name	Value	Description
1-0	BNKPWR[1:0]		Bank Power Mode - These bits describe the fall back power mode which the flash bank enters after the bank active grace period counter has timed out. The default (11 binary) is to remain active.
		00	Sleep (sense amplifiers and sense reference disabled)
		01	Standby (sense amplifiers disabled, but sense reference enabled)
		10	Reserved
		11	Activ e (both sense amplifiers and sense reference enabled)

6.4.2.5 Bank Access Control Register 2 (FMBAC2)

FMBAC2 is a half-word register. It controls wait state generation, bank sleep delay, and OTP sector protection. There is one FMBAC2 register for each bank in the flash module. The bank is selected via BANK[2:0] of the FMMAC2 register. As only one bank at a time can be selected by FMMAC2, only the register of the selected bank appears at this address.

Figure 6-6. Bank Access Control Register 2 (FMBAC2) [offset = 04h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OTP PROT DIS	BSLEEP[6:0]							WAIT3:0]				WAIT[3:0]			
RWP-0	RWP-1111111							RWP-0001				RWP-0001			

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

Table 6-9. Bank Access Control Register 2 (FMBAC2) Field Descriptions

Bit	Name	Value	Description
15	OTPPROTDIS		OTP Sector Protection Disable - When this bit is set, it enables programming of the OTP sector. This bit can be set only when PROTL1DIS = 1.
14-8	BSLEEP[6:0]		Bank Sleep - These bits contain the starting count value for the bank sleep down counter. While the bank is in sleep mode, the power mode management logic holds the bank sleep counter at this value. When the bank exits sleep power mode, the down counter delays from 0 to 127 SYSCLK cycles before putting the bank into active mode.
7-4	WAIT[7:4]		Wait State Counter - These bits contain the starting count value for the wait state down counter. The down counter delays from 0 to 15 SYSCLK cycles before indicating that data is available. For normal operation, these bits are set to 000 for single cycle standard read mode, or 001 for pipeline mode. Wait bits 7:4 must match wait bits 3:0.
3-0	WAIT[3:0]		Wait State Counter - For normal operation, these bits are set to 000 for single cycle standard read mode, or to 001 for pipeline mode. Wait bits 3:0 must match wait bits 7:4.

6.4.2.6 Bank Sector Enable Registers (FMBSEA and FMBSEB)

FMBSEA and FMBSEB are half-word registers. Together they provide one enable bit per sector for up to 32 sectors per bank. Each bank in the flash module has one FMBSEA and one FMBSEB register. The bank is selected via the BANK[2:0] bits of the FMMAC2 register. As only one bank at a time can be selected by FMMAC2, only the register for the bank selected appears at this address.

Figure 6-7. Bank Sector Enable Registers (FMBSEA and FMBSEB) [offset = 08h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BSE[15:0]															
RWP-0x0000															

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

Table 6-10. Bank Sector Enable Registers (FMBSEA and FMBSEB) Field Descriptions

Bit	Name	Value	Description
15–0	BSE[15:0]		Bank Sector Enable - When set, a bit enables the corresponding numbered sector for program or erase access. These bits can be set only when PROTL1DIS = 1

Figure 6-8. Bank Sector Enable Registers (FMBSEA and FMBSEB) [offset = 0Ch]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BSE[31:16]															
RWP-0x0000															

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

Table 6-11. Bank Sector Enable Registers (FMBSEA and FMBSEB) Field Descriptions

Bit	Name	Value	Description
15–0	BSE[31:16]		Bank Sector Enable - When set, a bit enables the corresponding numbered sector for program or erase access. These bits can be set only when PROTL1DIS = 1.

6.4.2.7 Bank Ready Register (FMBRDY)

FMBRDY is a half-word register. It allows the user to determine if the associated bank is ready for read access. FMBRDY is a local register. As a result, there is one for each flash bank present.

Figure 6-9. Bank Ready Register (FMBRDY) [offset = 10h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										BANK RDY	Reserved				
R-0										R-U	R-0				

R = Read, W = Write, S = Set, U = Undefined, -*n* = Value after reset

.

Table 6-12. Bank Ready Register (FMBRDY) Descriptions

Bit	Name	Value	Description
15–6	Reserved		Read values are zeros. Writes have no effect.
5	BANKRDY		Bank Ready - This is a read-only bit which allows software to determine if a bank is ready for access before the access is attempted.
4-0	Reserved		Read values are zeros. Writes have no effect.

6.4.2.8 Pump Ready Register (FMPRDY)

FMPRDY is a half-word register. It allows software to determine if the charge pump is ready for flash read access. FMPRDY is a global register. The TMS470Px F05 Flash-ECC has only one FMPRDY register.

Figure 6-10. Pump Ready Register (FMPRDY) [offset = 14h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PUMP RDY	Reserved								
R-0						R-U	R-0								

R = Read, W = Write, S = Set, U = Undefined, -*n* = Value after reset

Table 6-13. Pump Ready Register (FMPRDY) Field Descriptions

Bit	Name	Value	Description
15-10	Reserved		Reads return zeros and writes have no effect.
9	PUMPRDY		Pump Ready - This is a read-only bit which allows software to determine if the charge pump is ready for flash access before the access is attempted.
8-0	Reserved		Reads return zeros and writes have no effect.

6.4.2.9 Module Access Control register 1 (FMMAC1)

FMMAC1 is a half-word register. It supports charge pump sleep wait state generation and Level 1 protection. FMMAC1 is a global register. The flash module has only one, regardless of the number of banks present.

Figure 6-11. Module Access Control register 1 (FMMAC1) [offset = 18h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROTL1DIS	PSLEEP[14:0]														
RWP-0	RWP-0x28														

R = Read, W = Write, S = Set, U = Undefined, -*n* = Value after reset

.

Table 6-14. Module Access Control register 1 (FMMAC1) Field Descriptions

Bit	Name	Value	Description
15	PROTL1DIS		Level 1 Protection Disable - Setting this bit enables writing to the OTPROTDIS bits as well as to the Sector Enable registers, FMBSEA and FMBSEB, for all banks. Clearing this bit disables these accesses.
14–0	PSLEEP[14:0]		Pump Sleep - These bits contain the starting count value for the charge pump sleep down counter. While the charge pump is in sleep mode, the power mode management logic holds the charge pump sleep counter at this value. When the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP SYSCLK cycles before putting the charge pump into standby power mode (the flash module *can not* exit charge pump sleep mode directly to active mode).

6.4.2.10 Module Access Control Register 2 (FMMAC2)

FMMAC2 is a half-word register. It supports control port operations, charge pump fallback power mode, and charge pump standby wait state generation. FMMAC2 is a global register. The flash module has only one, regardless of the number of banks present.

Figure 6-12. Module Access Control Register 2 (FMMAC2) [offset = 1Ch]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSTDBY[10:0]											PMPPWR[1:0]		BANK[2:0]		
RWP-0x14											RWP-11		RWP-000		

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

.

Table 6-15. Module Access Control Register 2 (FMMAC2) Field Descriptions

Bit	Name	Value	Description
15–5	PSTDBY[10:0]		Pump Standby - These bits contain the starting count value for the charge pump standby down counter. While the charge pump is in standby mode, the power mode management logic holds the charge pump standby counter at this value. When the charge pump exits standby power mode, the down counter delays from 0 to PSTDBY SYSCLK cycles before putting the charge pump into active mode.
4-3	PMPPWR[1:0]		Flash Pump Fallback Power Mode - These bits select what power mode the charge pump enters after the pump active grace period (PAGP) counter has timed out.
		00	Sleep (all pump circuits disabled)
		01	Standby (only bandgap reference active)
		10	Reserved
		11	Active (all pump circuits active)
2-0	BANK[2:0]		Bank Enable - These bits select which bank is enabled for operations such as local register access, OTP sector access, and program/erase commands. BANK selects only one bank at a time from up to eight banks depending on the specific device being used. For example, a value of 000 binary selects Bank 0; 101 binary selects Bank 5.

6.4.2.11 Pump Active Grace Period Register (FMPAGP)

FMPAGP is a half-word register. It supports the pump active grace period delay value. FMPAGP is a global register. The flash module has only one, regardless of the number of banks present.

Figure 6-13. Pump Active Grace Period Register (FMPAGP) [offset = 20h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAGP[15:0]															
RWP-0000000000000000															

R = Read, W = Write, S = Set, U = Undefined, -*n* = Value after reset

.

Table 6-16. Pump Active Grace Period Register (FMPAGP) Field Descriptions

Bit	Name	Value	Description
31–16	PAGP[15:0]		Pump Active Grace Period. -This register contains the starting count value for the PAGP mode down counter. Any access to flash memory causes the counter to reload with the PAGP value. After the last access to flash memory, the down counter delays from 0 to 65535 SYSCLK cycles before entering one of the charge pump fallback power modes as determined by PMP-PWR[1:0] in the FMMAC2 register. See Section 6.3.5.2, *Power Mode Control* on page 213 for charge pump activation logic.

6.4.2.12 Module Status Register (FMMSTAT)

FMMSTST is a half-word-access read only register. This register indicates whether an erase or program operation has been completed, suspended, failed, or is in progress. FMSTAT is a global register. The flash module has only one, regardless of the number of banks present.

Figure 6-14. Module Status Register (FMMSTAT) [offset = 24h]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							BUSY	ERS	PGM	INV-DAT	CSTAT	3VSTAT	ESUSP	PSUSP	SLOCK
R-0							R-1->0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

R = Read, W = Write, S = Set, U = Undefined, *-n* = Value after reset

.

Table 6-17. Module Status Register (FMMSTAT) Field Descriptions

Bit	Name	Value	Description
15-9	Reserved		Reads return zeros and writes have no effect.
8	BUSY		Busy - When set, this bit indicates that a program, erase, or suspend operation is being processed, or that the module is in the process of being reset. Initially after reset, BUSY is set, then it is cleared after the flash module is ready for access.
7	ERS		Erase Active - When set, this bit indicates that the flash module is actively performing an erase operation. This bit is set when erasing begins and is cleared when erasing is complete. It is also cleared when the erase is suspended and set when the erase resumes.
6	PGM		Program Active - When set, this bit indicates that the flash module is currently performing a program operation. This bit is set when programming begins and is cleared when programming is complete. It is also cleared when programming is suspended and set when programming is resumes.
5	INVDAT		Invalid Data - When set, this bit indicates that the user attempted to program a “1” where a “0” was already present. This bit is cleared by the Clear Status command.
4	CSTAT		Command Status - When set, this bit informs the host CPU that the program, erase, or validate sector command failed. This bit is cleared by the Clear Status command.
3	VSTAT		VDD Status - When set, this bit indicates if the 3.3V power supply dipped below the lower limit allowable during a program or erase operation. This bit is cleared by the Clear Status command.
2	ESUSP		Erase Suspend - When set, this bit indicates that the flash module has received and processed an erase suspend command. This bit remains set until the erase resume command has been issued.
1	PSUSP		Program Suspend - When set, this bit indicates that the flash module has received and processed a program suspend command. This bit remains set until the program resume command has been issued.

Table 6-17. Module Status Register (FMMSTAT) Field Descriptions (Continued)

Bit	Name	Value	Description
0	SLOCK		Sector Lock Status - When set, this bit indicates that the operation was halted because the target sector was locked for erasing and programming either by the sector protect bit or by write protection key logic. This bit is cleared by the Clear Status command.

6.5 Application Information

6.5.1 Powering Down Flash for Halt Mode

To completely power down all of the flash banks and the flash pumps, the code must be executing from RAM or some memory other than flash when the CPU is halted. The host must first set the fallback power bits to sleep mode. Then, before the host enables halt mode, it must execute from RAM long enough to let the active grace period for all banks and the charge pump expire.

Note: When Putting All Banks Into Sleep or Standby Mode, Ensure That There Are No Accesses to Non-Existing Memory

If all banks are in sleep or standby mode and an access to a non-existing bank is performed, the CPU will hang. This can be avoided by setting the memory decoder to enable only the amount of flash that is physically implemented on the chip.

6.5.2 Setting a Different Number of Wait States for Each Bank

Typically, the number of wait states is set to the same value for all banks; one when in pipeline mode and zero when not in pipeline mode. If for any reason the number of wait states differs between banks, the bank with the higher number of wait states must be set before the bank with the lower number.

Frequency Modulated Zero-Pin Phase-Locked Loop (FMzPLL) Module

This reference guide describes the frequency modulated zero-pin phase-locked loop (FMzPLL) clock module that provides the clock to the system module.

Topic **Page**

7.1 Introduction

The FMzPLL clock module synthesizes the generally higher-frequency continuous system clock source, PLL_CLK, from an external resonator or crystal reference. The “z” in the name implies that no pins are required for an external loop filter.

The phase-locked loop (PLL) multiplies an external frequency reference to a higher frequency for internal use to avoid high-frequency signals on external package pins; removing higher frequencies from package pins helps to reduce electromagnetic interference (EMI). PLLs also avoid the use of crystals above 20 MHz. Such crystals usually operate in overtone mode and require an external tank circuit.

The PLL-based clock module has the following advantages over a simple oscillator:

- Lower external oscillator frequencies reduces EMI
- Frequency modulated output spreads EMI energy over a wider band
- Allows use of lower cost crystals and resonators and avoids tank circuits

Note:
The main disadvantage of the PLL over a simple oscillator is that it can be sensitive to noise if proper board decoupling/layout practices are not adhered to. Thus, the PLL requires more attention to system-level design to ensure low jitter and robust operation.

The following table defines specific terms as they apply to the module.

Table 7-1. Definitions

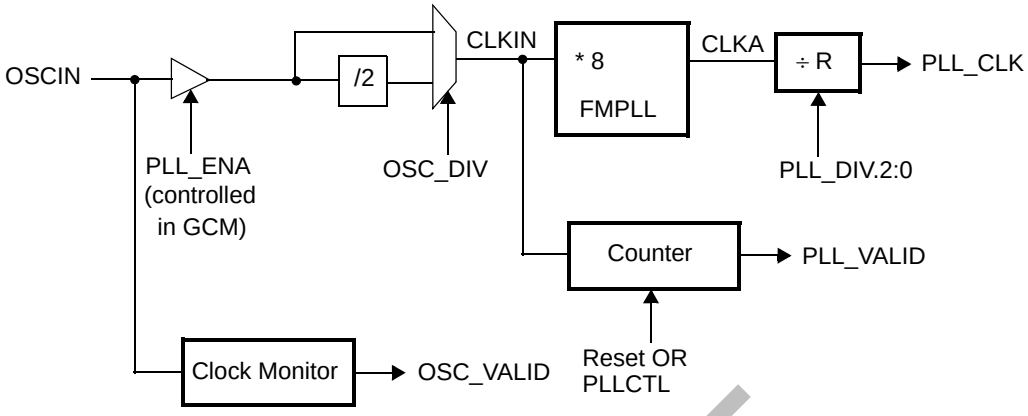
Term	Definition
Charge pump	A circuit that drives a constant current while a pulse-width modulated correction signal is active. Along with the loop filter, the charge pump converts a pulse-width modulated correction signal into an analog control voltage.
Cycle-to-cycle jitter	The maximum ± deviation of a clock edge with respect to its nominal position within a single clock period, expressed in nanoseconds or in a percentage of one clock period. This is sometimes called period jitter.
Electromagnetic interference (EMI)	The radio frequency noise radiated by a circuit that could disturb the proper operation of other equipment, or the radio frequency noise radiated by other equipment which could disturb the proper operation of the subject circuit.
Limp frequency	A slow clock that allows the application to shut down in case of a crystal/resonator failure.
Lock	The condition in which the output of a PLL is synchronized to the phase and frequency of its reference input.
Modulation depth	Percent of deviation from base frequency output of PLL.
Phase frequency detector (PFD)	A circuit that compares two signals in both phase and frequency and can distinguish when one is the harmonic or subharmonic of the other.
Phase-locked loop (PLL)	An oscillator circuit whose output frequency is typically an integer multiple of its reference input frequency.
Voltage controlled oscillator (VCO)	An oscillator whose output frequency is proportional to a control voltage input.

7.1.1 Purpose

The FMzPLL clock module synthesizes the generally higher-frequency continuous system clock source, PLL_CLK, from an external resonator/crystal reference. The oscillator circuit drives an external crystal/resonator, and the FMzPLL multiplies the CLKIN frequency by 8. The PLL output is subsequently divided by

a post-divider value (R), where R is an integer from 1 to 8. Optionally, the frequency can be modulated to reduce the electromagnetic emissions from the clocked capacitance. The FMzPLL allows the application to use a lower frequency external crystal/resonator. Figure 7-1 is a simplified block diagram of the FMzPLL module.

Figure 7-1. FMzPLL Module Block Diagram



7.1.2 Features

The main features of the FMzPLL clock module are:

- Operation in the non-modulation mode as well as FM modulation mode
- The oscillator circuit operates with both resonators and crystals
- OSCIN frequencies can range from 4 to 20 MHz
- Synthesizes PLL_CLK frequencies from 4 MHz up to the device limit
- $f_{PLL\_CLK} = 8 * f_{CLKIN} / R$, where R is any integer from 1 – 8
- $f_{CLKIN} = f_{OSCIN} / N$, where N is 1 or 2
- Programmable modulation depth (Δf)
- Programmable modulation frequency
- Phase-frequency detector (PFD) assures lock to the fundamental reference frequency
- Provides a user-option bit to reset the device if the resonator/crystal fails (not available on all devices; see the device-specific data sheet.)

7.1.3 Clock Module Pins

The following table describes the clock module's significant pins.

Table 7-2. Clock Module Pins

Pin	Description
OSCIN	Input for the crystal/resonator clock input
OSCOUT	Drives the crystal/resonator

The uses of these pins are described in Section 7.2.

The crystal oscillator requires two pins (OSCIN and OSCOUT) for the crystal and the external load capacitors. The load capacitors tie back to the nearest V_{SS} pin. The bias resistor of the oscillator is

integrated, so external resistors are unnecessary unless a series resistor is required by the resonator manufacturer, or if EMI data shows strong overtones of the crystal.

7.2 Operation

The resonator/crystal creates a reference frequency that is used as the input for the PLL. The PLL then multiplies the reference frequency by 8; the post-divider block scales this value by 1, 2, 3, 4, 5, 6, 7, or 8 resulting in PLL_CLK.

When the PLL is used in the frequency modulation mode, the programmable modulation block varies the PLL_CLK frequency symmetrically around the baseline frequency ($f_{baseline}$ = CLKIN*8). The frequency ranges from f+Δf to f-Δf in a period defined by 1/f_{mod}; the modulation waveform is a symmetrical ramp.

During power up or when exiting hibernate mode, platform devices start running from OSCIN. If the user switches the system clock from OSCIN to PLL_CLK too soon, a counter will hold off further software execution until the PLL has had time to lock. The counter delays releasing PLL_LOCKED by 4096 CLKIN cycles. When entering or leaving frequency modulation mode, the PLL output clock is held an additional 4096 CLKIN cycles.

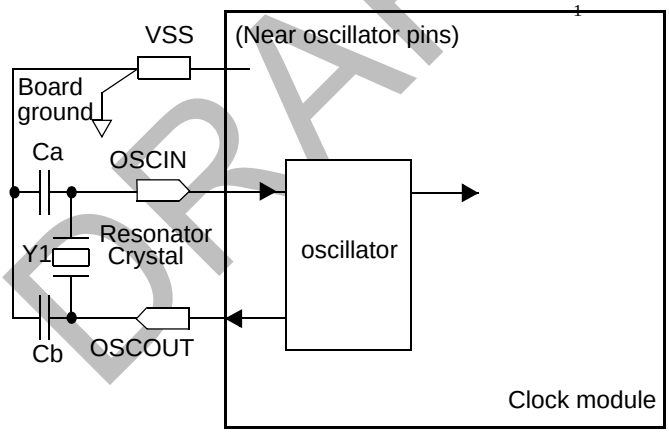
The clock monitor detects CLKIN edges. When the CLKIN frequency is outside a frequency range, the clock monitor sets the fail flag in the system module. Optionally, a system reset can be forced when such a failure is detected. This reset mechanism is disabled during power-up and during the low power modes snooze, sleep, and hibernate (when this oscillator is turned off).

The following sections explain the blocks that compose the FMzPLL clock module.

7.2.1 Resonator/Crystal Oscillator

The oscillator requires two external pins: OSCIN and OSCOUT, which are connected to the resonator/crystal and load capacitors (Figure 7-2). The oscillator is a single stage inverter held in bias by an integrated bias resistor. This resistor is disabled only during leakage test measurements and hibernate mode.

Figure 7-2. Reference Resonator/Crystal



1 To reduce EMI, keep all of these routes short and minimize loop areas.

Note: Vendor Validation of Resonators/Crystals

Texas Instruments strongly encourages each customer to submit device samples to the *resonator/crystal vendor* for validation. The vendor is equipped to determine what load capacitors will best tune their resonator/crystal to the microcontroller device for optimum start-up and operation over temperature and voltage extremes. These vendors also factor in margins for variations in the microcontroller processes.

The load capacitors should be grounded back to the nearest device ground pin with a private run as short as practical to minimize electromagnetic interference. Keep the loop area formed by the connection of these pins and components as small as possible. EMI is radiated by high-frequency current flowing around loops.

The larger the loop area, the more EMI energy is radiated. Keeping these oscillator traces tightly packed, minimizes loop area, and therefore, EMI.

Note: Driving the OSCIN Pin Directly
External frequency sources must be driven into the OSCIN pin, and the OSCOUT pin should be left open. Keep in mind that the OSCIN pin is 1.8V CMOS.

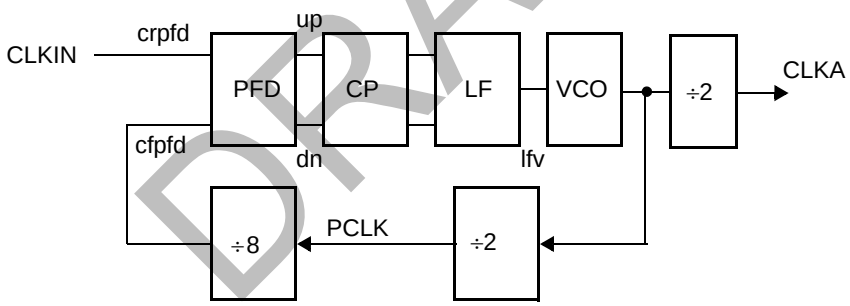
7.2.2 Phase-Locked Loop (PLL)

The PLL block consists of five logical sub-blocks:

- Phase-frequency detector (PFD)
  - Charge pump (CP)
  - Loop filter (LF)
  - Voltage controlled oscillator (VCO)
  - Feedback dividers (÷2 and ÷8)
  - Frequency modulator (shown in Figure 7-6)

Figure 7-3 illustrates these sub blocks in a basic PLL circuit without frequency modulation (FM). The VCO adjusts its frequency until the two signals into the PFD have the same phase and frequency. The feedback path (from VCO output to PFD cfpfd) divides the frequency of the feedback signal by 16; this feedback divider requires the VCO to generate a frequency 16 times greater than CLKIN. In the forward path (from VCO to PLL_CLK), an additional divide-by-2 block creates a 50% duty cycle and reduces the effective multiplier ratio to 8.

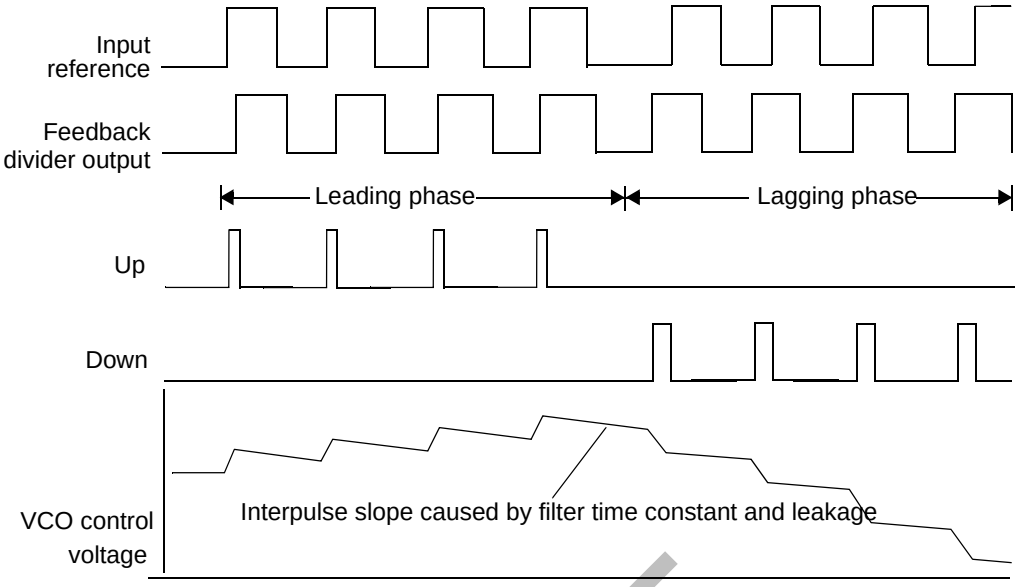
Figure 7-3. PLL Circuit without FM Section



7.2.2.1 Phase-Frequency Detector

The phase-frequency detector (PFD) compares the input reference phase/frequency to that of the feedback divider and generates two pulse-width modulated signals: an *up* pulse and a *down* pulse which drive a charge pump. The resulting charge, when integrated by the low-pass filter (LF) block, provides a VCO control voltage, as shown in Figure 7-4.

Figure 7-4. PFD Operation



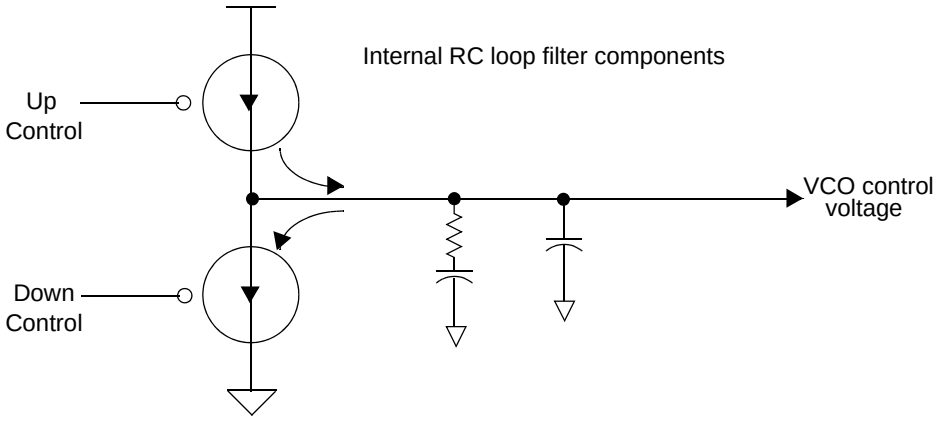
The frequency of the pulses is equal to the CLKIN frequency. The width of the up pulse and the down pulse depends on the difference in phase between the two PFD inputs. For example, when the reference input leads the feedback input by 10 ns, then an up pulse of approximately 10 ns is generated (see Figure 7-4). On the other hand, when the reference input lags the feedback input by 10 ns, then a down pulse of approximately 10 ns is generated. When the two inputs are exactly in phase, the up pulse and down pulse become essentially zero-width. These pulses are fed to the charge pump block, which meters charge into the low-pass loop filter.

The advantage of a phase-frequency detector over a phase-only detector is that PFDs cannot lock to a harmonic or subharmonic of the reference frequency. This property ensures that the output frequency of the VCO is always exactly 16 times the reference frequency after locking.

7.2.2.2 Charge Pump

The charge pump (CP), shown in Figure 7-5, consists of two gated current sources that either add or remove charge from the filter capacitors of the LF block, depending on the pulses coming from the PFD.

Figure 7-5. Charge Pump Functional Model



7.2.2.3 Loop Filter

Two components of the filter output signal are summed together: an integral component and a proportional component. The integral component maintains a DC level going to the VCO to set its frequency, and the proportional component makes the VCO track changes in phase. In the FMzPLL module, all components required for the filter are integrated.

7.2.2.4 Voltage Controlled Oscillator

The output frequency of the VCO is proportional to its input control voltage, which is generated by the charge pump via the integrated loop filter. If the VCO oscillates too slowly, the reference phase begins to lead the feedback phase at the PFD, which increases the control voltage to the VCO. Conversely, if the VCO oscillates too fast, the reference phase begins to lag the feedback phase, decreasing the control voltage to the VCO. These two actions keep the VCO running at the correct frequency multiple of the reference.

7.2.2.5 Frequency Modulation

During frequency modulation the modulation counter and charge pump 2 (CP2) modulate the VCO frequency at the loop filter (LF) (Figure 7-6). The output clock of the PLL changes frequency in a symmetrical ramp pattern, centered about the unmodulated output frequency (Figure 7-7).

Figure 7-6. PLL Modulation Block Diagram

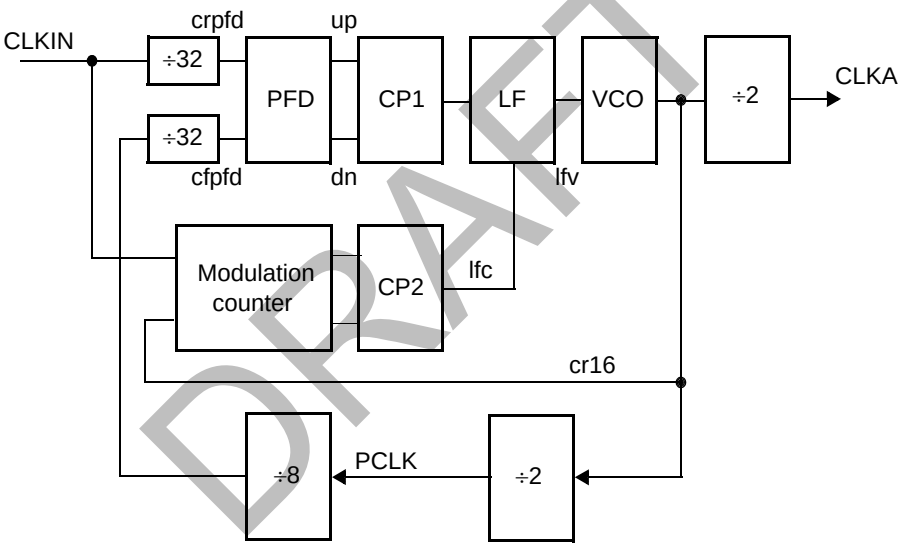
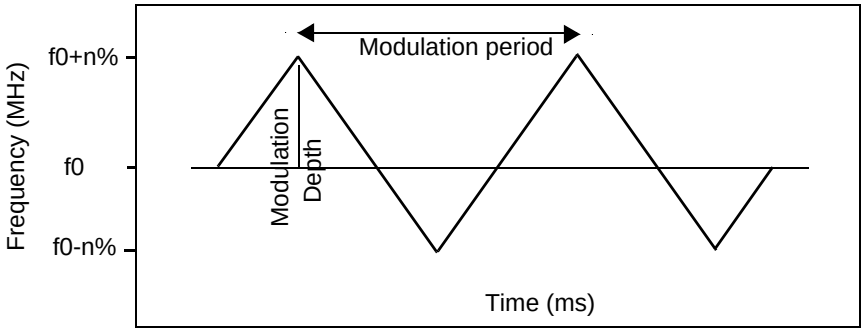


Figure 7-7. FM Mode Frequency vs. Time



Frequency modulation is set up and enabled from register PLLCTL2 (see Section 7.3.2). Modulation depth is controlled by PLL MOD_DEPTH[1:0] as 0.5%, 1%, 2%, or 4%.

Modulation frequency is CLKIN prescaled by PLL_MOD_FREQ[5:0] left-shifted by three bits. The modulation Frequency is given by (CLKIN³8) / (PLL_MOD_FREQ[5:0]).

Note: Effect of OSC_DIV on FM

The modulation frequency formula above refers to CLKIN; however, the frequency of CLKIN is either OSCIN or OSCIN÷2 depending on the state of the OSC_DIV bit (PLLCTL1.22).

After modulation depth and frequency have been setup, then FM mode can be enabled by setting the FM_ENA bit.

The frequency-modulating charge pump charges C1 in the 1st half of the modulating period and discharges it in the 2nd half (see Figure 7-7).

7.2.3 Clock Monitor

The clock monitor is intended to detect a gross failure of the oscillator system. If the crystal/oscillator should stop or slow down to a few hundred KHz because of some failure of the oscillator or its external components, the output of the clock monitor OSC_VALID becomes inactive setting the OSC_FAIL flag in the GLBSTAT register. This detection mechanism is disabled during oscillator startup from either power-on or from low power modes in which the oscillator is disabled.

If the ROF bit in the PLLCTL1 register is set when the oscillator fails, then a system reset occurs.

If a system reset occurs because of an oscillator failure, then the OSC_RST history bit in the SYSESR register is set. This bit is not cleared by a reset, allowing the user to determine why the reset occurred.

Note: Over-Frequency Detection

It is possible for the resonator to become damaged and speed up in frequency, and not be detected by the clock monitor circuit until the frequency is much greater than intended.

7.2.4 Clock Counter

The clock counter keeps PLL_CLK held off until the FMzPLL has time to lock each time it is enabled. If the user switches from an enabled clock to the FMzPLL clock before 4096 CLKIN cycles have expired, code execution will be stopped until the end of the 4096 cycle delay.

The clock counter only starts counting after the oscillator monitor has detected resonator/crystal oscillation and after the FMzPLL has been enabled.

When low power modes such as hibernate are exited, or when the frequency modulation (FM) mode is changed, PLL_CLK is also held off for 4096 oscillator cycles. Both of these mechanisms prevent code execution until the VCO frequency has restabilized.

7.2.5 Feedback Divider (Divide-by-16)

The feedback divider is a modulo-16 counter that divides the VCO output frequency by sixteen. The output of the divider becomes the *feedback phase*. When the VCO frequency is divided by sixteen, the reference frequency (CLKIN) is effectively multiplied by 16. This occurs because the VCO has to run 16 times as fast to produce a feedback signal with the same frequency as the reference signal.

7.2.6 Divide-by-2

After the VCO has generated a frequency 16 times the reference frequency, the VCO output is divided by 2 to produce a 50% duty cycle output clock, CLKA.

7.2.7 Divide-by-R Post-Divider

CLKA (now 8 times CLKIN) gets divided by R to produce the final PLL_CLK signal. R is any integer from 1 to 8 inclusive, and is programmed by bits PLL_DIV[2:0] in register PLLCTL1. Programming these bits allows a glitch-free transition from the current frequency to the frequency specified by the bits.

At power-up, the default of OSC_DIV is divide-by-one and the default of PLL_DIV is divide-by-eight so the initial frequency of PLL_CLK is equal to the resonator/crystal frequency.

7.3 Global Control Registers

The clock module has two registers (PLLCTL1 and PLLCTL2) located within the system module, plus it has four bits located in other global control registers of the system module.

The FMzPLL is off at power-on. It may be turned on by clearing the CLK_SR1_OFF bit in the CSDIS register of the System module.

The clock module generates the OSC_FAIL flag if a problem with the reference oscillator is detected. This flag is located in the GLBSTAT register.

If the ROF bit in the PLLCTL1 register is set when the oscillator fails, then a system reset occurs, and the OSC_RST history bit is set in the SYSESR register.

The following sections describe the two registers used in the system module. These registers support 8, 16, and 32-bit write accesses. The reset values for these registers are configurable at the device level. Please check the device specification for the bit reset states and actual physical address for the System Module registers as only the relative offset addresses are specified here. The PLL control registers are in the System Control Registers frame. (Section 5.1.26;Section 5.1.27).

7.3.1 PLL Control Register 1 (PLLCTL1)

Figure 7-8 illustrates this register and Table 7-3 provides the bit descriptions.

Figure 7-8. PLL Control Register 1 (PLLCTL1)[offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ROF	OSC_DIV	Reserved					
R-0								R/WP-0	R/WP-0	R-0					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PLL_DIV.2:0		
R-0													R/WP-111		

R = Read; WP = Write in privilege mode only; -*n* = Value after reset

Table 7-3. PLL Control Register 1 (PLLCTL1) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return zeros. Writes have no effect.
23	ROF		Reset on oscillator fail.
		0	Do not reset the system when the oscillator is out of range
		1	Reset the system when the oscillator is out of range. This reset mechanism is disabled during power-up and during low power modes, when the oscillator is either starting up, or is turned off.
22	OSC_DIV		Oscillator frequency divide select.
		0	$f_{CLKIN} = f_{OSCIN}$
		1	$f_{CLKIN} = f_{OSCIN}$ 32

Table 7-3. PLL Control Register 1 (PLLCTL1) Field Descriptions (Continued)

Bit	Name	Value	Description
21–3	Reserved		Reads return zeros. Writes have no effect.
2–0	PLL_DIV.2:0		PLL divider bits. The clock divider bits determine the divisor of the VCO output. Programming these bits causes a glitch-free transition from the current frequency to the frequency specified by the bits.
		000	$f_{PLL\_CLK} = f_{CLKA} \div 1$
		001	$f_{PLL\_CLK} = f_{CLKA} \div 2$
		010	$f_{PLL\_CLK} = f_{CLKA} \div 3$
		011	$f_{PLL\_CLK} = f_{CLKA} \div 4$
		100	$f_{PLL\_CLK} = f_{CLKA} \div 5$
		101	$f_{PLL\_CLK} = f_{CLKA} \div 6$
		110	$f_{PLL\_CLK} = f_{CLKA} \div 7$
		111	$f_{PLL\_CLK} = f_{CLKA} \div 8$

7.3.2 PLL Control Register 2 (PLLCTL2)

The PLLCTL2 register controls the frequency modulated mode of operation of the FMzPLL. The frequency modulation option is available for applications that have critical EMC considerations. Figure 7-9 illustrates this register and Table 7-4 provides the bit descriptions.

Figure 7-9. PLL Control Register 2 (PLLCTL2)[offset = 74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								FM_ ENA	Reserved						
R-0								R/WP-0	R-0						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PLL_MOD_FREQ[5:0]						Reserved						PLL_MOD_ DEPTH[1:0]	
R-0		R/W{P-0						R-0						R/W{P-0	

R = Read; WP = Write in privilege mode only; -*n* = Value after reset

Table 7-4. PLL Control Register 2 (PLLCTL2) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return zeros. Writes have no effect.
23	FM_ENA		Frequency modulation enable.
		0	PLL is in non-frequency modulation mode.
		1	PLL is in frequency modulation mode.
22–14	Reserved		Reads return zeros. Writes have no effect.
13–8	PLL_MOD_FREQ[5:0]		PLL modulation frequency. These bits set the PLL's modulation frequency when the PLL is in frequency modulation mode. This field must be set before enabling frequency modulation and must not be changed when frequency modulation mode is active. The modulation frequency is given by (CLKIN ÷ 8) / (PLL MODFREQ.5:0) **Note:** Refer the specific device datasheet to identify any limitations on the range of allowed values for the PLL MOD FREQ setting.
7-2	Reserved		Reads return zeros. Writes have no effect.

Table 7-4. PLL Control Register 2 (PLLCTL2) Field Descriptions (Continued)

Bit	Name	Value	Description
1-0	PLL_MOD_DEPTH.1:0		PLL modulation depth. These bits set the PLL's modulation depth (the maximum deviation as a percentage of the base frequency) when the PLL is in frequency modulation mode. This field must be set before enabling frequency modulation and must not be changed when frequency modulation mode is active.
		00	Modulation depth A selected
		01	Modulation depth B selected
		10	Modulation depth C selected
		11	Modulation depth D selected
			Note: The actual modulation depths achieved for each of the above four settings will be determined based on the PLL characterization. Please refer to the post-characterization release datasheet for these values.

7.4 Important Considerations

The following sections describe some important considerations for the use of the features provided by the FMzPLL module.

7.4.1 Frequency Modulation

Do not change the Modulation Frequency or the Modulation Depth after frequency modulation is enabled. These two values MUST be configured before frequency modulation is enabled.

When frequency modulation is either enabled or disabled, the FMzPLL waits for 4096 cycles of CLKIN to allow the PLL to reacquire lock. During this time, PLL_CLK is switched off.

7.4.2 Multiplication/Division

The PLL's multiplication ratio is fixed at 8. It cannot be changed except by modifying OSC_DIV.

The FMzPLL provides eight different divide ratios (R) ranging from 1 to 8. This divider value can be changed on the fly without the PLL needing to reacquire lock.

The FMzPLL's output clock frequency can be calculated by,

$f_{PLL\_CLK} = 8 * f_{CLKIN} / R$ where $R = \{1,2,...,8\}$

and

$f_{CLKIN} = f_{OSCIN} / N$ where $N = \{1,2\}$

7.4.3 PLL ON/OFF Control

After power-up, the PLL is disabled. The PLL can be turned ON/OFF by the CSDIS register of the System Module. Please refer to the TMS470Px Architecture specification for more details.

Once the PLL is switched ON, the FMzPLL wrapper waits for 4096 CLKIN cycles to allow the PLL to acquire lock. The PLL output clock is disabled during this time.

Vectored Interrupt Manager (VIM) Module

This section describes the behavior of the vectored interrupt manager (VIM) module of the TMS470Px family.

Topic **Page**

8.1 Overview

The vectored interrupt manager (VIM) provides hardware assistance for prioritizing and controlling the many interrupt sources present on a device. Interrupts are caused by events outside of the normal flow of program execution. Normally, these events require a timely response from the central processing unit (CPU); therefore, when an interrupt occurs, the CPU switches execution from the normal program flow to an interrupt service routine (ISR).

8.1.1 Interrupt Handling at the CPU

The ARM CPU provides two vectors for interrupt requests—fast interrupt requests (FIQ) and normal interrupt requests (IRQs). FIQs are higher priority than IRQs, and FIQ interrupts may interrupt IRQ interrupts.

The CPU may enable these interrupt request channels individually within the CPSR; CPSR bits 6 and 7 must be cleared to enablethe FIQ and IRQ interrupt requests to the CPU. When the CPU receives an interrupt request, the CPSR changes to either FIQ or IRQ mode. When an IRQ interrupt is received, the CPU disables other IRQ interrupts by setting CPSR bit 7. When an FIQ interrupt is received, the CPU disables both IRQ and FIQ interrupts by setting CPSR bits 6 and 7. After the interrupt is received by the CPU, the program counter jumps to the appropriate interrupt vector—0x18 for IRQ and 0x1C for FIQ.

Note:
The TMS470Px family does not support the HIVECT interrupt. (0xFFFF 00xx)

8.1.2 Interrupt Generation at the Peripheral

Interrupts begin when an event occurs within a peripheral module. Some examples of interrupt-capable events are expiration of a counter within a timer module, receipt of a character in a communications module, and completion of a conversion in an analog-to-digital converter (ADC) module. Some TMS470Px peripherals are capable of requesting interrupts on more than one interrupt channel.

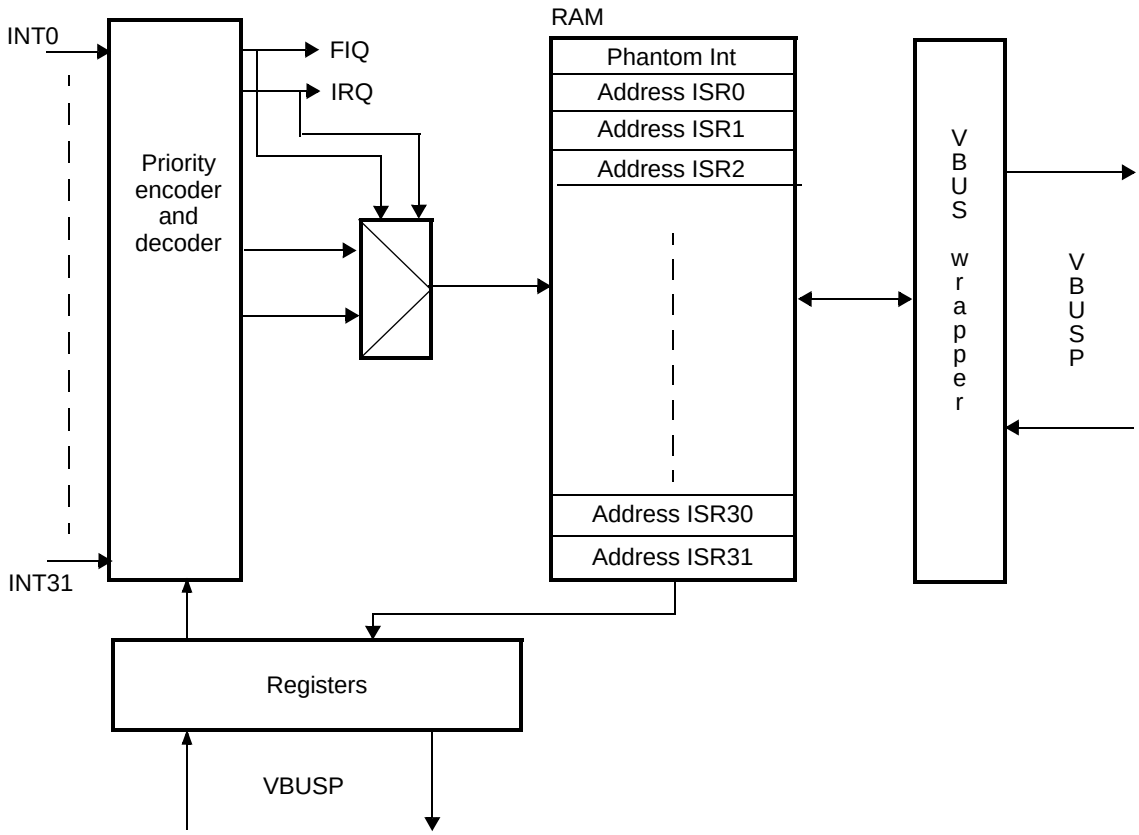
Interrupts are not always generated when an event occurs; the peripheral must make an interrupt request to the VIM based on the event occurrence. Typically, the peripheral contains:

- An interrupt flag bit for each event to signify the event occurrence
- An interrupt enable bit to control whether the event occurrence causes an interrupt request to the VIM.

8.2 Interrupt Management

A block diagram of the VIM is shown in Figure 8-1.

Figure 8-1. VIM Block Diagram



The VIM supports 32 interrupt request lines (INT[31:0]) from the peripherals. These peripheral interrupt requests are hardwired to each of the VIM interrupt request inputs, and these connections are device specific.

All interrupt requests (INT[31:0]) are mapped to a specific channel; this mapping is fully programmable and allows the software to reorder the interrupt priority. All requests pass through a synchronizer to prevent setup time violations; VIM samples these requests from the synchronizer every peripheral clock cycle. The VIM combines all the interrupt requests into two outputs: an FIQ request to the CPU and an IRQ request to the CPU.

The VIM performs the following functions:

- Maps the 32 interrupt requests to the 32 interrupt channels
  - Provides programmable priority for the request lines
  - Manages interrupt channels through masking
  - Prioritizes interrupt channels to the CPU
  - Provides the CPU with the address of the interrupt service routine (ISR) for each interrupt

The VIM provides the following different possibilities for software to handle interrupts:

1. TMS470R1x legacy code

   The CPU branches to 0x18 (IRQ) or 0x1C (FIQ) to execute the main ISR. The main ISR routine reads the offset register (IRQIVEC, FIQIVEC) to determine the source of the interrupt.

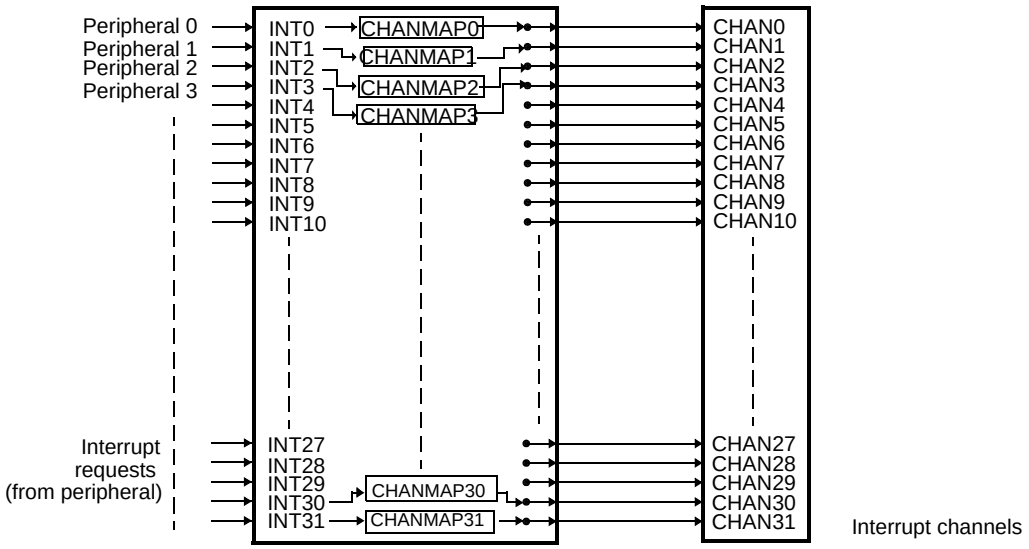
2. Vectored interrupts

   The CPU executes the instruction placed at 0x18 or 0x1C (IRQ or FIQ vector). The instruction should be a *LDR PC, [PC,#-0x1B0]*; the offset indicates the relative address of the interrupt vector register (IRQVECREG for IRQ interrupt, FIQVECREG for FIQ interrupt) within the vectored interrupt manager (VIM). This register will give the CPU the address of the Interrupt Service Routine (ISR) corresponding to the highest active IRQ.

8.2.1 VIM IRQ Management

The programmability of the VIM allows software to control the interrupt priority. A block diagram of the VIM interrupt requests arrangement from peripheral modules to the interrupt channels, shown in the default state following the reset, is provided in Figure 8-2. In the reset state, the VIM maps all of the interrupt requests in the system to their respective interrupt channels.

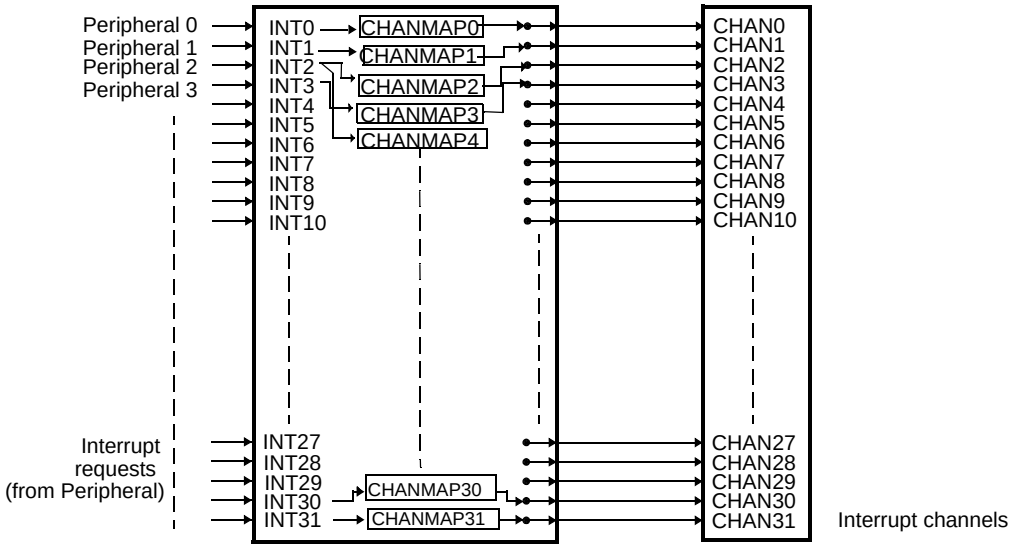
Figure 8-2. VIM in Default State



For each interrupt channel, CHAN[x], there is a corresponding mapping register bit field CHANMAPx[6:0], which determines which VIM request the interrupt channel maps to. With this scheme, the same request can be mapped to multiple channels.

Figure 8-3 shows the VIM INT2 is remapped to both Channel 2 and 4, and INT3 is mapped to channel 3.

Figure 8-3. VIM in a Programmed State



Note:

By mapping INT2 to channel 2 and channel 4 and mapping INT3 to channel 3, it is possible for the software to change the priority dynamically by changing the MASK register.
When channel 2 is enabled, the priority is:

a.INT0

b.INT1

c.INT2

d.INT3

e....

Disabling channel 2, the priority becomes:

a.INT0

b.INT1

c.INT3

d.INT2

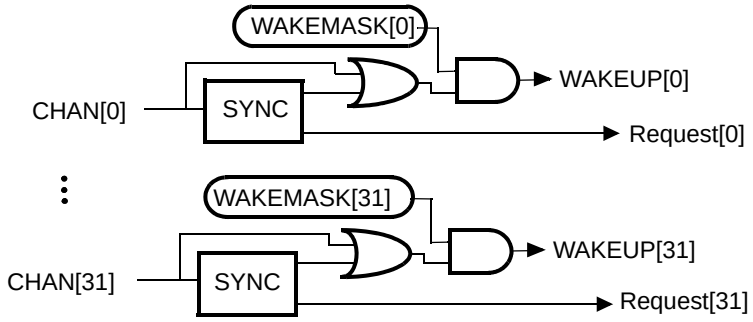
e....

8.2.2 VIM Wakeup Interrupt

The wakeup interrupts are used to come out of low power mode (LPM). Any interrupt can be used to wake up the device. After reset, all interrupts are set to wake up from LPM. However, the VIM can mask unwanted interrupt lines for wake-up by using the WAKEMASK register.

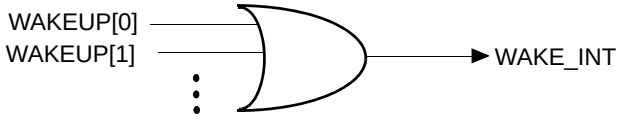
Figure 8-4 shows the implementation of each interrupt request. The request is synchronized before it reaches channel selection muxes.

Figure 8-4. Detail of the IRQ Input



The WAKEMASK registers will enable/disable an interrupt for wake-up from low-power mode. All wake-up interrupts are “ORed” into a single signal connected to the Global Clock Module (see Figure 8-5).

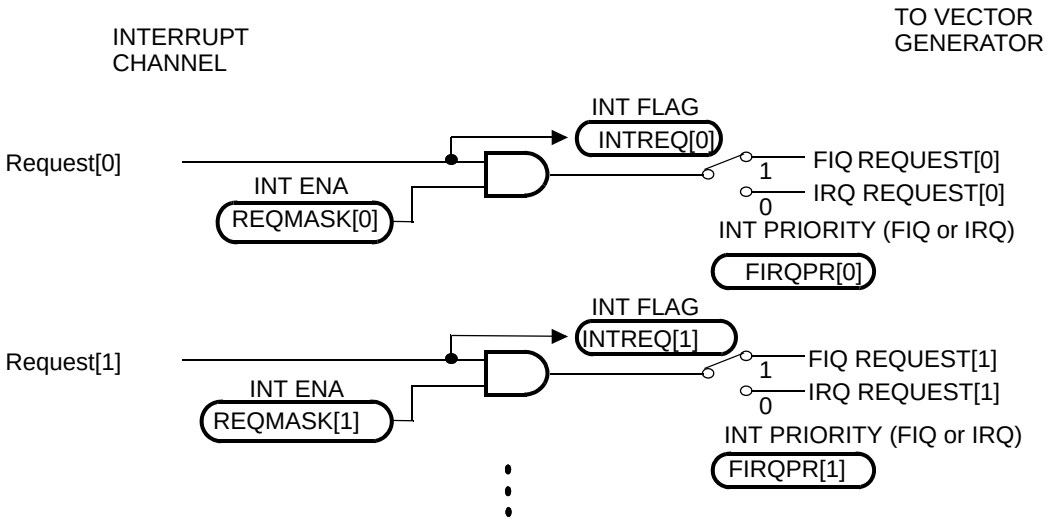
Figure 8-5. Wakeup Interrupt Generation



8.2.3 VIM Input Channel Management

On the input side, the VIM enables channels on a channel-by-channel basis (in the REQMASK registers); unused channels may be masked to prevent spurious interrupts. Each interrupt channel can be assigned to send either an FIQ or IRQ request to the CPU (in the FIQPR register). See Figure 8-6.

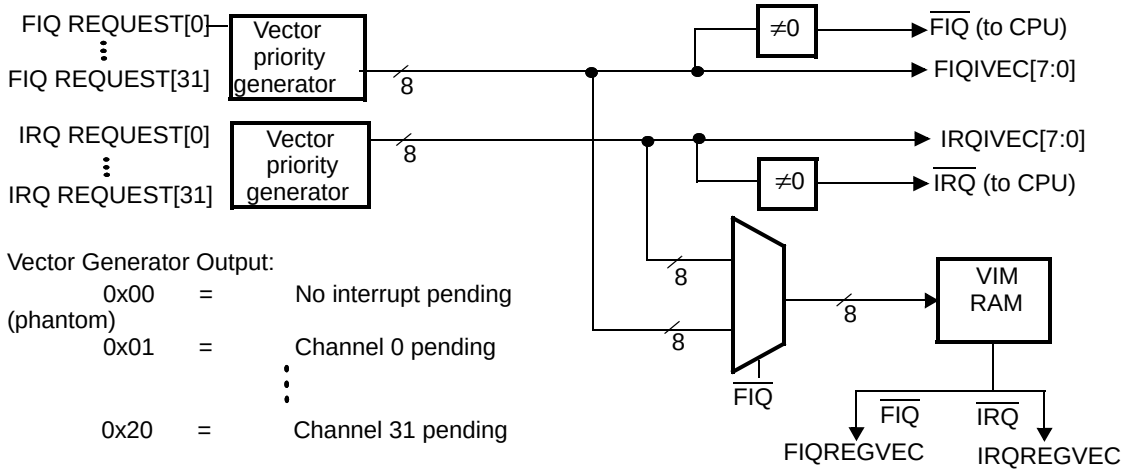
Figure 8-6. Channel Management Block Diagram



8.2.4 VIM Prioritization

The VIM prioritizes the received interrupts based upon a programmed prioritization scheme. The VIM can send two interrupt requests to the CPU simultaneously—one IRQ and one FIQ. If both interrupt types are enabled at the CPU level, then the FIQ has greater priority and is handled first. The VIM provides a default prioritization scheme, which sends the lowest numbered active channel (in each FIQ and IRQ) to the CPU. Within the FIQ and IRQ classes of interrupts, the lowest channel has the highest priority interrupt. However, the VIM provides a fully programmable priority scheme. Each request could be affected with a unique priority weight, which allows the VIM to decide which request has the highest priority at a given time. See Figure 8-7.

Figure 8-7. Priority, Vector, and ISR Address Generation Block Diagram



After the VIM has generated the vector corresponding to the highest active IRQ, it updates the FIQIVEC or the IRQIVEC register, depending on the class of interrupt. Then, it accesses its internal RAM using the vector value to fetch the address of the corresponding ISR. If the request is an FIQ class interrupt, the address read from memory, is written to the FIQVECREG register. If the request is an IRQ class interrupt, the address is written to the IRQVECREG register.

All of the interrupt registers are updated when a new high priority interrupt line becomes active.

8.3 VIM Operation

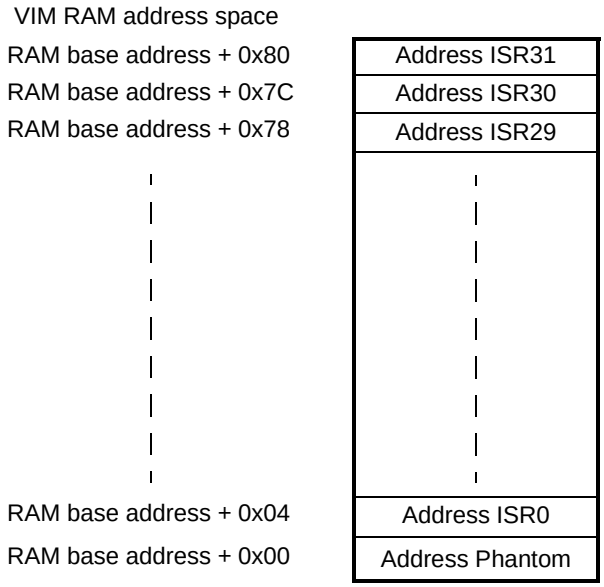
The following sections provide details about the operation of the VIM.

8.3.1 VIM Initialization

After reset, the VIM RAM is not initialized. Therefore, the CPU need to initialize all of the interrupt addresses into the RAM, before enabling the corresponding interrupt channel. This initialization is only required when vectored interrupts are used, legacy interrupt management does not need the RAM to be initialized.

The RAM is organized in 33 words of 32 bits. 32-bit, 16-bit, and 8-bit accesses are supported. Figure 8-8 shows the interrupt memory mapping. The RAM base address is 0xFFF8 2000h.

Figure 8-8. VIM Interrupt Address Memory Map



8.3.2 VIM RAM Arbitration

The VIM RAM is accessible by both the VBUS interface and the VIM internal state machine. If both are accessing the RAM at the same time, the VIM read always has the priority over the VBUS interface. This is to guaranty that the vector registers are updated in the same cycle that the IRQ or FIQ signal are propagated to the CPU.

8.3.3 VIM Response to Interrupts

The following sections describe how the VIM responds to interrupts.

8.3.3.1 Vectored Interrupt

When the CPU recognizes an interrupt request and responds, the program counter jumps to the appropriate interrupt vector. The interrupt vector should be a *LDR PC, [PC,#-0x1B0]* (see Example 1). The interrupt table reads the pending ISR address within the VIM memory from the vector. The address is written into the corresponding vector register (IRQVECREG for IRQ, FIQVECREG for FIQ). The CPU reads the content of the register and branches to the ISR. Example 1 illustrates the configuration for the exception vectors.

Example 1. Exception Vector Configuration for VIM Vector

```
          .sect ".intvecs"
00000000h b _RESET             ; RESET interrupt
00000004h b _UNDEF_INST_INT    ; UNDEFINED INSTRUCTION interrupt
```

```
00000008h b _SW_INT             ; SOFTWARE interrupt
0000000Ch b _ABORT_PREF_INT  ; ABORT (PREFETCH) interrupt
00000010h b _ABORT_DATA_INT  ; ABORT (DATA) interrupt
00000014h b #-8         ; Reserved
00000018h ldr pc,[pc,#-0x1B0]   ; IRQ interrupt
0000001Ch ldr pc,[pc,#-0x1B0]   ; FIQ interrupt
```

8.3.3.2 Legacy TMS470R1x

The VIM is also compatible with the TMS470R1x (CIM) module and provides the same interrupt registers. This mode could be used if legacy code needs to be reused, porting it from the TMS470R1x family. However, software thus imported will not benefit from the whole range of performance the VIM is capable of.

To port legacy software, the interrupt vector only needs to be a branch statement to a software interrupt table. The software interrupt table reads the pending interrupt from a vector offset register (FIQVEC[7:0] for FIQ interrupts and IRQVEC[7:0] for IRQ interrupts). All pending interrupts can be viewed in the INTREQ register.

Example 2 shows a fast response to the FIQ interrupt and can be applied to a system that has more than one channel assigned as a FIQ.

Example 2. How to Respond to FIQ With Short Latency

```
          .sect ".intvecs"
00000000h b _RESET    ; RESET interrupt
00000004h b _UNDEF_INST_INT; UNDEFINED INSTRUCTION interrupt
00000008h b _SW_INT   ; SOFTWARE interrupt
0000000Ch b _ABORT_PREF_INT; ABORT (PREFETCH) interrupt
00000010h b _ABORT_DATA_INT; ABORT (DATA) interrupt
00000014h b #-8       ; Reserved
00000018h b _IRQ_ENTRY_0; IRQ interrupt
                     ;**********************************
                     ; INTERRUPT PROCESSING AREA
                     ;**********************************
0000001Ch ldrb R8, [PC,#-0x21d]; FIQ INTERRUPT ENTRY
                     ; R8 used to get the FIQ index
                              ; with address pointer to the
                              ; first FIQ banked register
00000020h ldr PC, [PC, R8, LSL#2]; Branch to the indexed interrupt
                              ; routine. The prefetch
                              ; operation causes the PC to be 2
                              ; words (8 bytes) ahead of the
                              ; current instruction, so
                              ; pointing to _INT_TABLE.
00000024h nop         ; Required due to pipeline.

                     ;==================================
00000028h _INT_TABLE ; FIQ INTERRUPT DISPATCH
                     ;==================================

0000002Ch .word _FIQ_TABLE; beginning of FIQ Dispatch
00000030h .word _ISR1; dispatch to interrupt routine 1
00000034h .word _ISR2; dispatch to interrupt routine 2
          .
          .
```

Another way to improve the FIQ latency is to assign only one channel to the FIQ interrupt and to map the ISR code corresponding to this channel directly starting at 0x1C.

8.3.4 Emulation

The VIM continues to operate during debug mode; all registers are updated continuously. In debug mode, registers can be updated by the debugger independently of the CPU mode.

8.4 Capture Event Sources

The VIM can select any of the interrupt request sources to generate capture events for the real-time interrupt (RTI) module (see Figure 8-9). Two registers are available to select, for each capture event source, the interrupt request line after synchronization.

Figure 8-9. Capture Event Sources

8.5 Registers

This section details the VIM module registers, summarized in Figure 8-10. A detailed description of each register and its bits is also provided.

Each register begins on a word boundary. All registers are 32-bit, 16-bit and 8-bit accessible. The start address of the VIM module is 0xFFFF FE00.

Figure 8-10. VIM Control Register Summary

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xFDFC	Reserved															
	Reserved															
0xFE00h IRQIVEC Page 262	Reserved															
	Reserved								IRQIVEC[7:0]							
0xFE04h FIQIVEC Page 262	Reserved															
	Reserved								FIQIVEC[7:0]							
0xFE08h–FE0Ch	Reserved															
	Reserved															
0xFE10h FIRQPR0 Page 264	FIRQPR[31:16]															
	FIRQPR[15:0]															
0xFE20h INTREQ0 Page 265	INTREQ[31:16]															
	INTREQ[15:0]															
0xFE30h REQMSKSET0 Page 266	REQMASKSET[31:16]															
	REQMASKSET[15:0]															

Figure 8-10. VIM Control Register Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xFE40h REQMSKCLR0 Page 267	REQMASKCLR[31:16]															
	REQMASKCLR[15:0]															
0xFE50h WAKMSKSET0 Page 268	WAKEMASKSET[31:16]															
	WAKEMASKSET[15:0]															
0xFE60h WAKMSKCLR0 Page 269	WAKEMASKCLR[31:16]															
	WAKEMASKCLR[15:0]															
0xFE70h IRQVECREG Page 270	IRQ[31:16]															
	IRQVECREG[15:0]															
0xFE74h FIQVECREG Page 271	FIQVECREG[31:16]															
	FIQVECREG[15:0]															
0xFE78h CAPEVT Page 272	Reserved															
	Reserved															
0xFE7Ch	Reserved															
	Reserved															
0xFE80h– 0xFE9Ch CHANCTRL[0:7] Page 273	Reserved	CHANMAPx$_0$[6:0]							Reserved	CHANMAP1x$_1$[6:0]						
	Reserved	CHANMAPx$_2$[6:0]							Reserved	CHANMAPx$_3$[6:0]						

8.5.1 VIM Offset Vector Registers

The VIM offset register provides the user with the numerical index value that represents the pending interrupt with the highest precedence. The register IRQIVEC holds the index to the highest priority IRQ interrupt; the register FIQIVEC holds the index to the highest priority FIQ interrupt. The index can be used to locate the interrupt routine in a dispatch table, as shown in Table 8-1.

Table 8-1. Interrupt Dispatch

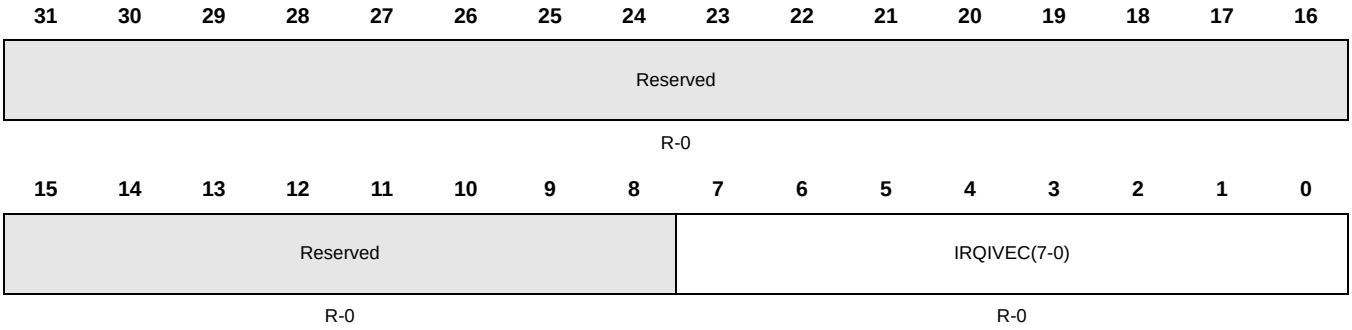
IVEC Index	Pending Interrupt
0x00	No interrupt
0x01	Channel 0
:	:
0x20	Channel 31

The VIM offset registers are read only. They are updated continuously by the VIM. When an interrupt is serviced, the offset vectors show the index for the next highest pending interrupt or 0x0 if no interrupt is pending.

8.5.2 IRQ Index Offset Vector Register (IRQIVEC)

The IRQ offset register provides the user with the numerical index value that represents the pending IRQ interrupt with the highest priority. Figure 8-11 and Table 8-2 describe this register.

Figure 8-11. IRQ Index Offset Vector Register (IRQIVEC) [offset = FE00h]



R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

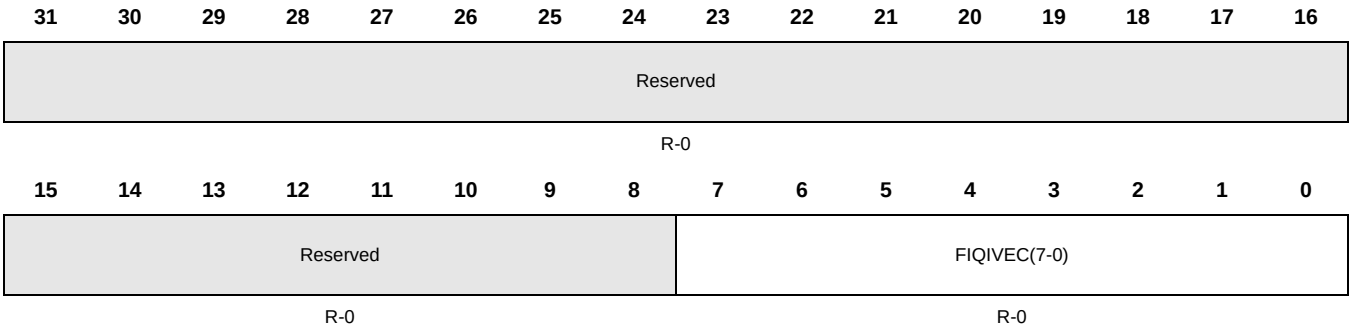
Table 8-2. IRQ Index Offset Vector Register (IRQIVEC) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads are undefined and writes have no effect.
7-0	IRQIVEC(7-0)	0–FFh	IRQ index vector. The least significant bits represent the index of the IRQ pending interrupt with the highest precedence, as shown in Table 8-1. When no interrupts are pending, the least significant byte of IRQIVEC is 0x00.

8.5.2.1 FIQ Index Offset Vector Registers (FIQIVEC)

The FIQIVEC register provides the user with a numerical index value that represents the pending FIQ interrupt with the highest priority. Figure 8-12 and Table 8-3 describe this register.

Figure 8-12. FIQ Index Offset Vector Register (FIQIVEC) [offset = FE04h]



R = Read in all modes; -n = value after reset

Table 8-3. FIQ Index Offset Vector Register (FIQIVEC) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads are undefined and writes have no effect.
7-0	FIQIVEC(7-0)	0–FFh	FIQ index offset vector. The least significant bits represent the index of the FIQ pending interrupt with the highest precedence, as shown in Table . When no interrupts are pending, the least significant byte of FIQIVEC is 0x00.

8.5.3 FIQ/IRQ Program Control Registers[0] (FIRQPR[0])

The FIQ/IRQ program control registers (FIRQPR[0]) determine whether a given interrupt request will be either FIQ or IRQ. Figure 8-13 and Table 8-4 describe these registers.

Figure 8-13. FIQ/IRQ Program Control Register 0 (FIRQPR0) [offset = FE10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIRQPR[31:16]															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FIRQPR[15:0]															
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-4. FIQ/IRQ Program Control Registers[0] (FIRQPR[0]) Field Descriptions

Bit	Name	Value	Description
31-0	FIRQPR[31:0]		FIQ/IRQ program control bits. These bits determine whether an interrupt request from a peripheral is of type FIQ or IRQ. Bit FIRQPRx[31:0] corresponds to request channel[31:0].
		0	Interrupt request is of IRQ type.
		1	Interrupt request is of FIQ type.

8.5.4 Pending Interrupt Read Location Registers[0] (INTREQ[0])

The pending interrupt register gives the pending interrupt requests. The register is updated every vbus clock cycle. Figure 8-14 and Table 8-5 describe this register.

Figure 8-14. Pending Interrupt Read Location Register 0 (INTREQ0) [offset = FE20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTREQ[31:16]															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTREQ[15:0]															
R-0															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-5. Pending Interrupt Read Registers[0] (INTREQ[0]) Field Descriptions

Bit	Name	Value	Description
31–0	INTREQ[31:0]		Pending interrupt bits. These bits determine whether an interrupt request is pending for the request channel between 0 and 31. The interrupt mask register does not affect the value of the interrupt pending bit. Bit INTREQx[31:0] corresponds to request channel[31:0].
		0	No interrupt event has occurred.
		1	An interrupt is pending.

8.5.5 Interrupt Mask Set Registers[0] (REQMSKSET[0])

The interrupt register mask selectively enables individual request channels. Figure 8-15 and Table 8-6 describe these registers.

Figure 8-15. Interrupt Mask Set Register 0 (REQMSKSET0) [offset = FE30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REQMASKSET[31:16]															
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REQMASKSET[15:0]															
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-6. Interrupt Mask Set Register[0] (REQMSKSET[0]) Field Descriptions

Bit	Name	Value	Description
31–0	REQMASKSET[31:0]		Request mask set bits. This vector determines whether the interrupt request channel is enabled. Bit REQMSKSETx[31:0] corresponds to request channel[31:0].
		0	*Read:* Interrupt request channel is disabled. *Write:* A write of 0 has no effect.
		1	*Read or Write:* The interrupt request channel is enabled.

8.5.6 Interrupt Mask Clear Registers[0] (REQMSKCLR[0])

The interrupt register mask selectively disables individual request channels. Figure 8-16 and Table 8-7 describe these registers.

Figure 8-16. Interrupt Mask Clear Register 0 (REQMSKCLR0) [offset = FE40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REQMASKCLR[31:16]															
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REQMASKCLR[15:0]															
R/WP-0															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-7. Request Mask Clear Register (REQMSKCLR) Field Descriptions

Bit	Name	Value	Description
31:0	REQMASKCLR[31:0]		Request mask clear bits. This vector determines whether the interrupt request channel is enabled. Bit REQMASKCLRx[31:0] corresponds to request channel[31:0].
		0	*Read:* Interrupt request channel is disabled. *Write:* A write of 0 has no effect.
		1	*Read:* The interrupt request channel is enabled. *Write:* The interrupt request channel is disabled.

8.5.7 Wake-Up Mask Set Registers[0] (WAKMSKSET[0])

The wake-up mask registers selectively enables individual wake-up interrupt request lines. Figure 8-17 and Table 8-8 describe these registers.

Figure 8-17. Wake-Up Mask Set Register 0 (WAKMSKSET0) [offset = FE50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WAKEMASKSET[31:16]															
R/WP-0xFFFF															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKEMASKSET[15:0]															
R/WP-0xFFFF															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-8. Wake-Up Mask Set Registers[0] (WAKMSKSET[0]) Field Descriptions

Bit	Name	Value	Description
31–0	WAKEMASKSET[31:0]		Wake-up mask set bits. This vector determines whether the wake-up interrupt line is enabled. Bit WAKEMASKSETx[31:0] corresponds to interrupt request channel[31:0].
		0	*Read:* Interrupt request channel is disabled. *Write:* A write of 0 has no effect.
		1	*Read or Write:* The interrupt request channel is enabled.

8.5.8 Wake-Up Mask Clear Registers[0] (WAKMSKCLR[0])

The wake-up mask register selectively disables individual wake-up interrupt request lines. Figure 8-18 and Table 8-9 describe these registers.

Figure 8-18. Wake-Up Mask Clear Register 0 (WAKMSKCLR0) [offset = FE60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WAKEMASKCLR[31:16]															
R/WP-0xFFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKEMASKCLR[15:0]															
R/WP-0xFFFF															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-9. Wake-Up Mask Clear Registers[0] (WAKMSKCLR[0]) Field Descriptions

Bit	Name	Value	Description
31–0	WAKEMASKCLR[31:0]		Wake-up mask clear bits. This vector determines whether the wake-up interrupt line is enabled. Bit WAKEMASKCLRx[31:0] corresponds to interrupt request channel[31:0].
		0	*Read:* Wake-up interrupt channel is disabled. *Write:* A write of 0 has no effect.
		1	*Read:* The wake-up interrupt channel is enabled. *Write:* The wake-up interrupt channel is disabled.

8.5.9 IRQ Interrupt Vector Register (IRQVECREG)

The interrupt vector register gives the address of the enabled and active IRQ interrupt. Figure 8-19 and Table 8-10 describe these registers.

Figure 8-19. IRQ Interrupt Vector Register (IRQVECREG) [offset = FE70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQVECREG(31-16)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQVECREG(15-0)															
R-0															

R = Read in all modes; WP = Write in privileged mode only; -n = value after reset

Table 8-10. IRQ Interrupt Vector Register (IRQVECREG) Field Descriptions

Bit	Name	Value	Description
31–0	IRQVECREG(31-0)	0–FFFF FFFFh	IRQ interrupt vector register. This vector gives the address of the ISR with the highest pending IRQ request. The CPU reads the address and branches to this location.

8.5.10 FIQ Interrupt Vector Register (FIQVECREG)

The interrupt vector register gives the address of the enabled and active FIQ interrupt. Figure 8-20 and Table 8-11 describe these registers.

Figure 8-20. IRQ Interrupt Vector Register (*FIQVECREG*) [offset = FE74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIQVECREG(31-16)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIQVECREG(15-0)															
R-0															

R = Read in all modes; WP = Write in privileged mode only; *-n* = value after reset

Table 8-11. FIQ Interrupt Vector Register (FIQVECREG) Field Descriptions

Bit	Name	Value	Description
31–0	FIQVECREG(31-0)	0–FFFF FFFFh	FIQ interrupt vector register. This vector gives the address of the ISR with the highest pending FIQ request. The CPU reads the address and branches to this location.

8.5.11 Capture Event Register (CAPEVT)

Figure 8-21 and Table 8-12 describe this register.

Figure 8-21. Capture Event Register (CAPEVT) [offset = FE78h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									CAPEVTSRC1(6-0)						
R-U									R/WP-0x0						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CAPEVTSRC0(6-0)						
R-U									R/WP-0x0						

R = Read, W = Write, WP = Write from privileged mode only, S = Set, U = Undefined, *-n* = Value after reset (see table)

Table 8-12. Capture Event Register (CAPEVT) Field Descriptions

Bit	Name	Value	Description
31–23	Reserved		Reads are indeterminate and writes have no effect.
22–16	CAPEVTSRC1[6:0]		Capture event source 1 mapping control. These bits determine which interrupt request maps to the capture event source 1 of the RTI:
		0000000	Interrupt request 0.
		0000001	Interrupt request 1.
		...	...
		0x1F	Interrupt request 31.
15–7	Reserved		Reads are indeterminate and writes have no effect.
6–0	CAPEVTSRC0[6:0]		Capture event source 0 mapping control. These bits determine which interrupt request maps to the capture event source 0 of the RTI:
		0000000	Interrupt request 0.
		0000001	Interrupt request 1.
		...	...
		0x1F	Interrupt request 31.

8.5.12 VIM Interrupt Control Register[0:31] (CHANCTRL[0:7])

Eight interrupt control registers control the 32 interrupt channels of the VIM. Each register controls four interrupt channels: each of them is indexed from 0 to 32. Table 8-13 shows the organization of all the registers and the reset value of each. Each four fields of the register has been named with a generic index that refers to the detailed register organization. Figure 8-22 and Table 8-14 describe these registers.

Table 8-13. Interrupt Control Registers Organization

Offset Address[1]	Mnemonic	CHANMAPx$_0$	CHANMAPx$_1$	CHANMAPx$_2$	CHANMAPx$_3$	Reset value
0x80	CHANCTRL0	CHANMAP0	CHANMAP1	CHANMAP2	CHANMAP3	0x00010203
0x84	CHANCTRL1	CHANMAP4	CHANMAP5	CHANMAP6	CHANMAP7	0x04050607
:	:	:	:	:	:	:
:	:	:	:	:	:	:
:	:	:	:	:	:	:
0x98	CHANCTRL6	CHANMAP24	CHANMAP25	CHANMAP26	CHANMAP27	0x18191A1B
0x9C	CHANCTRL7	CHANMAP28	CHANMAP29	CHANMAP30	CHANMAP31	0x1C1D1E1F

[1]The base address of the VIM is 0xFFFF_FE00h.

Figure 8-22. Interrupt Control Registers[0:31] (CHANCTRL[0:7]) [offset = 0x80 to 0x9Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	CHANMAPx$_0$[6:0]							Res	CHANMAPx$_1$[6:0]						
R-U	R/WP-see Table 8-13							R/WP-see Table 8-13							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	CHANMAPx$_2$[6:0]							Res	CHANMAPx$_3$[6:0]						
R-U	R/WP-see Table 8-13							R-U	R/WP-see Table 8-13						

R = Read, W = Write, WP = Write from privileged mode only, S = Set, U = Undefined, *-n* = Value after reset (see table)

Table 8-14. Interrupt Control Registers[0:31] (CHANCTL[0:7] Field Descriptions

Bit	Name	Value	Description
31	Reserved		Reads are indeterminate and writes have no effect.
30-24	CHANMAPx$_0$[6:0]		CHANMAPx$_0$(6–0). Interrupt CHANx$_0$ mapping control. These bits determine which interrupt request the priority channel CHANx$_0$ maps to:
		0000000	*Read:* Interrupt request 0 maps to channel priority CHANx$_0$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_0$ is set with the interrupt request.
		0000001	*Read:* Interrupt request 1 maps to channel priority CHANx$_0$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_0$ is set with the interrupt request.
		. . .	. . .
		0x1F	*Read:* Interrupt request 31 maps to channel priority CHANx$_0$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_0$ is set with the interrupt request.
23	Reserved		Reads are indeterminate and writes have no effect.

Table 8-14. Interrupt Control Registers[0:31] (CHANCTL[0:7] Field Descriptions (Continued)

Bit	Name	Value	Description
22–16	CHANMAPx$_1$[6:0]		CHANMAPx$_1$(6–0). Interrupt CHANx$_1$ mapping control. These bits determine which interrupt request the priority channel CHANx$_1$ maps to:
		0000000	*Read:* Interrupt request 0 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_1$ is set with the interrupt request.
		0000001	*Read:* Interrupt request 1 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_1$ is set with the interrupt request.
		. . .	. . .
		0x1F	*Read:* Interrupt request 31 maps to channel priority CHANx$_1$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_1$ is set with the interrupt request.
15	Reserved		Reads are indeterminate and writes have no effect.
14–8	CHANMAPx$_2$[6:0]		CHANMAPx$_2$(6–0). Interrupt CHANx$_2$ mapping control. These bits determine which interrupt request the priority channel CHANx$_2$ maps to:
		0000000	*Read:* Interrupt request 0 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_2$ is set with the interrupt request.
		0000001	*Read:* Interrupt request 1 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_2$ is set with the interrupt request.
		. . .	. . .
		0x1F	*Read:* Interrupt request 31 maps to channel priority CHANx$_2$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_2$ is set with the interrupt request.
7	Reserved		Reads are indeterminate and writes have no effect.

Table 8-14. Interrupt Control Registers[0:31] (CHANCTL[0:7] Field Descriptions (Continued)

Bit	Name	Value	Description
6–0	CHANMAPx$_3$[6:0]		CHANMAPx$_3$(6–0). Interrupt CHANx$_3$ mapping control. These bits determine which interrupt request the priority channel CHAN**x$_3$** maps to:
		0000000	*Read:* Interrupt request 0 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_3$ is set with the interrupt request.
		0000001	*Read:* Interrupt request 1 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_3$ is set with the interrupt request.
		. . .	. . .
		0x1F	*Read:* Interrupt request 31 maps to channel priority CHANx$_3$. *Write:* The default value of this bit after reset is given in Table 8-13. The channel priority CHANx$_3$ is set with the interrupt request.

Real-Time Interrupt (RTI) Module

This document describes the functionality of the real-time interrupt (RTI) module used on TMS470Px derivatives. It is specifically designed to support time-triggered operating systems and real-time operating systems.

Topic **Page**

9.1 Introduction and Feature Overview

This section describes the purpose, features, register mapping, and industry standard compliance.

9.1.1 Purpose

The real-time interrupt (RTI) module provides timer functionality for operating systems and for benchmarking code. The RTI module can incorporate several counters that define the timebases needed for scheduling in the operating system.

The timers also allow you to benchmark certain areas of code by reading the values of the counters at the beginning and the end of the desired code range and calculating the difference between the values.

9.1.2 Main Features

The RTI module has the following features:

- One counter block, counter block 0, for generating a timebase. The block consists of the following:
  - One 32-bit prescale counter, which can also be called the up counter
  - One 32-bit free running counter
  - Two capture registers for capturing the prescale and free running counter on a special event
- Four configurable compare registers for generating interrupts. Each register can work with counter block 0.
- Automatic update of compare register on compare match to generate periodic interrupts
- Fast enabling/disabling of interrupts
- Digital watchdog (DWD)
- RTI clock input selectable in the system module

9.1.3 Industry Standard Compliance Statement

This module is specifically designed to fulfill the requirements for OSEK (**O**ffene **S**ysteme und deren **S**chnittstellen für die **E**lektronik im **K**raftfahrzeug, or Open Systems and the Corresponding Interfaces for Automotive Electronics).

9.2 Module Operation

The following sections describe the operation of the RTI module.

9.2.1 Counter Operation

The RTI module has one counter block (counter block 0) for generating a timebase.

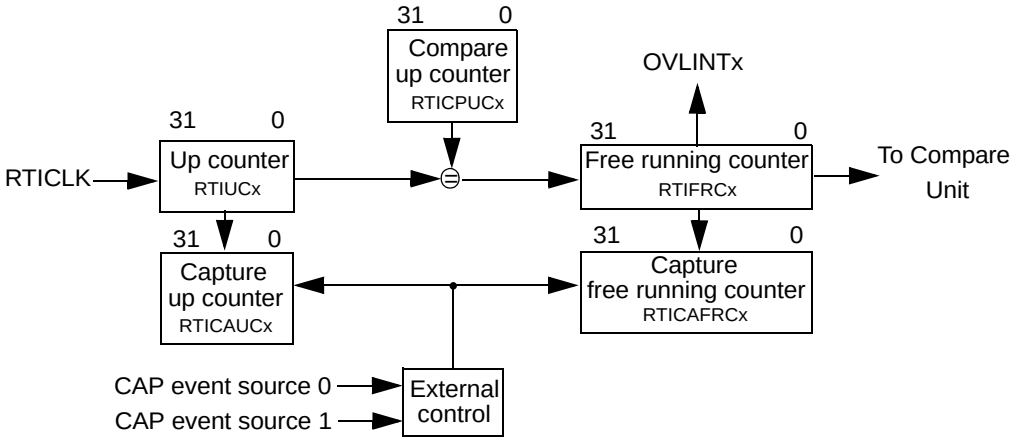
The block consists of the following:

- One 32-bit prescale counter (RTIUCx)
- One 32-bit free running counter (RTIFRCx)

The RTIUCx is driven by the RTICLK and counts up until the compare value in the compare up counter register (RTICPUCx) is reached. When the compare matches, RTIFRCx is incremented and RTIUCx is reset to 0. If RTIFRCx overflows, an interrupt is generated to the vectored interrupt manager (M3VIM). This process means that the value set in the compare up counter (RTICPUCx) prescales the RTI clock. The resulting formula for the frequency of the free running counter (RTIFRCx) is:

$f_{RTIFRCx} = f_{RTICLK}/(RTICPUCx+1)$

Figure 9-1. Counter Block Diagram



9.2.1.1 Counter Read Consistency

To ensure the consistency of the counters, when both counter values must be determined, RTIFRCx must be read first. This priority will ensure that at the CPU read cycle of RTIFRCx, the up counter value is stored in the counter register. The second read is done on the up counter register (RTIUCx), which then holds the value of the counter cycle of the corresponding read on the free running counter register (RTIFRCx).

9.2.1.2 Capture Feature

The block also provides a capture feature on external events. Two capture sources can trigger the capture event. Which source triggers the block is configurable. The sources originate from the M3VIM, to generate a capture event when one of the peripheral modules has generated an interrupt. The peripheral that can generate an event is configured in the vectored interrupt manager (M3VIM).

When an event is detected, RTIUCx and RTIFRCx are stored in the capture up counter (RTICAUCx) and capture free running counter (RTICAFRCx) registers. The read order of the captured values must be the same as the order of the actual counters. Therefore, RTICAFRCx must be read first and the RTICAUCx registers must be read after the RTICAFRCx value has been determined. While RTICAFRCx is read, the RTICAUCx value is loaded into a shadow register to ensure data consistency if another capture event occurs

during the two reads of the captured data. If the application fails to read the two registers before a second capture event happens, the previous data will be overwritten.

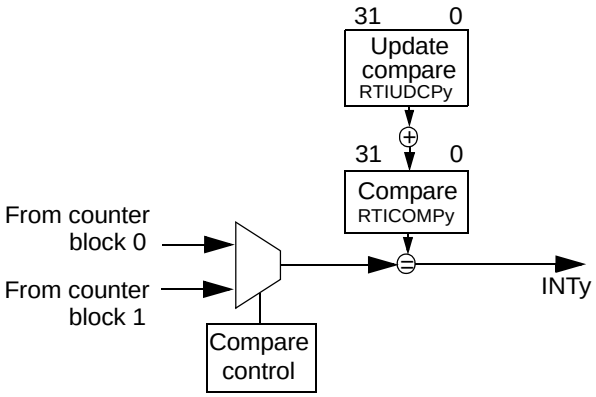
9.2.1.3 Interrupt Requests

To generate interrupt requests to the M3VIM, there are four compare registers (RTICOMPy). Each of the compare registers can be configured to work on either RTIFRC0 or RTIFRC1. When the counter value matches the compare value, an interrupt is generated. All compares provide an additional feature, with an automatic addition of the compare value with the value stored in the update compare (RTIUDCPy) register when the compare matches. This allows periodic interrupt requests to be generated without having to update the compare value by software. The period of the generated interrupt request is given by the formula:

$t_{COMPx} = t_{RTICLK}$ x (RTICPUCy+1) x RTIUDCPy

Another interrupt that can be generated is the overflow interrupt (OVLINTx) in case the RTIFRCx counter overflows.

Figure 9-2. Compare Block Diagram



The interrupts can be enabled in the RTISETINT register and disabled in the RTICLEARINT register. The RTIFLAG register shows the pending interrupts.

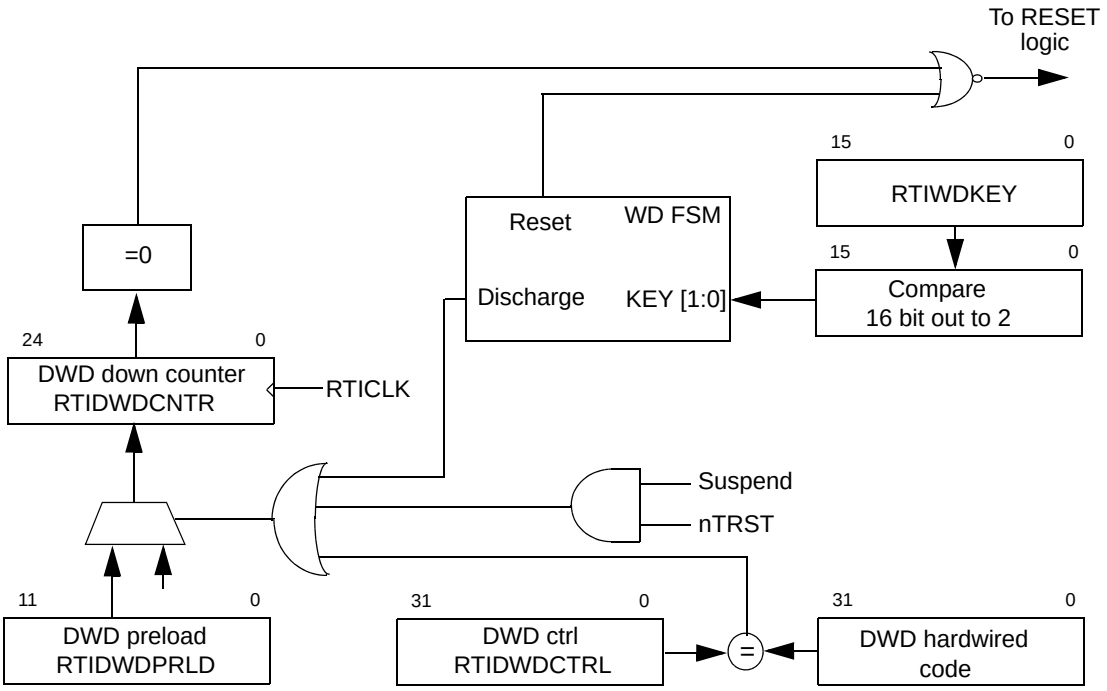
9.2.2 Clock Domain

For RTICLK definition, please see Section 1.5.2, "Clock Domains" in the system module.

9.2.3 Digital Watchdog (DWD)

The digital watchdog (DWD) generates resets to prevent runaway code. Figure 9-3 illustrates the DWD.

Figure 9-3. Digital Watchdog



Note:
It has to be taken into account that the write to the RTIWDKEY register takes 3 VCLK cycles.

9.2.3.1 DWD

Some applications might need to use a DWD instead of an AWD. The DWD also generates resets after a programmable period, or if no correct key sequence was written to the RTIWDKEY register.

The DWD is disabled by default. If it should be used, it must be enabled by writing a 32-bit value, which is the inverted value of the hardwired code in the module, to the RTIDWDCTRL register.

Note:
Once the DWD is enabled, it cannot be disabled.

If the correct key sequence is written to the RTIWDKEY register (0xE51A followed by 0xA35C), the 25-bit DWD down counter is reloaded with the left justified 12-bit preload value stored in RTIDWDPRLD. If the incorrect values are written, a watchdog reset will occur immediately. A reset will also be generated when the DWD down counter is decremented to 0.

While the device is in suspend mode (debug mode), the DWD down counter keeps the value it had when entering suspend mode.

The DWD down counter will be decremented with the RTICLK frequency.

The expiration time of the DWD down counter can be determined with the following equation:

$texp = (RTIDWDPRLD + 1) \times 2^{13}/RTICLK$

where

RTIDWDPRLD = 0...4095

Note:
It has to be taken into account that the write to the RTIWDKEY register takes 3 VCLK cycles.

9.2.4 Low Power Modes

Low power modes allow the trade off of the current used during low power versus functionality and fast wakeup response. All TMx70 low power modes have the following characteristics:

- CPU and system clocks are disabled.
- Flash banks and pump are in sleep mode.
- All peripheral modules are in low power modes and the clocks are disabled (exceptions to this may occur and would be documented in the specific device data sheet).

The TMS470Px is designed with flexibility in enabling and disabling clocks, so many different low-power modes are possible.

9.3 Control Registers

Figure 9-4 provides a summary of the registers. The registers support 8-bit, 16-bit, and 32-bit writes. The offset is relative to the associated peripheral select. See the following sections for detailed descriptions of the registers. The base address for the control registers is 0xFFFF FC00. The address locations not listed, are reserved..

Figure 9-4. RTI Registers

Offset Address / Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 RTIGCTRL Page 287	Reserved															
	COS	Reserved														CNT0 EN
0x08 RTICAPCTRL Page 288	Reserved															
	Reserved															CAP CNTR0
0x0C RTICOMPCTRL Page 289	Reserved															
	Reserved			COMP SEL3	Reserved			COMPS EL2	Reserved			COMPS EL1	Reserved			COMPS EL0
0x10 RTIFRC0 Page 289	FRC0(31–16)															
	FRC0(15–0)															
0x14 RTIUC0 Page 290	UC0(31–16)															
	UC0(15–0)															
0x18 RTICPUC0 Page 291	CPUC0(31–16)															
	CPUC0(15–0)															
0x20 RTICAFRC0 Page 292	CAFRC0(31–16)															
	CAFRC0(15–0)															

Figure 9-4. RTI Registers (Continued)

Offset Address / Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x24 RTICAUC0 Page 293	CAUC0(31–16)															
	CAUC0(15–0)															
0x50 RTICOMP0 Page 294	COMP0(31–16)															
	COMP0(15–0)															
0x54 RTIUDCP0 Page 295	UDCP0(31–16)															
	UDCP0(15–0)															
0x58 RTICOMP1 Page 296	COMP1(31–16)															
	COMP1(15–0)															
0x5C RTIUDCP1 Page 297	UDCP1(31–16)															
	UDCP1(15–0)															
0x60 RTICOMP2 Page 298	COMP2(31–16)															
	COMP2(15–0)															
0x64 RTIUDCP2 Page 299	UDCP2(31–16)															
	UDCP2(15–0)															
0x68 RTICOMP3 Page 300	COMP3(31–16)															
	COMP3(15–0)															

Figure 9-4. RTI Registers (Continued)

Offset Address / Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x6C RTIUDCP3 Page 301	UDCP3(31–16)															
	UDCP3(15–0)															
0x80 RTISETINT Page 302	Reserved														SET OVL0 INT	Reserved
	Reserved												SET INT3	SET INT2	SET INT1	SET INT0
0x84 RTICLEARINT Page 304	Reserved														CLEAR OVL0 INT	Reserved
	Reserved												CLEAR INT3	CLEAR INT2	CLEAR INT1	CLEAR INT0
0x88 RTIINTFLAG Page 306	Reserved														OVL0 INT	Reserved
	Reserved												INT3	INT2	INT1	INT0
0x90 RTIDWDCTRL Page 308	DWDCTRL(31–16)															
	DWDCTRL(15–0)															
0x94 RTIDWDPRLD Page 309	Reserved															
	Reserved															
	DWDPRLD(11–0)															
0x98 RTIWDSTATUS Page 310	Reserved															
	Reserved													KEYST	DWDST	Reserved
0x9C RTIWDKEY Page 311	Reserved															
	WDKEY(15–0)															

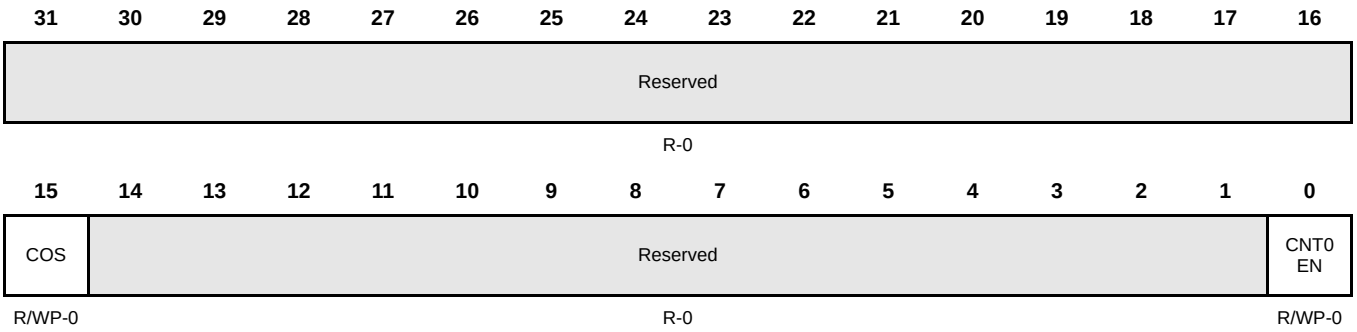
Figure 9-4. RTI Registers (Continued)

Offset Address / Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xA0 RTIDWDCNTR Page 313	Reserved							DWDCNTR(24–16)								
	DWDCNTR(15–0)															

9.3.1 RTI Global Control Register (RTIGCTRL)

The global control register starts/stops the counters. This register is shown in Figure 9-5 and described in Table 9-1.

Figure 9-5. RTI Global Control Register (RTIGCTRL) [offset = 0x00h]



R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-1. RTI Global Control Register (RTIGCTRL) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return 0 and writes have no effect.
15	COS		Continue on suspend. This bit determines if both counters are stopped when the device goes into suspend mode or if they continue counting.
		0	Counters are stopped while in suspend mode.
		1	Counters are running while in suspend mode.
14–1	Reserved		Reads return 0 and writes have no effect.
0	CNT0EN		Counter 0 enable. This bit starts and stops counter block 0 (RTIUC0 and RTIFRC0).
		0	Counter block 0 is stopped.
		1	Counter block 0 is running.

Note:
If the application uses the timebase circuit for synchronization between the communications controller and the operating system and the device enters debug mode, the synchronization may be lost depending on the COS setting in the RTI module and the debug mode behavior of the communications controller.

9.3.2 RTI Capture Control Register (RTICAPCTRL)

The capture control register controls the capture source for the counters. This register is shown in Figure 9-6 and described in Table 9-2.

Figure 9-6. RTI Capture Control Register (RTICAPCTRL) [offset = 0x08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CAP CNTR0
R-0															R/WP-0

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-2. RTI Capture Control Register (RTICAPCTRL) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return 0 and writes have no effect.
0	CAPCNTR0		Capture counter 0. This bit determines which external interrupt source triggers a capture event of RTIUC0 and RTIFRC0.
		0	Capture of RTIUC0/RTIFRC0 is triggered by capture event source 0.
		1	Capture of RTIUC0/RTIFRC0 is triggered by capture event source 1.

9.3.3 RTI Free Running Counter 0 Register (RTIFRC0)

The free running counter 0 register holds the current value of free running counter 0. This register is shown in Figure 9-7 and described in Table 9-3.

Figure 9-7. RTI Free Running Counter 0 Register (RTIFRC0) [offset = 0x10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRC0(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FRC0(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-3. RTI Free Running Counter 0 Register (RTIFRC0) Field Descriptions

Bit	Name	Value	Description
31–0	FRC0(31–0)	0–FFFF FFFFh	Free running counter 0. This registers holds the current value of the free running counter 0 (RTIFRC0) and will be updated continuously. A read of this counter returns the current value of the counter. The counter can be preset by writing (in privileged mode only) to this register. The counter increments then from this written value upwards. **Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC0 and RTIFRC0.**

9.3.4 RTI Up Counter 0 Register (RTIUC0)

The up counter 0 register holds the current value of prescale counter. This register is shown in Figure 9-8 and described in Table 9-4.

Figure 9-8. RTI Up Counter 0 Register (RTIUC0) [offset = 0x14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UC0(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UC0(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-4. RTI Up Counter 0 Register (RTIUC0) Field Descriptions

Bit	Name	Value	Description
31–0	UC0(31–0)	0–FFFF FFFFh	Up counter 0. This register holds the current value of the up counter 0 (RTIUC0) and prescales the RTI clock. It will be only updated by a previous read of free running counter 0 (RTIFRC0). This method of updating effectively gives a 64-bit read of both counters, without having the problem of a counter being updated between two consecutive reads on up counter 0 (RTIUC0) and free running counter 0 (RTIFRC0). A read of this counter returns the value of the counter at the time RTIFRC0 was read. A write to this counter presets it with a value. The counter then increments from this written value upwards. **Note: If counters must be preset, they must be disabled in the RTIGCTRL register to ensure consistency between RTIUC0 and RTIFRC0.** **Note: If the preset value is bigger than the compare value stored in register RTICPUC0, then it can take a long time until a compare matches, since RTIUC0 has to count up until it overflows.**

9.3.5 RTI Compare Up Counter 0 Register (RTICPUC0)

The compare up counter 0 register holds the value to be compared with prescale counter 0 (RTIUC0). This register is shown in Figure 9-9 and described in Table 9-5.

Figure 9-9. RTI Compare Up Counter 0 Register (RTICPUC0) [offset = 0x18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CPUC0(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CPUC0(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-5. RTI Compare Up Counter 0 Register (RTICPUC0) Field Descriptions

Bit	Name	Value	Description

31–0	CPUC0(31–0)	0–FFFF FFFFh	Compare up counter 0. This register holds the value that is compared with the up counter 0 (RTIUC0). When the compare shows a match, the free running counter 0 (RTIFRC0) is incremented. RTIUC0 is set to 0 when the counter value matches the RTICPUC0 value. The value set in this register prescales the RTI clock.
			If CPUC0 = 0, then
			f_{FRC0} = RTICLK/2^{32}
			If CPUC0 ≠ 0, then
			f_{FRC0} = RTICLK/(RTICPUC0+1)
			
			A read of this register returns the current compare value.
			
			A write to this register:
			If TBEXT = 0, the compare value is updated.
			If TBEXT = 1, the compare value is unchanged.

9.3.6 RTI Capture Free Running Counter 0 Register (RTICAFRC0)

The capture free running counter 0 register holds the free running counter 0 on external events. This register is shown in Figure 9-10 and described in Table 9-6.

Figure 9-10. RTI Capture Free Running Counter 0 Register (RTICAFRC0) [offset = 0x20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAFRC0(31–16)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAFRC0(15–0)															
R-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-6. RTI Capture Free Running Counter 0 Register (RTICAFRC0) Field Descriptions

Bit	Name	Value	Description
31–0	CAFRC0(31–0)	0–FFFF FFFFh	Capture free running counter 0. This register captures the current value of the free running counter 0 (RTIFRC0) when a event occurs, controlled by the external capture control block. A read of this register returns the value of RTIFRC0 on a capture event.

9.3.7 RTI Capture Up Counter 0 Register (RTICAUC0)

The capture up counter 0 register holds the current value of prescale counter 0 on external events. This register is shown in Figure 9-11 and described in Table 9-7.

Figure 9-11. RTI Capture Up Counter 0 Register (RTICAUC0) [offset = 0x24]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAUC0(31–16)															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAUC0(15–0)															
R-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-7. RTI Capture Up Counter 0 Register (RTICAUC0) Field Descriptions

Bit	Name	Value	Description
31–0	CAUC0(31–0)	0–FFFF FFFFh	Capture up counter 0. This register captures the current value of the up counter 0 (RTIUC0) when an event occurs, controlled by the external capture control block. **Note: The read sequence must be the same as with RTIUC0 and RTIFRC0. Therefore, the RTICAFRC0 register must be read before the RTICAUC0 register is read. This sequence ensures that the value of the RTICAUC0 register is the corresponding value to the RTICAFRC0 register, even if another capture event happens in between the two reads.** A read of this register returns the value of RTIUC0 on a capture event.

9.3.8 RTI Compare 0 Register (RTICOMP0)

The compare 0 register holds the value to be compared with the counters. This register is shown in Figure 9-12 and described in Table 9-8.

Figure 9-12. RTI Compare 0 Register (RTICOMP0) [offset = 0x50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP0(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP0(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-8. RTI Compare 0 Register (RTICOMP0) Field Descriptions

Bit	Name	Value	Description
31–0	COMP0(31–0)	0–FFFF FFFFh	Compare 0. This registers holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches the compare value, an interrupt is flagged. With this register it is also possible to initiate a DMA request, when the corresponding bit in RTISETINT is set. A read of this register will return the current compare value. A write to this register (in privileged mode only) will update the compare register with a new compare value.

9.3.9 RTI Update Compare 0 Register (RTIUDCP0)

The update compare 0 register holds the value to be added to the compare register 0 value on a compare match. This register is shown in Figure 9-13 and described in Table 9-9.

Figure 9-13. RTI Update Compare 0 Register (RTIUDCP0) [offset = 0x54h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UDCP0(31–16)															
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UDCP0(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-9. RTI Update Compare 0 Register (RTIUDCP0) Field Descriptions

Bit	Name	Value	Description
31–0	UDCP0(31–0)	0–FFFF FFFFh	Update compare 0. This register holds a value that is added to the value in the compare 0 (RTICOMP0) register each time a compare matches. This function allows periodic interrupts to be generated without software intervention. A read of this register will return the value to be added to the compare 0 register on the next compare match. A write to this register will provide a new update value.

9.3.10 RTI Compare 1 Register (RTICOMP1)

The compare 1 register holds the value to be compared to the counters. This register is shown in Figure 9-14 and described in Table 9-10.

Figure 9-14. RTI Compare 1 Register (RTICOMP1) [offset = 0x58]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP1(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP1(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-10. RTI Compare 1 Register (RTICOMP1) Field Descriptions

Bit	Name	Value	Description
31–0	COMP1(31–0)	0–FFFF FFFFh	Compare 1. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request, when the corresponding bit in RTISETINT is set. A read of this register will return the current compare value. A write to this register will update the compare register with a new compare value.

9.3.11 RTI Update Compare 1 Register (RTIUDCP1)

The update compare 1 register holds the value to be added to the compare register 1 value on a compare match. This register is shown in Figure 9-15 and described in Table 9-11.

Figure 9-15. RTI Update Compare 1 Register (RTIUDCP1) [offset = 0x5Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UDCP1(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDCP1(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-11. RTI Update Compare 1 Register (RTIUDCP1) Field Descriptions

Bit	Name	Value	Description
31–0	UDCP1(31–0)	0–FFFF FFFFh	Update compare 1. This register holds a value that is added to the value in the RTICOMP1 register each time a compare matches. This process allows periodic interrupts to be generated without software intervention. A read of this register will return the value to be added to the RTICOMP1 register on the next compare match. A write to this register will provide a new update value.

9.3.12 RTI Compare 2 Register (RTICOMP2)

The compare 2 register holds the value to be compared to the counters. This register is shown in Figure 9-16 and described in Table 9-12.

Figure 9-16. RTI Compare 2 Register (RTICOMP2) [offset = 0x60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP2(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP2(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-12. RTI Compare 2 Register (RTICOMP2) Field Descriptions

Bit	Name	Value	Description
31–0	COMP2(31–0)	0–FFFF FFFFh	Compare 2. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request, when the corresponding bit in RTISETINT is set. A read of this register will return the current compare value. A write to this register (in privileged mode only) will provide a new compare value.

9.3.13 RTI Update Compare 2 Register (RTIUDCP2)

The update compare 2 register holds the value to be added to the compare register 2 value on a compare match. This register is shown in Figure 9-17 and described in Table 9-13.

Figure 9-17. RTI Update Compare 2 Register (RTIUDCP2) [offset = 0x64h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UDCP2(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDCP2(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-13. RTI Update Compare 2 Register (RTIUDCP2) Field Descriptions

Bit	Name	Value	Description
31–0	UDCP2(31–0)	0–FFFF FFFFh	Update compare 2. This register holds a value that is added to the value in the RTICOMP2 register each time a compare matches. This process makes it possible to generate periodic interrupts without software intervention. A read of this register will return the value to be added to the RTICOMP2 register on the next compare match. A write to this register will provide a new update value.

9.3.14 RTI Compare 3 Register (RTICOMP3)

The compare 3 register holds the value to be compared to the counters. This register is shown in Figure 9-18 and described in Table 9-14.

Figure 9-18. RTI Compare 3 Register (RTICOMP3) [offset = 0x68h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP3(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP3(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-14. RTI Compare 3 Register (RTICOMP3) Field Descriptions

Bit	Name	Value	Description
31–0	COMP3(31–0)	0–FFFF FFFFh	Compare 3. This register holds a value that is compared with the counter selected in the compare control logic. If RTIFRC0 or RTIFRC1, depending on the counter selected, matches this compare value, an interrupt is flagged. With this register, it is possible to initiate a DMA request, when the corresponding bit in RTISETINT is set. A read of this register will return the current compare value. A write to this register will provide a new compare value.

9.3.15 RTI Update Compare 3 Register (RTIUDCP3)

The update compare 3 register holds the value to be added to the compare register 3 value on a compare match. This register is shown in Figure 9-19 and described in Table 9-15.

Figure 9-19. RTI Update Compare 3 Register (RTIUDCP3) [offset = 0x6Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UDCP3(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDCP3(15–0)															
R/WP-0															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-15. RTI Update Compare 3 Register (RTIUDCP3) Field Descriptions

Bit	Name	Value	Description
31–0	UDCP3(31–0)	0–FFFF FFFFh	Update compare 3. This register holds a value that is added to the value in the RTICOMP3 register each time a compare matches. This process makes it possible to generate periodic interrupts without software intervention. A read of this register will return the value to be added to the RTICOMP3 register on the next compare match. A write to this register will provide a new update value.

9.3.16 RTI Set/Status Interrupt Register (RTISETINT)

This register prevents the necessity of a read-modify-write operation if a particular interrupt should be enabled. This register is shown in Figure 9-20 and described in Table 9-16.

Figure 9-20. RTI Set/Status Interrupt Control Register (RTISETINT) [offset = 0x80h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													Reserved	SET OVL0 INT	Reserved
R-0													R/W-0	R/WP-0	R-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SET INT3	SET INT2	SET INT1	SET INT0
R-0												R/WP-0	R/WP-0	R/WP-0	R/WP-0

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-16. RTI Set/Status Interrupt Control Register (RTISETINT) Field Descriptions

Bit	Name	Value	Description
31–18	Reserved		Reads return 0 and writes have no effect.
17	SETOVL0INT		Set free running counter 0 overflow interrupt.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read or write:* The interrupt is enabled.
16–4	Reserved		Reads return 0 and writes have no effect.
7-4	Reserved		Reads return 0 and writes have no effect.
3	SETINT3		Set compare interrupt 3.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read or write:* The interrupt is enabled.
2	SETINT2		Set compare interrupt 2.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read or write:* The interrupt is enabled.
1	SETINT1		Set compare interrupt 1.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read or write:* The interrupt is enabled.

Table 9-16. RTI Set/Status Interrupt Control Register (RTISETINT) Field Descriptions (Continued)

Bit	Name	Value	Description
0	SETINT0		Set compare interrupt 0.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read or write:* The interrupt is enabled.

9.3.17 RTI Clear/Status Interrupt Register (RTICLEARINT)

This register prevents the necessity of a read-modify-write operation if a particular interrupt should be disabled. This register is shown in Figure 9-21 and described in Table 9-17.

Figure 9-21. RTI Clear/Status Interrupt Control Register (RTICLEARINT) [offset = 0x84h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CLEAR OVL0 INT	Reserved
R-0														R/WP-0	R-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CLEAR INT3	CLEAR INT2	CLEAR INT1	CLEAR INT0
R-0												R/WP-0	R/WP-0	R/WP-0	R/WP-0

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-17. RTI Clear/Status Interrupt Control Register (RTICLEARINT) Field Descriptions

Bit	Name	Value	Description
31–18	Reserved		Reads return 0 and writes have no effect.
17	CLEAROVL0 INT		Clear free running counter 0 overflow interrupt.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
16–4	Reserved		Reads return 0 and writes have no effect.
7-4	Reserved		Reads return 0 and writes have no effect.
3	CLEARINT3		Set compare interrupt 3.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
2	CLEARINT2		Set compare interrupt 2.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

Table 9-17. RTI Clear/Status Interrupt Control Register (RTICLEARINT) Field Descriptions (Continued)

Bit	Name	Value	Description
1	CLEARINT1		Set compare interrupt 1.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
0	CLEARINT0		Set compare interrupt 0.
		0	*Read:* The interrupt is disabled. *Write:* The corresponding bit is unchanged.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

9.3.18 RTI Interrupt Flag Register (RTIINTFLAG)

The corresponding flags are set at every compare match of the RTIFRCx and RTICOMPx values, whether the interrupt is enabled or not. This register is shown in Figure 9-22 and described in Table 9-18.

Figure 9-22. RTI Interrupt Flag Register (RTIINTFLAG) [offset = 0x88h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														OVL0 INT	Reserved
R-0														R/WP-0	R-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												INT3	INT2	INT1	INT0
R-0												R/CP-0	R/CP-0	R/CP-0	R/CP-0

R = Read, CP = Clear in privileged mode only, U = Undefined; -*n* = Value after reset

Table 9-18. RTI Interrupt Flag Register (RTIINTFLAG) Field Descriptions

Bit	Name	Value	Description
31–18	Reserved		Reads return 0 and writes have no effect.
17	OVL0INT		Free running counter 0 overflow interrupt flag. This bit determines if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* The bit is unchanged.
		1	*Read:* An interrupt is pending. *Write:* The bit is set to 0.
16–4	Reserved		Reads return 0 and writes have no effect.
3	INT3		Interrupt flag 3. These bits determine if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* The bit is unchanged.
		1	*Read:* An interrupt is pending. *Write:* The bit is set to 0.
2	INT2		Interrupt flag 2. These bits determine if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* The bit is unchanged.
		1	*Read:* An interrupt is pending. *Write:* The bit is set to 0.

Table 9-18. RTI Interrupt Flag Register (RTIINTFLAG) Field Descriptions (Continued)

Bit	Name	Value	Description
1	INT1		Interrupt flag 1. These bits determine if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* The bit is unchanged.
		1	*Read:* An interrupt is pending. *Write:* The bit is set to 0.
0	INT0		Interrupt flag 0. These bits determine if an interrupt is pending.
		0	*Read:* No interrupt is pending. *Write:* The bit is unchanged.
		1	*Read:* An interrupt is pending. *Write:* The bit is set to 0.

9.3.19 RTI Digital Watchdog Control Register (RTIDWDCTRL)

This register enables/disables the DWD. This register is shown in Figure 9-23 and described in Table 9-19.

Figure 9-23. RTI Digital Watchdog Control Register (RTIDWDCTRL) [offset = 0x90h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DWDCTRL(31–16)															
R/WP-0x5312h															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DWDCTRL(15–0)															
R/WP-0xACEDh															

R = Read, WP = Write in privileged mode only, -*n* = Value after reset

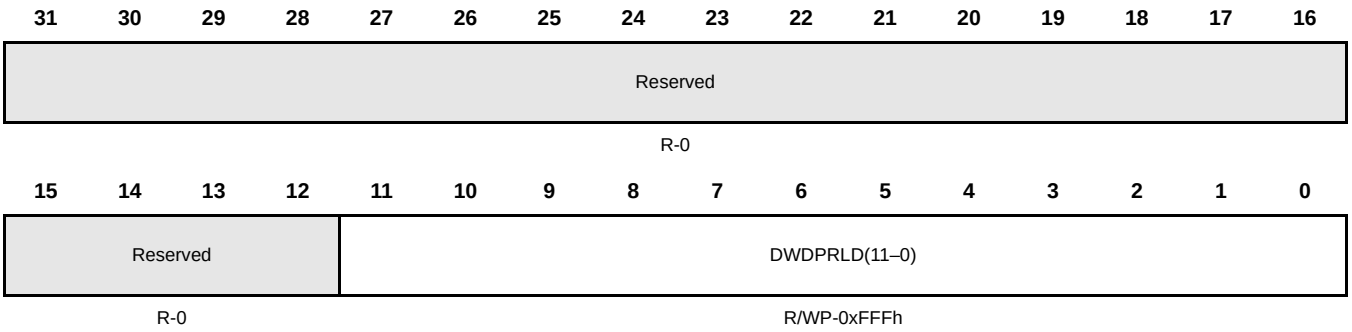
Table 9-19. RTI Digital Watchdog Control Register (RTIDWDCTRL) Field Descriptions

Bit	Name	Value	Description
31–0	DWDCTRL(31–0)		DWD control.
		0x5312ACED	*Read:* The DWD counter is disabled.
		Any other value	*Read:* The DWD counter is enabled.
			Write: By default, the DWD counter is disabled. Any write other than 0x5312ACED to the RTIDWDCTRL register enables the counter. This initial write can occur at any time during code execution. After the initial write has occurred, all other writes are ignored. TI recommends that the value 0xACED5312 be written to activate the counter.
			Note: The RTIDWDCTRL register is used to enable/disable the DWD functionality. The write once register allows this type of functionality. Writing the default value will disable the DWD. Writing an enable value will start the DWD. The write will only be enabled after a reset again.

9.3.20 RTI Digital Watchdog Preload Register (RTIDWDPRLD)

This register sets the expiration time of the DWD. This register is shown in Figure 9-24 and described in Table 9-20.

Figure 9-24. RTI Watchdog Preload Register (RTIDWDPRLD) [offset = 0x94h]



R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-20. RTI Watchdog Preload Register (RTIDWDPRLD) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return 0 and writes have no effect.
11–0	DWDPRLD(11–0)	0–FFFh	DWD preload value. *Read:* The current preload value is returned. *Write:* Set the preload value. The preload value must be written before enabling the DWD down counter. The expiration time of the counter can be calculated with the formula: t_{exp} = (RTIDWDPRLD + 1) x 2^{13}/RTICLK where RTIDWDPRLD = 0...4095

9.3.21 Watchdog Status Register (RTIWDSTATUS)

This register records the status of the DWD. The values of the following status bits will not be affected by a reset. Only the user can clear these bits. These bits are only intended for debug purposes. This register is shown in Figure 9-25 and described in Table 9-21.

Figure 9-25. RTI Watchdog Status Register (RTIDWDSTATUS) [offset = 0x98h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													KEYST	DWDST	Reserved
R-0													R/CP-x	R/CP-x	R-0

R = Read, CP = Clear in privileged mode only, -*n* = Value after reset, -*x* = indeterminate

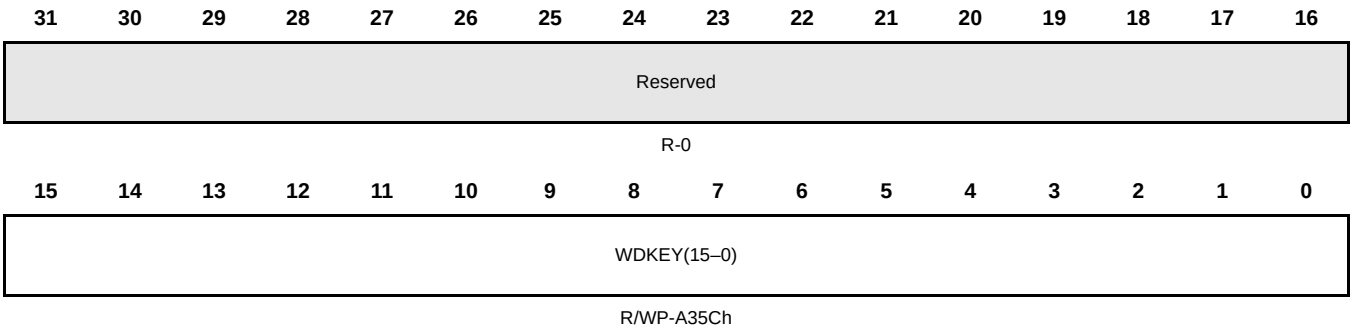
Table 9-21. RTI Watchdog Status Register (RTIDWDSTATUS) Field Descriptions

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	KEYST		Watchdog key status. This bit indicates a reset generated by a wrong key or key sequence written to the RTIWDKEY register.
		0	*Read:* No reset was generated. *Write:* The bit is unchanged.
		1	*Read:* A reset was generated. *Write:* The bit is set to 0.
1	DWDST		DWD status.
		0	*Read:* No reset was generated. *Write:* The bit is unchanged.
		1	*Read:* A reset was generated. *Write:* The bit is set to 0.
0	Reserved		Reads return 0 and writes have no effect.

9.3.22 RTI Watchdog Key Register (RTIWDKEY)

This register must be written with the correct written key values to discharge the external capacitor. This register is shown in Figure 9-26 and described in Table 9-22.

Figure 9-26. RTI Watchdog Key Register (RTIDWDKEY) [offset = 0x9Ch]



R = Read, WP = Write in privileged mode only, -*n* = Value after reset

Table 9-22. RTI Watchdog Key Register (RTIDWDKEY) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return 0 and writes have no effect.
15–0	WDKEY(15–0)	0–FFFFh	Watchdog key. These bits provide the key sequence location. Reads are indeterminate. A write of 0xE51A followed by 0xA35C in two separate write operations defines the key sequence and reloads the DWD. Writing any other value causes a reset, as shown in Table 9-23.

Note:
It has to be taken into account that the write to the RTIWDKEY register takes 3 VCLK cycles.

Table 9-23. Example of a WDKEY Sequence

Step	Value Written to WDKEY	Result
1	0x0A35C	No action
2	0x0A35C	No action
3	0x0E51A	WDKEY is enabled for reset by next 0x0A35C.
4	0x0E51A	WDKEY is enabled for reset by next 0x0A35C.
5	0x0E51A	WDKEY is enabled for reset by next 0x0A35C.
6	0x0A35C	Watchdog is reset.
7	0x0A35C	No action
8	0x0E51A	WDKEY is enabled for reset by next 0x0A35C.
9	0x0A35C	Watchdog is reset.

Step	Value Written to WDKEY	Result
10	0x0E51A	WDKEY is enabled for reset by next 0x0A35C.
11	0x02345	System reset; incorrect value written to WDKEY.

9.3.23 RTI Digital Watchdog Down Counter (RTIDWDCNTR)

This register provides the current value of the DWD down counter. This register is shown in Figure 9-27 and described in Table 9-24.

Figure 9-27. RTI Watchdog Down Counter Register (RTIDWDCNTR) [offset = 0xA0h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							DWDCNTR(24–16)								
R-0							R-1FFh								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DWDCNTR(15–0)															
R-FFFFh															

R = Read, *-n* = Value after reset

Table 9-24. RTI Watchdog Down Counter Register (RTIDWDCNTR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return 0 and writes have no effect.
24–0	DWDCNTR(24–0)	0–1FF FFFFh	DWD down counter. Reads return the current counter value.

Cyclic Redundancy Check Controller (CRC) Module

Topic Page

10.1 Overview

The CRC controller performs cyclic redundancy checks (CRC) to verify the integrity of the memory system. A signature representing the contents of the memory is obtained when the contents of the memory are read into the CRC controller. The CRC controller calculates the signature for a set of data and then compares the calculated signature value against a predetermined good signature value. The TMS470Px CRC controller provides one channel to perform CRC calculations on multiple memories in parallel, and it can be used on any memory system. Channel 1 can also be put into data trace mode. In data trace mode, the CRC controller compresses all data being read through the CPU read data bus.

10.2 TMS470Px CRC Features

The TMS470Px CRC controller offers:

- One channel to perform background signature verification on any memory sub-system
- Data compression on 8, 16, 32, and 64-bit data size
- Maximum-length parallel signature analysis (PSA) register constructed based on 64-bit primitive polynomials
- Programmable 20-bit pattern counter per channel to count the number of data patterns for compression
- Interface supported: AHB ARM7
- Data trace capability on the CPU data bus

10.3 Module Operation

The following sections describe how the module operates.

10.3.1 General Operation

A CRC controller has one channel. This channel has a memory-mapped PSA signature register.

A memory can be organized into multiple sectors with each sector consisting of multiple data patterns. A data pattern can be of 8, 16, 32, or 64-bit data. The CRC controller performs the signature calculation and compares the signature to a predetermined value. The PSA signature register compresses an incoming data pattern into a signature when it is written.

When one sector of data patterns is written into the PSA signature register, a final signature corresponding to that sector is obtained. The calculated signature and the predetermined signature are then compared to each other for signature verification.

The CRC controller also provides data trace capability. Channel 1 can perform data trace on the CPU data bus. During data trace, channel 1 monitors any data being read on the CPU data bus and compresses it. When data trace is enabled for channel 1, all circuits related to DMA request and interrupt generation and counters are disabled.

10.3.2 Modes

The CRC controller can operate in data trace mode.

10.3.2.1 Data trace mode

CRC channel 1 can be used to snoop the CPU read data bus. Data read on the CPU bus is converted into a 64-bit data value where the LSB or the MSB bits of the read data bus are padded with zeros. The PSA signature register concatenates the data value with zeros to create a 64-bit word based on the address of the read transaction. For example, the data value 0x12345678 is stored at addresses 0x0 and 0x4. If the CPU reads a word from address 0x0, the value 0x0000000012345678 will be used for the CRC signature generation. If the CPU reads a word from address 0x4, the value 0x1234567800000000 will be used for the signature generation. Each data pattern read by the CPU on its data bus is compressed in the PSA signature register. To enable the data trace on the read bus, 0x0 should be written in CH1_MODE bits. The CH1TRACEEN bit should be set to 0x1. Before the data compression begins, a seed value should be placed in the PSA signature register. To change the seed value, the CH1_MODE bits should be set to 0x00 and data trace bit should be disabled. During data trace mode, all interrupts and DMA request logic are inactive.

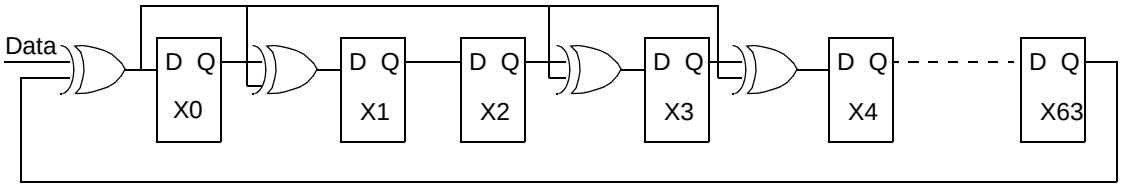
10.3.3 Register Definitions

10.3.3.1 PSA Signature Register

The 64-bit PSA signature register is based on the primitive polynomial shown in Equation 1 to produce the maximum length linear feedback shift register (LFSR) (more details of the 64-bit primitive polynomial can be found in Stahnke 1973).

$$f(x) = x^{64} + x^4 + x^3 + x + 1 \quad \text{(EQ 1)}$$

Figure 10-1. LFSR



The serial implementation of LFSF has a limitation that it requires *n* clock cycles to calculate the CRC values for an *n* bit data stream. It produces the same CRC value operating on a multi-bit data stream, as would occur if the CRC were computed one bit at a time over the whole data stream. The algorithm involves looping to simulate the shifting, and concatenating strings to build the equations after *n* shifts.

The parallel CRC calculation based on the above polynomial is illustrated in the (HDL) code below.

```
for i in 63 to 0 loop

   NEXT_CRC_VAL(0) := CRC_VAL(63) xor DATA(i);

     for j in 1 to 63 loop
        case j is
            when 1|3|4 =>
                 NEXT_CRC_VAL(j) :=
                    CRC_VAL(j - 1) xor CRC_VAL(63) xor DATA(i);
            when others =>
                 NEXT_CRC_VAL(j) := CRC_VAL(j - 1);
        end case;
     end loop;

   CRC_VAL := NEXT_CRC_VAL;

end loop;
```

1 The inner loop calculates the next value of each shift register bit after one cycle.
2 The outer loop simulates 64 cycles of shifting. The equation for each shift register bit is thus built before it is compressed into the shift register.
3 The MSB of the data is shifted in first.

PSA_REGL1[31:0] and PSA_REGH1[63:32] contain the calculated 64-bit PSA signature value for channel 1. The PSA signature register can be read from and written to. When a value is written, it can either compress the data or just capture the data depending on the state of CHx_MODE bits in the CRC_CTRL2 register. If CHx_MODE is set to data capture, a seed value can be planted in the PSA signature register without compression. The CRC channel can be planted with a different seed value before compression starts. Modes other than data capture mode will cause the data to be compressed by the PSA signature register when it is written. When the PSA signature register is read, it gives the calculated signature value.

The CRC controller supports doubleword, word, half word, and byte accesses to the PSA signature register. During a non-doubleword write access, all unwritten byte lanes are padded with zeros before compression.

There is a software reset for the PSA signature register channel. When set, the PSA signature register is reset to all zeros by writing to the CRC_CTRL0 register.

The PSA signature register is reset to zero under any of the following conditions:

- The system resets.
- A PSA software reset occurs.
- One sector of data patterns is compressed.

10.3.3.2 PSA Sector Signature Register

After one sector of data is compressed, the final resulting signature calculated by the PSA signature register is transferred to the PSA sector signature registers PSA_SECSIGHx and PSA_SECSIGLx. The PSA sector signature register is a read-only register.

10.3.3.3 Raw Data Register

The raw or uncompressed data written to the PSA signature register is also saved in the raw data register. The raw data registers, RAW_DATAREGHx and RAW_DATAREGLx, are read only.

10.3.4 Power-Down Mode

The CRC module can be put into power-down mode by setting the power-down control bit, PWDN, in the CRC_CTRL1 register. The module wakes up when the PWDN bit is cleared. See the TMS470Px device-specific data sheet for the actual implementation of your device.

10.3.5 Emulation

During emulation mode, a read access from any register returns the register contents to the bus and will not trigger or mask any event as it would in functional mode. This prevents the debugger from disturbing the runtime environment. For example, when the debugger reads the interrupt offset register during a screen refresh, the corresponding interrupt status flag will not be cleared. Time-out counters are stopped in emulation mode. When channel 1 is configured to data trace mode, the PSA signature register does not compress any data read on the CPU data bus in emulation mode. No CPU bus error will be generated if reading from unimplemented locations in emulation mode.

10.4 CRC Control Registers

All registers are on word boundaries. 64, 32, 16, and 8 bit write accesses are supported to all registers. Figure 10-2 shows the summary of the registers. The base address for the control registers is 0xFE00 0000.

Figure 10-2. CRC Register Summary

Offset Address Register(1)	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00h CRC_CTRL0 Page 323	Reserved															
	Reserved															CH1_PSA_SWREST
0x08h CRC_CTRL1 Page 324	Reserved															
	Reserved															PWDN
0x010h CRC_CTRL2 Page 325	Reserved															
	Reserved												CH1_TRACEEN	Reserved		CH1_MODE
0x060h PSA_SIGREGL1 Page 327	PSASIG1(31–16)															
	PSASIG1(15–0)															
0x064h PSA_SIGREGH1 Page 328	PSASIG1(63–48)															
	PSASIG1(47–32)															
0x070h PSA_SECSIGREGL1 Page 329	PSASECSIG1(31–16)															
	PSASECSIG1(15–0)															
0x074h PSA_SECSIGREGH1 Page 329	PSASECSIG1(63–48)															
	PSASECSIG1(47–32)															
0x078h RAW_DATAREGL1 Page 331	RAW_DATA1(31–16)															
	RAW_DATA1(15–0)															

1 The physical address (base + offset) of these registers is device specific. See the device-specific data sheet to verify the register base address.

Figure 10-2. CRC Register Summary (Continued)

Offset Address Register[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x07Ch RAW_DATA REGH1 Page 332	RAW_DATA1(63–48)															
	RAW_DATA1(47–32)															

1 The physical address (base + offset) of these registers is device specific. See the device-specific data sheet to verify the register base address.

10.4.1 CRC Global Control Register 0 (CRC_CTRL0)

Figure 10-3 and Table 10-1 describe this register.

Figure 10-3. CRC Global Control Register 0 (CRC_CTRL0) [offset = 00h)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CH1_PSA_SWREST
R-0															R/W-0

R = Read; W = Write; *-n* = Value after reset

Table 10-1. CRC Global Control Register 0 (CRC_CTRL0) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		These bits are read as zero, and writes have no effect.
0	CH1_PSA_SWREST		Channel 1 PSA software reset: When set, the PSA signature register is reset to all zeros. Software reset does not reset software reset bit itself. Therefore, the CPU is required to clear this bit by writing a 0.
		0	The PSA signature register is not reset.
		1	The PSA signature register is reset.

10.4.2 CRC Global Control Register (CRC_CTRL1)

Figure 10-4 and Table 10-2 describe this register.

Figure 10-4. CRC Global Control (CRC_CTRL1) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PWDN
R-0															R/W-0

R = Read; W = Write; -*n* = Value after reset

Table 10-2. CRC Global Control (CRC_CTRL1) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		These bits are read as zero, and writes have no effect.
0	PWDN		Power-down.
		0	CRC is not in power-down mode.
		1	CRC is in power-down mode.

10.4.3 CRC Global Control Register 2 (CRC_CTRL2)

The CRC_CTRL2 register is used to configure the CRC channel in data capture mode. Data capture mode is especially useful when it is used in conjunction with data trace (CH1_TRACEEN) for channel 1. The seed value can be planted in the PSA signature register during data capture mode by writing a seed value into the PSA signature register. The data trace enable bit is then set to snoop and compress any read transactions on the CPU bus. When the trace enable bit is set, CH1_MODE is automatically reset to data capture mode.

To change from one mode to another, the user should perform the following steps:

1. Assert software reset for the respective channel.
2. Change from the current active mode to data capture mode.
3. Change from data capture mode to the new mode.
4. Release the software reset.

Figure 10-5 and Table 10-3 describe this register.

Figure 10-5. CRC Global Control Register 2 (CRC_CTRL2) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											CH1_TRACEEN	Reserved		CH1_MODE	
R-0											R/W-0	R-0		R/W-0	

R = Read; W = Write; *-n* = Value after reset

.

Table 10-3. CRC Global Control Register 2 (CRC_CTRL2) Field Descriptions

Bit	Name	Value	Description
31–5	Reserved		These bits are read as zero, and writes have no effect
4	CH1_TRACEEN		Channel 1 trace enable. When set, the channel is put into data trace mode. The channel snoops on the CPU data bus for any read transactions. Any read data on the bus is compressed by the PSA signature register. When the CPU is in debug state, the PSA signature register does not compress any read data on the bus.
		0	Data trace is disabled.
		1	Data trace is enabled.
3–2	Reserved		These bits are read as zero, and writes have no effect

Table 10-3. CRC Global Control Register 2 (CRC_CTRL2) Field Descriptions (Continued)

Bit	Name	Value	Description
1–0	CH1_MODE		Channel 1 mode
		00	The CRC is in data capture mode. In this mode, the PSA signature register does not compress data when it is written. Any data written to the PSA signature register is simply captured by the PSA signature register without any compression. This mode can be used to plant a seed value into the PSA register.
		Other	Reserved.

10.4.4 PSA Signature Low Register 1 (PSA_SIGREGL1)

Figure 10-6 and Table 10-4 describe this register.

Figure 10-6. PSA Signature Low Register 1 (PSA_SIGREGL1) [offset = 60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSASIG1(31–16)															
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSASIG1(15–0)															
R/W-0															

R = Read; W = Write; *-n* = Value after reset

Table 10-4. PSA Signature Low Register 1 (PSA_SIGREGL1) Field Descriptions

Bit	Name	Value	Description
31–0	PSASIG1(31–0)	0–FFFF FFFFh	Channel 1 PSA signature low. These bits contain the PSASIG1(31–0) of the Channel 1 PSA signature.

10.4.5 PSA Signature High Register 1 (PSA_SIGREGH1)

Figure 10-7 and Table 10-5 describe this register.

Figure 10-7. PSA Signature High Register 1 (PSA_SIGREGH1) [offset = 64h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSASIG1(63–48)															
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSASIG1(47–32)															
R/W-0															

R = Read; *-n* = Value after reset

Table 10-5. PSA Signature High Register 1 (PSA_SIGREGH1) Field Descriptions

Bit	Name	Value	Description
31–0	PSASIG1(63–32)	0–FFFF FFFFh	Channel 1 PSA signature high. These bits contain the PSASIG1(63–32) of the Channel 1 PSA signature.

10.4.6 PSA Sector Signature Low Register 1 (PSA_SECSIGREGL1)

Figure 10-8 and Table 10-6 describe this register.

Figure 10-8. PSA Sector Signature Low Register 1 (PSA_SECSIGREGL1) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSASECSIG1(31–16)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSASECSIG1(15–0)															
R-0															

R = Read; W = Write; *-n* = Value after reset

Table 10-6. PSA Sector Signature Low Register 1 (PSA_SECSIGREGL1) Field Descriptions

Bit	Name	Value	Description
31–0	PSASECSIG1(31–0)	0–FFFF FFFFh	Channel 1 PSA sector signature low. These bits contain the value stored in the PSASECSIG1(31–0) register.

10.4.7 PSA Sector Signature High Register 1 (PSA_SECSIGREGH1)

Figure 10-9 and Table 10-7 describe this register.

Figure 10-9. PSA Sector Signature High Register 1 (PSA_SECSIGREGH1) [offset = 74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSASECSIG1(63–48)															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSASECSIG1(47–32)															
R-0															

R = Read; W = Write; *-n* = Value after reset

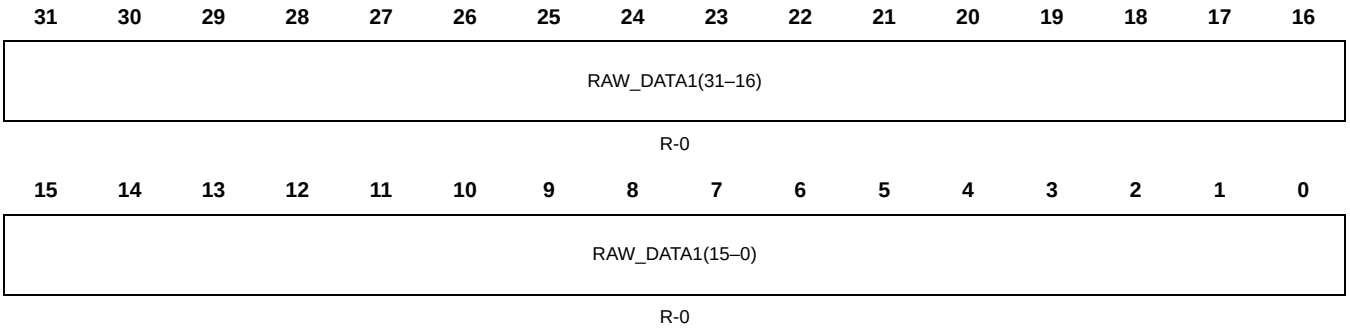
Table 10-7. PSA Sector Signature High Register 1 (PSA_SECSIGREGH1) Field Descriptions

Bit	Name	Value	Description
31–0	PSASECSIG1(63–32)	0–FFFF FFFFh	Channel 1 PSA sector signature high. These bits contain the value stored in the PSASECSIG1(63–32) register.

10.4.8 Raw Data Low Register 1 (RAW_DATAREGL1)

Figure 10-10 and Table 10-8 describe this register.

Figure 10-10. Raw Data Low Register 1 (RAW_DATAREGL1) [offset = 78h]



R = Read; W = Write; *-n* = Value after reset

Table 10-8. Raw Data Low Register 1 (RAW_DATAREGL1) Field Descriptions

Bit	Name	Value	Description
31–0	RAW_DATA1(31–0)	0–FFFF FFFFh	Channel 1 raw data low. These bits contain bits 31–0 of the uncompressed raw data.

10.4.9 Raw Data High Register 1 (RAW_DATAREGH1)

Figure 10-11 and Table 10-9 describe this register.

Figure 10-11. Raw Data High Register 1 (RAW_DATAREGH1) [offset = 7Ch)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RAW_DATA1(63–48)															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW_DATA1(47–32)															
R-0															

R = Read; W = Write; *-n* = Value after reset

Table 10-9. Raw Data High Register 1 (RAW_DATAREGH1) Field Descriptions

Bit	Name	Value	Description
31–0	RAW_DATA1(63–32)		Channel 1 raw data high. These bits contain bits 63–32 of the uncompressed raw data.

General-Purpose Input/Output (GIO) Module

The general-purpose input/output (GIO) module provides the TMS470Px family of devices with input/output (I/O) capability. The I/O pins are bidirectional and bit-programmable. The GIO module also supports external interrupt capability on up to 32 pins. Please check the device datasheet for the number of GIO pins in the device and the pins with interrupt capability.

11.1 Overview

The GIO module has the following features:

- Each I/O pin is controlled by bits in these registers:
  - Data direction (GIODIR; Section 11.5.11)
  - Data input (GIODIN; Section 11.5.12)
  - Data output (GIODOUT; Section 11.5.13)
  - Data set (GIODSET; Section 11.5.14)
  - Data clear (GIODCLR; Section 11.5.15)
  - Pull disable (GIOPULDIS; Section 11.5.16)
- The interrupts have the following characteristics:
  - Programmable interrupt detection either on both edges or on a single edge (set in GIOINTDET; Section 11.5.2)
  - Programmable edge-detection polarity, either rising or falling edge (set in GIOPOL register; Section 11.5.11)
  - Individual interrupt flags (set in GIOFLG register; Section 11.5.6)
  - Individual interrupt enables, set and cleared through GIOENASET and GIOENACLR registers respectively (Section 11.5.4)
  - Programmable interrupt priority, set through GIOLVLSET and GIOLVLCLR registers (Section 11.5.5)

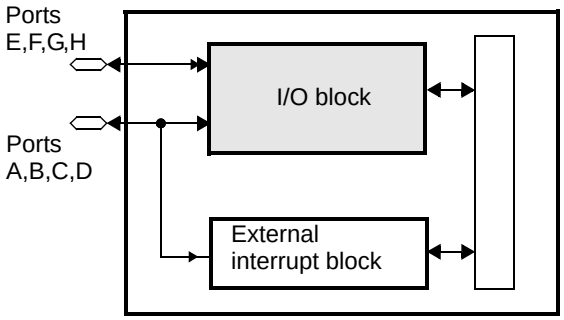
11.2 Functional Description of GIO Module

The GIO module (see Figure 11-1) comprises two separate components: an input/output (I/O) block and an external interrupt block. The GIO module can have up to 8 ports (A through H) with up to 8 pins (0 through 7) per port. The pins for ports E, F, G, and H handle the standard I/O functions and are connected directly to the VBUSP; see Figure 11-1. Ports A, B, C, and D can handle interrupts in addition to the standard I/O functions.

Note:
The number of GIO ports, the number of pins per port, and the pins with interrupt capability are device specific. Please check the device datasheet.

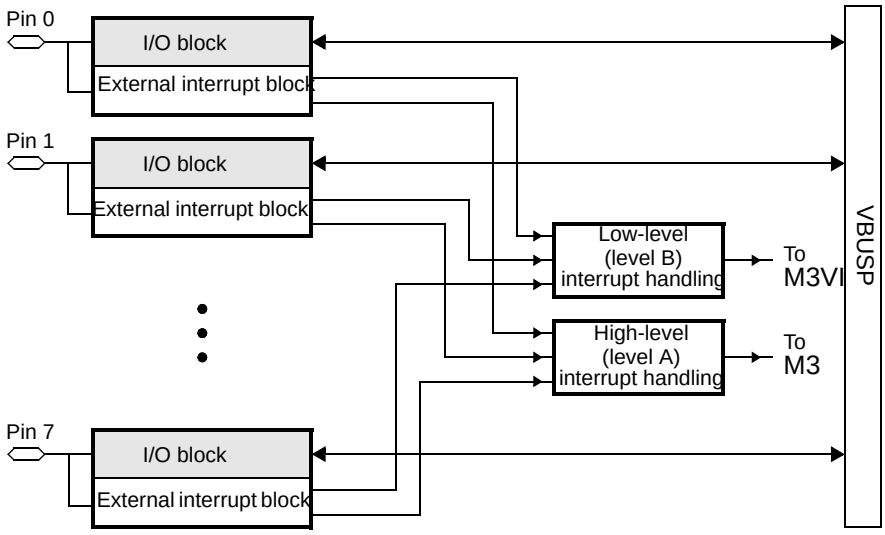
Note:
The device datasheet may specify the GIO ports as A, B, C, D etc., or as P0, P1, P2, P3 etc. Port A is equivalent to Port P0, Port B to Port P1 and so on.

Figure 11-1. GIO Module Diagram



The pins on port A (and optionally port B, C and D), shown in Figure 11-2, are all interrupt-capable pins and can be used to handle either general I/O functions or external interrupt signals. The pins are connected to both an I/O block and an external interrupt block. Each of the eight I/O blocks is connected to the VBUSP bus, whereas the interrupt blocks are physically attached to a single high-level-interrupt-handling block and to a single low-level-interrupt-handling block. The high-level-interrupt-handling block, which is also denoted as level A, and the low-level-interrupt-handling block, which is also denoted as level B, each send one signal to the vectored interrupt manager (M3VIM) in the system module for processing. The interrupt priority of level A and level B interrupt handling blocks can be re-programmed in the M3VIM.

Figure 11-2. GIO Port A / B / C / D Module Diagram



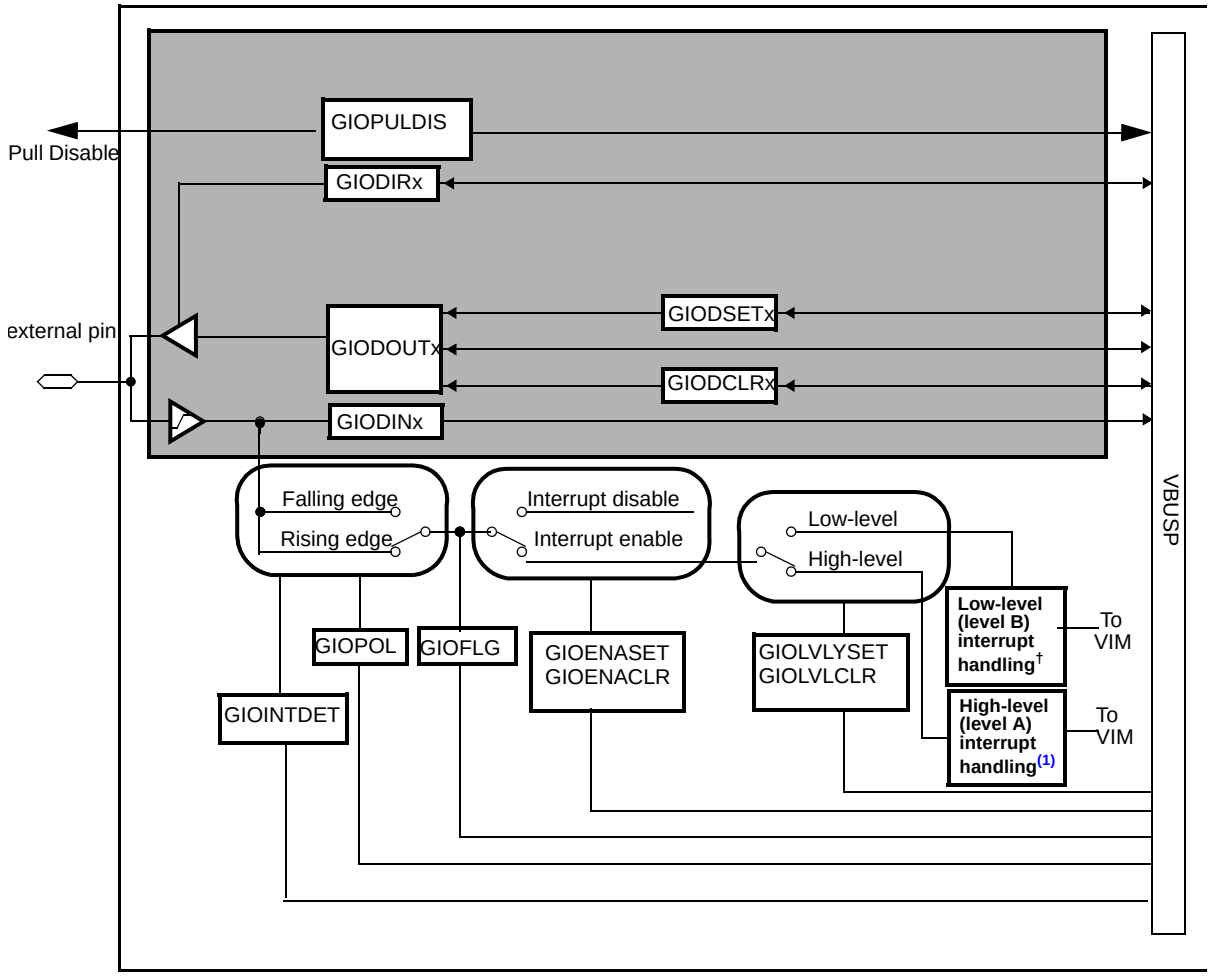
1 Not all devices have 32 external interrupts. The exact number of external interrupts is device specific; see the data sheet in for details.

11.2.1 GIO Block Diagram

The GIO block diagram (see Figure 11-3) represents the flow of information through a pin. The shaded area corresponds to the I/O block; the unshaded area corresponds to the external interrupt block.

Because ports E, F, G, and H are not interrupt-capable, the block diagram for those ports reduces to the shaded portion of the block diagram in Figure 11-3.

Figure 11-3. GIO Block Diagram



1 A single low-level-interrupt-handling block and a single high-level-interrupt-handling block service all of the interrupt-capable external pins, but only one pin can be serviced by an interrupt block at a time.

11.2.2 I/O Blocks

Each pin serviced by port A, B, C, D, E, F, G, and H contains its individual I/O block.

11.2.2.1 I/O Function

The GIO module sends data to the external pin through the output buffer and receives data from the external pin through the input buffer (see Figure 11-3). The associated registers are:

- GIODIR[7:0] (Section 11.5.11)—Controls the direction that information is sent. The GIODIR[7:0] register determines whether or not values in the data output register are sent to the external pin. The input buffer is always enabled. Therefore, the information that is sent to the external pin is also received in the input buffer.
- GIODOUT[7:0] (Section 11.5.13)—Controls what information is sent to the external pin when it is configured as an output. When the output buffer is enabled, writing values to the data output register (GIODOUT[7:0]) applies a voltage to the output pin. A low value (0) written to the data output register forces the pin to a low output voltage (V_{OL} or lower). A high value (1) written to the data output register forces the pin to a high output voltage (V_{OH} or higher)GIODIN[7:0] (Section 11.5.12)—Receives the information from the external pin. A high voltage (V_{IH} or greater) applied to the pin causes a high value (1) in the data input register (GIODIN[7:0]). When a low voltage (V_{IL} or less) is applied to the pin, the data input

register reads a low value (0). The V_{IH} and V_{IL} values are device specific and can be found in the device datasheet.

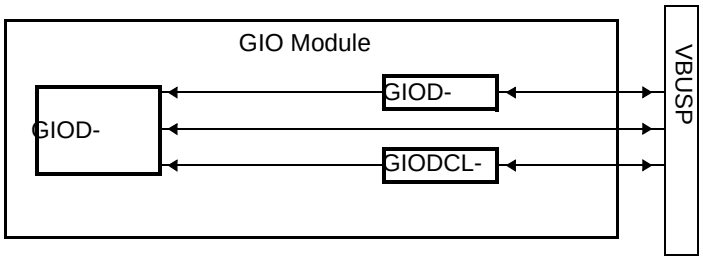
- GIOPULDIS[7:0] (Section 11.5.16)—Disables the pull control capability at pin. The pull functionality for the pin can be enabled or disabled in the GIOPULDIS[7:0] register.

11.2.2.2 Output Control Registers

When a GIO pin is configured as an output pin, the value in the data output register (GIODOUT[7:0]; Section 11.5.13) specifies the voltage applied to the external pin. The GIO module provides three ways of communicating with the data output register (see Figure 11-4).

- The control register bit can be written directly by writing to the data output register (GIODOUT[7:0]; Section 11.5.13).
- The data output register bit can be set to 1 using the data set register (GIODSET[7:0]; Section 11.5.14).
- The data output register bit can be cleared to 0 using the data clear register (GIODCLR[7:0]; Section 11.5.15).

Figure 11-4. Communication With the Data Output Register

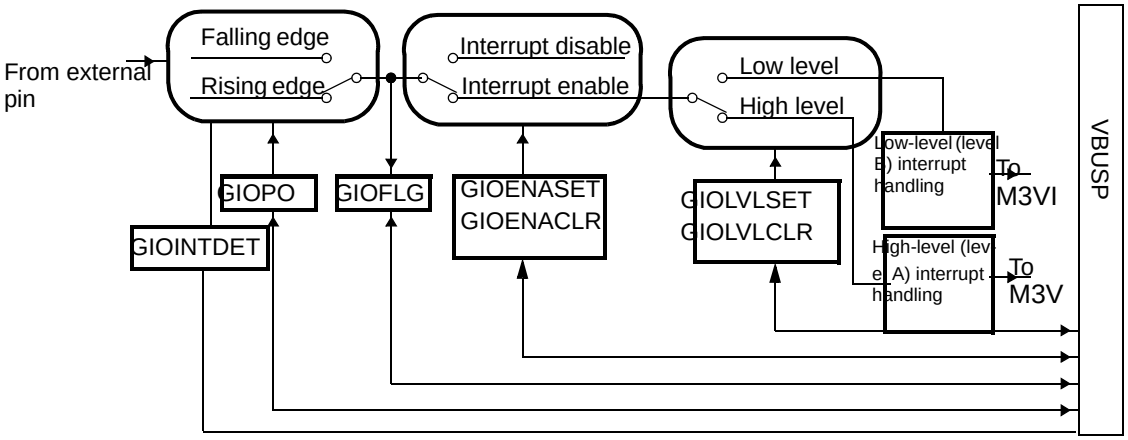


The GIODSET[7:0] and GIODCLR[7:0] registers allow improved handling of data. The data set and data clear registers preclude any possibility of a read-modify-write (RMW) operation. RMW operations are possible when the CPU reads a register, performs some action (for example, an OR operation), and then writes the values back into the register. Under such conditions, it is possible that the contents of the original register (GIODOUT[7:0]) can change (for example, an interrupt procedure) between the time when the CPU originally reads the register and the time when the CPU writes the new value. If the contents of the register changes between the time the CPU reads the register and the time it writes the new value to that register, then the CPU has ended up using an outdated register value as the basis for its operations, and therefore generates an erroneous result from those operations that also ends up being written back into the register. RMW operations can be avoided by using the GIODSET[7:0] and GIODCLR[7:0] registers.

11.2.3 External Interrupt Block

Each interrupt-capable pin connects to the GIO module's single low-level-interrupt-handling block and single high-level-interrupt-handling block. Depending on the priority, the interrupt signal is sent through the appropriate offset register to the vectored interrupt manager (M3VIM) in the system module (see Section 11.2.3.3). Figure 11-5 shows the external interrupt block.

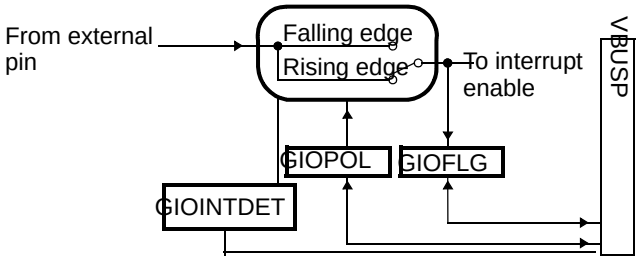
Figure 11-5. External Interrupt Block



11.2.3.1 Edge Detection and the Flag Register

The edge-detection hardware and flag register, like the input buffer, are always enabled. The GIOINTDET register (Section 11.5.2) specifies whether interrupt detection is on both edges or on a single edge only. If interrupt detection is on a single edge only, then the GIOPOL register (see Figure 11-6 and Section 11.5.3) is used to define whether a rising edge or a falling edge is recognized as an interrupt.

Figure 11-6. Edge Detection and Flag Register



A rising edge occurs when the voltage on a given pin transitions from a low value (V_{IL} or lower) to a high value (V_{IH} or higher). The voltage on the external pin must remain at the high level for at least one VCLK cycle to ensure recognition. (For an explanation of VCLK,).

A falling edge occurs when the voltage on the external pin transitions from a high value (V_{IH} or higher) to a low value (V_{IL} or lower). The voltage on the external pin must remain at the low level for at least one VCLK cycle to ensure recognition. (GIOPOL behaves differently in a low-power state. See Section 11.3.2 for more information.)

The GIOINTDET register (Section 11.5.2) is used to define whether the interrupt flag is set on both rising and falling edges or on a single edge only. Single edge polarity is defined in the polarity register (GIOPOL; Section 11.5.3).

The corresponding flag in the GIOFLG register (Section 11.5.6) is set when a transition appearing on the external pin matches the combination of edges chosen by the GIOINTDET and GIOPOL registers. For example, to set the flag on only a rising edge on pin 2 of GIO port A, clear the bit in the interrupt detection register (GIOINTDET[2] = 0), and set the bit in the polarity register (GIOPOL[2] = 1). Then, when the signal transition takes place, the GIO module will set the appropriate flag in the flag register (GIOFLG[2] = 1).

Note: Setting Flag With Interrupt Disabled

A flag can be set whether or not the interrupt is enabled. The flag register can then be polled instead of driving an interrupt. Additionally, the flag should not be set before

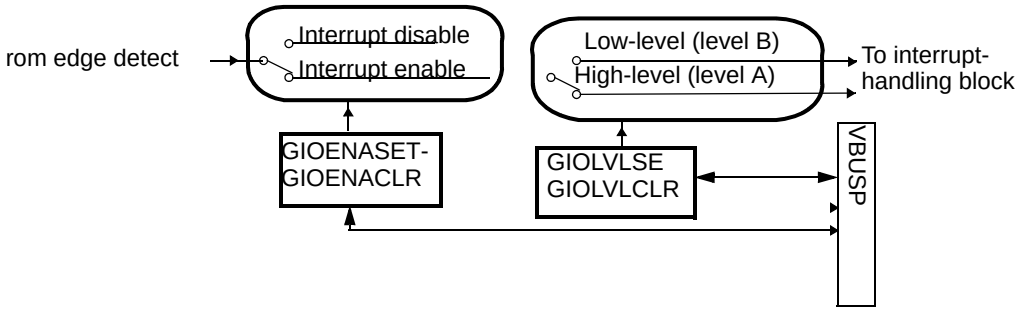
enabling the interrupt; specifically, the flag register should be cleared before enabling the interrupt.

The edge-detection hardware responds to voltages on the external pin and does not discriminate between the source of these voltages. Therefore, the interrupt will respond to the correct edge even when generated from a pin whose output buffer is enabled.

11.2.3.2 Interrupts and Interrupt Levels

The interrupt flag is set when an edge transition on the external pin is detected and matches the edge as chosen by the GIOINTDET and GIOPOL registers. An interrupt can be generated from the set flag if the interrupt is enabled (see Figure 11-7).

Figure 11-7. Interrupt Enable and Priority Level



The external interrupt can be enabled or disabled using the GIOENASET and GIOENACLR register (Section 11.5.4) bits respectively. These two registers are physically implemented as a single register. If the interrupt is enabled, the signal with an appropriate edge leads to an interrupt. If multiple interrupts occur simultaneously, the GIO module must prioritize the interrupts so that they can be handled in the proper order.

The order in which simultaneous GIO interrupts are processed is determined by the following criteria:

1. The GIO priority control registers (GIOLVLSET and GIOLVLCLR; Section 11.5.5) provide a software-implemented prioritization scheme. Each pin can be set as either a high-level (level A) or low-level interrupt (level B). By default in the Vectored Interrupt Manager (M3VIM), interrupts with level A are higher priority than those of level B and therefore are serviced first. However, the priority of level A and level B can be re-programmed in the M3VIM. Refer to the M3VIM module user's guide. The handling of interrupts with the same priority is determined by the interrupt with the lowest bit value having the highest priority. This prioritization is hardwired into the module.

Note: Wakeup Condition
GIOA interrupts are also used to awaken the device from the low power modes. The wakeup interrupt is level-based rather than edge-based.

Table 11-1 shows an example of how interrupt priorities are determined. Four interrupts occur simultaneously on GIO port A pins 3–0, and are handled as following:

1. The interrupts on pin 1 and pin 0 are both sent to the Vectored Interrupt Manager (M3VIM) for servicing because they have the lowest bit values among the high-level and low-level interrupts. Assuming that the M3VIM is set to service level A interrupt priority before level B interrupt priority, pin 1 is serviced first because it is the high-priority interrupt of the two.
2. Next, the interrupts on pin 3 and pin 0 are sent to the M3VIM for servicing. The interrupt on pin 3 is serviced because it is the only remaining high-priority interrupt.
3. The interrupt on pin 0 is serviced third because it has the lowest bit value.

The interrupt on pin 2 is serviced last because it is the only remaining interrupt.

Table 11-1. Determining Interrupt Priority

Pin (bit)	3	2	1	0
Priority	High Level (Level A)	Low Level (Level B)	High Level (Level A)	Low Level (Level B)
Order	2	4	1	3

11.2.3.3 High-Level-Interrupt Block and Low-Level-Interrupt Block

The interrupt-handling blocks each contain two registers: an offset register and an emulation register. The high-level interrupts are denoted as level A and consequently, the registers for the high-level-interrupt-handling block are GIOOFFA (Section 11.5.7) and GIOEMUA (Section 11.5.9); see Figure 11-8. Likewise, the registers for the low-level-interrupt-handling block (denoted as level B) are GIOOFFB (Section 11.5.8) and GIOEMUB (Section 11.5.10); see Figure 11-9.

Figure 11-8. High-Level-Interrupt-Handling Block

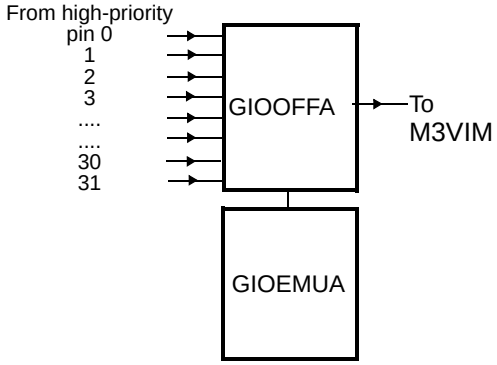
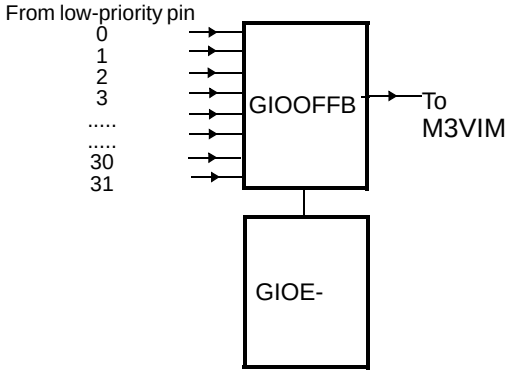


Figure 11-9. Low-Level-Interrupt-Handling Block



The read-only registers GIOOFFA and GIOOFFB generate a numerical offset value that represents the highest-priority pending external interrupt (see Table 11-2). The offset can be used to locate the position of the interrupt routine in the vector table. A read of the offset register clears the offset register and the corresponding flag bit in the GIOFLG register.

The high-level offset register, GIOOFFA, receives signals from each active interrupt that is configured as high-level. The GIOOFFA displays the high-level interrupt with the highest priority (that is, the interrupt that was generated by the lowest bit in the flag register). Similarly, the GIOOFFB displays the low-level interrupt with the highest priority.

The emulation control registers (GIOEMUA and GIOEMUB; Section 11.5.9 and Section 11.5.10) mirror the offset registers. The emulation registers contain a numerical offset value that represent the highest priority pending external interrupt (see Table 11-2). A read of the emulation registers does not clear any bit in any register. These registers allow the device emulator to read and display the offset register values without affecting interrupt execution.

Table 11-2. GIO Offset A or B Values and Corresponding Interrupt

GIOOFF(5–0)	Pending External Interrupt
000000	No interrupt
000001	Interrupt 0
000010	Interrupt 1
000011	Interrupt 2
000100	Interrupt 3
. . .	. . .
100000	Interrupt 31
100001–111111	Reserved

Table 11-3 illustrates how to identify the interrupt being serviced. In this example, interrupts are enabled for pins 0, 1 and 2 and all three pins are set to the same interrupt level (same priority). This example assumes that interrupts on pins 0 and 2 occur simultaneously. Reading the flag control register, GIOFLG, returns a value of xxxxx101, indicating that interrupt flags have been set for pins 0 and 2. The first five values are indeterminate because only pins 0, 1 and 2 are interrupt-enabled. Reading the offset B control register, GIOOFFB, returns a value of 00000001, indicating that the interrupt on pin 0 is currently being processed (see Table 11-2).

Table 11-3. Reading the Offset Register to Determine Serviced Interrupt

Bits	7	6	5	4	3	2	1	0
GIOENA	Reserved					1	1	1
GIOPOL	Reserved					1	0	0
GIOFLG	Reserved					1	0	1
GIOLVL	Reserved					0	0	0
GIOOFFA	0	0	0	0	0	0	0	0
GIOEMUA	0	0	0	0	0	0	0	0
GIOOFFB	0	0	0	0	0	0	0	1
GIOEMUB	0	0	0	0	0	0	0	1

11.2.3.4 Special Considerations for Interrupts

Please note that interrupts are subject to the following special considerations:

- On devices where fewer than 32 interrupts are available, the unused control register bits are reserved.
- To use the enabled pins as interrupts, the I/O function of the pins are typically set as input. If the pin's I/O function is set as output, the signal feeds directly into the input. In this case, interrupts are only generated when the device toggles the data output register, thereby creating the appropriate interrupt edge. See Table 11.6.4.
- The interrupt flag can be set even though the interrupt is not enabled. Therefore, you must clear the flag register before enabling the interrupts to ensure that a spurious interrupt is not generated. See Section 11.6.4.

- If interrupts are enabled when GIODIN[7:0] is read, the bits corresponding to the enabled interrupts must be masked to avoid ambiguous results. For example, if GIO port A pins [2:0] are configured as interrupts and pins [7:3] are configured as inputs (V_{IH} applied), then a read of GIODIN0 will read 11111xxx, where the x values are interrupt levels and not inputs. Mask the input register against (in this case) 11111000. See section 11.6.1, *Example: Setting Interrupts and Configuring Pins for Output*, page 375.

11.3 Device Modes of Operation

The GIO module behaves differently in different modes of operation. There are two main modes:

- Emulation mode
- Power-down mode (Low-power mode)
  - Module level power-down
  - Device level power-down

11.3.1 Emulation Mode

Emulation mode is used by debugger tools to stop the CPU at breakpoints to read registers. When the device is in emulation mode, it pulls the suspend signal high.

Note: Emulation Mode
Emulation mode is a mode of operation of the device and is separate from the GIO emulation registers (GIOEMUA and GIOEMUB).

During emulation mode:

- External interrupts are not captured because the M3VIM is unable to service interrupts.
- Any register can be read without affecting the state of the system.
- A write to a register affects the state of the system.

11.3.2 Power-Down Mode (Low-Power Mode)

In the power-down mode, the clock signal to the GIO module is disabled. Thus, there is no switching and the only current draw comes from leakage current. The GIO module has two power-down modes: module-level power down and device-level power down. In both these power-down modes (low-power modes), interrupt pins become level-sensitive rather than edge-sensitive. The polarity bit changes function from falling edge and rising edge to low and high. A corresponding level on an interrupt pin pulls the module out of low-power mode.

11.3.2.1 Module-Level Power Down

The GIO module can be placed into a power down state by disabling the GIO peripheral module via the appropriate bit in the peripheral power down register. Please refer to the Peripheral Central Resource Register for details.

11.3.2.2 Device-Level Power Down

The entire device can be placed in one of the pre-defined low-power modes: doze, snooze, sleep, or hibernate. See the device datasheet for details.

11.4 Pullup/Pulldown Function

GIO module pins can have either an active pullup or active pulldown that makes it possible to leave the pins unconnected externally when the pins are input pins. The pull capability is enabled by programming the GIOPULDIS[7:0] register (Section 11.5.16). By enabling the pull capability, the default pull on the GIO pins, as indicated in the device datasheet, are enabled. The default pullup/pulldown functionality of all GIO pins at system reset is fixed by design. Please see the TMS470Px device-specific data sheet for the reset state of the pullups/pulldowns and for the current supplied by the pullups/pulldowns.

Note:
It is possible to disable the GIO Input buffer altogether by configuring the pin direction to input via the GIODIR[A-H][7:0], disabling the pin's pull functionality using GIOPDIS[7:0], and setting the pin's corresponding bit in the GIOPSEL[7:0] to 0. Once an input buffer is disabled, the external signal can not be read from the buffer internally.

11.5 GIO Control Registers

Table 11-4 shows the summary of the GIO registers. It assumes eight ports designated A–H (four of them interrupt capable ports), with eight pins per port. As an example, port B bits [2:0] of the GIODIR register would be represented as GIODIR[B][2:0]. Each port has a set of registers that control the I/O function of the pins for that port - they are GIODIR[A-H][7:0], GIODIN[A-H][7:0], GIODOUT[A-H][7:0], GIODSET[A-H][7:0], GIODCLR[A-H][7:0], GIOPULDIS[A-H][7:0], .

The registers are accessible in 8-, 16-, and 32-bit reads or writes. Consult the device-specific data sheet to verify the pin configuration. The start address for the GIO module is 0xFFF7 BC00.

Table 11-4. GIO Control Register Summary

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 GIOGCR0 Page 349	Reserved															
	Reserved															RESET
0x08 GIOINTDET Page 350	GIOINTDET 3[7:0]								GIOINTDET 2[7:0]							
	GIOINTDET 1[7:0]								GIOINTDET 0[7:0]							
0x0C GIOPOL Page 352	GIOPOL 3[7:0]								GIOPOL 2[7:0]							
	GIOPOL 1[7:0]								GIOPOL 0[7:0]							
0x10 GIOENASET Page 354	GIOENASET 3[7:0]								GIOENASET 2[7:0]							
	GIOENASET 1[7:0]								GIOENASET 0[7:0]							
0x14 GIOENACLR Page 354	GIOENACLR 3[7:0]								GIOENACLR 2[7:0]							
	GIOENACLR 1[7:0]								GIOENACLR 0[7:0]							
0x18 GIOLVL SET Page 357	GIOLVLSET 3[7:0]								GIOLVLSET 2[7:0]							
	GIOLVLSET 1[7:0]								GIOLVLSET 0[7:0]							
0x1C GIOLVLCLR Page 357	GIOLVLCLR 3[7:0]								GIOLVLCLR 2[7:0]							
	GIOLVLCLR 1[7:0]								GIOENACLR 0[7:0]							

1 See the specific device data sheet to verify the base address of the GIO registers.

Table 11-4. GIO Control Register Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x20 GIOFLG Page 361	GIOFLG 3[7:0]								GIOFLG 2[7:0]							
	GIOFLG 1[7:0]								GIOFLG 0[7:0]							
0x24 GIOOFFA Page 363	Reserved															
	Reserved										GIOOFFA[5:0]					
0x28 GIOOFFB Page 364	Reserved															
	Reserved										GIOOFFB[5:0]					
0x2C GIOEMUA Page 365	Reserved															
	Reserved										GIOEMUA[5:0]					
0x30 GIOEMUB Page 366	Reserved															
	Reserved										GIOEMUB[5:0]					
0x34, 54, 74, 94, B4, D4, F4, 114 GIODIR[7:0] Page 367	Reserved															
	Reserved								GIODIR0[A-H][7:0]							
0x38, 58, 78, 98, B8, D8, F8, 118 GIODIN[7:0] Page 368	Reserved															
	Reserved								GIODIN0[A-H][7:0]							
0x3C, 5C, 7C, 9C, BC, DC, FC, 11C GIODOUT[7:0] Page 369	Reserved															
	Reserved								GIODOUT0[A-H][7:0]							

1 See the specific device data sheet to verify the base address of the GIO registers.

Table 11-4. GIO Control Register Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x40, 60, 80, A0, C0, E0, 100, 120 GIOSET[7:0] Page 370	Reserved															
	Reserved								GIOSET[A-H][7:0]							
0x44, 64, 84, A4, C4, E4, 104, 124 GIOCLR[7:0] Page 371	Reserved															
	Reserved								GIOCLR[A-H][7:0]							
0x4C, 6C, 8C, AC, CC, EC, 10C, 12C GIOPULDIS[7:0] Page 372	Reserved															
	Reserved								GIOPULDIS[A-H][7:0]							
0x50, 70, 90, B0, D0, F0, 110, 130 GIOPSL[7:0] Page 373	Reserved															
	Reserved								GIOPSL[A-H][7:0]							

1 See the specific device data sheet to verify the base address of the GIO registers.

11.5.1 GIO Global Control Register (GIOGCR0)

The GIOGCR0 register contains one bit that controls the module reset status. Writing a zero (0) to this bit puts the module in a reset state. After system reset, this bit must be set to 1 before normal operations can begin on this module. Figure 11-10 and Table 11-5 describe this register.

Figure 11-10. GIO Global Control Register (GIOGCR0) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RESET
R-0															R/WP-0

R = Read in all modes; WP = Write in privileged mode only; -n = Value after reset

.

Table 11-5. GIO Global Control Register (GIOGCR0) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zeros and writes have no effect.
0	RESET		GIO reset.
		0	The GIO is in reset state.
		1	The GIO is operating normally.

11.5.2 GIO Interrupt Detect Register (GIOINTDET)

The GIOINTDET register provides the flexibility to either ignore the polarity of the edges that are recognized as an interrupt, in which case both rising and falling edges are recognized, or recognizing the interrupt on specifically a rising or falling edge as determined by the GIOPOL register (Section 11.5.3). To ensure recognition of the signal as an edge, the signal must maintain the new level for at least one VCLK cycle. Figure 11-11 and Table 11-6 describe this register.

Figure 11-11. GIO Interrupt Detect Register (GIOINTDET) [offset = 08h]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-6. GIO Interrupt Detect Register (GIOINTDET) Field Descriptions

Bit	Name	Value	Description
31–24	GIOINTDET 3[7:0]		Interrupt detection select for pins GIOD[7:0].
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL; Section 11.5.3).
		1	The flag sets on both the rising and falling edges on the corresponding pin.
23–16	GIOINTDET 2[7:0]		Interrupt detection select for pins GIOC[7:0].
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL; Section 11.5.3).
		1	The flag sets on both the rising and falling edges on the corresponding pin.
15–8	GIOINTDET 1[7:0]		Interrupt detection select for pins GIOB[7:0].
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL; Section 11.5.3).
		1	The flag sets on both the rising and falling edges on the corresponding pin.

Table 11-6. GIO Interrupt Detect Register (GIOINTDET) Field Descriptions (Continued)

Bit	Name	Value	Description
7–0	GIOINTDET 0[7:0]		Interrupt detection select for pins GIOA[7:0].
		0	The flag sets on either a falling or a rising edge on the corresponding pin, depending on the polarity setup in the polarity register (GIOPOL; Section 11.5.3).
		1	The flag sets on both the rising and falling edges on the corresponding pin.

11.5.3 GIO Interrupt Polarity Register (GIOPOL)

The GIOPOL register controls the polarity—rising edge (low to high) or falling edge (high to low)—that sets the flag. To ensure recognition of the signal as an edge, the signal must maintain the new level for at least one VCLK cycle. When the device is in low power mode, the interrupts are no longer triggered by an edge, but instead by a level. Therefore, in low power mode, the GIOPOL register controls the ***level***, high or low, which will trigger the interrupt. Figure 11-12 and Table 11-7 describe this register.

Figure 11-12. GIO Interrupt Polarity Register (GIOPOL) [offset = 0Ch]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-7. GIO Interrupt Polarity Register (GIOPOL) Field Descriptions

Bit	Name	Value	Description
31–24	GIOPOL 3[7:0]		Interrupt polarity select for pins GIOD[7:0].
			User or privileged mode:
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (doze, snooze, sleep or hibernate):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.
23–16	GIOPOL 2[7:0]		Interrupt polarity select for pins GIOC[7:0].
			User or privileged mode:
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (doze, snooze, sleep or hibernate):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.

Table 11-7. GIO Interrupt Polarity Register (GIOPOL) Field Descriptions (Continued)

Bit	Name	Value	Description
15–8	GIOPOL 1[7:0]		Interrupt polarity select for pins GIOB[7:0].
			User or privileged mode:
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (doze, snooze, sleep or hibernate):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.
7–0	GIOPOL 0[7:0]		Interrupt polarity select for pins GIOA[7:0].
			User or privileged mode:
		0	The flag is set on the falling edge on the corresponding pin.
		1	The flag is set on the rising edge on the corresponding pin.
			Low-power mode (doze, snooze, sleep or hibernate):
		0	The interrupt is triggered on the low level.
		1	The interrupt is triggered on the high level.

11.5.4 GIO Interrupt Enable Registers (GIOENASET and GIOENACLR)

The GIOENASET and GIOENACLR register controls which interrupt-capable pins are configured as interrupts. These two registers are physically implemented as a single register. If the interrupt is enabled, the signal with an appropriate edge leads to an interrupt.

11.5.4.1 GIOENASET Register

Figure 11-13 and Table 11-8 describe this register.

Note: Enabling Interrupt at the Device Level
Two bits, corresponding to the GIO high level (level A) and low level (level B) interrupts, must be set within the vectored interrupt manager (M3VIM) in the interrupt mask register (REQMASK;) to enable the appropriate interrupts. Additionally, the ARM CPU must be configured to recognize interrupt requests (IRQ/FIQ); .

Figure 11-13. GIO Interrupt Enable Register (GIOENASET) [offset = 10h]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-8. GIO Interrupt Enable Register (GIOENASET) Field Descriptions

Bit	Name	Value	Description
31–24	GIOENASET 3[7:0]		Interrupt enable for pins GIOD[7:0]
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
23–16	GIOENASET 2[7:0]		Interrupt enable for pins GIOC[7:0]
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
15–8	GIOENASET 1[7:0]		Interrupt enable for pins GIOB[7:0]
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.

Table 11-8. GIO Interrupt Enable Register (GIOENASET) Field Descriptions (Continued)

Bit	Name	Value	Description
7–0	GIOENASET 0[7:0]		Interrupt enable for pins GIOA[7:0]
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.

11.5.4.2 GIOENACLR Register

This register disables the interrupt. Figure 11-14 and Table 11-9 describe this register.

Figure 11-14. GIO Interrupt Enable Register (GIOENACLR) [offset = 14h]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-9. GIO Interrupt Enable Register (GIOENACLR) Field Descriptions

Bit	Name	Value	Description
31–24	GIOENACLR 3[7:0]		Interrupt disable for pins GIOD[7:0].
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* Writing a one to this bit disables the interrupt.
23–16	GIOENACLR 2[7:0]		Interrupt disable for pins GIOC[7:0].
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* Writing a one to this bit disables the interrupt.
15–8	GIOENACLR 1[7:0]		Interrupt disable for pins GIOB[7:0].
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* Writing a one to this bit disables the interrupt.
7–0	GIOENACLR 0[7:0]		Interrupt disable for pins GIOA[7:0].
		0	*Read:* The interrupt is disabled. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* Writing a one to this bit disables the interrupt.

11.5.5 GIO Interrupt Priority Registers (GIOLVLSET and GIOLVLCLR)

The GIOLVLSET and GIOLVLCLR registers configure the interrupts as high-level (level A) or low-level (level B) going to the M3VIM. Each interrupt is individually configured.

- The high-level interrupts are recorded to GIOOFFA and GIOEMUA.
- The low-level interrupts are recorded to GIOOFFB and GIOEMUB.

11.5.5.1 GIOLVLSET Register

The GIOLVLSET register is used to configure an interrupt as a high-level interrupt going to the M3VIM. An interrupt can be configured as a high level interrupt by writing a 1 into the corresponding bit of the GIOLVLSET register. Writing a zero has no effect. Figure 11-15 and Table 11-10 describe this register.

Figure 11-15. GIO Interrupt Priority Register (GIOLVSLSET) [offset = 18h]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-10. GIO Interrupt Priority Register (GIOLVSLSET) Field Descriptions

Bit	Name	Value	Description
31-24	GIOLVSLSET 3[7:0]		GIO high priority interrupt for pins GIOD[7:0].
		0	*Read:* The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA.
23-16	GIOLVSLSET 2[7:0]		GIO high priority interrupt for pins GIOC[7:0].
		0	*Read:* The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA.

Table 11-10. GIO Interrupt Priority Register (GIOLVSLSET) Field Descriptions (Continued)

Bit	Name	Value	Description
15-8	GIOLVSLSET 1[7:0]		GIO high priority interrupt for pins GIOB[7:0].
		0	*Read:* The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA.
7-0	GIOLVSLSET 0[7:0]		GIO high priority interrupt for pins GIOA[7:0].
		0	*Read:* The interrupt is a low-level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB. *Write:* Writing a zero to this bit has no effect.
		1	*Read or write:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA.

11.5.5.2 GIOLVLCLR Register

The GIOLVLCLR register is used to configure an interrupt as a low level interrupt going to the M3VIM. Figure 11-16 and Table 11-11 describe this register.

Figure 11-16. GIO Interrupt Priority Register (GIOLVLCLR) [offset = 1Ch]

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

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-11. GIO Interrupt Priority Register (GIOLVLCLR) Field Descriptions

Bit	Name	Value	Description
31–24	GIOLVLCLR 3[7:0]		GIO low priority interrupt for pins GIOD[7:0].
		0	*Read:* The interrupt is a low-level interrupt. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA. *Write:* The interrupt is set as a low level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB.
23–16	GIOLVLCLR 2[7:0]		GIO low priority interrupt for pins GIOC[7:0].
		0	*Read:* The interrupt is a low-level interrupt. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA. *Write:* The interrupt is set as a low level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB.
15–8	GIOLVLCLR 1[7:0]		GIO low priority interrupt for pins GIOB[7:0].
		0	*Read:* The interrupt is a low-level interrupt. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA. *Write:* The interrupt is set as a low level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB.

Table 11-11. GIO Interrupt Priority Register (GIOLVLCLR) Field Descriptions (Continued)

Bit	Name	Value	Description
7–0	GIOLVLCLR 0[7:0]		GIO low priority interrupt for pins GIOA[7:0].
		0	*Read:* The interrupt is a low-level interrupt. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The interrupt is set as a high level interrupt. The high-level interrupts are recorded to GIOOFFA and GIOEMUA. *Write:* The interrupt is set as a low level interrupt. The low-level interrupts are recorded to GIOOFFB and GIOEMUB.

11.5.6 GIO Interrupt Flag Register (GIOFLG)

The GIOFLG register contains flags indicating that the transition edge (as set in GIOINTDET;Section 11.5.2 and GIOPOL; Section 11.5.3) has occurred. The flag is also cleared by reading the appropriate offset register; see Section 11.2.3.3. Figure 11-17 and Table 11-12 describe this register.

Figure 11-17. GIO Interrupt Flag Register (GIOFLG) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GIOFLG 3[7:0]								GIOFLG 2[7:0]							
R/WC-0								R/WC-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GIOFLG 1[7:0]								GIOFLG 0[7:0]							
R/WC-0								R/WC-0							

R = Read in all modes; WC = Write clears the bit, -n = Value after reset

.

Table 11-12. GIO Interrupt Flag Register (GIOFLG) Field Descriptions

Bit	Name	Value	Description
31–24	GIOFLG 3[7:0]		GIO flag for pins GIOD[7:0].
		0	*Read:* A transition has not occurred since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A transition has occurred since the last clear. *Write:* The corresponding bit is cleared to 0.
			Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register
23–16	GIOFLG 2[7:0]		GIO flag for pins GIOC[7:0].
		0	*Read:* A transition has not occurred since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A transition has occurred since the last clear. *Write:* The corresponding bit is cleared to 0.
			Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register
15–8	GIOFLG 1[7:0]		GIO flag for pins GIOB[7:0].
		0	*Read:* A transition has not occurred since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A transition has occurred since the last clear. *Write:* The corresponding bit is cleared to 0.
			Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register

Table 11-12. GIO Interrupt Flag Register (GIOFLG) Field Descriptions (Continued)

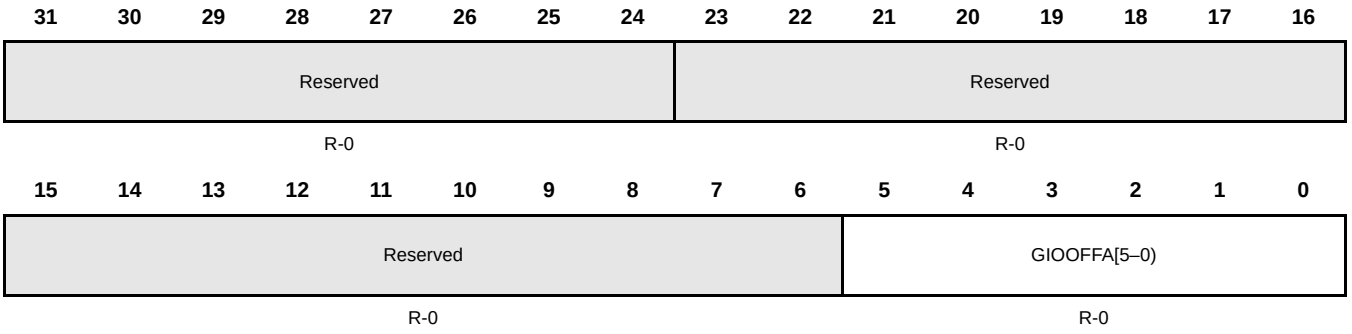
Bit	Name	Value	Description
7–0	GIOFLG 0[7:0]		GIO flag for pins GIOA[7:0].
		0	*Read:* A transition has not occurred since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The selected transition on the corresponding pin has occurred. *Write:* The corresponding bit is cleared to 0. **Note: This bit is also cleared by a read to the corresponding bit in the appropriate offset register**

11.5.7 GIO Offset A Register (GIOOFFA)

The GIOOFFA register provides a numerical offset value that represents the pending external interrupt with high priority. The offset value can be used to locate the position of the interrupt routine in a vector table in application software.Figure 11-18 and Table 11-13 describe this register.

Note:
Reading this register clears it and the corresponding flag bit in the GIOFLG register; Section 11.5.6. However, in emulation mode, a read to this register does not clear the corresponding flag bit.

Figure 11-18. GIO Offset A Register (GIOOFFA) [offset = 24h]



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

Table 11-13. GIO Offset A Register (GIOOFFA) Field Descriptions

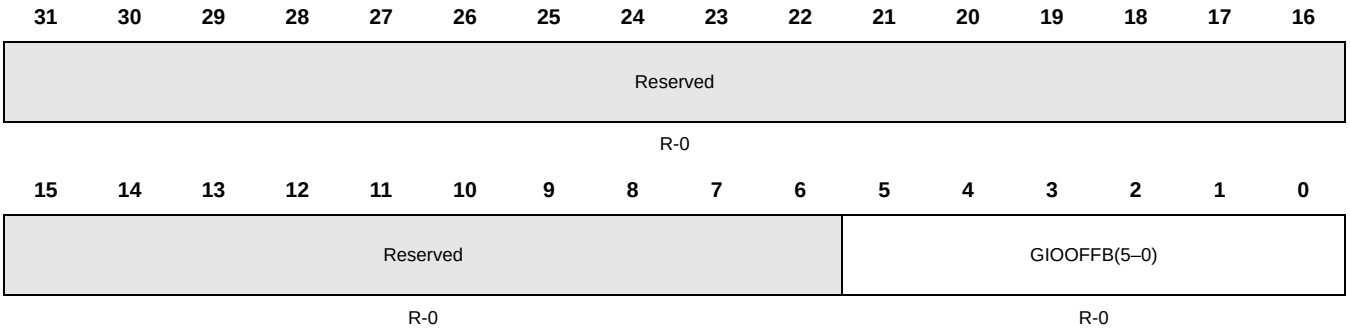
Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–0	GIOOFFA(5–0)		GIO offset A. These bits index the currently pending high-priority interrupt. This register and the flag bit (in the GIOFLG register) are also cleared when this register is read, except in emulation mode.
		000000	No interrupt is pending.
		000001	Interrupt 0 is pending with a high priority.
		. . .	. . .
		100000	Interrupt 31 is pending with a high priority.
		100001–111111	Reserved

11.5.8 GIO Offset B Register (GIOOFFB)

The GIOOFFB register provides a numerical offset value that represents the pending external interrupt with low priority. The offset value can be used to locate the position of the interrupt routine in a vector table in application software. Figure 11-19 and Table 11-14 describe this register.

Note:
Reading this register clears it and the corresponding flag bit in the GIOFLG register; Section 11.5.6. However, in emulation mode, a read to this register does not clear the corresponding flag bit.

Figure 11-19. GIO Offset B Register (GIOOFFB) [offset = 28h]



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

Table 11-14. GIO Offset B Register (GIOOFFB) Field Descriptions

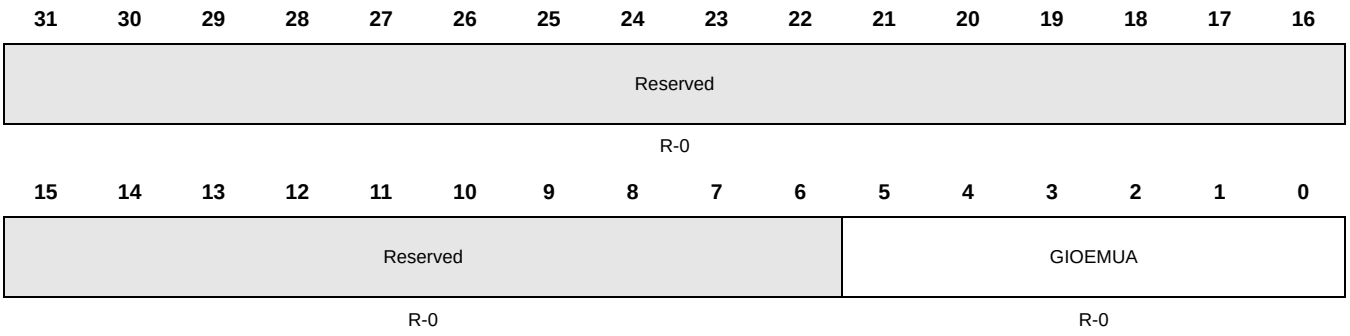
Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–0	GIOOFFB		GIO offset register B. These bits index the currently pending low-priority interrupt. This register and the flag bit (in the GIOFLG register) are also cleared when this register is read, except in emulation mode.
		000000	No interrupt is pending.
		000001	Interrupt 0 is pending with a low priority.
		. . .	. . .
		100000	Interrupt 31 is pending is pending with a low priority.
		100001–111111	Reserved

11.5.9 GIO Emulation A Register (GIOEMUA)

The GIOEMUA register is a read-only register, and is provided for use by the debug monitor in normal operation. The contents of this register are identical to the contents of GIOOFFA. Figure 11-20 and Table 11-15 describe this register.

Note:
The corresponding flag in the GIOFLG register (Section 11.5.6) is not cleared when the GIOEMUA register is read.

Figure 11-20. GIO Emulation A Register (GIOEMUA) [offset = 2Ch]



R = Read in all modes; n = Value after reset

Table 11-15. GIO Emulation A Register (GIOEMUA) Field Descriptions

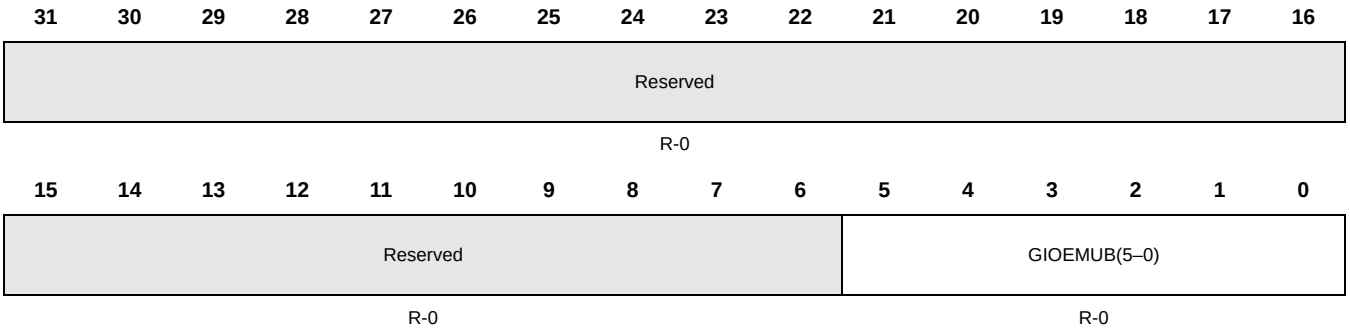
Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–0	GIOEMUA		GIO emulation register A. These bits index the currently pending high-priority interrupt.
		000000	No interrupt is pending.
		000001	Interrupt 0 is pending with a high priority.
		. . .	. . .
		100000	Interrupt 31 is pending with a high priority.
		100001–111111	Reserved

11.5.10 GIO Emulation B Register (GIOEMUB)

The GIOEMUB register is a read-only register, and is provided for use by the debug monitor in normal operation. The contents of this register are identical to the contents of GIOOFFB. Figure 11-21 and Table 11-16 describe this register.

Note:
The corresponding flag in the GIOFLG register (Section 11.5.6) is not cleared when the GIOEMUB register is read.

Figure 11-21. GIO Emulation B Register (GIOEMUB) [offset = 30h]



R = Read in all modes; -n = Value after reset

.

Table 11-16. GIO Emulation B Register (GIOEMUB) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–0	GIOEMUB		GIO emulation register B. These bits index the currently pending low-priority interrupt.
		000000	No interrupt is pending.
		000001	Interrupt 0 is pending with a low priority.
		. . .	. . .
		100000	Interrupt 31 is pending with a low priority.
		100001–111111	Reserved

11.5.11 GIO Data Direction Registers [A-H][7:0] (GIODIR[A-H][7:0])

The GIODIR register controls whether the pins of a given port are configured as inputs or outputs. Figure 11-22 and Table 11-17 describe this register.

Figure 11-22. GIO Data Direction Registers [A-H][7:0] (GIODIR[A-H][7:0])

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GIODIR[A-H][7:0]							
R-0								R/W-0							

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

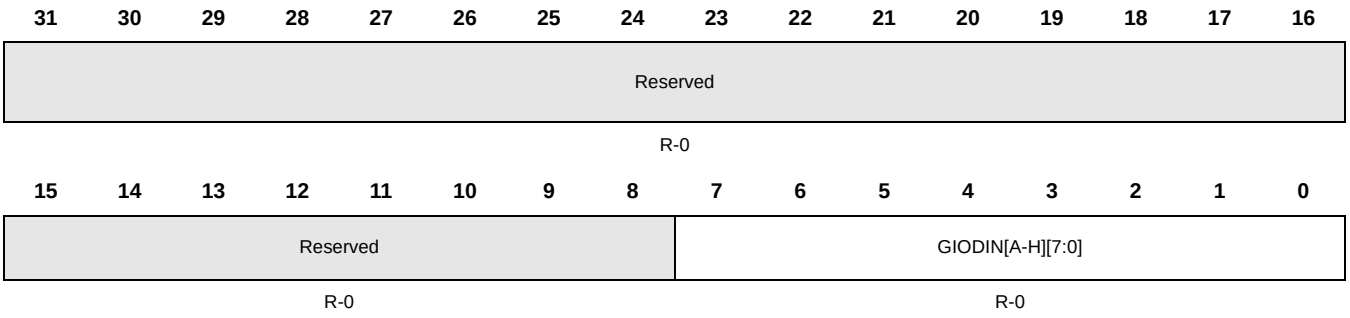
Table 11-17. GIO Data Direction Registers [A-H][7:0] (GIODIR[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	GIODIR[A-H][7:0]		GIO data direction of ports [A-H], pins[7:0].
		0	The output buffer is disabled.
		1	The output buffer is enabled.

11.5.12 GIO Data Input Registers [A-H][7:0] (GIODIN[A-H][7:0])

Values in the GIODIN register reflect the current state (high = 1 or low = 0) on the pins of the port. Figure 11-23 and Table 11-18 describe this register.

Figure 11-23. GIO Data Input Registers [A-H][7:0] (GIODIN[A-H][7:0])



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

Table 11-18. GIO Data Input Registers [A-H][7:0] (GIODIN[A-H][7:0]) Field Descriptions

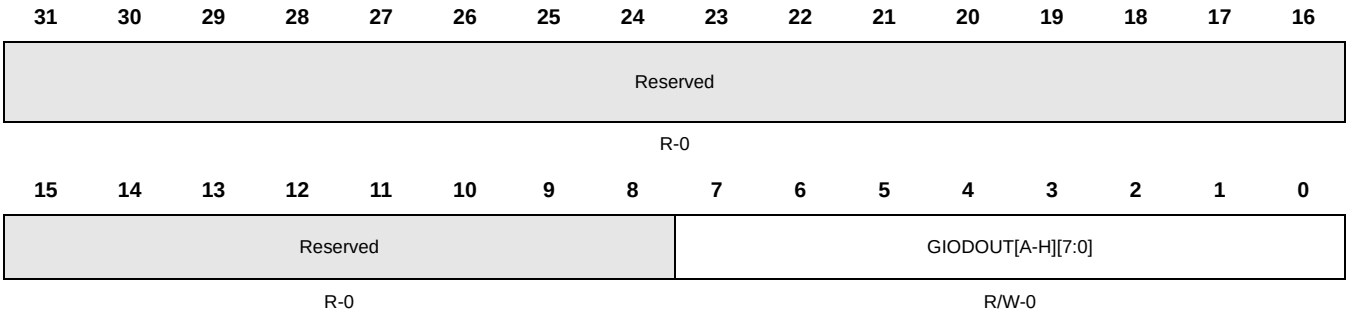
Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7-0	GIODIN[A-H][7:0]		GIO data input for ports [H-A], pins[7:0].
		0	The pin is at logic low (0); input voltage is V_{IL} or lower.
		1	The pin is at logic high (1); input voltage is V_{IH} or higher.

11.5.13 GIO Data Output Registers [A-H][7:0] (GIODOUT[A-H][7:0])

Values in the GIODOUT register specify the output state (high = 1 or low = 0) of the pins of the port when they are configured as outputs. Figure 11-24 and Table 11-19 describe this register.

Note:
Values in the GIODSET register, Section 11.5.14, set the data output control register bits to 1 regardless of the current value in the GIODOUT bits.

Figure 11-24. GIO Data Output Registers [A-H][7:0] (GIODOUT[A-H][7:0])



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

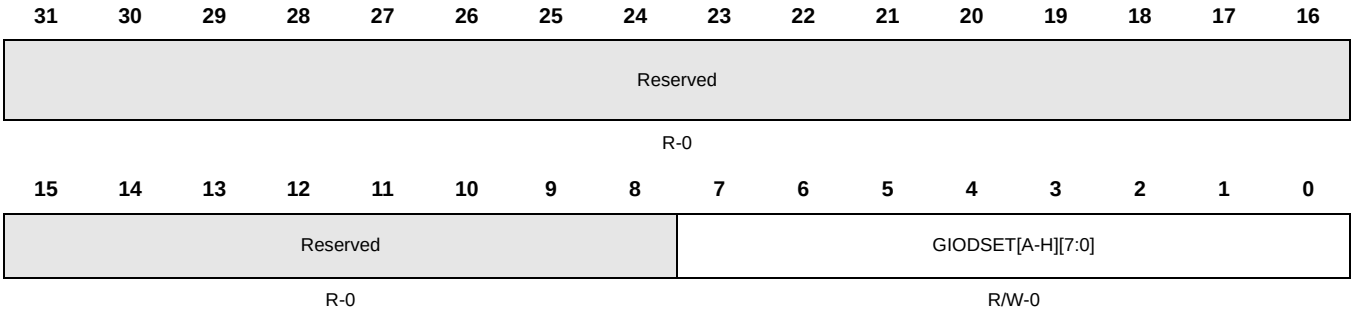
Table 11-19. GIO Data Output Registers [A-H][7:0] (GIODOUT[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7-0	GIODOUT[A-H][7:0]		GIO data output of ports [A-H], pins[7:0].
		0	The pin is at logic low (0); output voltage is V_{OL} or lower.
		1	The pin is at logic high (1)

11.5.14 GIO Data Set Register [A-H][7:0] (GIODSET[A-H][7:0])

Values in this register set the data output control register bits to 1 regardless of the current value in the GIODOUT bits. The contents of this register reflect the contents of GIODOUT. Figure 11-25 and Table 11-20 describe this register.

Figure 11-25. GIO Data Set Registers [A-H][7:0] (GIODSET[A-H][7:0])



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

.

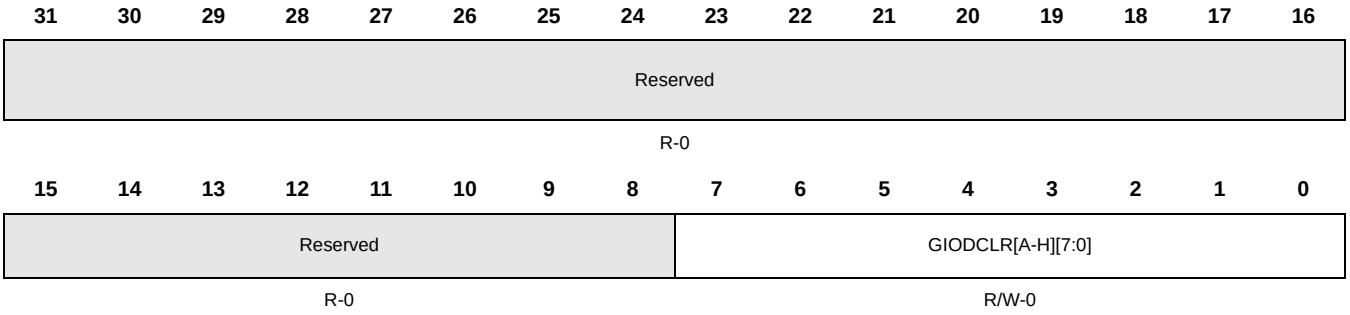
Table 11-20. GIO Data Set Registers [A-H][7:0] (GIODSET[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7-0	GIODSET[A-H][7:0]		GIO data set for ports [A-H], pins[7:0].
		0	*Read:* The pin is at logic low (0); output voltage is V_{OL} or lower. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The pin is at logic high (1); output voltage is V_{OH} or higher. *Write:* The corresponding bit in GIODOUT[7:0] is set to 1.

11.5.15 GIO Data Clear Registers [A-H][7:0] (GIODCLR[A-H][7:0])

Values in this register clear the data output register (Section 11.5.14) bit to 0 regardless of its current value. The contents of this register reflect the contents of GIODOUT. Figure 11-26 and Table 11-21 describe this register.

Figure 11-26. GIO Data Clear Registers [A-H][7:0] (GIODCLR[A-H][7:0])



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

Table 11-21. GIO Data Clear Registers [A-H][7:0] (GIODCLR[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7-0	GIODCLR[A-H][7:0]		GIO data clear for ports [A-H], pins[7:0].
		0	*Read:* The pin is at logic low (0); output voltage is V_{OL} or lower. *Write:* Writing a zero to this bit has no effect on the corresponding bit in GIODOUT[7:0].
		1	*Read:* The pin is at logic high (1); output voltage is V_{OH} or higher. *Write:* The corresponding bit in GIODOUT[7:0] is cleared to 0.

11.5.16 GIO Pull Disable Registers [A-H][7:0] (GIOPULDIS[A-H][7:0])

Values in this register enable or disable the pull control capability of the pins. Figure 11-27 and Table 11-22 describe this register.

Figure 11-27. GIO Pull Disable Registers [A-H][7:0] (GIOPULDIS[A-H][7:0])

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GIOPULDIS[A-H][7:0]							
R-0								R/W-x							

R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset depends upon the device

.

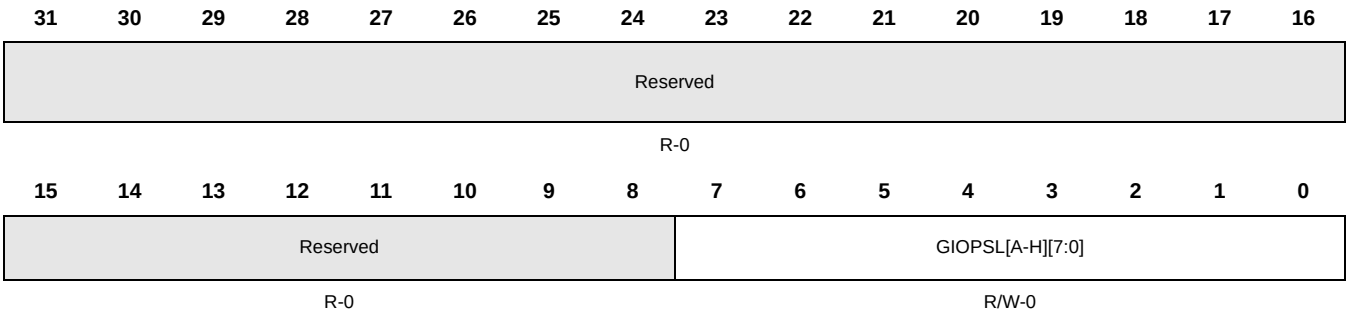
Table 11-22. GIO Pull Disable Registers [A-H][7:0] (GIOPULDIS[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	GIOPULDIS[A-H][7:0]		GIO pull disable for ports [A-H], pins[7:0]. Please refer to the device-specific datasheet for the reset state of these bits.
		0	The pull functionality is enabled.
		1	The pull functionality is disabled.

11.5.17 GIO Pull Select Register [A-H][7:0] (GIOPSL[A-H][7:0])

Values in this register will disable or enable the input buffer when the pull logic is disabled. Figure 11-28 and Table 11-23 describe this register.

Figure 11-28. GIO Pull Select Registers [A-H][7:0] (GIOPSL[A-H][7:0])



R = Read in all modes; W = Write in user and privileged modes; -n = Value after reset

Table 11-23. GIO Pull Select Registers [A-H][7:0] (GIOPSL[A-H][7:0]) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	GIOPSL[A-H][7:0]		GIO pull select for ports [A-H], pins[7:0].
		0	Input buffer for GIO is disabled if PULLDIS = 1.
		1	Input buffer for GIO is enabled if PULLDIS = 1.

11.6 Applications

The application examples in this section assume a typical configuration of five I/O functional ports, in which only Port A has three external interrupt-capable pins, 2:0. Table 11-24 illustrates the sample GIO register.

Table 11-24. Example GIO Register Set Showing Reserved Bits

Register	Address	7	6	5	4	3	2	1	0
GIOGCR	0x00								
GIOINTDET	0x08								
GIOPOL	0x0C								
GIOENASET	0x10								
GIOENACLR	0x14								
GIOLVLSET	0x18								
GIOLVLCLR	0x1C								
GIOFLG	0x20								
GIOOFFA	0x24								
GIOOFFB	0x28								
GIOEMUA	0x2C								
GIOEMUB	0x30								
GIODIRA	0x34								
GIODINA	0x38								
GIODOUTA	0x3C								
GIODSETA	0x40								
GIODCLRA	0x44								
GIOPULDISA	0x4C								
GIODIRB	0x54								
GIODINB	0x58								
GIODOUTB	0x5C								
GIODSETB	0x60								
GIODCLRB	0x64								
GIOPULDISB	0x6C								
GIODIRC	0x74								
GIODINC	0x78								
GIODOUTC	0x7C								
GIODSETC	0x80								
GIODCLRC	0x84								
GIOPULDISC	0x8C								
GIODIRD	0x94								
GIODIND	0x98								
GIODOUTD	0x9C								
GIODSETD	0xA0								
GIODCLRD	0xA4								
GIOPULDISD	0xAC								
GIODIRE	0xB4								
GIODINE	0xB8								

Table 11-24. Example GIO Register Set Showing Reserved Bits (Continued)

GIODOUTE	0xBC								
GIODSETE	0xC0								
GIODCLRE	0xC4								
GIOPULDISE	0xCC								
GIODIRF	0xD4								
GIODINF	0xD8								
GIODOUTF	0xDC								
GIODSETF	0xE0								
GIODCLRF	0xE4								
GIOPULDISF	0xEC								
GIODIRG	0xF4								
GIODING	0xF8								
GIODOUTG	0xFC								
GIODSETG	0x100								
GIODCLRG	0x104								
GIOPULDISG	0x10C								
GIODIRH	0x114								
GIODINH	0x118								
GIODOUTH	0x11C								
GIODSETH	0x120								
GIODCLRH	0x124								
GIOPULDISH	0x12C								

11.6.1 Example: Setting Interrupts and Configuring Pins for Output

The following code demonstrates how to set the interrupts. In the example, two of the interrupts are enabled, and the third pin of port A is configured for output. R2 keeps the same value, and the other registers act as temporary storage for addresses and values.

```
GIO_LOC     .word 0xFFF7BC00       ;device specific address for the GIO registers.
GIOENASET   .equ     0x0C            ;setting up equate statements
GIOPOL      .equ    0x08
GIOFLG      .equ    0x1C
GIOPRYSET   .equ    0x14
GIOPRYCLR   .equ    0x18
GIOOFFA     .equ    0x20
GIOEMUA     .equ    0x28
GIOOFFB     .equ    0x24
GIOEMUB     .equ    0x2C
GIODIRA     .equ    0x30
GIODINA     .equ    0x34
GIODOUTA    .equ    0x38
```

```
GIODSETA   .equ    0x3C
GIODCLRA   .equ    0x40


GIOPULDISA .equ    0x48


   LDR R2, GIO_LOC              ;loads GIO base address into register 2.
                                ; **SET THE POLARITY OF THE INTERRUPTS.**
   MOV R3, #GIOPOL              ; loads GIOPOL offset address into register 3.
   MOV R4, #0x02                ; loads a value of 0x02 into register 4.
                                ; Sets bit 1.
   STR R4, [R2, R3]             ; sets interrupt 0 to trigger on falling edge
                                ; and interrupt 1 to trigger on rising edge.
                                ; **SET THE PRIORITY ON BOTH INTERRUPTS AS LOW**
   MOV R3, #GIOPRYCLR           ; loads the GIOPRY address into register 3.
   MOV R4, #0x03                ; loads a value of 0 into register 4.
   STR R4, [R2, R3]             ; priority on interrupts 0 and 1 is set LOW.
                                ; ** SET PORT A FOR OUTPUT (EXCEPT FOR INTERRUPTS
                                ; WHICH MUST BE CONFIGURED AS INPUTS)**
   MOV R3, #GIODIRA             ; loads GIODIRA offset address into register 3.
   MOV R4, #0xFC                ; loads a binary value of 11111100
   STR R4, [R2, R3]             ; sets pins 0 and 1 as input to insure proper
                                ; interrupt behavior pin 2 configured as
                                ; output
   MOV R3, #GIOFLG              ; loads GIOFLG offset address into register 3.
   MOV R4, #0xFF                ; sets 32 bits.
   STR R4, [R2, R3]             ; clears all bits of the flag register.
 ; **ENABLE THE FIRST TWO INTERRUPTS 1:0.**
  MOV R3, #GIOENA ; loads GIOENA offset address into register 3.
  MOV R4, #0x03 ; sets first 2 bits 1:0
  STR R4, [R2, R3] ; enables the first two interrupts
```

11.6.2 Example: Toggling Output Buffers

The following code demonstrates how to toggle the output buffer port B using the GIODSET and GIODCLR buffers.

```
GIO_LOC    .word   0xFFF7BC00   ;device specific address for the GIO registers.
GIOENA     .equ    0x04       ;setting up equate statements
...
GIODIRB   .equ    0x58
GIODINB    .equ    0x5C
GIODOUTB   .equ    0x60
GIODSETB   .equ    0x64
GIODCLRB   .equ    0x68
```

```
   LDR R2, GIO_LOC              ; loads absolute address of the GIO memory
                                ; location (device specific).
                                ; **SET ALL BITS IN GIODOUTB AS 1.**
   MOV R3, #GIODSETB            ; loads offset address of GIODSETB
   MOV R4, #0xFF                ; loads a binary value of 11111111
   STR R4, [R2, R3]             ; sets all bits of GIODOUTB.
                                ; **CONFIGURE PORT 1 AS OUTPUT.**
   MOV R3, #GIODIRB             ; loads offset address of GIODIRB.
   MOV R4, #0xFF                ; loads a binary value of 11111111.
   STR R4, [R2, R3]             ; configures port 1 as output
                                ; **TOGGLE BITS 0, 2, 4, 6.**
   MOV R3, #GIODSETB            ; loads offset address of GIODSETB
   MOV R4, #GIODCLRB            ; loads GIODCLRB offset address
   MOV R5, #0x55                ; loads a binary value of 01010101
TOGGLE
   STR R5, [R2, R4]             ; clears GIODOUTB bits 0, 2, 4, 6
   STR R5, [R2, R3]             ; sets GIODOUTB bits 0, 2, 4, 6
     B TOGGLE        ; loops back to TOGGLE
```

11.6.3 Example: Clearing Interrupt Flags and Setting Interrupts

The following code demonstrates how interrupt flags should be cleared before interrupts are enabled. In this example, all three interrupt-capable pins are set as interrupts.

```
GIO_LOC     .word   0xFFF7BC00   ;device specific address for GIO registers
GIOENASET   .equ    0x0C       ;setting up equate statements
...
GIOFLG      .equ    0x1C



   LDR R2, GIO_LOC              ;loads absolute address of the GIO memory
   MOV R3, #GIOFLG              ;loads GIOFLG offset address into R3.
   MOV R4, #0x07                ;loads a value of 00000111 into R4.
   STR R4, [R2, R3]             ;clears the interrupt-capable bits of the
                                ;GIOFLG control register.

   MOV R3, #GIOENA              ;loads GIOENA offset address into R3.
   MOV R4, #0x07                ;loads 00000111 into R4.
   STR R4, [R2, R3]             ;enables pins 2:0 of port A as interrupts.
```

11.6.4 Example: Reading Port B Input Register

The following code demonstrates how to read the input register of port A when interrupts are enabled. Pin 0 is set as an interrupt, and pins 2 and 1 are configured as inputs.

```
GIO_LOC.word 0xFFF7BC00          ;device specific address for GIO registers
```

```
GIOENA .equ    0x04              ;setting up equate statements
...
GIODIRB .equ 0x58
GIODINB .equ 0x5C
GIODOUTB.equ 0x60
GIODSETB.equ 0x64
GIODCLRB .equ 0x68
   LDR R2, GIO_LOC               ;loads absolute address of the GIO memory
                                 ;**MASK OUTPUTS AND INTERRUPTS SO INPUT IS UNAMBIGUOUS.**
   MOV R3, #GIODINB              ;loads the offset address of GIODINB.
   LDR R4, [R2, R3]              ;loads the value in GIODINB register into R4.
   MOV R3, #0xFE                 ;loads 11111110 into R3. This value is used
                                 ;to mask the input so that only the input
                                 ;values are read. The 1's appear in the places
                                 ;where the input register is reading input
                                 ;voltages.
   AND R4, R4, R3                ;loads masked input value into R4.
```

High-End Timer (HET) Module

This section describes the high-end timer (HET). The HET is a software-controlled timer with a dedicated specialized timer micromachine and a set of 21 instructions. The HET micromachine is connected to a port of input/output (I/O) pins.

Topic **Page**

12.1 Features

The HET for the TMS470Px device family has the following features:

- High-level timer functions such as period and pulse width measurement
- 32 input/output (I/O) channels for timer functions such as capture, compare, pulse width measurements (PWMs), and general purpose I/O pins
- 24 high resolution (HR) hardware channels associated with 24 of the 32 I/O channels
- 8 loop resolution (LR) hardware channels associated with 8 of the 32 I/O channels
- User-programmable loop resolution and HR clocks
- User-programmable micromachine with a reduced instruction set (21 instructions) for build-time functions
- Multiple 20-bit virtual counters for timers, event counters, and angle counters
- Dual port RAM with capacity for 64 words of 96 bits (expandable to 256 words, see device-specific datasheet) to optimize the number of cycles per instruction (most instructions require one cycle)
- Shadow registers for CPU communication
- Minimal CPU code overhead and required interrupts
- 35 interrupt sources with two individually programmable levels
- HR I/Os and coarse resolutions implemented by sub loops for multiple resolution capability
- Conditional program execution based on pin conditions and compares
- Software breakpoint capability on each instruction
- Synchronously operates with VCLK2
- Possibility of using multiple HETs synchronized on the same resolution clock
- Digital and analog loopback mode

12.2 Overview

The HET is a third-generation Texas Instruments (TI) advanced intelligent timer module.

This HET module provides sophisticated timing functions for real-time applications such as car engine management. The new HR hardware channels allow greater accuracy for widely used timing functions such as period and pulse measurements, output compare, and PWMs.

The reduced instruction set, based mostly on very generic and comprehensive instructions, has improved the definition and development cycle time of an application and its derivatives. With the HET breakpoint feature combined with various stop capabilities, the TMSx70 HET is designed for easy software application debug.

12.2.1 Timer Module Structure and Execution

The timer consists of a specialized micromachine that operates a reduced instruction set at the same speed as VCLK2. Most of the data, arithmetic, and logical unit (ALU) are 25 bits wide. Two 20-bit registers and one 25-bit register are available to manipulate information such as time, event counts, and angle values. System performance is improved by a wide instruction format (96 bits) that allows the CPU to fetch the opcode and data in one VCLK2 cycle, thus increasing the speed at which data can be processed. The typical operations performed in the ALU are additions (count), compares, and magnitude compares (higher or same).

Each instruction includes an 8-bit field for specifying the address of the next instruction to be executed. This means that an application program sequence is not controlled by a program counter (PC), but by the actual content of each instruction. This arrangement offers greater flexibility in going back and forth in the memory during program execution.

The interface to the host CPU is based on both communication memory and control registers. The communication memory includes timer instructions (program and data). This memory is typically initialized by the CPU after reset before the timer starts execution. Once the timer program is loaded into the memory, the CPU starts the timer execution, and only data parameters can then be read or written into the timer memory. The control registers include bits for selecting timer clock, configuring I/O pins, and controlling the timer module.

The programmer implements timer functions by combining instructions in specific sequences. For instance, a single count instruction sequence implements a timer. A PWM would need a two-instruction sequence: count and compare. A complex time function may include many instructions in the sequence. The total timer program is a set of time functions executed sequentially, one after the other. Reaching the end, the program must roll to the first function so that it behaves as a loop.

The time for a loop to execute is referred to as a *loop resolution clock cycle*. When the HET rolls over to the first function (i.e., instruction), the timer waits for the resolution clock to restart the execution of the loop to ensure that only one loop is executed for each loop resolution clock cycle.

The execution time of this main loop is the sum of all the cycles required for the instructions that the loop executes. The main loop must be completed within the loop resolution clock. Therefore, you must make sure that the timer functions implemented in the algorithm are executed within the resolution clock. Otherwise, the program will behave unpredictably because some instructions will not be executed each time through the loop.

This requirement creates a strong link between the accuracy of the timer functions and the number of functions (the number of instructions) the timer can perform. To overcome this limitation, some of the most commonly used instructions can access the HR hardware. This access allows pulse or period measurements, time compare, and PWM output waveforms at the resolution of the HR clock instead of the loop resolution clock.

Updating parameters of compare instructions (time, event, or angle compare values) can be controlled by the timer itself using built-in move instructions. These built-in move instructions actually transfer new data from the CPU into the active compare field of an instruction synchronously with the resolution clock. This synchronization method makes it unnecessary to use interrupts to avoid such problems as incorrect pulse widths.

12.2.2 Major Advantages

In addition to classic time functions such as input capture or multiple independent PWMs, higher-level time functions can be easily implemented in the timer program main loop. Higher-level time functions include angle driven wave forms, angle and time-driven pulses, and input pulse width modulation (PWM) duty cycle measurement.

Because of these high-level functions, data exchanges with the CPU are limited to the fundamental parameters of the application (periods, pulse widths, angle values, etc.); and the real-time constraints for parameter communication are dramatically minimized; for example, few interrupts are required and asynchronous parameter updates are allowed.

The reduced instruction set and simple execution flow control make it simple and easy to develop and modify programs. Simple algorithms can embed all the flow control inside the HET program itself. More complex algorithms can take advantage of the CPU access to the HET RAM. With this, the CPU program can make calculations and can modify the timer program flow by changing the data and control fields of the HET RAM. CPU access to the HET RAM also improves the debug and development of timer programs. The CPU program can stop the HET and view the contents of the program, control, and data fields that reside in the HET RAM.

Finally, the modular structure provides maximum flexibility to address a wide range of applications. The timer resolution can be selected from two cascaded prescalers to adjust the loop resolution and HR clocks. The 32 I/O pins can provide any combination of input, period or pulse capture, and output compare, including 24 HR channels. The standard memory structure allows module configuration from 64 words to 256 words of timer program memory.

12.2.3 Performance

Most instructions execute in one cycle, but a few take two or three cycles. The average cycle per instruction (CPI) measured on various complex benchmarks is approximately 1.3.

The HET can generate many complex output waveforms without CPU interrupts. Where special algorithms are needed following a specific event (e.g., missing teeth or a short/long input signal), a minimal number of interrupts to the CPU are needed. The minimal interrupts frees the CPU bandwidth to perform other tasks.

12.2.4 Instructions Features

The TMS470Px HET has the following instructions features:

- All standard resolution outputs are synchronized to the loop resolution clock to avoid the variability associated with the instruction placement in the HET program.
- HET uses a RISC-based specialized timer micromachine to carry out a set of 21 instructions.
- Instructions implemented in a VLIW format (96 bits wide).
- The HET program execution is self-driven by external or internal events, branching to special routines based on input edges or output compares.

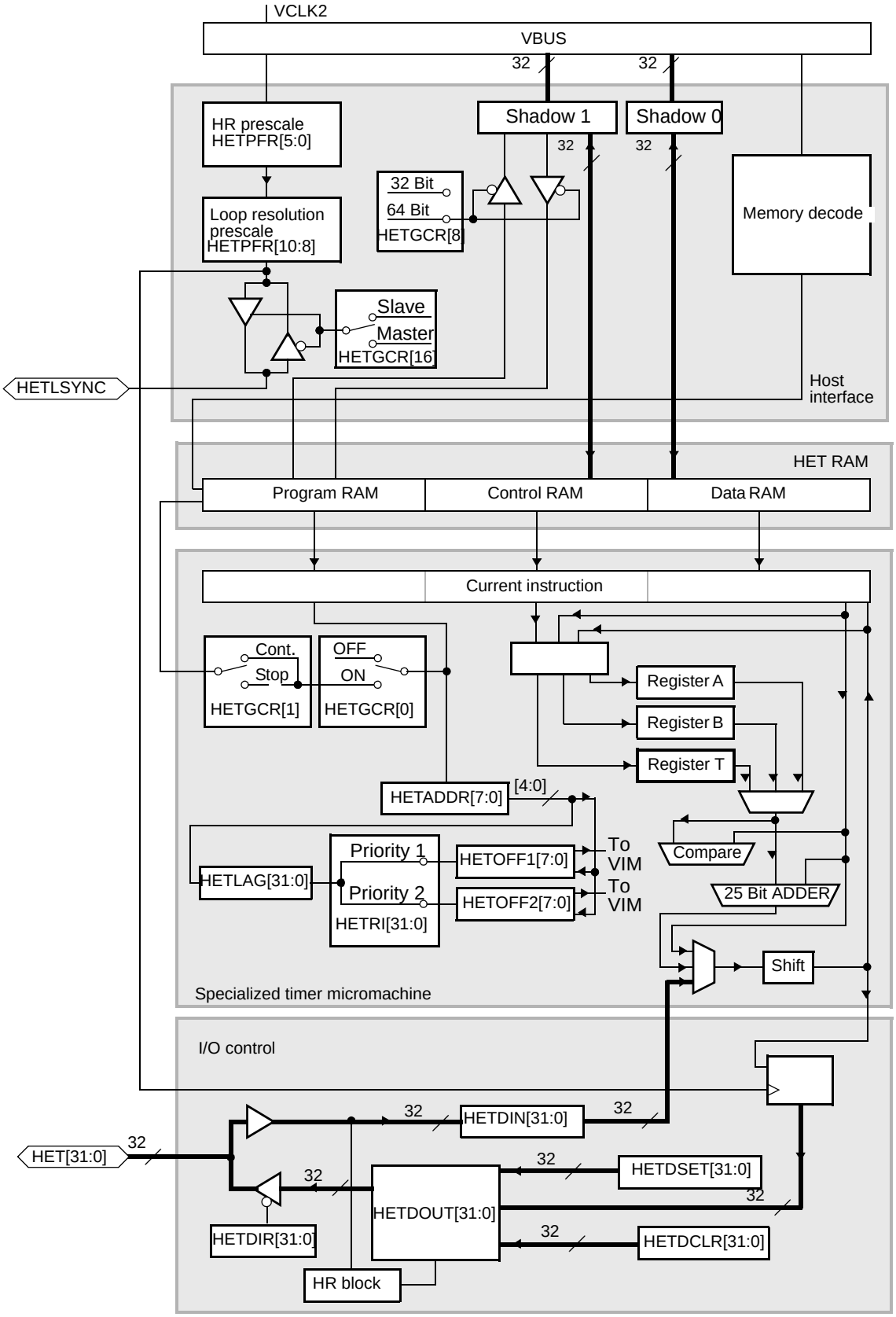
12.2.5 Block Diagram

The HET module (see Figure 12-1) comprises four separate components:

1. Host interface
2. HET RAM
3. Specialized timer micromachine
4. I/O control

The HET is attached to an I/O port of up to 32 pins. Please see the device-specific data sheet for details on the number of HET pins available.

Figure 12-1. HET Block Diagram



12.3 HET Functional Description

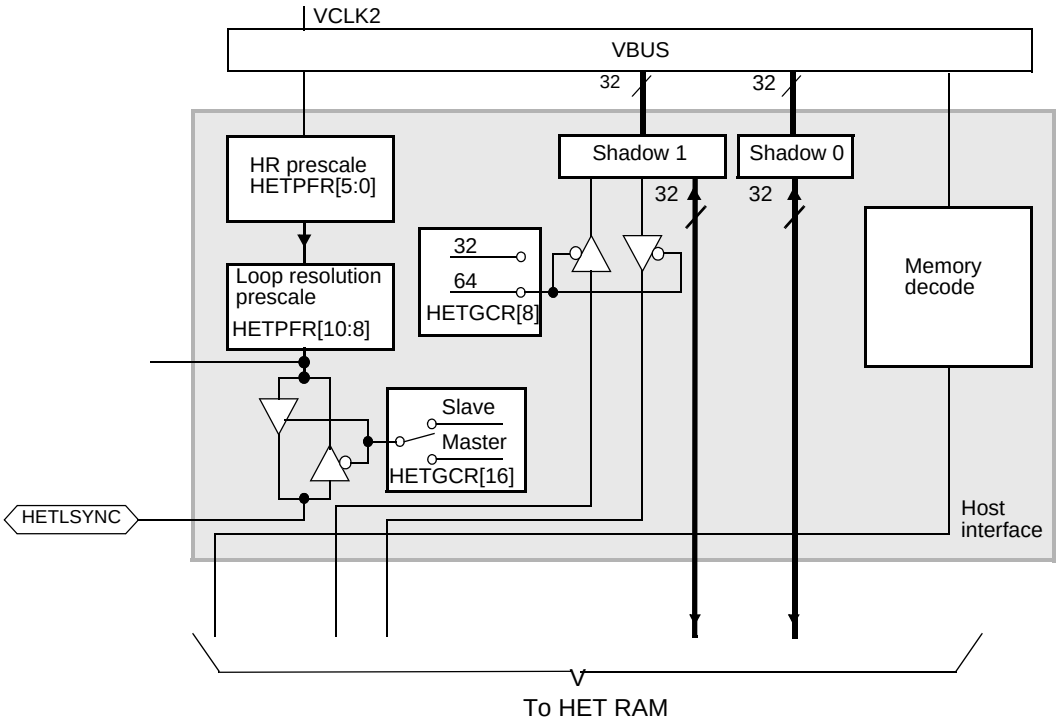
The HET contains a host interface to the CPU and RAM into which HET code is loaded. The HET code is executed by the specialized timer micromachine. The I/O control provides an interface to external pins respectively.

12.3.1 Host Interface

The host interface controls all communications between timer-ram and CPU (see Figure 12-2). It includes the following components:

- Two 32-bit shadow registers (shadow register 0 and 1)
- A bit (HETGCR[8]; Section 12.5.1) for controlling how the shadow registers are written/read by the CPU
- Two user programmable 6-bit(HETPFR[5:0]; Section 12.5.2) and 3-bit (HETPFR[10:8]) clock pre scalers for HR and loop resolution clocks
- A bit (HETGCR[16]; Section 12.5.1) for controlling whether the HET is configured as a master or a slave

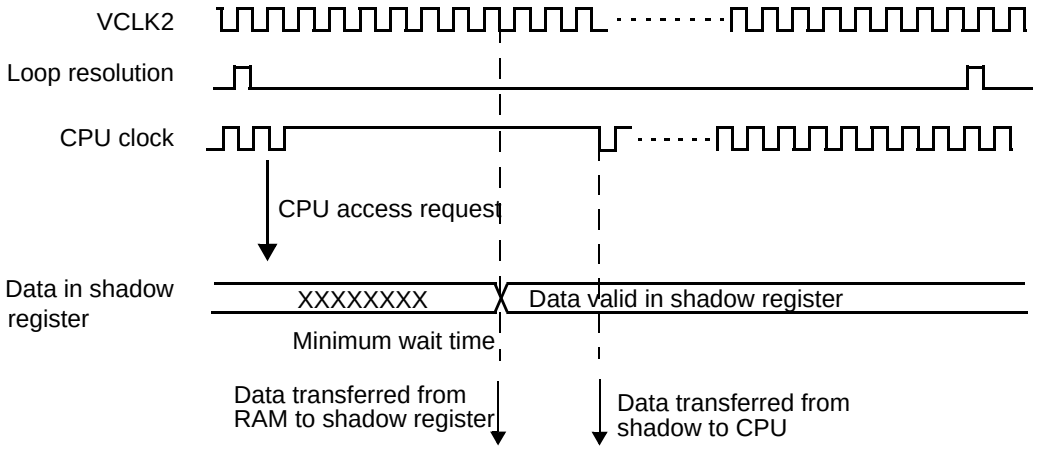
Figure 12-2. Host Interface



12.3.1.1 Memory Control

Each CPU access to the HET RAM will take seven VCLK2 cycles in addition to the cycles needed on the peripheral bus, which will be dependent on the HCLK to VCLK ratio programmed in the device.

Figure 12-3. HET RAM Accesses (Example of a 10-Wait Cycle Read Operation)



Each time the CPU performs an access to the HET RAM, one or two cycles of the HET program are dedicated to the CPU access since the CPU and the HET cannot access the HET RAM at the same time. A 32-bit read takes one cycle and a 64-bit takes two cycles. One of the most common uses of a 64-bit read is for the PCNT instruction. The 64-bit read gives you the control field (previous period) and data field (current period) at the same instant in time. The total number of available time slots for the HET must be computed by subtracting the time slots consumed by the CPU from the number of cycles available in a loop resolution. In the worst case of continual CPU access to the HET, the HET loses one time slot (or two cycles in case of auto read/clear of PCNT) out of every eight.

Example:

An example HET configuration allows 24 time slots in a loop resolution (by the prescalers). The memory controller is programmed with seven wait-state to the HET. In the case of continuous CPU access, three cycles per resolution will be dedicated to the CPU, leaving 21 time slots really available for the HET application program.

12.3.1.2 Shadow Registers

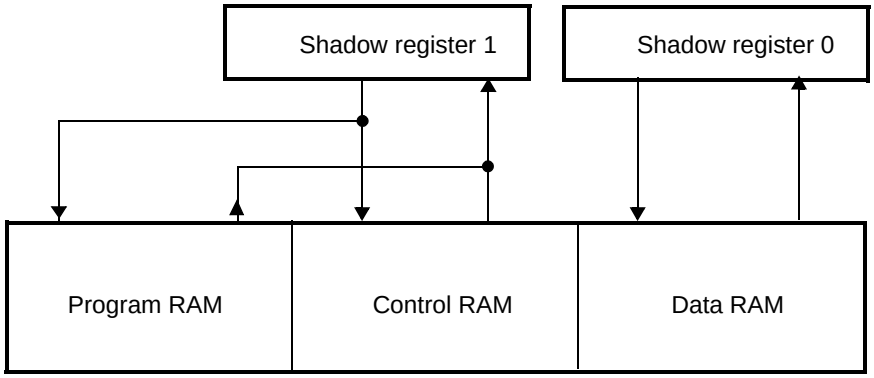
The two shadow registers allow the CPU to access the program, control, or data field in the timer-RAM without interrupting execution of a timer instruction. Shadow register operations are transparent to you. See Figure 12-4.

Shadow registers allow the timer instructions to access the timer control field and data field 64 bits at a time. To be active, this 64-bit mode must be set in the global configuration register.

The timer-RAM cannot be accessed by byte or half-word (16-bit data).

- Shadow register 0 is assigned to data field access.
- Shadow register 1 shares program field and control field accesses.

Figure 12-4. Shadow Registers with RAM Units



12.3.1.3 CPU Access to Timer-RAM

- 32-bit access (HETGCR[8] = 0; Section 12.5.1) (program, control and data field)
  - **Write**: When the CPU executes STR, it first writes 32-bit CPU data into relevant shadow registers. The HET asserts then 7 additional VCLK2 wait cycles to the R2V. The CPU resumes its execution due to the write buffer implemented in the R2V. Should another HET access occur while the waitstates are active, the CPU will be put into waitstate for the remaining wait cycles. At the end of the current timer instruction, the shadowed data moves into timer selected RAM.
  - **Read**: When the CPU executes LDR, the HET asserts then 7 additional VCLK2 wait cycles to the CPU. At the end of the current instruction, the 32-bit timer data are loaded into relevant shadow registers. At the end of the wait time, the shadowed data moves onto the data bus.
- 64-bit access (HETGCR[8] = 1; Section 12.5.1)
  - **Write**: For a 64-bit write, two STR CPU instructions are required. The control field must be written first, followed by the data field. For the first STR instruction, the CPU writes 32-bit CPU data into shadow register 1. The HET asserts then 7 additional VCLK2 wait cycles to the R2V, but the CPU can still remain executing the next instructions. For the second STR instruction, the CPU writes 32-bit CPU data into shadow register 0. The CPU will be put into waitstates, should the previous wait cycles still be active. The HET asserts then 7 additional VCLK2 wait cycles to the CPU for the second access. At the end of the current timer instruction, all 64-bit shadowed data moves into timer RAM.
  - **Read**: For a 64-bit read, two LDR CPU instructions are executed. The control field must be read first, followed by the data field. For the first LDR instruction, the HET asserts then 7 additional VCLK2 wait cycles to the CPU. At the end of the current timer instruction, the timer loads 32-bit control field into shadow register 1 and the 32-bit data field into shadow register 0 (both in the same cycle). At the end of the wait time, the shadowed data from the control field move onto the data bus. For the second LDR instruction, the HET asserts then 7 additional VCLK2 wait cycles to the CPU. The CPU then moves the 32-bit of shadow register 0 (loaded from the data field at the first LDR) onto the data bus.

Note:
Interrupts:
During the time the 64-bit access bit is set, it is recommended to disable all interrupts.
Background**:**
- Avoid an interrupt routine that causes a delay between the first and the second LDR instruction.
- Disabling the interrupts is a must if an interrupt routine also reads the HET RAM locations.

- Automatic read/clear of the HET RAM data field

The HET provides a feature allowing to automatically clear the data field after a 64-bit read operation of the control and data fields. This feature is implemented via the control bit, which is located in the control field (bit C21). This is a static bit that can be used by any instruction.

1. Set the control bit in the selected instruction.
2. Set the 64-bit access bit in the global configuration register.
3. Perform a 64-bit read access with the CPU.
4. Loads shadow registers 1 and 0 with the control and data fields of the memory, and then clears the data field [D24:D0] of the instruction. This sequence requires two time slots.

This feature is mainly used for PCNT instruction. You read a period or pulse value, which is then cleared by the HET itself. This means that you do not need to take a new value as long as the period or pulse data is still cleared.

12.3.1.4 Memory Selects

The start address of the HET RAM is device dependent. Please see the device specific datasheet for further information.

12.3.1.5 Emulation Mode

Emulation mode, used by the software debugger, is specified in the global configuration register. When the debugger hits a breakpoint, the CPU sends a suspend signal to the modules. Two modes of operation are provided: suspend and ignore suspend.

- Suspend

  When a suspend is issued, the timer operation stops at the end of the current timer instruction. However, the CPU accesses to the timer RAM or control registers are freely executed.

- Ignore suspend

  The timer RAM ignores the suspend signal and operates real time as normal. Wait states asserted by the timer while the suspend signal is active are executed as normal.

12.3.1.6 Power-Down

The HET can be put into either local or global low power mode. Global low power mode is asserted by the system and is not controlled by the HET. During global low power mode, all clocks to the HET are turned off so the module is completely inactive.

Local low power mode can be asserted in two ways. One possibility is by setting the POWERDOWN (HETGCR [24]; Section 12.5.1) bit; setting this bit stops the clocks to the HET internal logic (state machine), but the HET registers continue to be clocked. Another way to stop the clock to the HET state machine and the registers is to put the peripheral select the HET is connected to into power down mode (write 1 to corresponding PSPWRDWNSETx register bit).

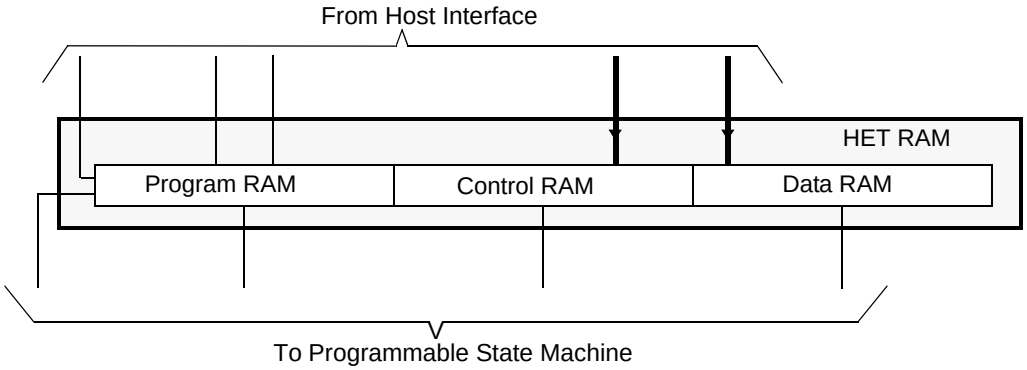
Exiting local power mode can be achieved by resetting to 0 the POWERDOWN (HETGCR [24]; Section 12.5.1) bit, to enable the clocks of the HET internal logic (state machine) or by writing a 1 to the appropriate PSPWRDWNCLRx registers (Section 12.5.1) if this scheme was used for putting the module into power down mode.

12.3.2 HET RAM

The timer RAM uses dual port access. This means that one RAM address may be written while another address is read. This feature is especially useful for the HET architecture because the prefetch is done during the last cycle of each instruction and the timer RAM allows writing back data into the current instruction while reading the next instruction. The RAM words are 96 bits wide, which are split into three 32-bit fields (program, control, and data) to fit with the CPU data bus. See Figure 12-5.

Note: RAM
The RAM supports word accesses only.

Figure 12-5. HET RAM



12.3.2.1 Memory Map

Table 12-1 describes the HET memory map.

Table 12-1. HET Memory Map

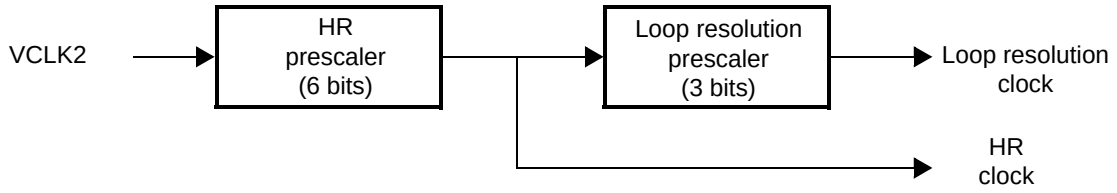
Instruction Address	Program Field Address	Control Field Address	Data Field Address	Reserved Address
00h	XX000h	XX004h	XX008h	XX00Ch
01h	XX010h	XX014h	XX018h	XX01Ch
02h	XX020h	XX024h	XX028h	XX02Ch
:	:	:	:	:
:	:	:	:	:
3Fh	XX3F0h	XX3F4h	XX3F8h	XX3FCh
40h	XX400h	XX404h	XX408h	XX40Ch
:	:	:	:	:
FFh	XXFF0h	XXFF4h	XXFF8h	XXFFCh

12.3.3 Time Base

Two prescalers generate the timer resolution clock for the program loop and the HR clock for the HR I/O counters. See Figure 12-6. The prescalers consist of the following:

- A 6-bit prescaler dividing VCLK2 by a user-defined HR prescale divide rate (hr) stored in the 6-bit HR prescale factor code (with a linear increment of codes). See Table 12-2.
- A 3-bit prescaler dividing the HR clock by a user-defined loop-resolution prescale divide rate (lr) stored in the 3-bit loop-resolution prescale factor code (with a power of 2 increment of codes). See Table 12-3.

Figure 12-6. Prescaler Configuration



The LR is typically determined by the number of cycles that the HET requires to complete the worst-case application software loop plus CPU accesses. Therefore, you must choose the prescaler numbers so that the number of time slots in any software loop is greater than the number of cycles required in a loop. Typically, you choose the smallest HR prescaler number that will still allow the appropriate number of cycles.

- Time slots available (Ts)

  Ts = [HR prescale divide rate (Hr) * loop resolution prescale divide rate (Lr)] -1

- HR clock period (HRP)

  *HRP = Hr * VCLK2 period* (Tclk) = Hr / VCLK2

- Loop resolution clock period (LRP)

  LRP = loop resolution prescale divide rate (LR) * HR clock period (HRP) = Lr * Hr *Tclk = (Hr * Lr)/ VCLK2

Table 12-2. HR Prescale Factor Codes (Continued)

HR Prescale Factor Code						HR Prescale Divide Rate
5	**4**	**3**	**2**	**1**	**0**	
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
:	:	:	:	:	:	:
:	:	:	:	:	:	:
1	1	1	1	0	1	62
1	1	1	1	1	0	63
1	1	1	1	1	1	64

Table 12-3. Loop Resolution Prescale Factor Codes

Loop Resolution Prescale Factor Code				Loop-Resolution Prescale Divide Rate
ordinal	**2**	**1**	**0**	
0	0	0	0	1
1	0	0	1	2
2	0	1	0	4
3	0	1	1	8
4	1	0	0	16

Loop Resolution Prescale Factor Code				Loop-Resolution Prescale Divide Rate
ordinal	2	1	0	
5	1	0	1	32
6	1	1	0	reserved
7	1	1	1	reserved

Note:
When the loop-resolution prescale divide rate is smaller than 32, the non-relevant bits of the HR data fields (non-relevant LSBs) will be one of the following; see also Table 12-4:

•Written as 0 for HR capture

•Or interpreted as 0 for HR compare

For more information about this, see the descriptions of the ECMP (Section 12.6.3.10), MCMP (Section 12.6.3.12), PCNT (Section 12.6.3.15), PWCNT (Section 12.6.3.16), and WCAP (Section 12.6.3.21) instructions.

Table 12-4. Interpretation of the 5-Bit HR Data Field

Loop Resolution Prescale divide rate (Lr)	Bits of the HR data field(1)					HRP Cycles delay range
	• B[4]	• B[3]	• B[2]	• B[1]	• B[0]	
1	X	X	X	X	X	0
2	V	X	X	X	X	0 to 1
4	V	V	X	X	X	0 to 3
8	V	V	V	X	X	0 to 7
16	V	V	V	V	X	0 to 15
32	V	V	V	V	V	0 to 31
Weight factor as fraction of HR Cycles in one loop	1/2	1/4	1/8	1/16	1/32	

1 V = Valid bit; X = Non-relevant bit (ignored)

Example:

HETPFR[31:0] register = 0x300 → Lr prescale divide rate = Lr = 8 (→ 8 Hr cycles in one loop).

Assumption: HR data field = 0x14 = 10100b

Lr = 8 → Bits B[1] and B[0] are ignored → Hr delay = 101b = 5 Hr Cycles

or by using the calculation with weight factors:

Hr Delay = Lr * (B[4] * 1/2 + B[3] * 1/4 + B[2] * 1/8 + B[1] * 1/16 + B[0] * 1/32)
= 8 * (1 * 1/2 + 0 * 1/4 + 1 * 1/8)
= 5 Hr cycles

12.3.3.1 Determining Loop Resolution

As an example, consider an application that requires HR I/Os with a resolution of 62.5 ns, and 8 standard IOs at 2 μs, and needs 60 time slots for HET application programs.

Considering a VCLK2 frequency of 32 MHz, the following divide-by rates can be programmed using the following prescaler factor codes:

- Divide-by rate of 2 for the HR clock, giving 62.5 ns (code = 000001 = 0x1):
  HRP = Hr / VCLK2 = 2 /32 Mhz = 62.5 ns
- Divide-by rate of 32 for the loop resolution clock, giving 2 μs (code = 101 = 0x5)
  LRP = Lr * HRP = 32 * 62.5 ns = 2 us
- Number of available time slots: Ts = (Hr * Lr) -1 = (2 * 32)-1 = 63
- Hr =2, Lr = 32. Then, the corresponding code inside the HETPFR[31:0] register (Section 12.5.2) will be: 0x00000501

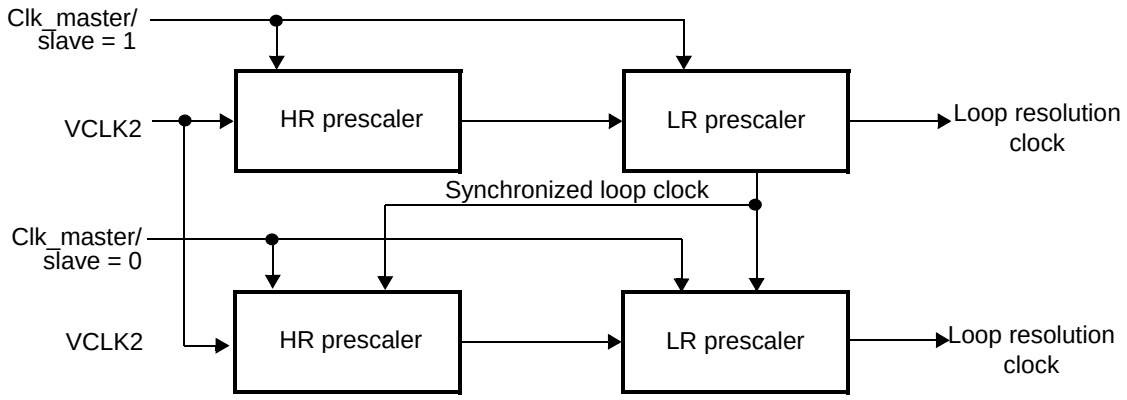
If in the example above, the resolution needed moves from 2 μs to 1 μs, then only 31 time slots will be available.

Or, if the number of time slots needed are 90, HRP then moves to 93.75 ns.

12.3.3.2 Multi-HETs Resolution Synchronization

In some applications configured with multiple HETs, the resolutions must be synchronized. This is typically the case for configurations that use a unique angle reference shared by several HETs. See Figure 12-7.

Figure 12-7. Multi-HETs Configuration—Synchronization of Resolutions



Note: Synchronization
The loop clock signal is two VCLK2 cycles ahead of the resolution clock.

The HET provides a synchronization mechanism for multiple timers. The Clk_master/slave (HETGCR[16]; Section 12.5.1) configures the HET in master or slave mode (default is slave mode). A HET in master mode provides a signal to synchronize the prescalers of the slave HET. The slave HET synchronizes its loop resolution to the loop resolution signal sent by the master. The slave does not require this signal after it receives the first synchronization signal. However, any time the slave receives the resynchronization signal from the master, the slave must resynchronize.

Follow these steps to synchronize multiple HETs.

1. Select one HET as master by writing in the clk_master/slave bit.
2. Configure the HR and LR prescaler control registers and all other control registers (direction, interrupts, etc.) of the master HET.
3. Configure the HR and LR prescaler control registers and all other control registers (direction, interrupts, etc.) of the slave HET. The loop resolution of the master **must be the same** as the slave's loop resolution.
4. Turn on all slave HETs. They will wait for the synchronization from the master HET to start the program execution.
5. Turn on the master HET.

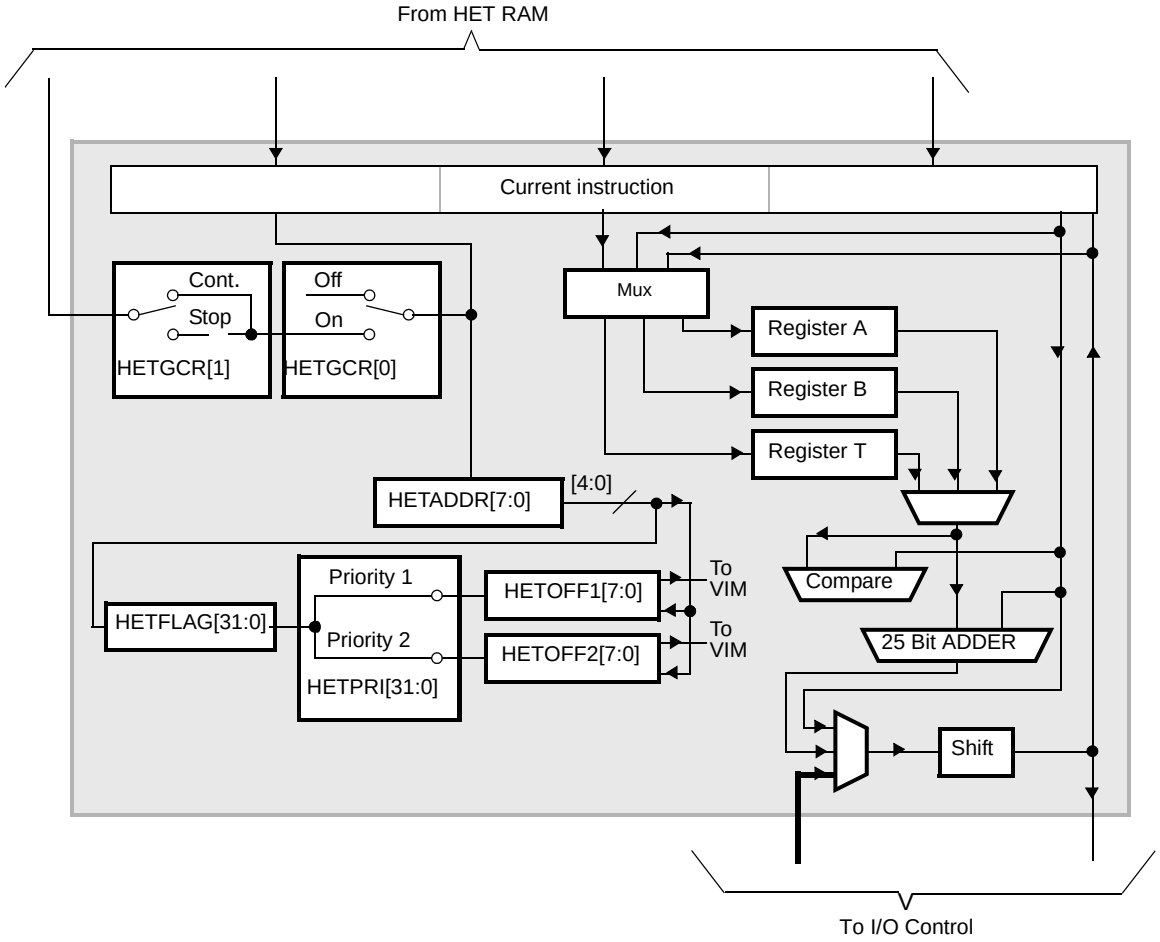
12.3.4 Specialized Timer Micromachine

The HET has its own instruction set, detailed in Section 12.5.6. The timer micromachine reads each instruction from the HET RAM. The program and control fields contain the instructions for how the specialized timer micromachine executes the command. For most instructions, the data field stores the information that needs to be manipulated.

The specialized timer micromachine executes the instructions stored in the HET RAM sequentially. The HET program execution is self-driven by external or internal events. This means that input edges or output compares may force the program to branch to special routines using an indexed addressing mode.

Figure 12-8 shows the major operations that the HET can carry out: compares, captures, angle functions, additions, and shifts. The HET is also capable of special operations, which the user can construct using the major operations. The HET contains three registers (A, B, and T) used to hold compare or counter values to be used by the HET instructions. Data may be taken from the registers or the data field for manipulation; likewise, the data may be returned to the registers or the data field.

Figure 12-8. Specialized Timer Micromachine



12.3.4.1 Time Slots and Resolution Loop

Each instruction requires a specific number of cycles or time slots to execute. The resolution specified in the prescaler registers determines the timer accuracy. All input captures, event counts, and output compares are executed once in each resolution loop. HR captures and compares are possible (up to VCLK2 accuracy) on the HR I/O pins. For more information, see Section 12.3.5.

12.3.4.2 Program Loop Time

The program loop time is the sum of all cycles used for instruction execution and CPU accesses. This time may vary from one loop to another depending on the number of CPU accesses or special routine execution.

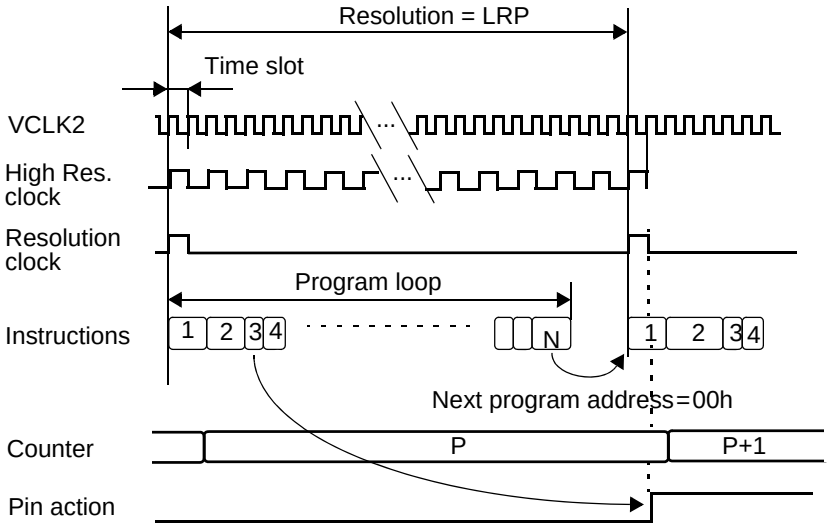
The timer program restarts on every resolution loop. The start address is fixed at RAM **address 00h**. The longest loop in a program must fit within one resolution loop time to guarantee complete accuracy.

The last instruction of a program points to the start address, (next program address= 00h).

If a program loop is shorter than the selected resolution, the program waits until end of resolution before starting the next loop.

The timing diagram (see Figure 12-9) illustrates the program flow execution.

Figure 12-9. Program Flow Timings

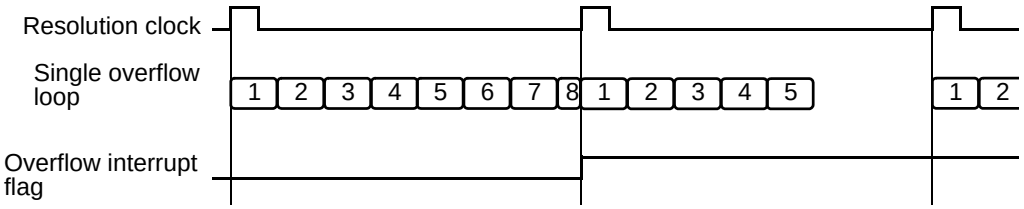


12.3.4.3 Program Overflow

If the number of time slots used in a program loop exceeds the number available in one resolution, the timer sets the program overflow interrupt flag located in the exceptions interrupt control register. To maintain synchronization of the I/Os, this condition should never occur in a normal operation. During the software debug phase, this flag may be used to debug the programmed loop resolution value. See Figure 12-10.

In case of program overflow, the HET sequence is started again at address 00h.

Figure 12-10. Use of the Overflow Interrupt Flag



Note: Limitation on instruction falling on a program overflow

The pin action will not be taken if the instruction that caused the pin action falls on a program overflow. All other instructions such as updating the A, B, and T registers and the RAM will still be performed.

12.3.4.4 Instruction Execution Sequence

The execution of a HET program begins with the first occurrence of the loop resolution clock after the HET is turned on. At the first and subsequent occurrences of the loop resolution, the instruction at location address 00h is prefetched. The program execution begins at the occurrence of the loop resolution clock and continues executing the instructions until the program branches to 00h location. The instruction is prefetched at location 00h and execution flag is reset. The HET goes into a wait state until the occurrence of the loop resolution clock and resumes normal execution. If a IDLE state is asserted (i.e., CPU access) at the beginning of the program, (at the occurrence of the loop resolution clock), the execution of the program is postponed by one or two VCLK2 cycles.

Single and multi-cycle instructions prefetch the next instruction in the last cycle of the instruction. SCNT, the only three cycles instruction (see Section 12.6.3.19), fetches the next instruction always in the third cycle, which is the last cycle of the instruction. ECMP, as other single cycle instruction (see Section 12.6.3.10), fetches the next instruction while executing the current instruction. By prefetching the instruction at the next or conditional address, the instruction is latched by the instruction registers and decoded in the same cycle.

The use of a dual-ported RAM permits dual access to the instruction memory. A single read and a single write can be performed in the VCLK2 cycle. In fact, a read and write to the same location can also be performed in the same VCLK2 cycle. The write operation is performed, regardless of the prefetching of instruction, in the cycle required by the instruction. Single cycle instruction can perform the following actions in one cycle:

1. Latch the current instruction.
2. Decode and execute the latched instruction.
3. Perform a write instruction to the instruction RAM.
4. Prefetch the instruction at the next or conditional address.

Multi-cycle instructions perform step 1 and 2 in the first cycle. Step 3 is executed in the cycle required by the instruction. Some instructions will perform multiple writes to the RAM, although in different cycles. Step 4 above is always performed in the last cycle of the instruction.

The number of instructions executed by a program must be between two loop resolution clocks. The program executes its first instruction in the high phase of the loop resolution clock and stops the execution of the program before the occurrence of the LClock_Last. If the total time of execution for the program exceeds the limit, the program overflow flag is asserted and the execution of the program is stopped. The execution of the program is resumed by prefetching the instruction at location address 00h.

Restrictions

Note:
The size of a HET program should be greater than one instruction. If there is only one instruction, this instruction will be executed twice.

Note:
Any MOVE instruction (see Section 12.6.3.13 and Section 12.6.3.14) that modifies a field in the next instruction to be executed will incur a wait cycle.

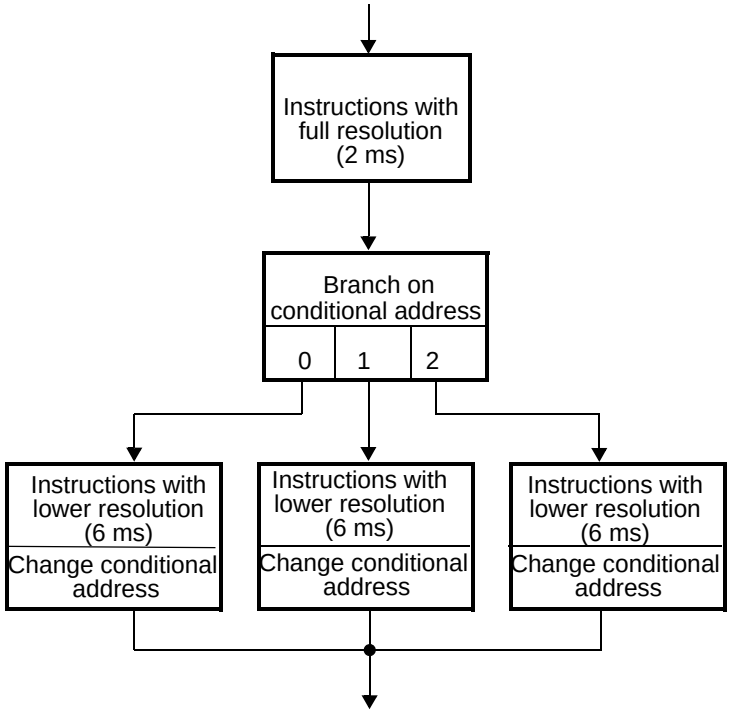
12.3.4.5 Multi-Resolution Scheme

When full resolution is not required, special program sequences with lower resolution can be used to optimize the use of time slots.

A lower-resolution sequence consists of instructions that are not executed on every resolution loop but only on every *n* resolutions. Unconditional branch instructions and an index sequence, using a MOV64 instruction (see Section 12.6.3.14) in each low resolution loop, is required to control this particular program flow. See Figure 12-11.

Note:
HR instructions must be placed in the main (full resolution) loop to ensure proper operation.

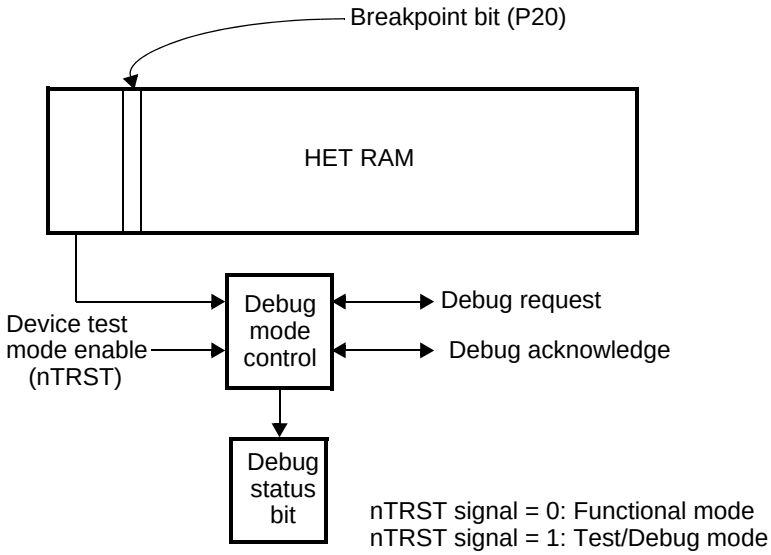
Figure 12-11. Multi-Resolution Operation Flow



12.3.4.6 Debug Capability

The HET has built-in debug capability in its architecture that allows you to more easily debug your HET program. See Figure 12-12.

Figure 12-12. Debug Control Configuration



This debug capability uses the breakpoint bit that is included as part of each HET instruction (bit P20 of any instruction). This bit can be programmed to a 1 to cause the HET program to freeze after the current instruction is executed. This action will also cause all internal HET state machines to freeze and send a debug request to the CPU. During the HET debug (i.e., HET breakpoint reached), you may still access the contents of all HET control registers. This includes a register that contains the current HET address. This allows you to determine which instruction caused the breakpoint condition. If the breakpoint bit is then programmed to a 0, the instruction will then be executed and resume program operation.

When the device test mode is enabled and a breakpoint is reached in the HET program, the debug status bit (HETGCR[2]) will be set and the debug request signal will be sent to the CPU. The CPU will wait for the CPU instruction boundary and then send the debug acknowledge signal back to the HET. At this time the debugger may clear the debug status bit by writing a 1 to clear the bit in the HET peripheral frame. This will allow the HET to exit its debug mode when the debug acknowledge signal is deasserted by the CPU.

For example, an equality compare (ECMP) instruction could be used to branch to an instruction with the breakpoint bit set when the compare is equal. A software capture word (WCAP) instruction could be used to branch to an instruction with the breakpoint bit set when a certain external pin condition exists. When a breakpoint instruction is executed, the HET freezes and the CPU enters debug mode.

The HET also supports the direct assertion of debug mode from the CPU. If the CPU goes into debug mode, it will signal this to the HET. If the ignore suspend bit is not set, the HET freezes all state machines and allows CPU accesses just as it does during the HET-initiated debug mode described above. If the ignore suspend bit is set, the HET will resume its execution, although the CPU is in debug mode.

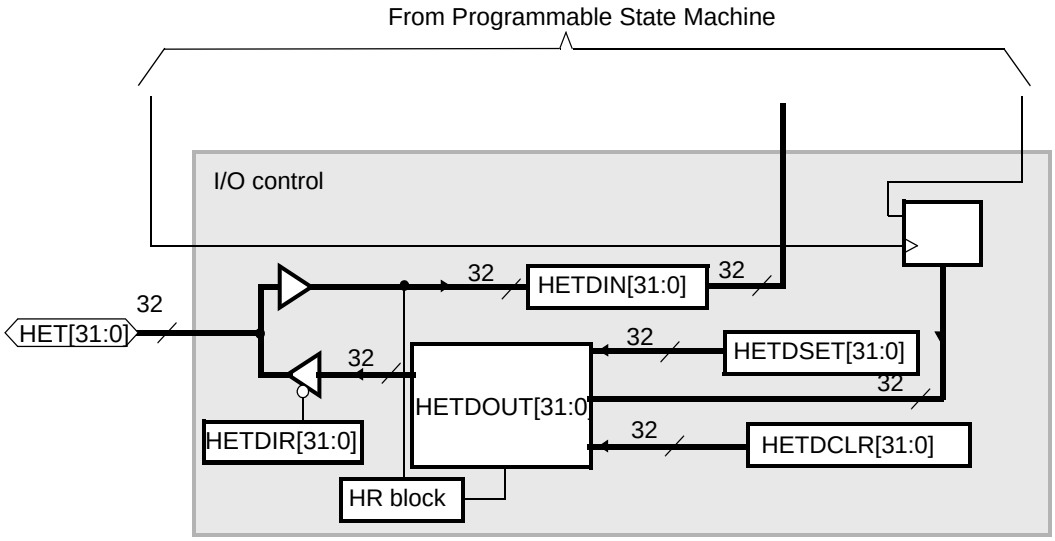
12.3.5 I/O Control

The HET has a port capable of up to 32 pins, but not all might be available on each device. All the HET pins available are programmable as either inputs or outputs.

Note:
Refer to the device configuration to determined the implemented pins.

The 32 I/Os are identically structured and are connected to pins HET[31] to HET[0]. This structure allows any HET instruction to use these I/Os with a loop resolution accuracy. See Figure 12-13 for an illustration of the I/O control. Among these 32 I/Os, 24 have an additional HR structure (pins HET[23] to HET[0]) based on the HR clock.

Figure 12-13. I/O Control



The CPU can use general purpose I/Os, pins HET [31] to HET [0], for reading using the control register HETDIN (Section 12.5.13), and for writing using HETDOUT (Section 12.5.14), HETDSET (Section 12.5.15), or HETDCLR (Section 12.5.16).

Note: Using HETDSET and HETDCLR

The HETDCLR and HETDSET registers (Section 12.5.16 and Section 12.5.15) can be used to minimize the number of accesses to the peripheral to modify the output register and output pins. Actually, when the application needs to set or to reset *n* pins without changing the value of the other pins, the first possibility is to read HETDOUT, modify the content (AND, OR, etc.), and write the result into HETDOUT. This solution will take, in the best case, eight Cycles (LDR,AND/OR, STR), and could be interrupted by a function modifying the same register, which will result in a data coherency problem.

Using the HETDSET or HETDCLR registers, the application program must write the mask value (same mask value for the first option) to the register to set or reset the desired pins. This operation will take only three cycles (STR) and is not interruptible.

Example (C program): Set pins using the two methods.

```
unsigned int MASK; /* Variable that content the bit mask   */
volatile unsigned int *HETDOUT,*HETDSET/* Pointer to HET registers */
...
*HETDOUT = *HETDOUT | MASK;/* Read-modify-write of HETDOUT */
*HETDSET = MASK;   /* Set the pin without reading HETDOUT */
```

12.3.5.1 Loop Resolution Structure

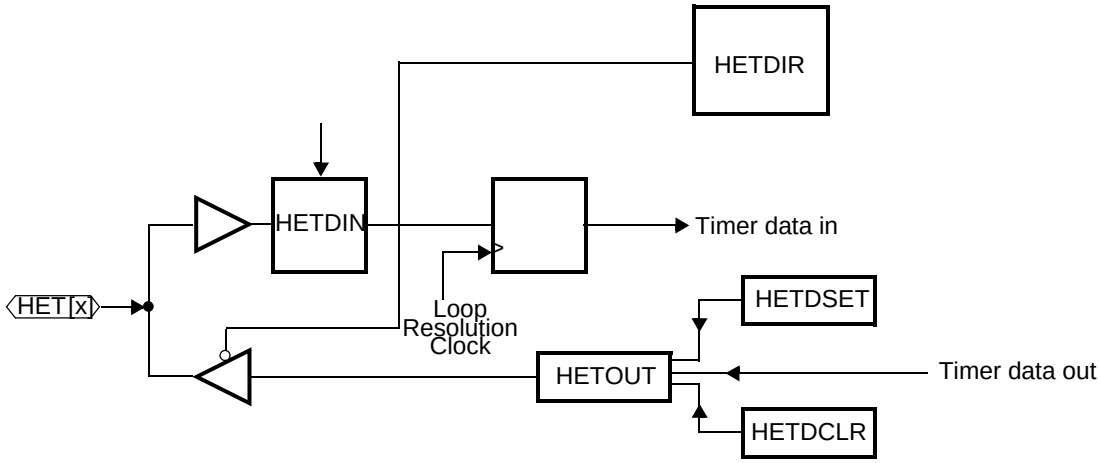
The HET uses the pins HET [31:0] as inputs and/or outputs. By using the I/O register of the HET, the CPU can interact with the HET program flow.

When an action (set or reset) is taken on a pin by the HET program, the HET will modify the pin at the rising edge of the next resolution clock.

When an event occurs on a HET I/O pin, it is taken into account at the next rising edge of the resolution clock.

The structure of each pin is shown in Figure 12-14.

Figure 12-14. HET Loop Resolution Structure for Each Bit



12.3.5.2 HR Structure

Among the 32 I/Os, 24 have the special HR structure (pins HET [23:0]) based on the HR clock. See Figure 12-15. The HR clock frequency is programmed through register HETPFR[5:0]. See Section 12.5.2.

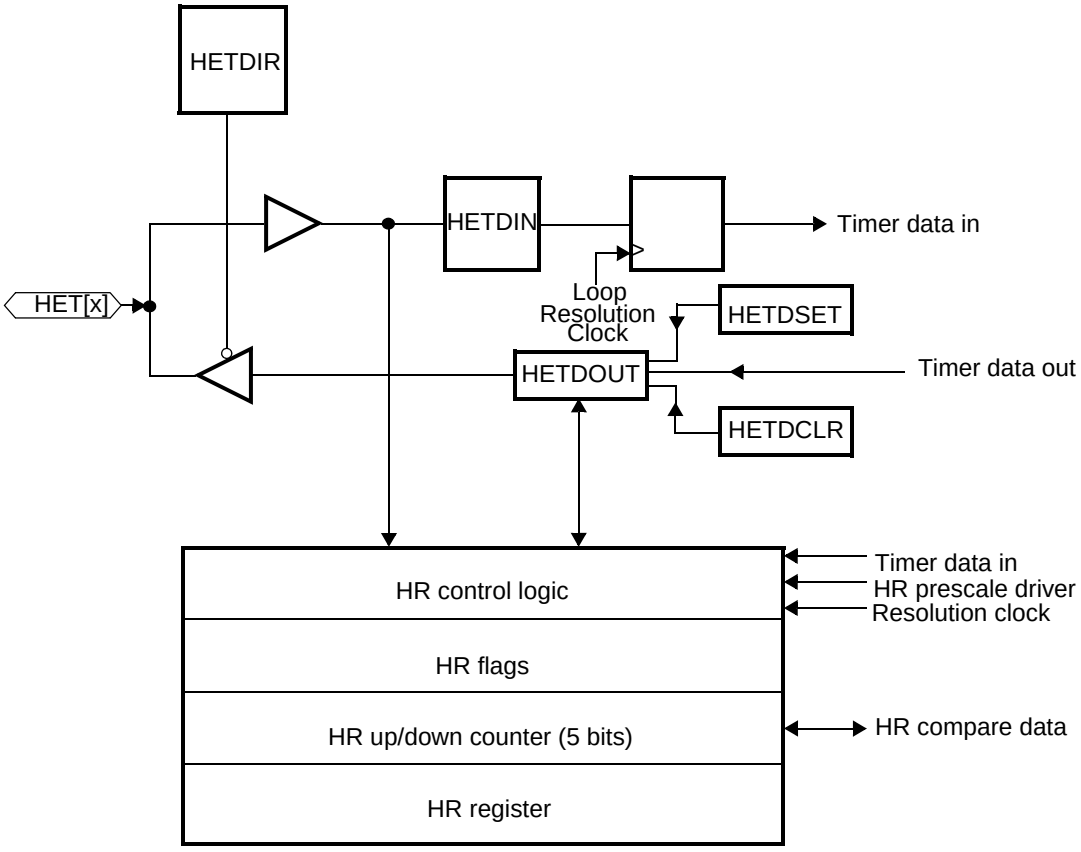
In addition to the standard I/O structure pins, HET [23:0] have special HR hardware so that these pins can be used as HR input captures (using PCNT or WCAP) or HR output compares (using ECMP, MCMP or PWCNT).

12.3.5.3 HR Block Diagram

Each time an HR instruction is executed on a given pin, the HR block for that pin is programmed (which HR function to perform and on which edges it should take an action) with the information in the instruction. The HR structure for each pin decodes the pin select field of the instruction and programs its HR structure if it matches.

Note:
Only one HR function is allowed per resolution structure. To enforce this restriction, the architecture shown below is programmed only once per resolution loop. This occurs when the first instruction is executed. You must ensure that only one instruction accesses a given pin in HR mode. The HR structure takes the information from the first instruction and ignores the remaining instructions.

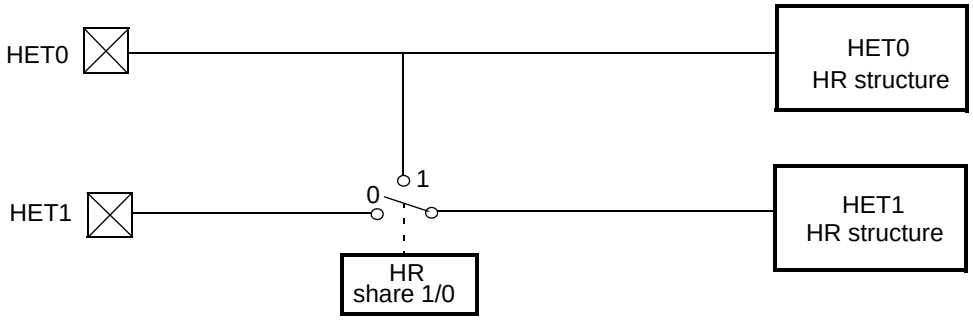
Figure 12-15. HR I/O Architecture



12.3.5.4 HR Structures Sharing

The HR share control register bits (see Section 12.5.10) allow two HR structures to share the same pin **for input capture only**. If these bits are set, the HR structures N and N+1 are connected to pin N. In this structure, pin N+1 remains available for general purpose input/output. See Figure 12-16.

Figure 12-16. Example of HR Structure Sharing for HET Pins 0/1



The following program gives an example how the HR share feature(HET0 HR structure and HET [1] HR structure shared) can be used for the PCNT instruction:

```
L00 PCNT { next=L01, type=rise2fall, pin=CC0 }
L01 PCNT { next=L00, type=fall2rise, pin=CC1 }
```

The HET [1] HR structure is also connected to the HET [0] pin. The L00_PCNT data field captures a high pulse and the L01_PCNT captures a low pulse on the **same** pin (HET [0] pin).

12.3.5.5 XOR-Shared HR Structure

Usually the HET design allows only one HR structure to cause HR edges on a pin configured as output pin. According to Section 12.3.5.10,the duration between two HR edges (both of which do not line up with the LR clock) cannot be smaller than 1 LRP. To eliminate this constraint, the HETXOR register allows a logical XOR of the output signals of two consecutive HR structures; see Figure 12-17. In this way, it is possible to generate pulses smaller than the loop resolution clock since both edges are now based on the HR clock. This is especially useful for a symmetrical PWM. See Figure 12-18.

This apparatus consists of a XOR gate that is connected to the outputs of the HR structure of two consecutive pins. In this structure, pin N+1 remains available for general purpose input/output.

Figure 12-17. XOR-shared HR I/O

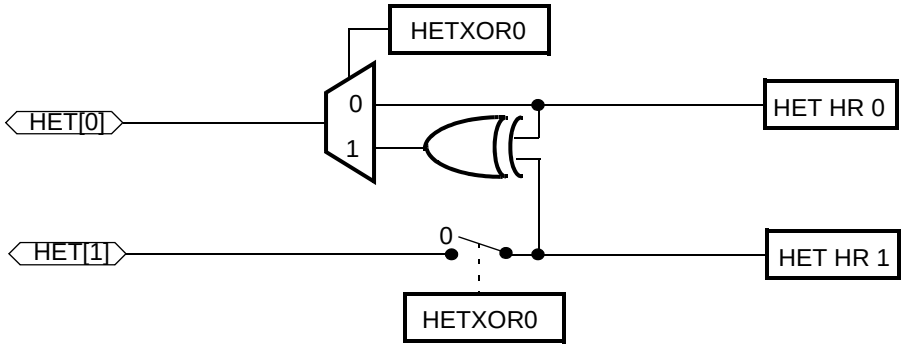
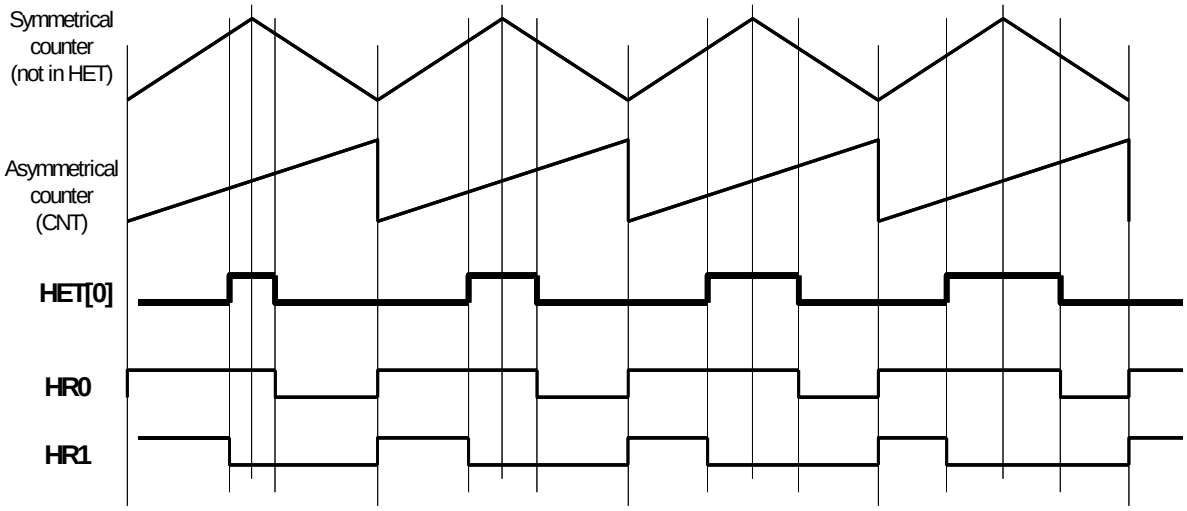


Figure 12-18. Symmetrical PWM with XOR-sharing Output



The following HET program gives an example for **one** channel of the symmetrical PWM. The generated timing is given in Figure 12-18.

```
MAXC .equ 22
A_   .equ 0  ; HR structure HR0
B_   .equ 1  ; HR structure HR1
```

```
CN CNT   { next=EA, reg=A, max=MAXC }

EA ECMP  { next=EB, cond_addr=MA, hr_lr=HIGH, en_pin_action=ON, pin=A_,
           action=PULSELO, reg=A, data=17, hr_data=19 }
MA MOV32 { next=EB, remote=EA, type=IMTOREG&REM, reg=NONE, data=17, hr_data=19 }

EB ECMP  { next=CN, cond_addr=MB, hr_lr=HIGH, en_pin_action=ON, pin=B_,
           action=PULSELO, reg=A, data=5, hr_data=13 }
MB MOV32 { next=CN, remote=EB, type=IMTOREG&REM, reg=NONE, data=5, hr_data=13 }
```

The HET settings and output signal calculation for this example program are as follows:

- Pin HET[0] and HET[1] are XOR-shared.
- HETPFR[31:0] register = 0x501 → Lr = 32 and Hr = 2 → time slots Ts = 64
- PWM period (determined by CNT_max field) = (22+1) * LRP = 736 HRP
- Length of high pulse of (HET[0] XOR HET[1]) =

  LH = (17*LRP+19*HRP)-(5*LRP+13*HRP)

  With LR = 32, there is LRP = 32 * HRP, therefore,

  LH = (563-173) * HRP = 390 HRP
- Duty cycle (DC) = LH / PWM_period = 390 HRP / (736*HRP) = 53.0%

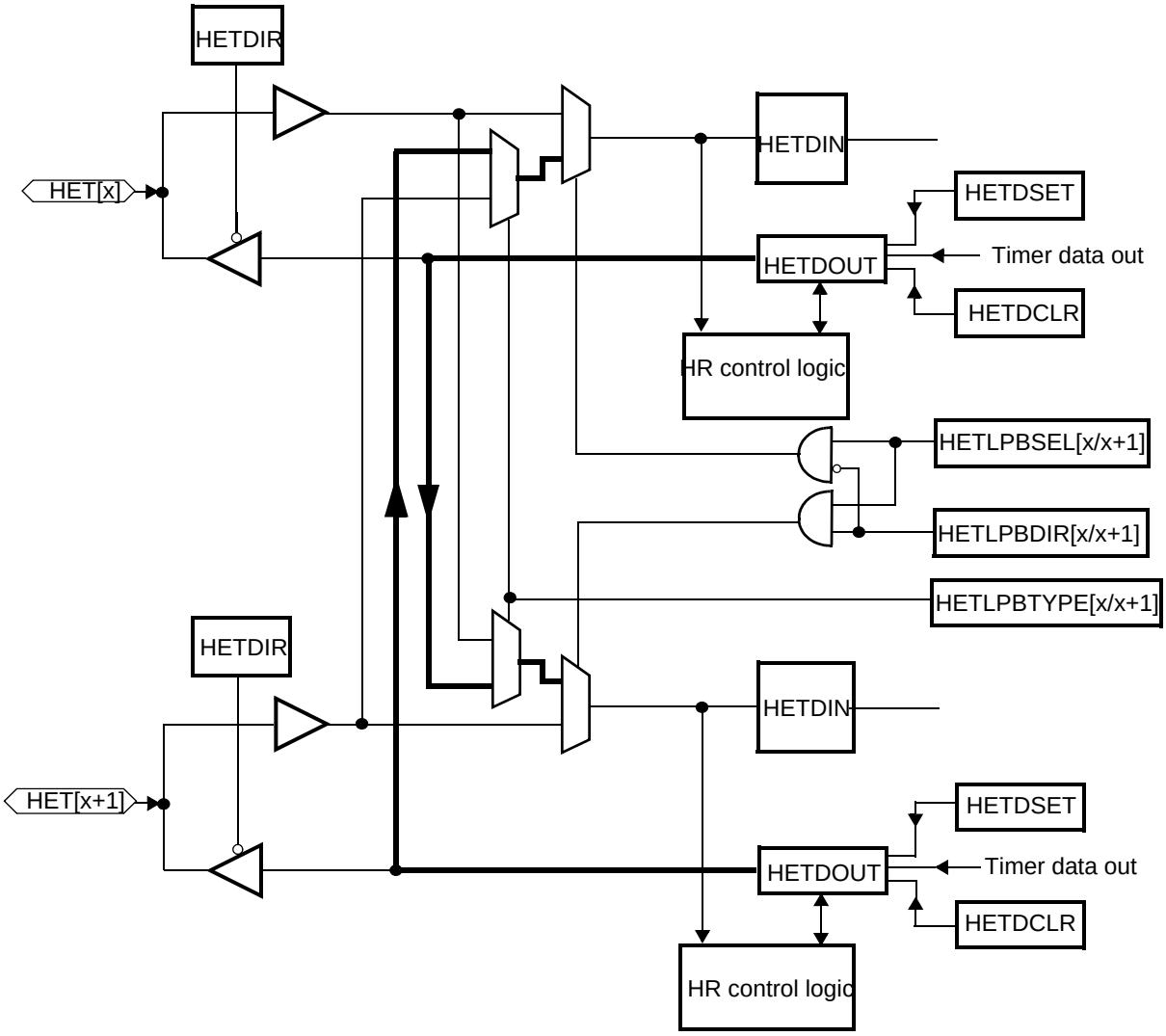
Figure 12-18 shows the implementation of the XOR-shared feature. The first two waveforms (symmetrical counter and CNT) show a symmetrical counter and an asymmetrical counter. The symmetrical counter is shown only to highlight the axis of symmetry and is not implemented in the HET. The asymmetrical counter, which is implemented with a CNT instruction, needs to be set to the period of the symmetrical counter. The next two waveforms (HR [0] and HR [1]) show the output of the HR structures, which are the inputs for the XOR gate to create the PWM output on pin HET[0]. Notice that the pulses of signal HET[0] are centered about the axis of symmetry.

12.3.5.6 Loopback Modes

It is possible to output a signal on one pin (e.g., with ECMP; Section 12.6.3.10) and measure back the signal on the same pin (e.g., with ECMP). However, if HR is used to output the signal, it is not possible to measure back the signal with HR, since the same HR logic cannot be used for both signal generation and signal measurement at the same time. It is only possible to measure back with LR.

The HET, however, provides two different loopback modes (digital loopback and analog loopback) to allow the signal to also be measured in HR.

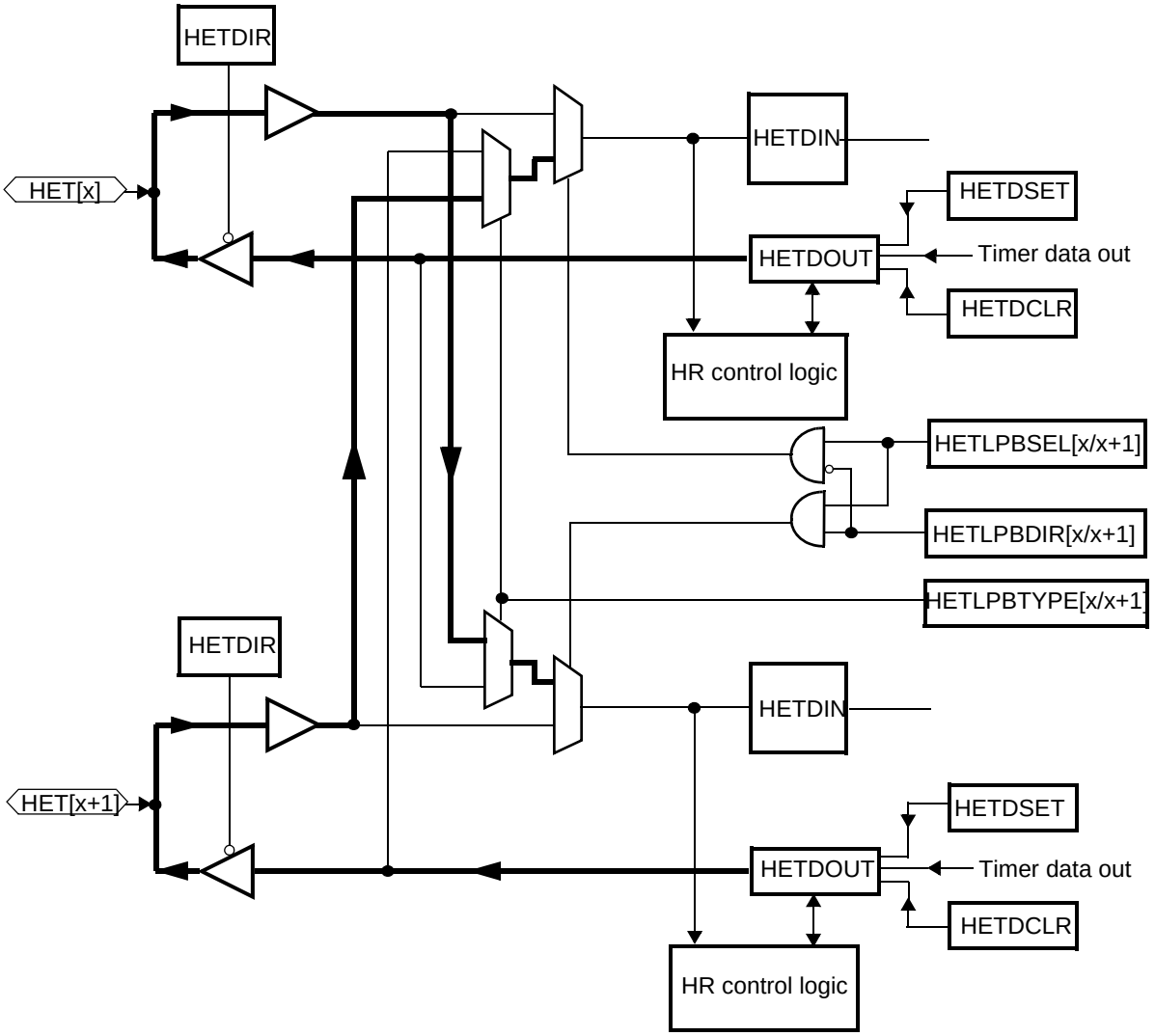
Figure 12-19. Digital Loopback I/O



To select digital loopback mode, the corresponding HETLPBSEL bit (Section 12.5.19) has to be set to 1 and HETLPBTYPE (Section 12.5.19) has to be set to 0. Depending on the setting of the HETLPBDIR bit (Section 12.5.20), the direction will go from HET[x] to HET[x+1] or from HET[x+1] to HET[x]. In digital loopback mode, the input of the output buffer will be connected to the input of HR logic and the HETDIN register.

Depending on the HETDIR bit setting, the signal to be fed back can be output on the pin or suppressed. The pin structure that is used to output the signal can still be used as general-purpose input if the HETDIR bit is set so that input mode is selected. The pin structure that measures back the generated signal can still use its pin as general-purpose output pin.

Figure 12-20. Analog Loopback I/O



To select analog loopback mode, the corresponding HETLPBSEL bit (Section 12.5.19) has to be set to 1 and HETLPBTYPE (Section 12.5.19) has to be set to 1. Depending on the setting of the HETLPBDIR bit (Section 12.5.20), the direction will go from HET[x] to HET[x+1] or from HET[x+1] to HET[x]. In analog loopback mode, the output of the input buffer will be connected to the input of HR logic and the HETDIN register.

The HETDIR bit has to be set to 1 so that the signal to be fed back can be output on the pin. The pin structure that measures back the generated signal can still use its pin as a general-purpose output pin.

12.3.5.7 HR/Low Resolution Bit

Four HR instructions—ECMP (Section 12.6.3.10), MCMP (Section 12.6.3.12), PWCNT (Section 12.6.3.16), and WCAP (Section 12.3.5.13)—have a dedicated hr_lr bit (HR/LR) allowing operation either in HR mode or in LR mode by ignoring the HR field. The hr_lr bit is P(7) (program field bit 7). By default, the hr_lr bit value is 0, which implies HR operation mode. However, setting this bit to 1 allows the use of several HR instructions on a single HR pin. Only one instruction will be allowed to operate in HR mode (i.e., the bit set to 0), but the other instructions can be used in LR mode (i.e., the bit set to 1).

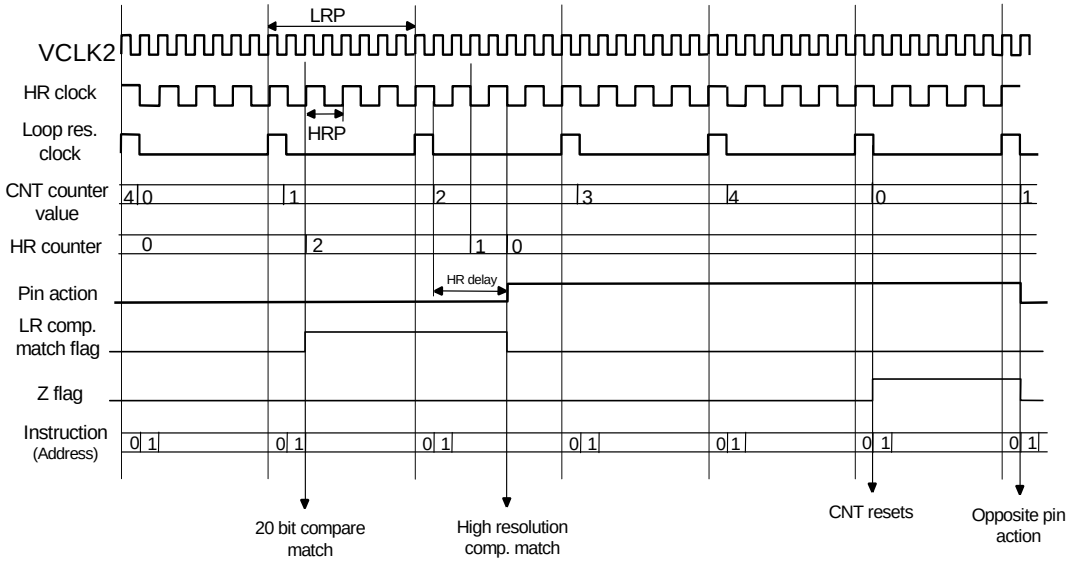
12.3.5.8 ECMP Execution Example (in HR Mode)

First is an example of an HR ECMP with a linear prescale of /2 and a loop resolution prescale of /4. With a 32-MHz VCLK2 clock, these parameters result in a 62.5 ns HR clock and a 0.25us resolution loop clock. This allows eight time slots for the program, which in this case will consist of one CNT and one ECMP instruction (Section 12.6.3.7 and Section 12.6.3.10). The ECMP instruction is loaded into the HET RAM with a 25-bit compare value, the lower 5 bits representing the HR compare value.

When the 20-bit (low resolution) compare matches, the LR comp match flag will be set in the HR hardware for the corresponding pin and the HR compare value will be loaded from the five lower bits of the instruction data field to the HR counter. At the next LR clock, the HR counter will count down at the HR clock frequency and perform the set action when it reaches zero.

Figure 12-21 shows the execution timing when the 20-bit compare value is 1 and the HR compare value is 2.

Figure 12-21. ECMP Execution Timings



The following HET program gives an example for a one channel PWM. The generated timing is given in Figure 12-21.

```
HETPFR[31:0] register = 0x201 → lr=4 and hr=2 → time slots ts = 7
L00   CNT  {next=01h, reg=A, irq=OFF, max = 4 }
L01   ECMP {next=00h, en_pin_action=ON, cond_addr= 00h, pin=CC0, action=PULSEHI, reg=A,
irq=OFF, data= 1, hr_data = 0x10 }
; 20 bit compare value is 1 and the HR compare value is 2
```

Note: ECMP Opposite Actions
ECMP opposite actions are always synchronized to the loop resolution clock.

Capture/compare pins can be selected individually with instructions ECMP (Section 12.6.3.10), SCMP (Section 12.6.3.18), MCMP (Section 12.6.3.12), PWCNT (Section 12.6.3.16), MOV64 (Section 12.6.3.14), DADM64 (Section 12.6.3.8), and RADM64 (Section 12.6.3.17). A pin already defined as an output compare can be selected as an internal input in the instructions BR (Section 12.6.3.6), CNT (Section 12.6.3.7), SHFT (Section 12.6.3.20), and WCAP (Section 12.6.3.21).

12.3.5.9 MCMP Execution Example

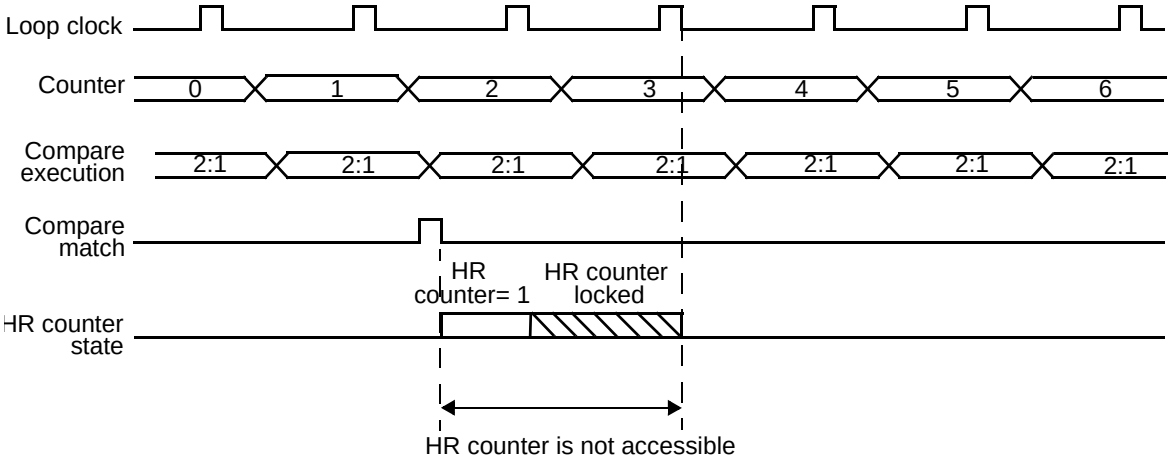
The MCMP instruction (see Section 12.6.3.12 for detailed information) can also be used in HR mode. Operation with the MCMP instruction is exactly the same as for the ECMP instruction except that the 20-bit

low resolution is now the result of a magnitude compare rather than of an equality compare. Otherwise, operation is the same as for the 20-bit match setting the HR flag and loading the HR counter. The HR counter starts to decrement after the next resolution clock and performs the specified pin action. See Figure 12-22.

12.3.5.10 Limitation on ECMP and MCMP Operations in HR Mode

There is some limitation on the ECMP (Section 12.6.3.21) and MCMP (Section 12.6.3.12) operations in the HR mode, which Figure 12-22 illustrates.

Figure 12-22. ECMP Limitation Timing Diagram



ECMP or MCMP compares its own data field value with a register value that is loaded by a CNT instruction. The compare is a 20-bit comparison, even if ECMP or MCMP has an HR value. If this 20-bit comparison matches and if the HR mode is used, the HR compare value is loaded in the dedicated HR counter. The pin action happens in the next loop resolution cycle. The HR counter is locked during all this time, as shown in Figure 12-22.

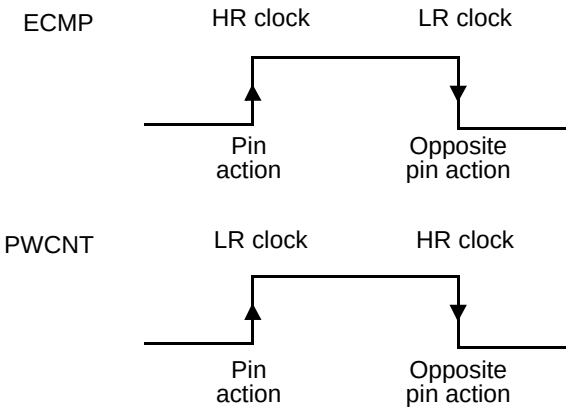
In other words, when a compare matches, one loop resolution cycles must pass before a new compare (in HR mode) can occur.

Note:
If it is necessary to have a more accurate pulse time, using the HR XOR-shared features with two HR pins will work around this limitation.

12.3.5.11 PWCNT Execution Example (in HR Mode)

The PWCNT instruction (Section 12.6.3.16) may also be used in HR mode to generate pulse outputs with HR width. The PWCNT instruction operates conversely to the ECMP instruction; see Figure 12-23. For PWCNT, the opposite pin action is synchronous with the HR clock and for ECMP the pin action is synchronous with the HR clock. The PWCNT pin action is synchronous with the loop resolution clock.

Figure 12-23. High/Low Resolution Modes for ECMP and PWCNT



12.3.5.12 PCNT Execution Example (in HR Mode)

In conjunction with HR I/O pins, the PCNT instruction (Section 12.6.3.15) can capture an HR measurement of the high/low pulse time or periods of the input. As shown in Figure 12-24, at the first marker, the input goes high and the HR counter immediately begins to count. The counter increments and rolls over until the falling edge, where it captures the counter value into the HR capture register (the second marker). The PCNT instruction begins counting when the synchronized input signal goes high and captures both the 20-bit data field and the HR capture register into RAM when the synchronized input falls (third marker).

Note:
The HR capture value written into RAM is shifted appropriately depending on the loop resolution prescale divide rate (lr).

Figure 12-24 shows what happens when the capture edge arrives *after* the HR counter overflows. This causes the incremented value to be captured by the PCNT instruction.

Figure 12-24. PCNT Instruction Timing (With Capture Edge After HR Counter Overflow)

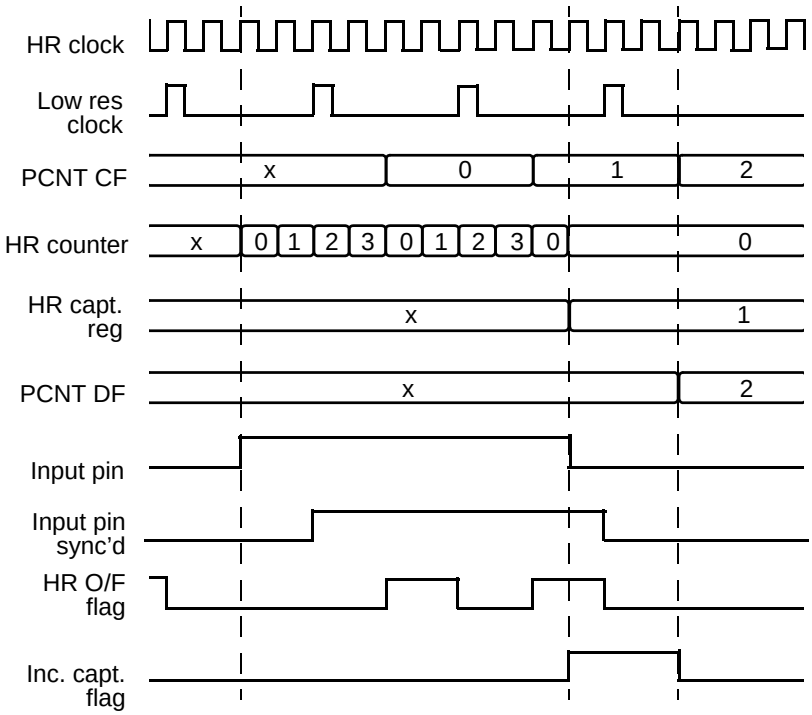
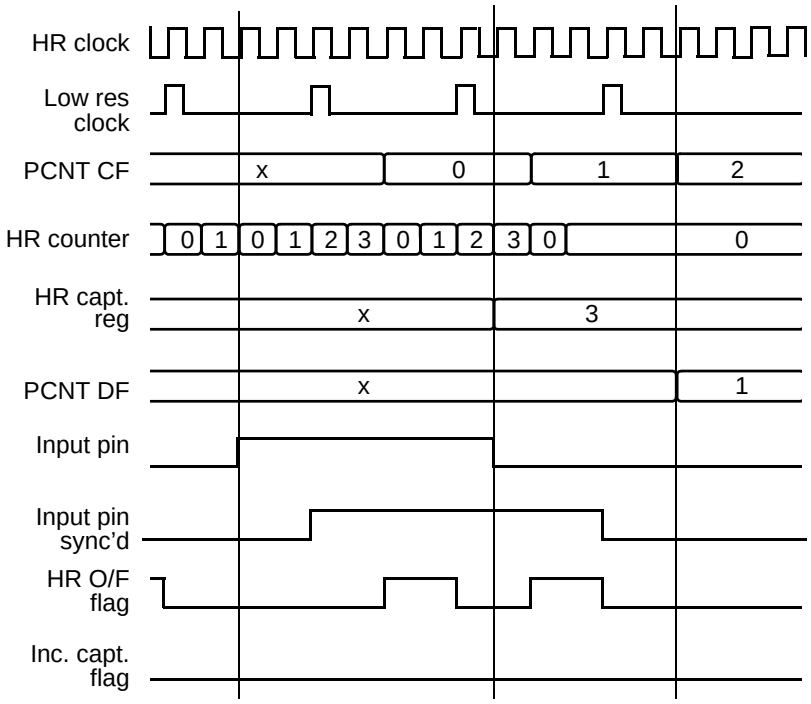


Figure 12-25 shows what happens when the capture edge arrives *before* the HR counter overflows. This causes the non-incremented value to be captured by the PCNT instruction.

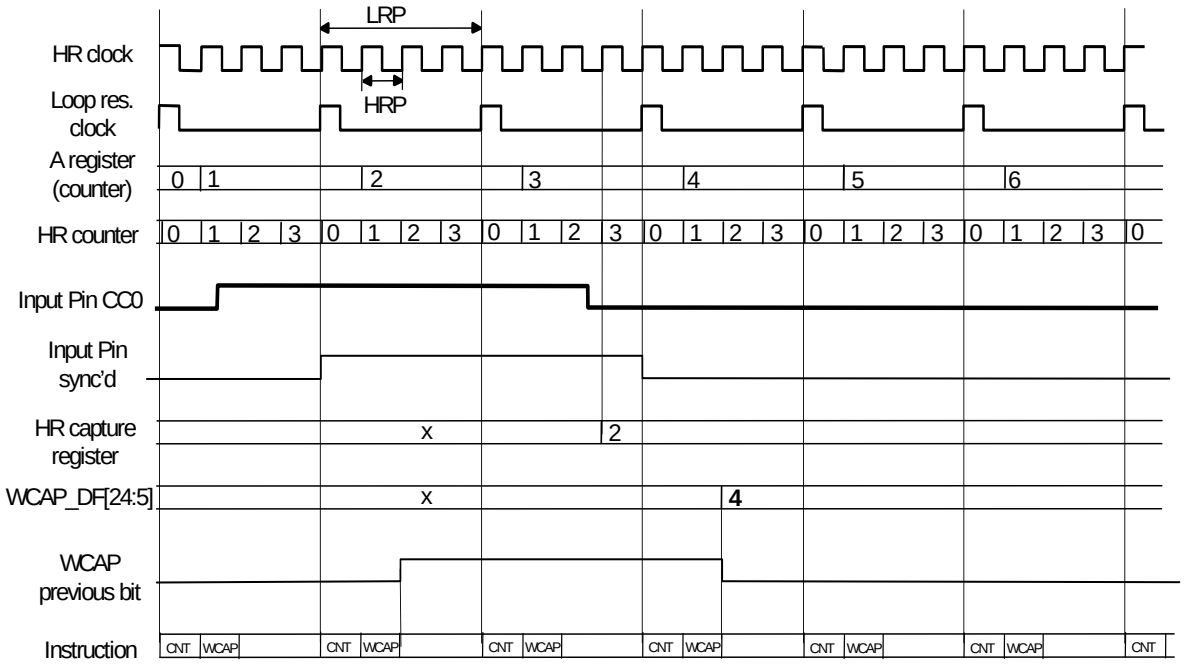
Figure 12-25. PCNT Instruction Timing (With Capture Edge Before HR Counter Overflow)



12.3.5.13 WCAP Execution Example (in HR Mode)

The WCAP instruction (Section 12.6.3.21) can be implemented for HR I/O. In this case the HR counter is always enabled (using the count always flag in the HR structure) and is synchronized with the resolution loop. When the specified edge is detected, the current value of the HR counter is captured in the HR capture register and written into the RAM after the next WCAP execution. The WCAP instruction effectively time stamps a free run timer saved in a register (for example, register A shown in Figure 12-26).

Figure 12-26. WCAP Instruction Timing



The following HET program gives an example for a one channel capture input using the WCAP instruction. The generated timing is given in Figure 12-26.

```
HETPFR_register = 0x0200 → lr = 4, hr = 1, ts = 3

L00 CNT  {reg=A, max=0fffffh}
L01 WCAP {next=L00, cond_addr=L00, hr_lr=high, reg=A, event= FALL, pin=CC0,
          data=0}
```

In the timing of the example, the WCAP is configured to capture the counter when a **falling** edge occurs. The WCAP data field (WCAP_DF) is complete in the loop succeeding the loop, in which the edge occurred. In the figure example, the current value of the counter (4) is captured to WCAP_DF[24:5] and the value of the HR capture register (2) is transferred to the valid bits (according the lr prescaler) of WCAP_DF[4:0]. Therefore, in the example 0x090 is captured to WCAP_DF[24:0].

12.3.6 Interrupts and Exceptions

HET interrupts are generated by any instruction that has an interrupt enable bit in its instruction format. When the interrupt condition in an instruction is true and the interrupt enable bit of that instruction is set, an interrupt flag is then set in the HETFLG register. The address code for this flag is determined by the five LSBs of the current timer program address.

Note:
Enabling or disabling the interrupt generation of certain instructions (BR, ECNT, SHFT, WCAP) when the HET is turned on could falsify the operation of these instructions. This effect is due to the possible modification of the PRV bit by the HET during the read/modify/write of the CPU to modify the control field.

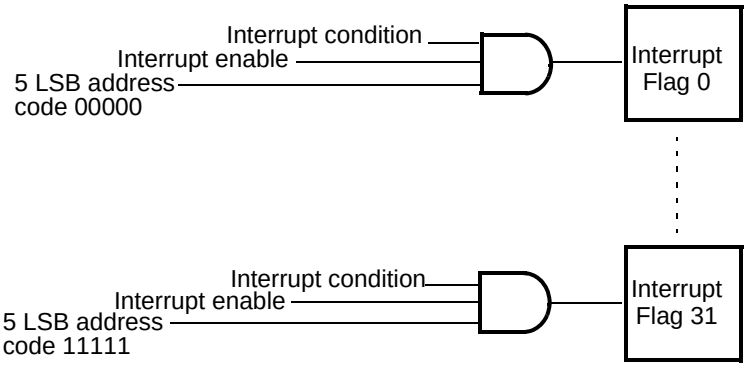
Table 12-5. Interrupt Sources and Corresponding Offset Values in Registers HETOFFx

Source No.	Offset Value
no interrupt	0
Instruction 0, 32, 64, 96, 128, ..., 224	1
Instruction 1, 33, 65, 97, 129,..., 225	2
:	:
Instruction 31, 63, 95, 127,..., 255	32
Program overflow	33
APCNT underflow:	34
APCNT overflow	35

To execute the interrupt service routine, the main CPU must first determine which source created the interrupt request. It does that by reading the offset register (HETOFFx), which gives the number of the source. Reading the offset register will automatically clear the source flag that created the request. However, if it is decided not to use the offset register to check the interrupt source, the flag should still be cleared after the interrupt has been serviced.

The instructions capable of generating interrupts are listed in Table 12-32 in Section 12.6.

Figure 12-27. Interrupt Servicing



Each interrupt source is associated with a priority level (level 1 or level 2). When multiple interrupts with the same priority level occur during the same loop resolution, the lowest flag bit is serviced first. Two interrupt requests are generated by the HET for the vectored interrupt manager module (VIM). Figure 12-27 shows how the interrupts are serviced.

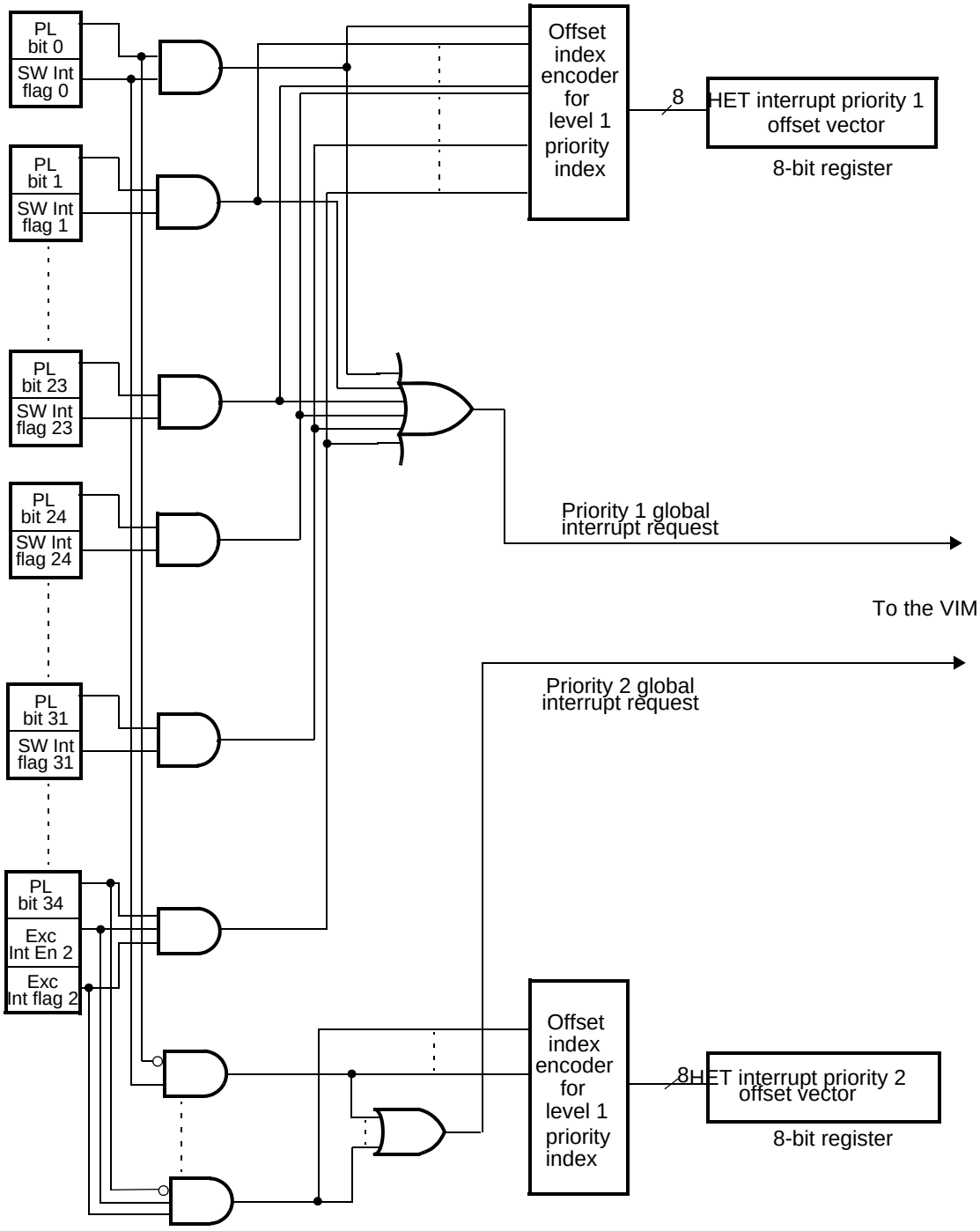
The HET can generate three types of exception from the following:

- Program overflow
- APCNT underflow (see Section 12.4.1.2)
- APCNT overflow (see Section 12.4.1.3)

12.3.7 Hardware Priority Scheme

If two or more software interrupts are pending on the same priority level, the offset value will show the one with the highest priority. The interrupt with the highest priority is the one with the lowest offset value. This scheme is hard-wired in the offset encoder. See Figure 12-28.

Figure 12-28. Interrupt Flag/Priority Level Architecture



12.4 Angle Functions

Engine management systems require an angle-referenced time base to synchronize signals to the engine toothed wheel. The HET has a method to provide such a time base:

- A software angle counter for low-end engine systems. The reference is created by the HET using three dedicated instructions with fractional angle steps equal to /8, /16, /32, /64.

12.4.1 Software Angle Generator (SWAG)

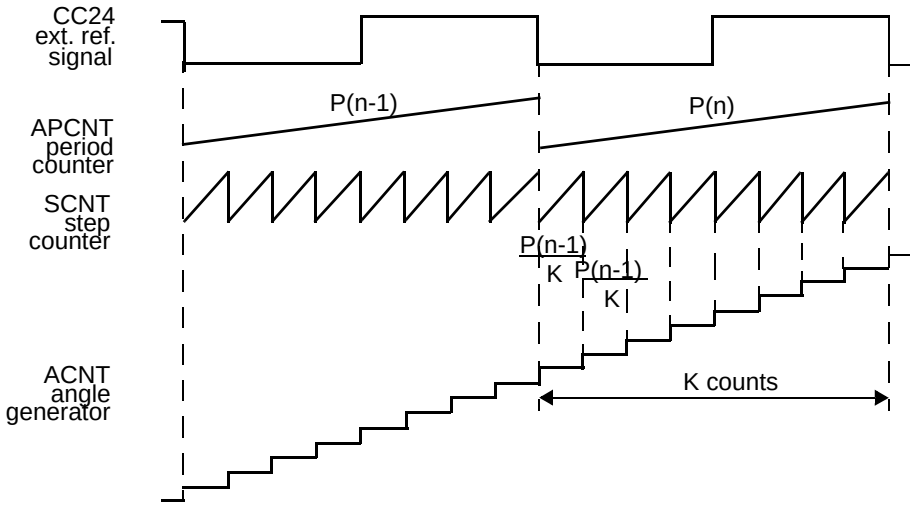
The HET provides three specialized count instructions to generate an angle-referenced time base synchronized to an external reference signal (the toothed wheel signal) that defines angular reference points.

The time base is used to generate fractional angle steps between the reference points. The step width K (= 8, 16, 32, or 64) programmed by you defines the angle accuracy of the time base. These fractional steps are then accumulated in an angle counter to form the absolute angle value.

The first counter APCNT (Section 12.6.3.5) incremented on each resolution clock measures the periods P(n) of the external signal. The second counter SCNT (Section 12.6.3.19) counts by step K up to the previous period value P(n-1), measured by APCNT, and then recycles. The resulting period of SCNT is the fraction P(n-1) / K. The third counter ACNT (Section 12.6.3.2) accumulates the fractions generated by SCNT.

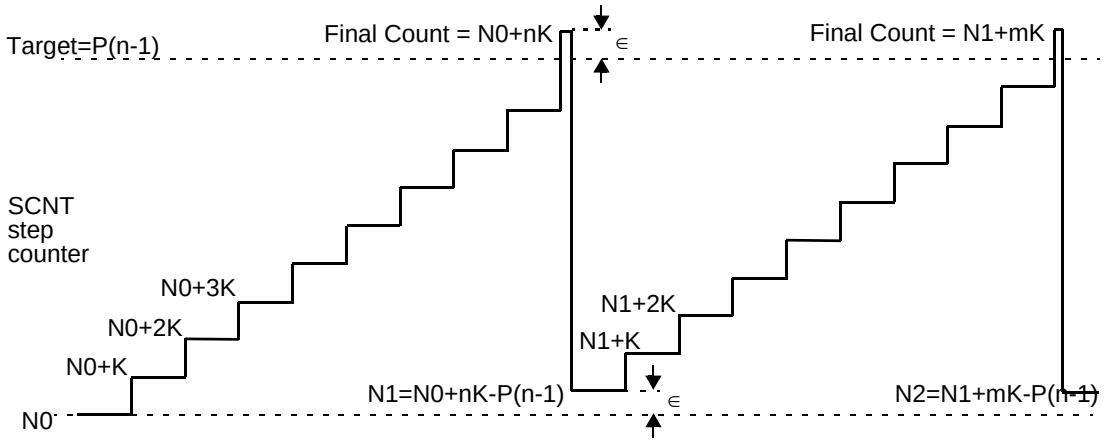
A HET timer program can only have one angle generator. Figure 12-29 illustrates the basic operation of APCNT, SCNT, and ACNT.

Figure 12-29. Operation of HET Count Instructions



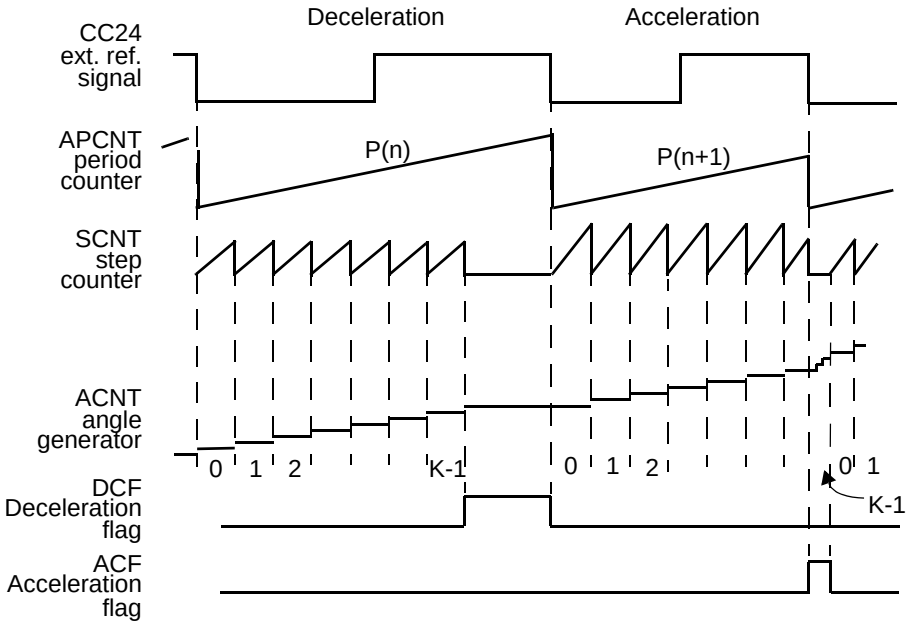
Because of stepping, the final count of SCNT does not usually exactly match the target value P(n-1). Figure 12-30 illustrates how SCNT compensates for this feature by starting each cycle with the remainder (final count - target) of the previous cycle.

Figure 12-30. SCNT Count Operation



ACNT detects period variations of the external signal measured by APCNT and compensates related counting errors. A period increase is flagged in the deceleration flag. A period decrease is flagged in the acceleration flag. If no variation is flagged, ACNT increments the counter value each time SCNT reaches its target. If acceleration is detected, ACNT increments the counter value on each timer resolution (fast mode). If deceleration is detected, ACNT is stopped. Figure 12-31 illustrates how the compensations for acceleration and deceleration operate.

Figure 12-31. ACNT Period Variation Compensations



12.4.1.1 Singularities

Singularities (gaps, in this case, from missing teeth in a toothed wheel) in the external reference signal can be masked. The start and end of singularities are defined by gap start and gap end values specified in SCNT (Section 12.6.3.19) and ACNT (Section 12.6.3.2). When ACNT reaches gap start or gap end, it sets/resets the gap flag.

While the gap flag is set, new periods of the external reference signal are ignored for angle computation. SCNT uses the last period measured by APCNT just before gap start.

Figure 12-32 and Figure 12-33 illustrate the behavior of the angle generator during a gap after a deceleration or acceleration of the HET.

Figure 12-32. HET Timings Associated with the Gap Flag (ACNT Deceleration)

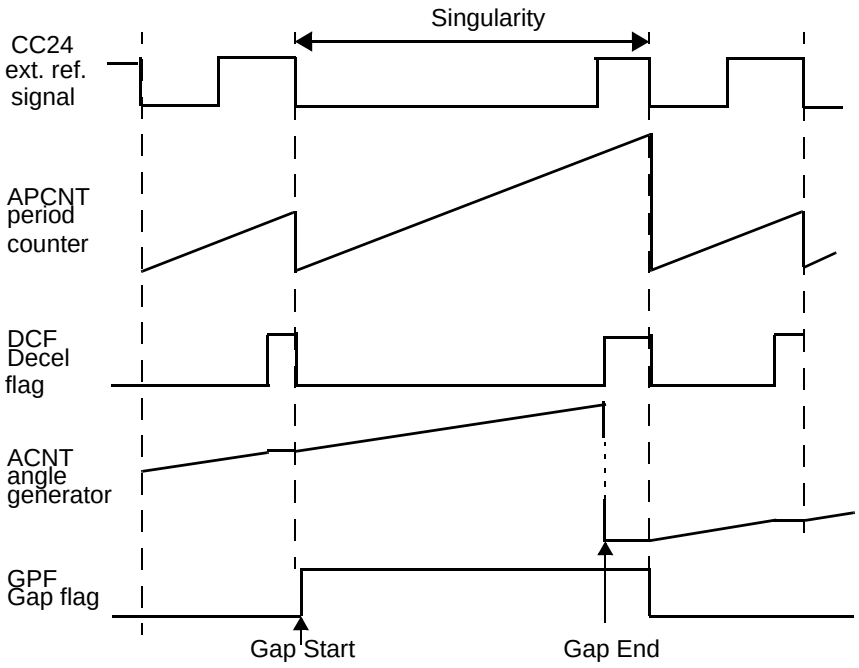
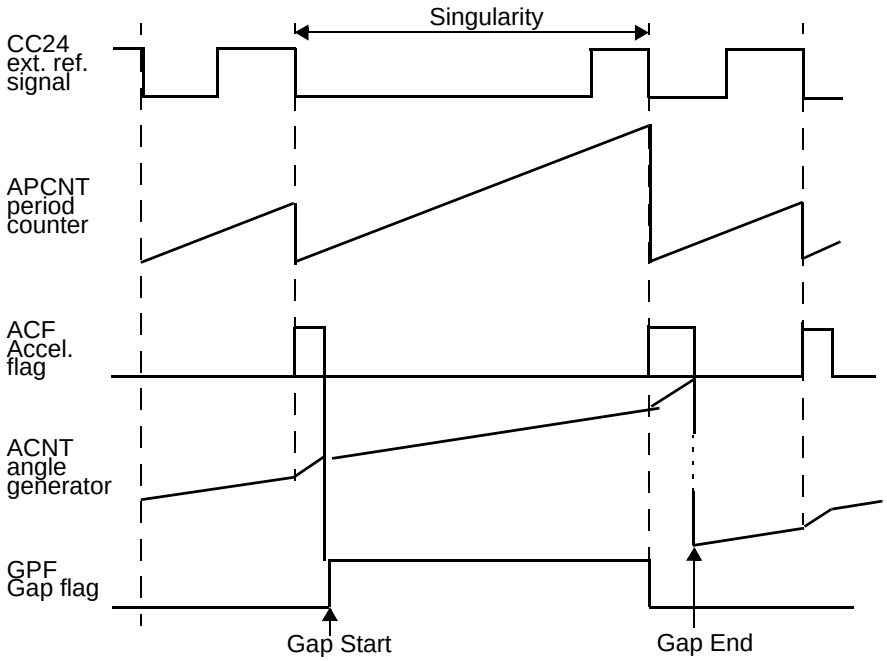


Figure 12-33. HET Timings Associated with the Gap Flag (ACNT Acceleration)



12.4.1.2 APCNT Underflow

The fastest valid external signal APCNT (Section 12.6.3.5) can accept must satisfy the following condition:

$$\text{Step Width K} < \frac{\text{Period Min.}}{\text{Resolution (LRP)}}$$

This condition fixes the maximum possible step width once the minimum period and the resolution of an application are specified.

If a period value accidentally falls below the minimum allowed, APCNT stops the capture of these periods and sets the APCNT underflow interrupt flag located in the exceptions interrupt control register. In such a situation, SCNT and ACNT continue to be executed using the last valid period captured by APCNT.

12.4.1.3 APCNT Overflow

The slowest valid external signal APCNT (Section 12.6.3.5) can measure must satisfy the following condition:

$$\frac{\text{Period Max}}{\text{Resolution}} < 1048575$$

When this limit is reached (APCNT count equals all 1s), APCNT stays at a maximum count (stops counting). APCNT remains in this position until the next specified capture edge is detected on the selected pin and sets the APCNT overflow interrupt flag located in the exceptions interrupt control register. In this situation, SCNT and ACNT continue to be executed using the maximum APCNT period count.

12.5 HET Control Registers

Accesses to the peripheral registers can be done in byte, half-word, or word format.

For the registers having a read/clear, read/set structure, the update is as follows:

- Writing a 1 performs the action (set or clear depending on the structure).
- Writing a 0 has no effect (no change for the bit).

Table 12-6. Read/Write Conventions

Action	Description
R	Read
W	Write
C	Clear
S	Set
n	Value after reset
RWP	Read in all modes, write in privileged mode only

Figure 12-34 presents a summary of the HET registers.

The base address for the control registers is 0xFFF7 A400

Figure 12-34. HET Register Summary

Offset Address Register(1)(2)	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00h HETGCR Page 420	Reserved							Power Down	Reserved							CLK Master/ Slave
	Reserved							64-Bit Access	Reserved					Debug Status Flag	Ignore Susp.	Turn On/Off
0x04 HETPFR Page 423	Reserved															
	Reserved							LRPres.Factor(2–0)			Reserved			HRPres.Factor(5–0)		
0x08 HETADDR Page 425	Reserved															
	Reserved								HETADDR(7–0)							
0x0C HETOFF1 Page 426	Reserved															
	Reserved								OFFSET1(7–0)							

1 The physical address (base + offset) of these registers is device specific. See the device-specific data sheet to verify the register base address.

2 The HETXOR register is only available for certain TMS470Px derivatives. Refer to the device-specific data sheet.

Figure 12-34. HET Register Summary

Offset Address Register[1][2]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x10 HETOFF2 Page 427	Reserved															
	Reserved								OFFSET2(7–0)							
0x14 HETEXC1 Page 428	Reserved							APCNT OVRFL ENA	Reserved							APCNT UNRFL ENA
	Reserved							PRGM OVRFL ENA	Reserved					APCNT OVRFL PRY	APCNT UNRFL PRY	PRGM OVRFL PRY
0x18 HETEXC2 Page 429	Reserved															
														APCNT OVRFL FLAG	APCNT UNRFL FLAG	PRGM OVRFL FLAG
0x1C HETPRY Page 430	HETPRY[31:16]															
	HETPRY[15:0]															
0x20 HETFLG Page 431	HETFLAG[31:16]															
	HETFLAG[15:0]															
0x2C HETHRSH Page 432	Reserved															
	Reserved		HR SHARE 23/22		HR SHARE 21/20		HR SHARE 19/18		HR SHARE 17/16		HR SHARE 15/14		HR SHARE 13/12		HR SHARE 11/10		HR SHARE 9/8		HR SHARE 7/6		HR SHARE 5/4		HR SHARE 3/2		HR SHARE 1/0
0x30 HETXOR Page 433	Reserved															
	Reserved		XOR SHARE 23/22		XOR SHARE 21/20		XOR SHARE 19/18		XOR SHARE 17/16		XOR SHARE 15/14		XOR SHARE 13/12		XOR SHARE 11/10		XOR SHARE 9/8		XOR SHARE 7/6		XOR SHARE 5/4		XOR SHARE 3/2		XOR SHARE 1/0
0x34 HETDIR Page 434	HETDIR[31:16]															
	HETDIR[15:0]															

1 The physical address (base + offset) of these registers is device specific. See the device-specific data sheet to verify the register base address.
2 The HETXOR register is only available for certain TMS470Px derivatives. Refer to the device-specific data sheet.

Figure 12-34. HET Register Summary

Offset Address Register[1][2]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x38 HETDIN Page 435	HETDIN[31:16]															
	HETDIN[15:0]															
0x3C HETDOUT Page 436	HETDOUT[31:16]															
	HETDOUT[15:0]															
0x40 HETDSET Page 437	HETDSET[31:16]															
	HETDSET[15:0]															
0x44 HETDCLR Page 438	HETDCLR[31:16]															
	HETDCLR[15:0]															
0x4C HETPULDIS Section 12.5.17	HETPULDIS[31:16]															
	HETPULDIS[15:0]															
0x50h HETPSL Section 12.5.18	HETPSL[31:16]															
	HETPSL[15:0]															
0x60h HETLPBSEL Section 12.5.19	HETLP BTYPE 31/30	HETLP BTYPE 29/28	HETLP BTYPE 27/26	HETLP BTYPE 25/24	HETLP BTYPE 23/22	HETLP BTYPE 21/20	HETLP BTYPE 19/18	HETLP BTYPE 17/16	HETLP BTYPE 15/14	HETLP BTYPE 13/12	HETLP BTYPE 11/10	HETLP BTYPE 9/8	HETLP BTYPE 7/6	HETLP BTYPE 5/4	HETLP BTYPE 3/2	HETLP BTYPE 1/0
	HETLP BSEL 31/30	HETLP BSEL 29/28	HETLP BSEL 27/26	HETLP BSEL 25/24	HETLP BSEL 23/22	HETLP BSEL 21/20	HETLP BSEL 19/18	HETLP BSEL 17/16	HETLP BSEL 15/14	HETLP BSEL 13/12	HETLP BSEL 11/10	HETLP BSEL 9/8	HETLP BSEL 7/6	HETLP BSEL 5/4	HETLP BSEL 3/2	HETLP BSEL 1/0
0x64h HETLPBDIR Section 12.5.20	Reserved															
	HETLP BDIR 31/30	HETLP BDIR 29/28	HETLP BDIR 27/26	HETLP BDIR 25/24	HETLP BDIR 23/22	HETLP BDIR 21/20	HETLP BDIR 19/18	HETLP BDIR 17/16	HETLP BDIR 15/14	HETLP BDIR 13/12	HETLP BDIR 11/10	HETLP BDIR 9/8	HETLP BDIR 7/6	HETLP BDIR 5/4	HETLP BDIR 3/2	HETLP BDIR 1/0

1 The physical address (base + offset) of these registers is device specific. See the device-specific data sheet to verify the register base address.
2 The HETXOR register is only available for certain TMS470Px derivatives. Refer to the device-specific data sheet.

12.5.1 Global Configuration Register (HETGCR)

Figure 12-35 and Table 12-7 illustrate the HETGCR register.

Figure 12-35. Global Configuration Register (HETGCR) [offset = 0x00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							Power Down	Reserved							clk_ master /slave
R-0							R/W-0	R-0							R/W-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							64-bit Access	Reserved					Debug Status Flag	Ignore Suspend	Turn On/Off
R-0							R/W-0	R-0					R/C-0	R/W-0	R/W-0

R = Read; W = Write; C = Clear; -*n* = value after reset

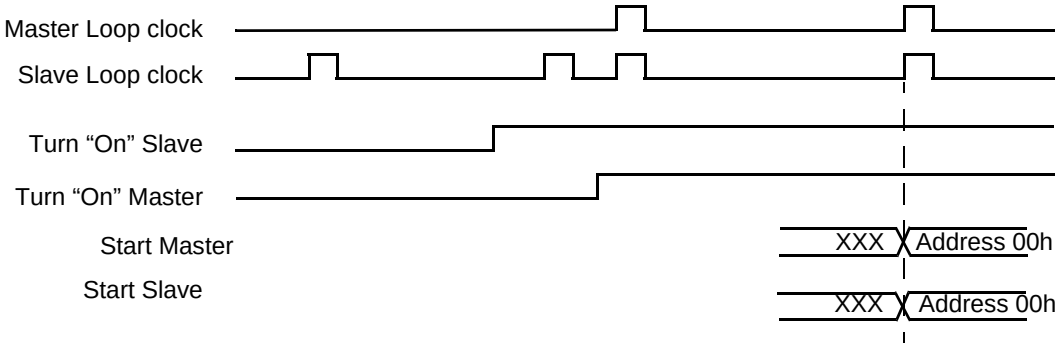
Table 12-7. Global Configuration Register (HETGCR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return 0 and writes have no effect.
24	Power-Down		HET power-down mode bit. When Power-Down = 1, the HET internal clocks are stopped. This leaves the module in a static state in which it consumes the lowest possible current. To release the power-down mode, this bit must be written to 0; after setting turn-off, you must delay until the end of the timer program loop before putting the HET in power-down mode. This can be done by waiting until the HET PROGRAM ADDRESS becomes zero.
		0	HET clocks are running.
		1	HET clocks are stopped (power-down mode).
23–17	Reserved		Reads return 0 and writes have no effect.
16	clk_master/slave		This bit is used to synchronize multi-HETs. If set (HET is master), the HET outputs a signal to synchronize the prescalers of the slave HET. By default, this bit is reset, which means a slave configuration.
		0	HET is configured as a slave.
		1	HET is configured as a master.
			Note: This bit must be set to 1 for single-HET configuration.
15–9	Reserved		Reads return 0 and writes have no effect.
8	64-bit Access		Any operation mode (read/write):
		0	Timer RAM is accessed 32 bits at a time
		1	Timer RAM is accessed 64 bits at a time
7–3	Reserved		Reads return 0 and writes have no effect.

Table 12-7. Global Configuration Register (HETGCR) Field Descriptions (Continued)

Bit	Name	Value	Description
2	Debug Status Flag		This flag is set when the device test mode is set and a breakpoint is reached in the HET program. A debug request signal is asserted to the main CPU. Any operation mode (read): 0 *Read:* No HET instruction with a breakpoint has been reached since the flag was last cleared. *Write:* Writing a 0 to this bit has no effect. 1 *Read:* A HET instruction with a breakpoint has been reached since the flag was last cleared. *Write:* The bit is cleared.
1	Ignore Suspend		When ignore suspend = 0, the timer operation is stopped on suspend (the current timer instruction is completed). Timer RAM can be freely accessed during suspend. When set to 1, the suspend is ignored and the HET continues operating. 0 HET stops when a software breakpoint is encountered. 1 HET ignores software breakpoints.
0	Turn On/Off		When turn on/off = 0, the timer program stops executing. Turn-off is automatically delayed until the current timer program loop is completed. After enabling Turn-off, you must delay until the end of the timer program loop before putting the HET in power-down mode. This can be done by waiting until the HET PROGRAM ADDRESS becomes zero.Turn-off does not affect the content of the timer RAM, ALU registers, or control registers. Turn-off resets all flags. See Figure 12-36. 0 The HET is off. **Note: Turn-off does not affect the state of the pins. You must set/reset the timer pins when they are turned off, or re-initialize the timer RAM and control registers before a reset. After a CPU reset, the timer is turned off by default.** 1 The HET is on. **Note: When turn off/on equals 1, timer program execution starts synchronously to the Loop clock. In case of multiple HETs configuration, the slave HETs are waiting for the loop clock to come from the master before starting execution. Then, the timer address points automatically address 00h (corresponding to program start).**

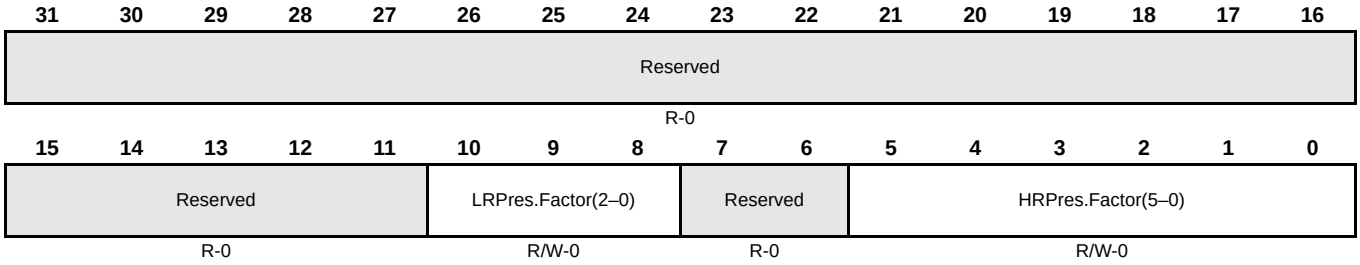
Figure 12-36. Multiple HETs Turn-On Sequence



12.5.2 Prescale Factor Register (HETPFR)

Figure 12-37 and Table 12-11 present the HETPFR.

Figure 12-37. Prescale Factor Register (HETPFR) [offset = 04h]



R = Read; WP = Write in any mode; *-n* = Value after reset

Table 12-8. Prescale Factor Register (HETPFR) Field Descriptions

Bit	Name	Value	Description
31–11	Reserved		Reads return 0 and writes have no effect.
10–8	LRPres.Factor(2–0)	0–7h	Loop resolution prescale factor code. The binary code programmed in the register gives the loop resolution (LR). The LR pre-scale factor code and the HR pre-scale factor code define the number of time slots available. See Table 12-9 for the loop resolution encoding format, calculated by: *Time Slots Available = HR Prescale Divide rate X Loop Resolution Prescale Divide rate*
8–6	Reserved		Reads return 0 and writes have no effect.
5–0	HRPres.Factor(5–0)	0–3Fh	HR prescale factor code. The binary code programmed in the register gives the HR. See Table 12-10 for the HR encoding format.

Table 12-9. Loop Resolution Encoding Format

Loop Resolution Prescale Factor Code			Loop-Res Prescale Divide Rate
2	1	0	
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	Reserved
1	1	1	Reserved

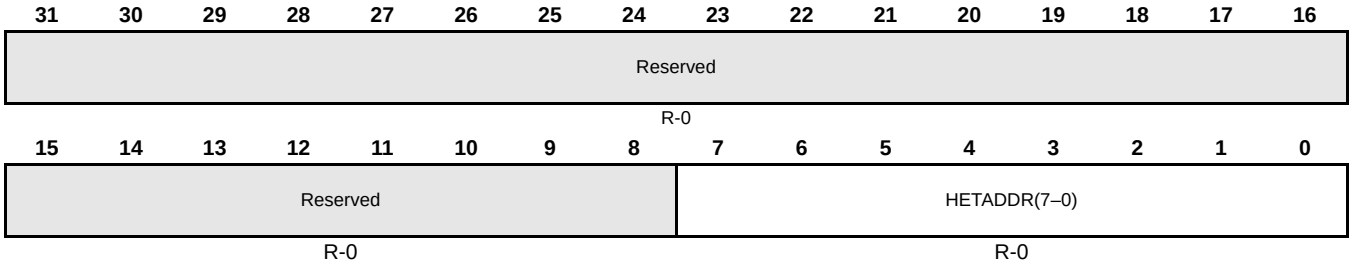
Table 12-10. HR Encoding Format

HR Prescale Factor code						HR Prescale Divide Rate
5	4	3	2	1	0	
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
0	0	0	1	0	0	5
:	:	:	:	:	:	:
:	:	:	:	:	:	:
1	1	1	1	0	1	62
1	1	1	1	1	0	63
1	1	1	1	1	1	64

12.5.3 HET Current Address Register (HETADDR)

Figure 12-38 and Table 12-11 present the register for HETADDR.

Figure 12-38. HET Current Address Register (HETADDR) [offset = 08h]



R = Read; *-n* = Value after reset

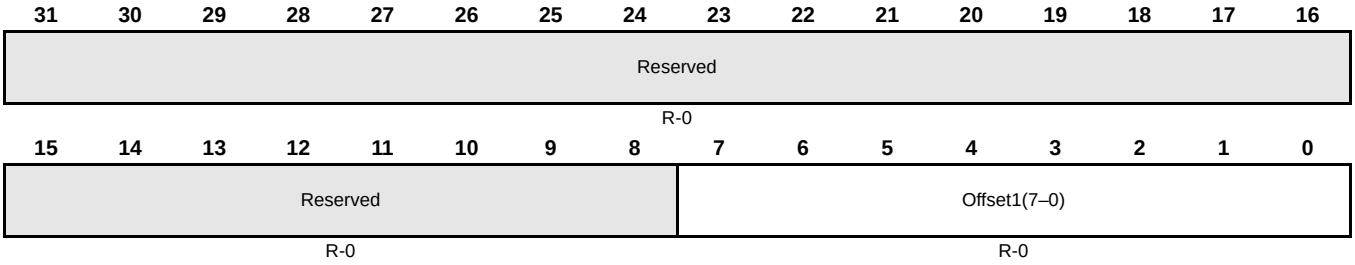
Table 12-11. HET Current Address Register (HETADDR) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return 0 and writes have no effect.
7–0	HETADDR(7–0)	0–FFh	HET address. A read of these bits in any operation mode returns the current HET program address register.

12.5.4 Offset Index Priority Level 1 Register (HETOFF1)

Figure 12-39 and Table 12-12 present the HETOFF1 register.

Figure 12-39. Offset Index Priority Level 1 Register (HETOFF1) [offset = 0Ch]



R = Read; -*n* = Value after reset

Table 12-12. Offset Index Priority Level 1 Register (HETOFF1) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return 0 and writes have no effect.
7–0	Offset(7–0)	0–FFH	Offset. HETOFF1[7:0] indexes the currently pending high-priority interrupt. Offset values and sources are listed below and the interrupt encoding format is presented in Table 12-13. *Any operation mode (read):* Read of these bits determines the pending HET interrupt; the corresponding flag (in the HETFLG) is also cleared. *Emulation mode (read):* Read of these bits determine the pending HET interrupt but the corresponding flag is not cleared.

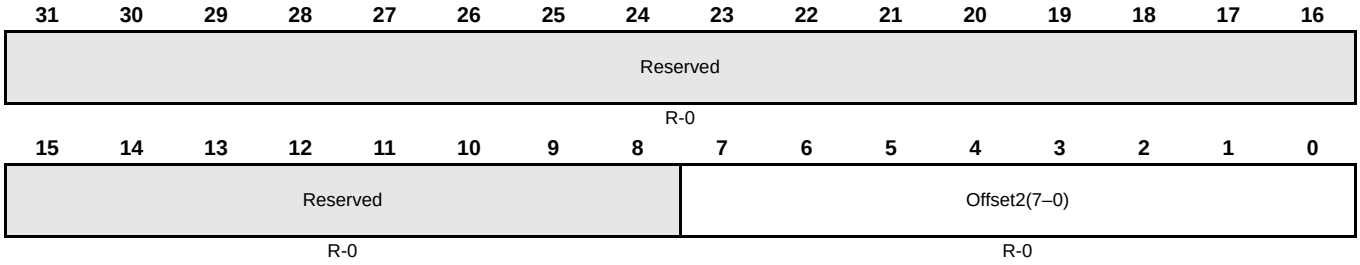
Table 12-13. Interrupt Offset Encoding Format

Source No.	Offset Value
no interrupt	0
Instruction 0, 32, 64...	1
Instruction 1, 33, 65...	2
:	:
Instruction 31, 63, 95...	32
Program Overflow	33
APCNT underflow:	34
APCNT overflow	35

12.5.5 Offset Index Priority Level 2 Register (HETOFF2)

Figure 12-40 and Table 12-14 describe the HETOFF2 register.

Figure 12-40. Offset Index Priority Level 2 Register (HETOFF2) [offset = 10h]



R = Read; *-n* = Value after reset

Table 12-14. Offset Index Priority Level 2 Register (HETOFF2) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return 0 and writes have no effect.
7–0	Offset2(7–0)	0–FFh	HETOFF2(7–0) indexes the currently pending low-priority interrupt. Offset values and sources are listed in Table 12-13. *Any operation mode (read):* A read of these bits in an operation mode determines the pending HET interrupt; the corresponding flag (in the HETFLG) is also cleared. *Emulation mode (read):* A read of these bits in emulation mode determines the pending HET interrupt, but the corresponding flag is not cleared.

12.5.6 Exception Control Register 1 (HETEXC1)

Figure 12-41 and Table describe the HETEXC1.

Figure 12-41. Exception Control Register (HETEXC1) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							APCNT Ovrfl Ena	Reserved							APCNT Undrfl Ena
R-0							R/W-0	R-0							R/W-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							Prgm Ovrfl Ena	Reserved					APCNT OVRfl Pry	APCNT Undrfl Pry	Prgm Ovrfl Pry
R-0							R/W-0	R-0					R/W-0	R/W-0	R/W-0

R = Read; W = Write; *-n* = Value after reset

Table 12-15. Exception Control Register (HETEXC1) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return 0 and writes have no effect.
24	APCNT Ovrfl Ena		APCNT overflow enable.
		0	The APCNT overflow exception is not enabled.
		1	The APCNT overflow exception is enabled.
23–17	Reserved		Reads return 0 and writes have no effect.
16	APCNT Undrfl Ena		APCNT underflow enable.
			Any operation mode (read/write):
		0	APCNT underflow exception is not enabled.
		1	APCNT underflow exception is enabled.
15–9	Reserved		Reads return 0 and writes have no effect.
8	Prgm Ovrfl Ena		Program overflow enable.
		0	The program overflow exception is not enabled.
		1	The program overflow exception is enabled.
7–3	Reserved		Reads return 0 and writes have no effect.
2–0	Exception Priority Level bits		Used to select the priority of any of the three potential exception sources.
		0	The exception priority level is 2.
		1	The exception priority level is 1.

12.5.7 Exception Control Register 2 (HETEXC2)

Figure 12-42 and Table 12-16 describe the HETEXC2.

Figure 12-42. Exception Control Register 2 (HETEXC2) [offset = 18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved													APCNT Ovrfl flg	APCNT Undfl flg	Prgm Ovrfl flg
R-0													R/C-0	R/C-0	R/C-0

R = Read; C = Clear; -*n* = Value after reset

Table 12-16. Exception Control Register 2 (HETEXC2) Field Descriptions

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	APCNT Ovrfl flg		APCNT overflow flag
		0	*Read:* No exception has occurred since the flag was cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An exception has occurred since the flag was cleared. *Write:* The bit is cleared.
1	APCNT Undrfl flg		APCNT underflow flag
		0	*Read:* Exception has not occurred since the flag was cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* Exception has occurred since the flag was cleared. *Write:* The bit is cleared.
0	Prgrm Overfl flg		Program overflow flag
		0	*Read:* No exception has occurred since the flag was cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An exception has occurred since the flag was cleared. *Write:* The bit is cleared.

12.5.8 Interrupt Priority Register (HETPRY)

Figure 12-43 and Table 12-17 describe the HETPRY register.

Figure 12-43. Interrupt Priority Register (HETPRY) [offset = 1Ch]

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

R = Read; WP = Write in privileged mode only; -*n* = Value after reset

Table 12-17. Offset Index Priority Level 2 Register (HETOFF2) Field Descriptions

Bit	Name	Value	Description
31–0	HETPRY[31:0]		HET priority level bits. These bits are used to select the priority of any of the 32 potential interrupt sources coming from the instructions.
		0	The software priority level is 2.
		1	The software priority level is 1.

12.5.9 HET Interrupt Flag Register (HETFLG)

Figure 12-44 and Table 12-18 describe the HETFLG.

Figure 12-44. HET Interrupt Flag Register (HETFLG) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HETFLAG[31:16]															
R/C-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HETFLAG[15:0]															
R/C-0															

R = Read; C = Clear; -*n* = Value after reset

Table 12-18. HET Interrupt Flag Register (HETFLG) Field Descriptions

Bit	Name	Value	Description
31–0	HETFLAG[31:0]		Interrupt flag. These bits are set when an interrupt condition has occurred and when the interrupt enable bit (in the instruction) is set. The flag is also set by the five LSBs of the instruction address that generated the interrupt. To clear the flag, these bits must be set to 1 or the reading of the corresponding HETOFFx register will automatically clear the flag.
		0	*Read:* No HET instruction with an interrupt has been reached since the flag was last cleared. *Write:* The bit is unchanged.
		1	*Read:* A HET instruction with an interrupt has been reached since the flag was last cleared. *Write:* The bit is cleared.

12.5.10 HR Share Control Register (HETHRSH)

Figure 12-45 and Table 12-19 describe the HETHRSH register.

Figure 12-45. HR Share Control Register (HETHRSH) [offset = 2Ch]

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

R = Read; W = Write; -*n* = Value after reset

Table 12-19. HR Share Control Register (HETHRSH) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return 0 and writes have no effect.
11–0	HR Sharex		HR share bits. These bits enable the share of the same pin for two HR structures. For example, if bit HR share 1/0 is set, the pin HET0 will then be connected to both HR structures 0 and 1.
		0	Pins are not shared.
		1	Pins are shared.
			If HR share bits are used, pins not connected to HR structures can be accessed as general inputs/outputs

12.5.11 HR XOR-Share Control Register (HETXOR)

Figure 12-46 and Table 12-20 describe the HETXOR register.

Figure 12-46. HR XOR-Share Control Register (HETXOR) [offset = 30h]

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

R = Read; W = Write; *-n* = Value after reset

Table 12-20. HR XOR-Share Control Register (HETXOR) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved		Reads return 0 and writes have no effect.
11–0	HR XOR-Sharex		HR XOR-share bits These bits enable the XOR-share of the same pin for two HR structures. For example, if bit HR XOR-share 1/0 is set, the pin HET0 will then be commanded by a logical XOR of both HR structures 0 and 1.
		0	Pins are not XOR-shared.
		1	Pins are XOR-shared.
			If HR share bits are used, pins not connected to HR structures can be accessed as general inputs/outputs

12.5.12 HET Direction Register (HETDIR)

Figure 12-47 and Table describe the HETDIR register.

Figure 12-47. HET Direction Register (HETDIR) [offset = 34h]

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

R = Read; W = Write; *-n* = Value after reset

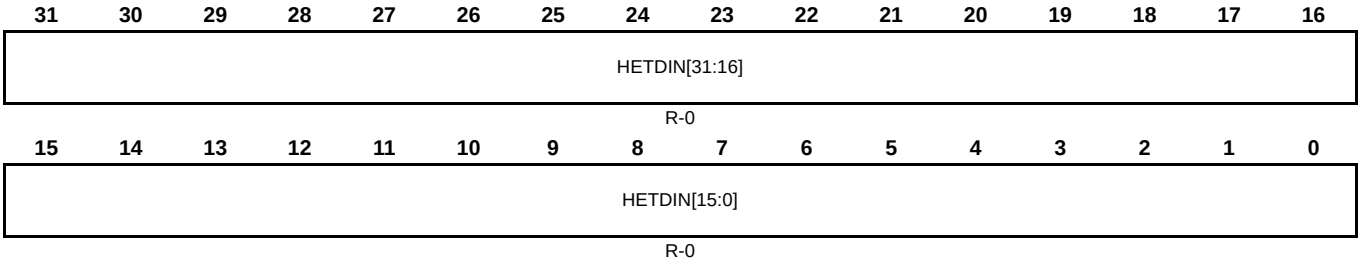
Table 12-21. HET Direction Register (HETDIR) Field Descriptions

Bit	Name	Value	Description
31–0	HET DIR[31:0]		Input/output direction These bits are used to select the direction of the HET pins.
		0	HET pin configured as input.
		1	HET pin configured as output.

12.5.13 HET Data Input Register (HETDIN)

Figure 12-48 and Table 12-22 describe the HETDIN register.

Figure 12-48. HET Data Input Register (HETDIN) [offset = 38h]



R = Read; *-n* = Value after reset

Table 12-22. HET Data Input Register (HETDIN) Field Descriptions

Bit	Name	Value	Description
31–0	HET DIN[31:0]		HET data input. Read operation mode (writes have no effect):
		0	The pin is at logic low (0).
		1	The pin is at logic high (1).

12.5.14 HET Data Output Register (R-Write) (HETDOUT)

Figure 12-49 and Table 12-23 describe the HETDOUT register.

Figure 12-49. HET Data Output Register (R-Write) (HETDOUT) [offset = 3Ch]

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

R = Read; W = Write; *-n* = Value after reset

Table 12-23. HET Data Output Register (R-Write) (HETDOUT) Field Descriptions

Bit	Name	Value	Description
31–0	HETDOUT[31:0]		HET data output. This bit is used to set/reset the HET pins. A read to this register gives the value of the corresponding HETDSET register (Section 12.5.14).
		0	The pin is at logic low (0).
		1	The pin is at logic high (1).

12.5.15 HET Data Set Register (R-Set) (HETDSET)

Figure 12-50 and Table 12-24 describe the HETDSET register.

Figure 12-50. HET Data Set Register (R-Set) (HETDSET) [offset = 40h]

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

R = Read; S = Set; *-n* = Value after reset

Table 12-24. HET Data Set Register (R-Set) (HETDSET) Field Descriptions

Bit	Name	Value	Description
31–0	HETDSET[31:0]		HET data set. These bits are used to set the HET pins to the HIGH level. A read of HETDSET reflects the contents of the HETDOUT register (Section 12.5.14).
		0	*Read:* The pin is at logic low (0). *Write:* The corresponding bit in HETDOUT is unchanged.
		1	*Read:* The pin is at logic high (1). *Write:* The corresponding bit in HETDOUT is set to 1.

12.5.16 HET Data Clear Register (R-Clear) (HETDCLR)

Figure 12-51 and Table 12-25 describe the HETDCLR register.

Figure 12-51. HET Data Clear Register (R-Clear) (HETDCLR) [offset = 44h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HETDCLR[31:16]															
R/C-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HETDCLR[15:0]															
R/C-0															

R = Read; C = Clear; -*n* = Value after reset

Table 12-25. HET Data Clear Register (R-Clear) (HETDCLR) Field Descriptions

Bit	Name	Value	Description
31–0	HETDCLR[31:0]		HET data clear. These bits are used to clear the HET pins to 0. A read of HETDCLR reflects the contents of HETDOUT (Section 12.5.14).
		0	*Read:* The pin is at logic low (0). *Write:* The corresponding bit in HETDOUT is unchanged.
		1	*Read:* The pin is at logic high (1). *Write:* The corresponding bit in HETDOUT is set to 1.

12.5.17 HET Pull Disable Register (HETPULDIS)

This register enables or disables the pull control function of the pins. Figure 12-52 and Table 12-26 describe this register.

Figure 12-52. HET Pull Disable Register (HETPULDIS) [offset = 4Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HETPULDIS[31:16]															
R/W-x															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HETPULDIS[15:0]															
R/W-x															

R = Read; W = Write; *-x* = Device specific tie off after reset (please read device-specific pin description)

Table 12-26. HET Pull Disable Register (HETPULDIS) Field Descriptions

Bit	Name	Value	Description
31–0	HETPULDIS[31:0]		HET pull disable.
		0	The pull function is enabled.
		1	The pull function is disabled.

12.5.18 HET Pull Select Register (HETPSL)

Values in this register will disable or enable the input buffer when the pull logic is disabled.

Figure 12-53 and Table 12-27 describe this register.

Figure 12-53. HET Pull Select Register (HETPSL) [offset = 50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HETPSL[31:16]															
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HETPSL[15:0]															
R/W-0															

R = Read; W = Write; -*n* = Value after reset

Table 12-27. HET Pull Select Register (HETPSL) Field Descriptions

Bit	Name	Value	Description
31–0	HETPSL[31:0]		HET pull select
		0	Input buffer for HET is disabled if PULLDIS = 1.
		1	Input buffer for HET is enabled if PULLDIS = 1.

12.5.19 HET Loopback Pair Select Register (HETLPBSEL)

This register selects whether the loopback mode on the dedicated pins should be enabled and analog or digital loopback can be configured. Figure 12-54 and Table 12-28 describe this register.

Figure 12-54. HET Loopback Pair Select Register (HETLPBSEL) [offset = 60h]

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

R = Read; W = Write; *-n* = Value after reset

Table 12-28. HET Loopback Pair Select Register (HETLPBSEL) Field Descriptions

Bit	Name	Value	Description
31–16	HETLPBTYPE[2x+1/2x]		Loopback type select
		0	Digital loopback is selected.
		1	Analog loopback is selected.
15–0	HETLPBSEL[2x+1/2x]		Loopback pair select
		0	No loopback mode selected.
		1	High resolution structures of pin 2x and 2x+1 are connected together. The direction is given by the HETLPBDIR register.

12.5.20 HET Loopback Pair Direction Register (HETLPBDIR)

This register selects the direction of the loopback mode on the dedicated pins. The loopback direction is independent of the HETDIR register setting. Figure 12-55 and Table 12-29 describe this register

Figure 12-55. HET Loopback Pair Direction Register (HETLPBDIR) [offset = 64h]

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

R = Read; W = Write; *-n* = Value after reset

Table 12-29. HET Loopback Pair Direction Register (HETLPBDIR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return 0 and writes have no effect.
15–0	HETLPBDIR[2x+1/2x]		Loopback direction select.
		0	HR structure 2x is input and 2x+1 is output.
		1	HR structure 2x+1 is input and 2x is output.

12.6 Instruction Set

Table 12-30 presents a list of the instructions in the HET instruction set. Table 12-31 presents the flags generated by the instructions, and Table 12-32 lists the instructions as to whether they are interrupt capable or not. The following sections discuss the HET instruction set in detail.

Table 12-30. Instruction Summary

Abbreviation	Instruction Name	Opcode	Sub-Opcode	Cycles	Reference
ACMP	Angle Compare	Ch	-	1	Page 449
ACNT	Angle Count	9h	-	2	Page 451
ADCNST	Add Constant	5h	-	2	Page 454
ADM32	Add Move 32	4h	C5=1	1 or 2	Page 455
APCNT	Angle Period Count	Eh	-	1 or 2	Page 459
BR	Branch	Dh	-	1	Page 462
CNT	Count	6h	-	1 or 2	Page 464
DADM64	Data Add Move 64	2h	-	2	Page 467
DJZ	Decrement and Jump if -zero	Ah	P[6-5]=10	1	Page 469
ECMP	Equality Compare	0h	C[6-5]=00	1	Page 471
ECNT	Event Count	Ah	P[6-5]=01	1	Page 474
MCMP	Magnitude Compare	0h	C[6-5]=1X	1	Page 476
MOV32	Move 32	4h	C5=0	1 or 2	Page 479
MOV64	Move 64	1h	-	1	Page 483
PCNT	Period/Pulse Count	7h	-	1	Page 485
PWCNT	Pulse Width Count	Ah	P[6-5]=11	1	Page 489
RADM64	Register Add Move 64	3h	-	1	Page 491
SCMP	Sequence Compare	0h	C[6-5]=01	1	Page 494
SCNT	Step Count	Ah	P[6-5]=00	3	Page 497
SHFT	Shift	Fh	-	1	Page 499
WCAP	Software Capture Word	Bh	-	1	Page 502

Table 12-31. FLAGS Generated by Instruction

	Flag Name	Set / Reset by	Used by
Z	Z flag	CNT, APCNT, PCNT, ACNT, SCNT, SHFT	ACMP, ECMP, SCMP, MCMP, ACNT, BR, SHFT
X	X flag	ACMP	SCMP
SWF 0-1	Step width flags	SCNT	ACNT
NAF	New angle flag	ACNT	NAF_global
NAF_global	New angle flag (global)	NAF	ACNT, CNT, ECNT, BR, ECMP, MCMP

	Flag Name	Set / Reset by	Used by
ACF	Acceleration flag	ACNT	SCNT, ACNT
DCF	Deceleration flag	ACNT	SCNT, ACNT
GPF	Gap flag	ACNT	APCNT, ACNT

The instructions capable of generating SW interrupts are listed in Table 12-32.

Table 12-32. Interrupt-Capable Instructions

Interrupt-Capable Instructions	Non-Interrupt-Capable Instructions
ACMP	ADCNST
ECMP	ADM32
SCMP	DADM32
MCMP	MOV32
CNT	MOV64
ECNT	RADM64
ACNT	SCNT
APCNT	
PWCNT	
PCNT	
DJZ	
WCAP	
SHFT	
BR	

12.6.1 Abbreviations

Abbreviations marked with a star (*) are available only on specific instructions.

BRK	Defines the software breakpoint for the device software debugger. Default: OFF. Location: Program field [20]
Next	Defines the program address of the next instruction in the program flow. This value may be a label or an 8-bit unsigned integer. Default: Current instruction + 1. Location: Program field [19:12]
Remote(1)	Determines the 8-bit address of the remote address for the instruction. Default: Current instruction + 1. Location: Program field [7:0]
Control	Determines whether the immediate data field [24:0] is cleared when it is read. When the bit is not set, reads do not clear the immediate data field. Default: OFF. Location: Control field [21]
En_pin_action(1)	Determines whether the selected pin is ON so that the action occurs on the chosen pin. Default: OFF. Location: Control field [20]
Cond_addr(1)	Conditional address: (Optional) Defines the address of the next instruction when the condition occurs. Default: Current address + 1. Location: Control field [19:12]
Pin(1)	Pin Select: Selects the pin on which the action occurs. Enter the pin number.(1) Default: pin 0. Location: Control field [11:7] except PCNT,
U	Reading a bit marked with U will return an indeterminate value.

1 The format CC{pin number} is also supported.

12.6.2 Encoding Formats and Bits

Table 12-33 through Table 12-38 provide information about encoding formats and bits.

Table 12-33. PIN Encoding Format

MSB				LSB	Pin Select
0	0	0	0	0	Selects HET0
0	0	0	0	1	Selects HET1
(each pin may be selected by writing its number in binary)					
1	1	1	1	0	Selects HET30
1	1	1	1	1	Selects HET31

Reg(1)	Register select: Selects the register for data comparison and storage. Default: No register (none). Location: Control field [2:1] except CNT

1 The format CC{pin number} is also supported.

Table 12-34. Register Bit Field Encoding Format

Register[1]	C[2]	C[1]
A	0	0
B	0	1
T	1	0
None	1	1

1 The register bits field could be placed either in the program field (CNT), or in the control field (all others' instructions use register field).

Action[1] (Two action option) Either sets or clears the pin
Default: Clear.
Location: Control field [4]

1 The format CC{pin number} is also supported.

Table 12-35. PIN Action Bit Field (2 options)

Action	C[4]
Clear	0
Set	1

Action[1] (Four-action option) Either sets, clears, pulse high or pulse low on the pin. Pulse high occurs when the pin is set on the compare and toggles at the overflow.
Default: Clear.
Location: Control field [4:3]

1 The format CC{pin number} is also supported.

Table 12-36. PIN Action Bit Field (4 options)

Action	C[4](1)	C[3](2)
Clear	0	0
Set	1	0
Pulse Low	0	1
Pulse High	1	1

1 Bit C[4] is also called enable pin action.
2 C[3] is also called opposite pin action.

hr_lr(1) Specifies high/low data resolution. If the hr_lr field is high, the instruction implements the hr_data field (when the action is carried out on an HR pin). If the hr_lr field is low, the hr_data field is ignored.

Default: High.
Location: Program field [7]

1 The format CC{pin number} is also supported.

Table 12-37. High-Low Resolution Bit Field

hr_lr	Prog. field [7]
Low	1
High	0

prv(1) Specifies the initial value defining the previous pin-level bit for the first edge detect performed by the instruction. The edge detect is performed by comparing the current pin value to the value stored in the previous pin-level bit. A value of ON sets the previous pin-level bit to 1. A value of OFF sets the initial value of the previous (prv) bit to 0. After the initial comparison, the value of the prv bit is set or reset by the system.
Default: OFF.
Location: Control field [20]

cntl_val(1) Available for the DADM64, MOV64, and RADM64, this bits field allows you to specify the replacement value for the remote control field.

comp_mode(1) Specifies the compare mode. This field is used with the 64-bit move instructions. The field ensures that the sub-opcodes are moved correctly.
Default: ECMP.
Location: Control field [6:5]

1 The format CC{pin number} is also supported.

Table 12-38. Comp_mode Bit Field

comp_mode	C[6]	C[5]
ECMP	0	0
SCMP	0	
MCMP	1	
ACMP	1	

12.6.3 Instruction Description

The following sections provide the details for each instruction.

12.6.3.1 ACMP (Angle Compare)

Syntax

```
ACMP {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}]
pin={pin number}
[action={CLEAR | SET}]
reg={A | B | T | NONE}
[irq ={OFF | ON}]
data={20-bit unsigned integer}
}
```

Opcode Ch, [P11:P8]

Table 12-39. ACMP Program Field (P31:P0)

31–22	21	20	19–12	11–8	7–0
0	Res	BRK	Next program address	Opcode	Reserved
10	1-U	1	8	4	8

Table 12-40. ACMP Control Field (C31:C0)

31–22	21	20	19–12	11–7	6	5	4	3	2–1	0
0	Control	En. pin action	Conditional address	Pin select	Res.	Cout prv	Pin action	Res.	Register select	Int. ena.
10	1	1	8	5	1	1	1	1	1	1

Table 12-41. ACMP Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles One

Register modified Selected register (A, B, or T)

Description The purpose of the comparison is to assert pin action when the angle compare value lies between the old counter value and the new counter value (held in the selected register). Since the angle increment

varies from one loop resolution clock to another, an exact equality test cannot be applied. Instead, the following inequality is used to determine the occurrence of a match:

Old counter value < *Angle compare value* ≤ *New counter value*

This is done by performing following comparisons:

Selected register value minus angle increment < angle compare value
Angle compare value ≤ selected register value

register — Register B is recommended for typical applications with ACMP.

irq — Specifies whether or not an interrupt is generated. Specifying ON generates an interrupt when the edge state is satisfied and the gap flag is set. Specifying OFF prevents an interrupt from being generated. Default: OFF.

data — Specifies the 20-bit angle compare value.

Execution

```
X = 0;
If (Data field value ≤ Selected register value)
    Cout = 0;
else
    Cout = 1;
If (Z == 0 AND (Selected register value - Angle Inc. < Data field
value) AND Cout == 0)
- or -
     (Z == 1 AND (Cout_prv == 1 OR Cout == 0)))
    {
    X = 1;
    If (Enable Pin Action == 1)
          Selected Pin = Pin Action AT next loop resolution clock;
    If (Interrupt Enable == 1)
          SW interrupt flag = 1;
    Jump to Conditional Address;
    }
else
    Jump to Next Program Address;
Cout_prv = Cout (always executed)
```

Note: Carry-Out Signal (Cout)
Cout is the carry-out signal of the adder. Even if it is not a flag, it is valid all along ACMP instruction execution.

Angle inc. = NAF_global or hardware angle generator 4-bit input.

12.6.3.2 ACNT (Angle Count)

Syntax

```
ACNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
edge={RISING | FALLING}
[irq={OFF | ON}]
[control={OFF | ON}]
[prv={OFF | ON}]
gapend=20-bit unsigned integer
[data=20-bit unsigned integer]
}
```

Opcode: 9h, [P11:P8]

Table 12-42. ACNT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–1	0
0	Res	BRK	Next program address	Opcode	Edge-select.	Reserved	Int. ena.
10	1-U	1	8	4	1	6	1

Table 12-43. ACNT Control Field (C31:C0)

31–22	21	20	19–0
0	Control	Prv.	Gap End
10	1	1	20

Table 12-44. ACNT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles: Two, as follows:

- First cycle: Angle increment condition and gap end comparison.
- Second cycle: Gap start comparison.

Register Modified: Register B (angle value)

Description: This instruction defines a specialized virtual timer used after SCNT and APCNT to generate an angle-referenced time base that is synchronized to an external signal (that is, a toothed wheel signal). ACNT

uses pin HET24 exclusively**.** The edge select must be the same as the HET24 edge which was selected in the previous APCNT.

ACNT refers to the same step width selection that the previous SCNT saved in flags SWF0 and SWF1 (see information on SCNT).

ACNT detects period variations of the external signal measured by APCNT and compensates related count errors.

A period increase is flagged in the deceleration flag (DCF). A period decrease is flagged in the acceleration flag (ACF). If no variation is detected, ACNT increments the counter value each time SCNT reaches its target.

If acceleration is detected, ACNT increments the counter value on each timer resolution. If deceleration is detected ACNT does not increment and is thus saturated.

ACNT also specifies the gap end angle value defining the end value of a gap range in ACNT where period measurements in APCNT are temporarily stopped to mask singularities in the external signal. ACNT uses register A containing gap start and register B to store the counter value.

Edge — Specifies the edge for the input capture pin (HET[24]).

Action	P7	Edge Select
Rising	1	Detects a rising edge of HET24
Falling	0	Detects a falling edge of HET24

Irq — ON generates an interrupt when the edge state is satisfied and the gap flag is set. OFF prevents an interrupt from being generated.
Default: OFF.

gapend — Defines the 20-bit end value of a gap range. The start value is defined in the SCNT instruction.
GAPEND = (Step Value * (# of teeth on the toothed wheel + # of missing teeth)) -1

data — Specifies the 20-bit initial count value for the data field.
Default: 0.

Note: Target Edge Field
The target edge field represents the three LSBs of data field register in case of step width = 8, four LSBs for step width = 16, five LSBs for step width = 32 and six LSBs for step width = 64.

Execution

```
Increment Condition: ((Z = 1 AND DCF = 0) OR ACF = 1)
Pin Edge Condition: Specified edge detected on HET24
Target Edge Condition: (Target Edge field in data field = 0) AND (An-
gle Increment condition is true) AND (GPF = 0)

If (Angle Increment Condition) is false
    {
    NAF_global = 0;
    Register B = Data field register;
    }

else
    {
    NAF_global = 1;
    If (Counter value != GapEnd - 1)
          {
```

```
            Register B = Data field register + 1;
            Data Field Register = Counter value + 1;
            }
      else
            {
            Register B = 0;
            Data Field Register = 0;
            If (ACF == 0)
              DCF = 1;
            }
      }
Z = 0;
If (Data field register == GapStart)
      {
      GPF = 1;
      If (Target Edge condition is true)
            {
            ACF = 0;
            If ((specified edge is not detected on pin HET24) AND (data
            field register != 0) AND (ACF == 0))
              DCF = 1;
            }
      If (specified edge is detected on pin HET24)
            {
            DCF = 0;
            If ((data field register != 0) AND (DCF == 0))
              ACF = 1;
            If (GPF == 1)
              {
              GPF = 0;
              Z = 1;
              If (Interrupt Enable == 1)
                SW interrupt flag = 1;
              }
            }
      }
Prv bit = Current HET24 pin level;
```

12.6.3.3 ADCNST (Add Constant)

Syntax

```
ADCNST {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
remote={label | 8-bit unsigned integer}
min_off=20-bit unsigned integer
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

Opcode 5h, [P11:P8]

Table 12-45. ADCNST Program Field (P31:P0)

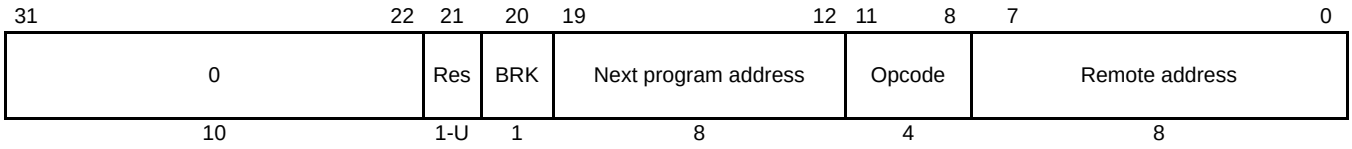


Table 12-46. ADCNST Control Field (C31:C0)

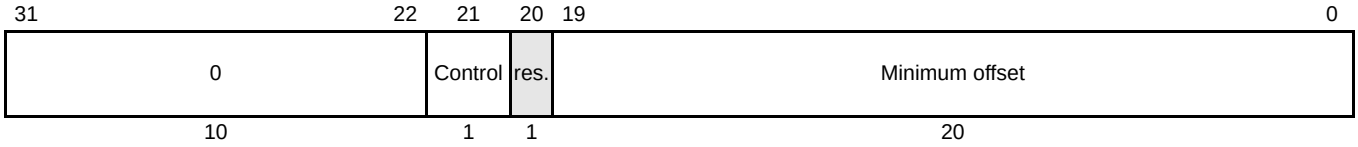
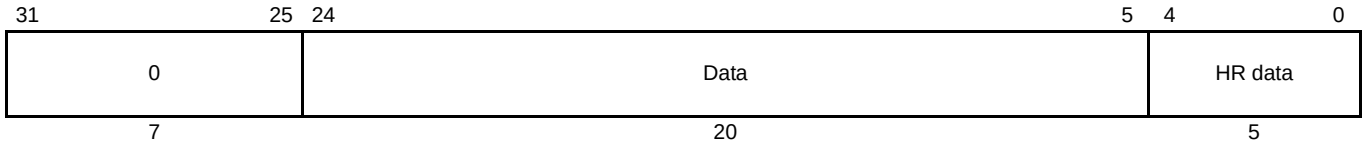


Table 12-47. ADCNST Data Field (D31:D0)



Cycles Two

Register Modified Register T (implicitly)

Description ADCNST is an extension of ADMOV32. ADCNST first checks whether the data field value at the remote address is zero; it then performs different adds and moves on the result. ADCNST is typically used to extend the counter value of PWCNT.

- min_off — A 20-bit constant value that is added to the data field value if the remote data field is null.
- data — A 20-bit value that is always added to the remote data field.
  Default: 0.
- hr_data — Five least significant bits of the data addition to the remote data field.
  Default: 0.

Table 12-56 and Table 12-57 illustrate the behavior of ADCNST if the remote data field is or is not zero.

Figure 12-56. ADCNST Operation If Remote Data Field[24:5] Is Not Zero

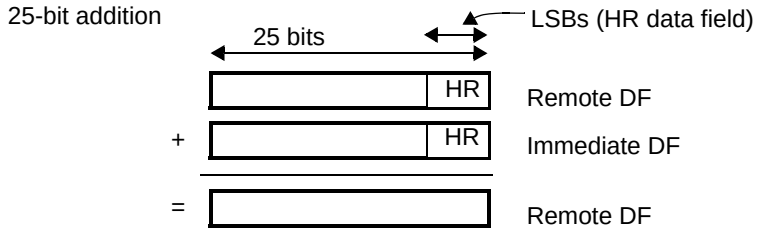
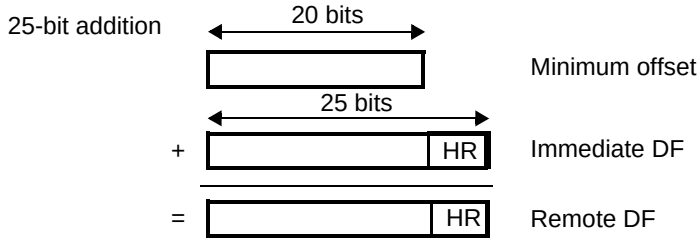


Figure 12-57. ADCNST Operation if Remote Data Field [24:5] Is Zero




```
Execution If (Remote Data Field Value [24:5] != 0)
        Remote Data Field = Immediate Data Field + Remote Data Field;
    else
        Remote Data Field = Immediate Data Field + min. offset(bits
        C19:C0);
    Jump to Next Program Address;
```

12.6.3.4 ADM32 (Add Move 32)

Syntax

ADM32 {
[brk={OFF | ON}]
[next={*label* | *8-bit unsigned integer*}]
remote={*label* | *8-bit unsigned integer* }
[control={OFF | ON}]
[init={OFF | ON}]
type={IM&REGTOREG | REM&REGTOREG |
IM&REMTOREG | IM&REGTOREM}
reg={A | B | T | NONE}
data=*20-bit unsigned integer*
[hr_data=5-*bit unsigned integer*]
}

Opcode 4h, [P11:P8];

Sub-Opcode *1h, [C5]*

Table 12-48. ADM32 Program Field (P31:P0)

31 22	21	20	19 12	11 8	7 0
0	Res	BRK	Next program address	Opcode	Remote address
10	1-U	1	8	4	8

Table 12-49. ADM32 Control Field (C31:C0)

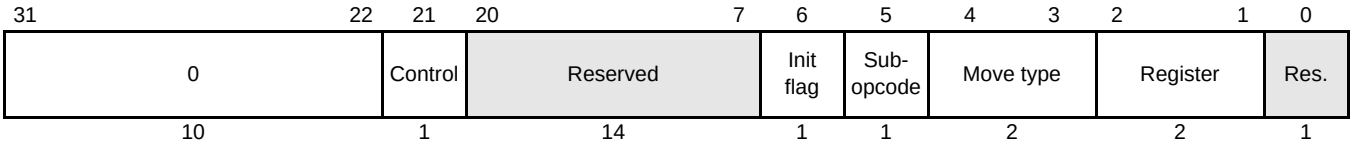
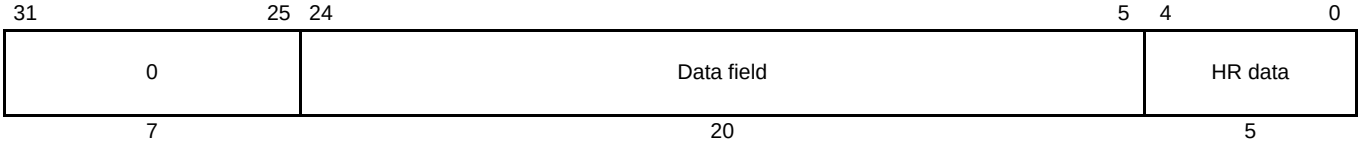


Table 12-50. ADM32 Data Field (D31:D0)



Cycles — One or two cycles (see Table 12-51)

Register Modified — Selected register (A, B, or T)

Description — This instruction modifies the selected ALU register or data field values at the remote address depending on the move type. The modified value results from adding the immediate or remote data field to the ALU register or the remote data field, depending on the move type. Table description shows the C2 and C1 bit encoding for determining which register is selected.

.

init — (Optional) Determines whether or not system flags are initialized. A value of ON reinitializes the following system flags to these states:
Acceleration flag (ACF) = 0
Deceleration flag (DCF) = 1
Gap flag (GPF) = 0
New angle flag (NAF) = 0
A value of OFF results in no change to the system flags.
Default: OFF

type — Specifies the move type to be executed.

Table 12-51. Move Types for ADM32

Type	C4	C3	Add			Destination(s)	Cycles
IM&REGTOREG	0	0	Imm. data field	+	Reg. A, B, or T	Register A, B, or T	1
REM&REGTOREG	0	1	Remote data field	+	Reg. A, B, or T	Register A, B, or T	2
IM&REMTOREG	1	0	Imm. data field	+	Remote data field	Register A, B, or T	2
IM&REGTOREM	1	1	Imm. data field	+	Reg. A, B, or T	Remote data field	1

If selected register is T, the operation is a 25-bit Addition/move. If A or B register is selected, it is limited to 20-bit operation since A and B only support 20-bit.

data — Specifies the 20-bit integer value for the immediate data field.

hr_data — Specifies the 5 least significant bits of the immediate data field.
Default: 0.

Execution

```
switch (C4:C3)
     {
     case 00:
            Selected register = Selected register + Immediate Data Field;
```

```
        case 01:
              Selected register = Selected register + Remote Data Field;
        case 10:
              Selected register = Immediate Data Field + Remote Data Field;
        case 11:
              Remote Data Field = Selected register + Immediate Data Field;
        }
    If (Init Flag == 1)
        {
        ACF = 0;
        DCF = 1;
        GPF = 0;
        NAF = 0;
        }
    else
        All flags remain unchanged;
    Jump to Next Program Address;
```

Register Modified — Selected register (A, B or T)

Figure 12-58 through Figure 12-61 illustrate the ADM32 operation for various cases.

Figure 12-58. ADM32 Add and Move Operation for IM&REGTOREM (Case 00)

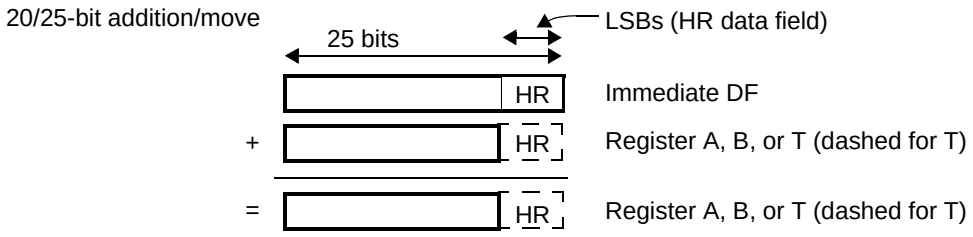


Figure 12-59. ADM32 Add and Move Operation for REM&REGTOREG (Case 01)

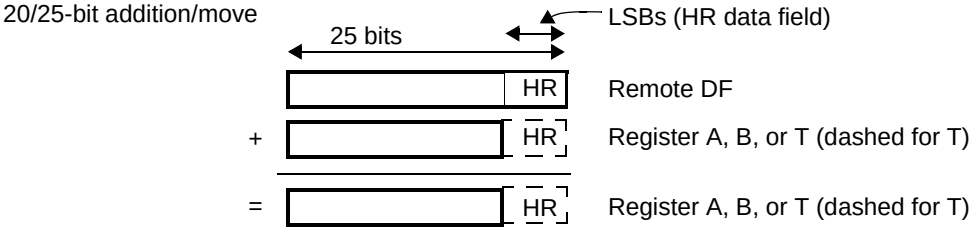


Figure 12-60. ADM32 Add and Move Operation for IM&REMTOREG (Case 10)

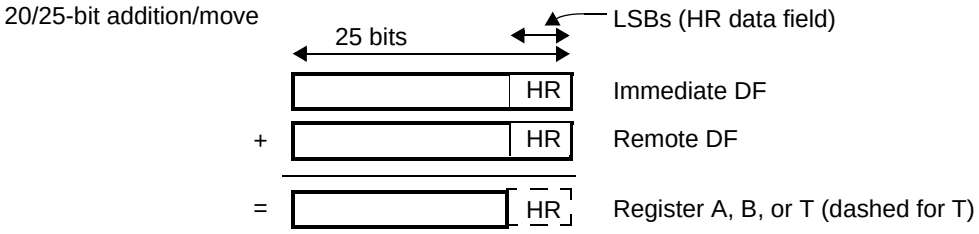
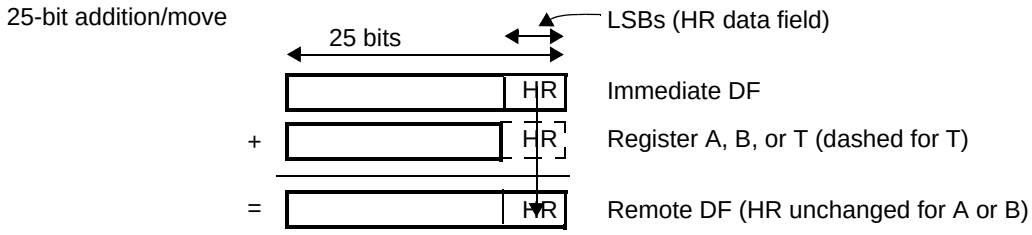


Figure 12-61. ADM32 Add and Move Operation for IM&REGTOREM (Case 11)



12.6.3.5 APCNT (Angle Period Count)

Syntax

```
APCNT {
[brk={OFF| ON}]
[next={label | 8-bit unsigned integer}]
[irq={OFF | ON}]
type={FALL2FALL | RISE2RISE}
[control={OFF | ON}]
prv={OFF | ON}
period=20-bit unsigned integer
[data=20-bit unsigned integer]
}
```

Opcode: Eh, [P11:P8]

Table 12-52. APCNT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–5	4–0
0	Res	BRK	Next program address	Opcode	Int. ena.	Edge select	Reserved
10	1-U	1	8	4	1	2	5

Table 12-53. APCNT Control Field (C31:C0)

31–22	21	20	19–0
0	Control	Prv.	Period count
10	1	1	20

Table 12-54. APCNT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles: One or two cycles
One cycle (normal operation) two cycles (edge detected)

- Cycle 1: edge detected (normal operation)
- Cycle 2: edge detected and GPF = 1 and underflow condition is true

Register Modified — Register A and T (implicitly)

Description — This instruction is used before SCNT and ACNT to generate an angle-referenced time base synchronized to an external signal (that is, a toothed wheel signal). It is assumed that the pin and edge selections are the same for APCNT and ACNT.

APCNT is restricted to pin HET24. The toothed wheel must then be connected to pin HET24.

APCNT uses the gap flag (GPF) defined by ACNT to start or stop captures in the period count field [C19:C0]. When GPF=1, the previous period value is held in the control field and in register T. When GPF= 0, the current period value is captured in the control field and in register T.

APCNT uses the step width flags (SWF0 and SWF1) defined by SCNT to detect period durations shorter than one step and then disables capture.

The edge select encoding is shown in Table 12-55.

irq — ON generates an interrupt when the edge state is satisfied. OFF prevents an interrupt from being generated.
Default: OFF.

type — Specifies the edge type that triggers the instruction.
Default: Fall2Fall.

Table 12-55. Edge Select Encoding for APCNT

type	P6	P5	Selected Condition
Fall2Fall	1	0	Falling edge
Rise2Rise	1	1	Rising edge

period — Contains the 20-bit count value from the previous APCNT period.

data — 20-bit value serving as a counter.
Default: 0.

Execution

```
Z = 0;
If (Data field register != FFFFFh)
     {
     Register A = Data field register + 1;
     Data field register = Data field register + 1;
     }
elseIf (specified edge not detected on HET24)
     {
     Register A = FFFFFh;
     APCNT Ovflw flag = 1;
     }
If (specified edge detected on HET24)
     {
     Z = 1;
     If (Data field register == FFFFFh)
            {
            Register A = FFFFFh;
            Register T = FFFFFh;
            Period count = FFFFFh;
            }
     elseIf (GPF == 0 AND Data Field register ≥ Step width)
```

```
                            {
                            Register A = Data field register + 1;
                            Register T = Register A;
                            Period count = Register T;
                            If (Interrupt Enable == 1)
                              SW interrupt flag = 1;
                            }
                      If (GPF == 1)
                            Register T = Period count;
    If (Data Field register < Step width)
                            {
                            Register T = Period count;
                            APCNT Undflw flag = 1;
                            Data field register = 00000h;
                            }
                      }
                  else
                      Register T = Period count;
                  Prv bit = Current HET24 pin level;
                  Jump to Next Program Address;
```

12.6.3.6 BR (Branch)

Syntax

```
BR {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[prv={OFF | ON}]
cond_addr={label | 8-bit unsigned integer}
[pin= {pin number}]
event={NOCOND | FALL | RISE | BOTH | ZERO | NAF | LOW | HIGH}
[irq={OFF | ON}]
}
```

Opcode: Dh, [P11:P8]

Table 12-56. BR Program Field (P31:P0)

31–22	21	20	19–12	11–8	7–0
0		BRK	Next program address	Opcode	Reserved
10	1-U	1	8	4	8

Table 12-57. BR Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–4	3–1	0
0	Control	Prv.	Conditional address	Pin select	Branch cond.	Reserved	Int. ena.
10	1	1	8	5	3	3	1

Table 12-58. BR Data Field (D31:D0)

31–25	24–0
0	Reserved
7	25

Cycles: One

Register Modified: None

Description: This instruction executes a jump to the conditional address [C19:C12] on a pin or a flag condition and can be used with all pins.

An edge is detected by comparing the current pin level sampled at loop start to the previous pin level stored in the Prv. (Previous) bit [C20].

event: Specifies the event that triggers a jump to the indexed program address.
Default: FALL

Table 12-59 provides the branch condition encoding.

Table 12-59. Branch Condition Encoding for BR

event	C6	C5	C4	Branch Condition
NOCOND	0	0	0	Always
FALL	0	0	1	On falling edge on the selected pin

event	C6	C5	C4	Branch Condition
RISE	0	1	0	On rising edge on selected pin
BOTH	0	1	1	On rising or falling edge on selected pin
ZERO	1	0	0	If Zero flag is set
NAF	1	0	1	If NAF_global flag is set
LOW	1	1	0	On LOW level on selected pin
HIGH	1	1	1	On HIGH level on selected pin

irq ON generates an interrupt when the event occurs that triggers the jump. If irq is set to OFF, no interrupt is generated.
Default: OFF.

Execution

```
If (Condition is true)
    {
    If (Interrupt Enable == 1)
          SW interrupt flag = 1;
    Jump to Conditional Address;
    }
else
    Jump to Next Program Address;
Prv. bit = Selected Pin level; (Always executed)
```

12.6.3.7 CNT (Count)

Syntax

```
CNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[angle_count={OFF | ON}]
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
[control={OFF | ON}]
max=20-bit unsigned integer
[data=20-bit unsigned integer]
}
```

Opcode 6h, [P11:P8]

Table 12-60. CNT Program Field (P31:P0)

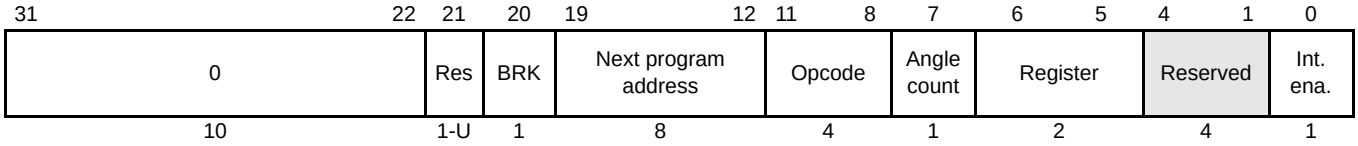


Table 12-61. CNT Control Field (C31:C0)

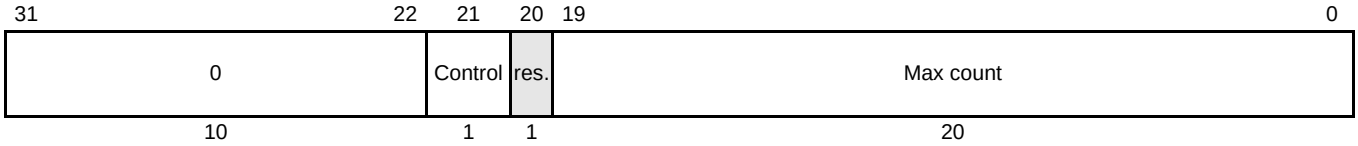
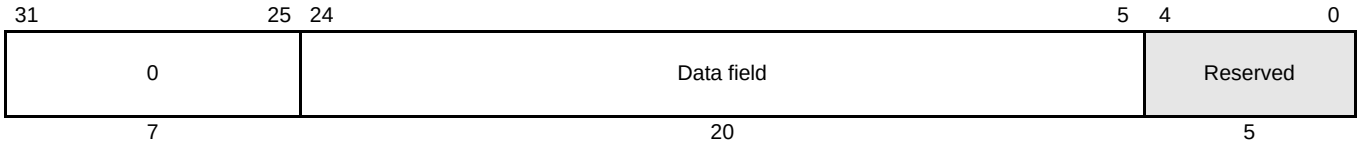


Table 12-62. CNT Data Field (D31:D0)



Cycles One or two
One cycle (time mode), two cycles (angle mode)

Register Modified Selected register (A, B or T)

Description This instruction defines a virtual timer. The counter value stored in the data field [D24:5] is incremented unconditionally on each resolution when in time mode (angle count bit [P7] = 0). When

the count reaches the maximum count specified in the control field, the counter is reset. It takes one cycle in this mode.

In angle mode (angle count bit [P7] = 1), CNT needs data from the software angle generator (SWAG). For the SWAG, the angle increment value will be 0 or 1. It takes two cycles in this mode.

Table 12-34 shows the P6 and P5 encoding for selecting the required ALU register

.

angle_ count	Specifies when the counter is incremented. A value of ON causes the counter value to be incremented only if the new angle flag is set (NAF_global = 1). A value of OFF increments the counter each time the assembler encounters the CNT instruction. Default value for this field is OFF.
irq	ON generates an interrupt when the counter overflows to zero. The interrupt is not generated until the data field is reset to zero. If irq is set to OFF, no interrupt is generated. Default: OFF.
max	Specifies the 20-bit integer value that defines the maximum count value allowed in the data field. When the count in the data field is equal to max, the data field is reset to 0 and the Z system flag is set to 1.
data	Specifies the 20-bit integer value serving as a counter. Default: 0.

Execution

```
Z = 0;

If (Angle Count (bit P7 == 1))
     {
     If (NAF_global == 0)
            Jump to Next Program Address;
     else
            {
            If ((Immediate Data Field + Angle Increment) ≥Max count)
              {
              Z = 1;
              Selected register = ((Immediate Data Field + Angle Inc.)
              - Max count);
              Immediate Data Field = ((Immediate Data Field + Angle Inc.)
              - Max count);
              If (Interrupt Enable == 1)
                SW interrupt flag = 1;
              }
            else
              {
              Selected register = Immediate Data Field + Angle Incre-
              ment;
              Immediate Data Field = Immediate Data Field + Angle Incre-
              ment;
              }
            }
     }
else (Time mode (bit P7 == 0))
     {
     If (Immediate Data Field == Max count)
```

```
                {
                Z = 1;
                Selected register = 00000;
                Immediate Data Field = 00000;
                If (Interrupt Enable == 1)
                  SW interrupt flag = 1;
                }
        else
                {
                Selected register = Immediate Data Field + 1;
                Immediate Data Field = Immediate Data Field + 1;
                }
        }
Jump to Next Program Address;
```

12.6.3.8 DADM64 (Data Add Move)

Syntax

DADM64 {
[brk={OFF | ON}]
[next=*{label | 8-bit unsigned integer}]*
remote=*{label | 8-bit unsigned integer}*
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr=*{label | 8-bit unsigned integer}]*
[pin=pin number]
comp_mode={ECMP | SCMP | MCMP|ACMP}
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | T | NONE}
[irq={OFF | ON}]
data=*20-bit unsigned integer*
[hr_data=5-*bit unsigned integer]*
}

-or-

Syntax

DADM64 {
[brk={OFF | ON}]
[next=*{label | 8-bit unsigned integer}]*
remote=*{label | 8-bit unsigned integer}*
[control={OFF | ON}]
cntl_val=*21-bit unsigned integer*
data=*20-bit unsigned integer*
[hr_data=*5-bit unsigned integer]*
}

Opcode *2h, [P11:P8]*

Table 12-63. DADM64 Program Field (P31:P0)

31–22	21	20	19–12	11–8	7–0
0	Res	BRK	Next Program Address	Opcode	Remote Address
10	1-U	1	8	4	8

Table 12-64. DADM64 Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4–3	2–1	0
0	Control	Enable Pin Action	Conditional Address	Pin Select	Comp. Mode	Action	Register	Int. Ena.
10	1	1	8	5	2	2	2	1

Table 12-65. DADM64 Data Field (D31:D0)

31–25	24–5	4–0
0	Data Field	High Res. Data Field
7	20	5

Cycles	Two
Register Modified	Register T (implicitly)
Description	This instruction modifies the data field and the control field at the remote address. The remote data field value is not just replaced, but is added with the DADM64 data field.

DADM64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similarly to the format of the DADM64 control field. A second syntax, in which the entire 21-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar to the DADM64 control field. Either syntax may be used, but you must use one or the other but not a combination of syntaxes.

Figure 12-62 shows the DADM64 add and move operation.

Figure 12-62. DADM64 Add and Move Operation

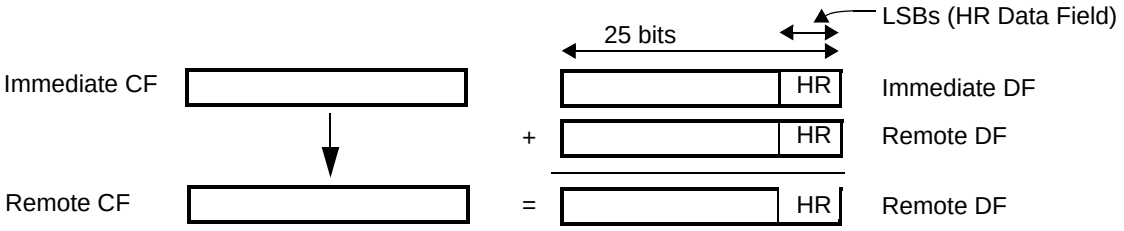


Table 12-66. DADM64 Control Field Description

Control Field	Description
Control	maintains the control field for the remote instruction
en_pin_ action	maintains the control field for the remote instruction
cond_ addr	maintains the control field for the remote instruction
pin	maintains the control field for the remote instruction
register	maintains the control field for the remote instruction
action	maintains the control field for the remote instruction
irq	maintains the control field for the remote instruction
data	Specifies the 20-bit initial value for the data field.
hr_data	Five least significant bits of the 25 bit data field. Default: 0
cntl_val	Specifies the 21 least significant bits of the Control field.

Execution

```
Remote Data Field = Remote Data Field + Immediate Data Field;
Remote Control Field = Immediate Control Field;
Jump to Next Program Address;
```

12.6.3.9 DJZ (Decrement and Jump if Zero)

Syntax[(1)]

```
DJZ{
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
cond_addr={label | 8-bit unsigned integer}
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
[data=20-bit unsigned integer]
}
```

Opcode Ah, [P11:P8];

Sub-opcode *[P6-P5]=10*

Table 12-67. DJZ Program Field (P31:P0)

31 – 22	21	20	19 – 12	11 – 8	7	6 – 5	4 – 0
0	Res	BRK	Next program address	Opcode	Res.	Sub-opcode	Reserved
10	1-U	1	8	4	1	2	5

Table 12-68. DJZ Control Field (C31:C0)

31 – 22	21	20	19 – 12	11 – 3	2 – 1	0
0	Control	res.	Conditional address	Reserved	Register	Int. ena.
10	1	1	8	9	2	1

Table 12-69. DJZ Data Field (D31:D0)

31 – 25	24 – 5	4 – 0
0	Data	Reserved
7	20	5

Cycles One

Register Modified Selected register (A, B, or T)

Description This instruction defines a virtual down counter used for delayed execution of certain instructions (to generate minimum on/off times). When DJZ is executed with counter value not zero, the counter value is decremented. If the counter value is zero, the counter remains zero until it is reloaded with a non-zero value. The program flow can be modified when down counter value is zero by using the conditional address.

1 DJNZ is also supported syntax. The functionality of the two instruction names is identical.

cond_addr This field is not optional for the DJZ instruction.

irq — ON generates an interrupt when the data field reaches zero. No interrupt is generated when the bit is OFF.
Default: OFF.

data — Specifies the 20-bit integer value used as a counter. This counter is decremented each time the DJZ instruction is executed until the counter reaches 0.
Default: 0.

Execution

```
If (Data value != 0)
     {
     Selected register = Data field value - 1;
     Down Counter value = Data field value - 1;
     Jump to Next Program Address;
     }
else
     {
     Selected register = 00000h;
     If (Interrupt Enable == 1)
           SW interrupt flag = 1;
     Jump to conditional Address;
     }
```

12.6.3.10 ECMP (Equality Compare)

Syntax

```
ECMP {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[hr_lr={HIGH | LOW}]
[angle_comp={OFF | ON}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}]
pin=pin number
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | T | NONE}
[irq={OFF | ON}]
data=20-bit unsigned integer
[hr_data={5-bit unsigned integer}]
}
```

Opcode 0h, [P11:P8];

Sub-Opcode *00h, [C6:C5]*

Table 12-70. ECMP Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6	5–0
0	Res	BRK	Next program address	Opcode	hr_lr	Angle comp.	Reserved
10	1-U	1	8	4	1	1	6

Table 12-71. ECMP Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4–3	2–1	0
0	Control	Enable pin action	Conditional address	Pin select	Sub-opcode	Action	Register	Int. ena.
10	1	1	8	5	2	2	2	1

Table 12-72. ECMP Data Field (D31:D0)

31–25	24–5	4–0
0	Data	HR data
7	20	5

Cycles One

Register Modified Register T if selected

Description ECMP can use all pins. This instruction compares a 20-bit data value stored in the data field (D24–D5) to the value stored in the selected ALU register (A, B, or T).

If T-register is selected, and if the 20-bit data field matches, ECMP updates T-register with the 25-bit value (D24-D0).

If an HR pin is chosen (HET23-HET0) and the hr_lr bit is cleared, pin action will occur after a delay from the next resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is programmed in

the data field (D4–D0). In the case of a non-HR pin, the pin action will be taken on the next loop resolution clock.

The behavior of the pins is governed by the four action options in bits C4:C3. ECMP uses the zero flag to generate opposite pin action (synchronized to the loop resolution clock).

angle_comp	Determines if an angle compare is performed. A value of ON causes the comparison to be performed only if the new angle flag is set (NAF = 1). If OFF is specified, the compare is then performed regardless of the state of the new angle flag. Default: OFF.
irq	Specifies whether or not an interrupt is generated. A value of ON sends an interrupt if register and data field values are equivalent. If OFF is selected, no interrupt is generated. Default: OFF.
data	Specifies the value for the data field. This value is compared with the selected register.
hr_data	Specifies the HR pin delay. Default: 0.

Execution

```
If (Angle Comp. bit == 0 OR (Angle Comp. bit == 1 AND NAF_global ==
1))
     {
     If (Selected register value == Immediate data field value)
           {
           If (hr_lr bit == 0)
             {
             If (Enable Pin action == 1)
               Selected Pin = Pin Action AT next loop resolution clock
               + HR delay;
             }
           else
             {
             If (Enable Pin action == 1)
               Selected Pin = Pin Action AT next loop resolution clock;
             }
           If ( (Z_flag == 1) AND (Opposite action == 1) )
             {
             If (Enable Pin Action == 1)
               {
                   Selected Pin = opposite Pin Action AT next
                   loop resolution clock;
               }
             }
           If (Interrupt Enable == 1)
             SW interrupt flag = 1;
           If (register T is selected)
             T register = Compare value (25 bit);
           Jump to Conditional Address;
           }
     elseIf (Z == 1 AND Opposite action == 1)
           {
           If (Enable Pin action == 1)
             Selected Pin = opposite Pin Action AT next loop resolution
             clock;
```

```
            Jump to Next Program Address;
            }
      else
            Jump to Next Program Address;
      }
If (Angle Comp. bit == 1 AND NAF_global == 0)
      Jump to Next Program Address;
```

12.6.3.11 ECNT (Event Count)

Syntax

```
ECNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}]
pin= pin number
event={NAF | FALL | RISE |BOTH | ACCUHIGH | ACCULOW}
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
data=20-bit unsigned integer
}
```

Opcode Ah;

Sub-Opcode *01h, [P6-P5]*

Table 12-73. ECNT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–5	4–0
0	Res	BRK	Next program address	Opcode	Res.	Sub-opcode	Reserved
10	1-U	1	8	4	1	2	5

Table 12-74. ECNT Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–4	3	2–1	0
0	Control	Prv.	Conditional address	Pin select	Event	Res.	Register	Int. en.
10	1	1	8	5	3	1	2	1

Table 12-75. ECNT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles: One cycle

Register Modified: None

Description: This instruction defines a specialized 20-bit virtual counter used as an event counter or pulse accumulator (see Table 12-76). The counter value is stored in the data field [D24:D5].

When an event count condition is specified, the counter value is incremented on a pin edge condition or on the NAF_global condition (NAF_global is defined in ACNT).

An edge detect is performed by comparing the current pin level, sampled at loop start, to the previous pin level stored in Prv. bit. The ECNT instruction updates the Prv. bit each time an edge is detected regardless of whether the edge is selected.

This instruction can be used with all pins.

event: The event that triggers the counter.

Table 12-76. Event Encoding Format for ECNT

event	C6	C5	C4	Count Conditions	Mode	Int. Available
NAF	0	0	0	NAF flag is Set	Angle counter	Y
FALL	0	0	1	Falling edge on selected pin	Event counter	Y
RISE	0	1	0	Rising edge on selected pin	Event counter	Y
BOTH	0	1	1	Rising and Falling edge on selected pin	Event counter	Y
ACCUHIGH	1	0	-	while pin is high level	Pulse accumulation	N
ACCULOW	1	1	-	while pin is low level	Pulse accumulation	N

irq: ON generates an interrupt when event in counter mode occurs. No interrupt is generated with OFF.
Default: OFF.

data: 20-bit integer value serving as a counter.
Default: 0.

Execution

```
If (Event count condition is true according to bits [C6:C4] (see table 1-17))
{
    Selected register = Immediate Data Field + 1;
    Immediate Data Field = Immediate Data Field + 1;
    If (Interrupt Enable == 1)
          SW interrupt flag = 1;
    Jump to Conditional Address;
    }
else
    Jump to Next Program Address;
Prv. bit = Selected Pin level; (Always executed)
```

12.6.3.12 MCMP (Magnitude Compare)

Syntax

```
MCMP {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[hr_lr={LOW |HIGH}]
[angle_comp={OFF | ON}]
[savesub={OFF | ON}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 4-bit unsigned integer}]
pin= pin number
order={REG_GE_DATA | DATA_GE_REG}
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | T | NONE}
[irq={OFF | ON}]
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

Opcode 0h, [P11–P8];

Sub-Opcode *1h, C6*

Table 12-77. MCMP Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6	5	4	3–0
0	Res	BRK	Next program address	Opcode	hr_lr	Angle comp.	res.	Save sub.	res.
10	1-U	1	8	4	1	1	1	1	4

Table 12-78. MCMP Control Field (C31:C0)

31–22	21	20	19–12	11–7	6	5	4–3	2–1	0
0	Control	Enable pin action	Conditional address	Pin select	Sub-op code	Order	Action	Register	Int. ena.
10	1	1	8	5	1	1	2	2	1

Table 12-79. MCMP Data Field (D31:D0)

31–25	24–5	4–0
0	Data	HR data
7	20	5

Cycles One

Register Modified Register T (implicitly)

Description This instruction compares the magnitude of the 20-bit data value stored in the data field (D24-D5) and the 20-bit value stored in the selected ALU register (A, B, or T). The register select coding for C2 and C1 are shown in Table 12-34.

Note: The Difference Between Compare Values
The difference between the two data **values must not exceed** (2^{19}) - 1.

If an HR pin is chosen (HET23-HET0) and the hr_lr bit is reset, pin action will occur after a delay from the next resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is programmed in the data field (D4-D0). In the case of a non-HR pin, the pin action will be taken on the next resolution clock.

When the data value matches, an output pin can be set or reset according to the pin action bit (C4). The pin will not change states if the enable pin action bit (C20) is reset.

MCMP uses the zero flag set to generate opposite pin action (synchronized to the loop resolution clock). The save sub bit (P4) provides the option to save the result of a subtraction into register T.

angle_comp — Determines whether or not an angle compare is performed. A value of ON causes the comparison to be performed only if the new angle flag is set (NAF = 1). If OFF is specified, the compare is then performed regardless of the state of the new angle flag.
Default: OFF.

savesub — When set, the comparison result is saved into the T register (upper 20 bits).
Default: OFF.

order — Specifies the order of the operands for the comparison.

Table 12-80. Magnitude Compare Order for MCMP

Order	C5	Description
REG_GE_DATA	0	Evaluates to true if the register value is greater than or equal to the data field value.
DATA_GE_REG	1	Evaluates to true if the data field value is greater than or equal to the register value.

irq — Specifies whether or not an interrupt is generated. A value of ON sends an interrupt if the register and data field values are equivalent. If OFF is selected, no interrupt is generated.

data — Specifies the value for the data field. This value is compared with the selected register.

hr_data — HR pin display. The default value for an unspecified bit is 0.

Execution

```
If (Angle Comp. bit == 0 OR (Angle Comp. bit == 1 AND NAF_global ==
1))
     {
     If ((C5 == 1 AND (Immediate data value - Selected register) ≥ 0) OR
     (C5 == 0 AND (Selected register - Immediate data value) ≥ 0))
            {
            If (hr_lr bit == 0)
              {
              If (Enable Pin action == 1)
                Selected Pin = Pin Action AT next loop resolution clock
                + HR delay;
              }
            else
              {
              If (Enable Pin action == 1)
                Selected Pin = Pin Action AT next loop resolution clock;
              }
            If ( (Z_flag == 1) AND (Opposite action == 1) )
```

```
                {
                If (Enable Pin Action == 1)
                  {
                     Selected Pin = opposite Pin Action AT next
                     loop resolution clock;
                  }
                }
              If (Interrupt Enable == 1)
                SW interrupt flag = 1;
              If (Save sub. == 1)
                {
                If ([C5] == 1)
                  T register = Immediate data value - Selected register;
                If ([C5] == 0)
                  T register = Selected register - Immediate data value;
                }
              Jump to Conditional Address;
              }
        elseIf (Z == 1 AND Opposite action == 1)
              {
              If (Enable Pin action == 1)
                Selected Pin = opposite Pin Action AT next loop resolution
                clock;
              Jump to Next Program Address;
              }
        else
              Jump to Next Program Address;
        }
  If (Angle Comp. bit == 1 AND NAF_global == 0)
        Jump to Next Program Address;
```

12.6.3.13 MOV32 (Data Move 32)

Syntax

```
MOV32 {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
remote={label | 8-bit unsigned integer}
[control={OFF | ON}]
[init={OFF | ON}]
type={IMTOREG | IMTOREG&REM | REGTOREM | REMTOREG}
reg={A | B | T | NONE}
[data=20-bit unsigned integer]
[hr_data=5-bit unsigned integer]
}
```

Opcode: 4h, [P11:P8];

Sub-Opcode: *C5=0*

Table 12-81. MOV32 Program Field (P31:P0)

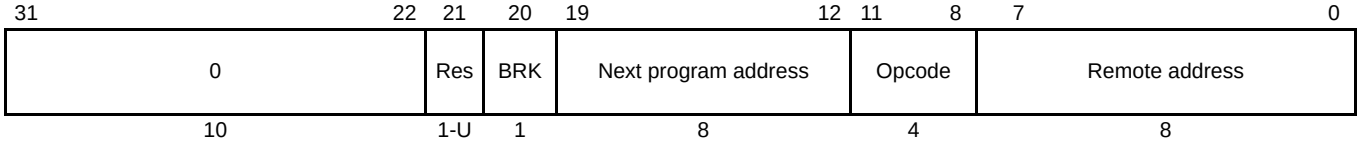


Table 12-82. MOV32 Control Field (C31:C0)

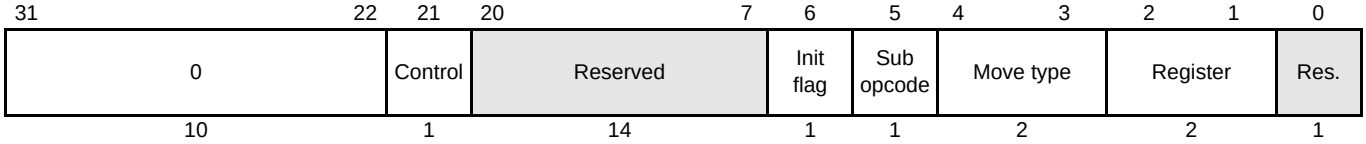
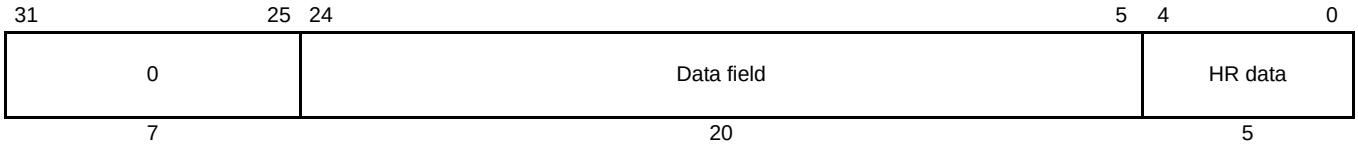


Table 12-83. MOV32 Data Field (D31:D0)



Cycles — One or two cycles

Register Modified — Selected register (A, B or T)

Description — MOV32 replaces the selected ALU register and/or the data field values at the remote address location depending on the move type. Table 12-34 shows the C2 and C1 bit encoding for selecting the desired register.

Table 12-84 shows the [C4:C3] bit encoding for selecting the destination for a data move. Figure 12-63 through Figure 12-66 illustrate these operations.

If *no register* is selected, the move is not executed, except for configuration C4:C3 = 01, where the remote data field is written with the immediate data field value.

remote — Determines the location of the remote address.
Default: Current instruction + 1.

init — (Optional) Determines whether or not system flags are initialized. A value of ON reinitializes the following system flags to these states:

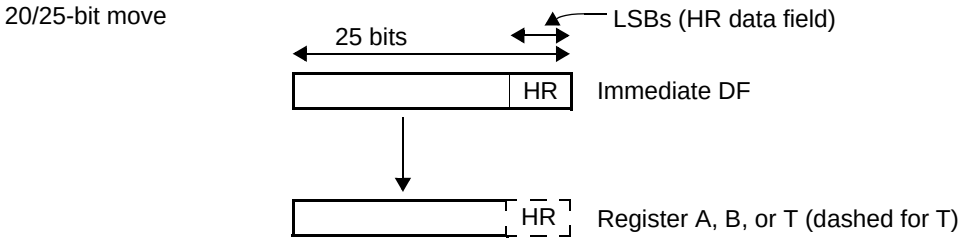
Acceleration flag (ACF) = 0
Deceleration flag (DCF) = 1
Gap flag (GPF) = 0
New angle flag (NAF) = 0
A value of OFF results in no change to the system flags.

type — Specifies the move type to be executed.

Table 12-84. Move Type Encoding Selection

Move Type	C4	C3	Source	Destination(s)	Cycles
IMTOREG	0	0	Immediate data field	Register A, B, or T	1
IMTOREG&REM	0	1	Immediate data field	Remote data field & register A, B, or T	1
REGTOREM	1	0	Register A, B, or T	Remote data field	1
REMTOREG	1	1	Remote data field	Register A, B, or T	2

Figure 12-63. MOV32 Move Operation for IMTOREG (Case 00)



reg — Specifies which register (A, B, T, or NONE) is involved in the move. A register (A, B, or T) must be specified for every move type except IMTOREG&REM. If *NONE* is used with move type IMTOREG&REM, the assembler executes a move from the immediate data field to the remote data field. If *NONE* is used with any other move type, no move is executed.

data — Specifies a 20-bit integer value to be written to the remote data field or selected register.

hr_data — (Optional) HR pin delay. The default value for an unspecified bit is 0.

Figure 12-64. MOV32 Move Operation for IMTOREG&REM (Case 01)

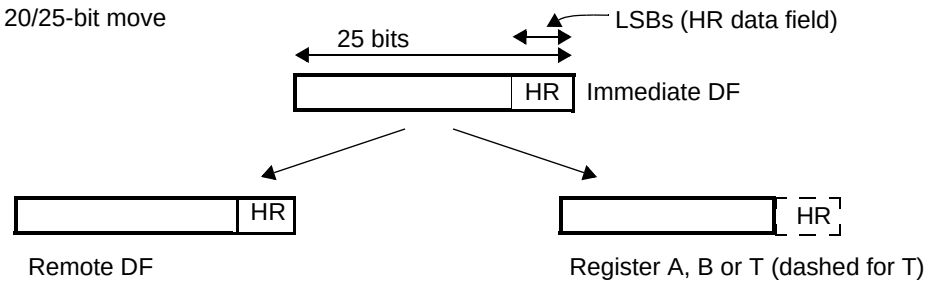


Figure 12-65. MOV32 Move Operation for REGTOREM (Case 10)

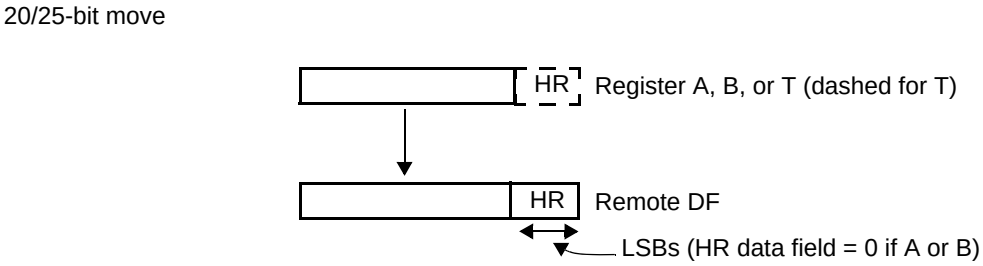
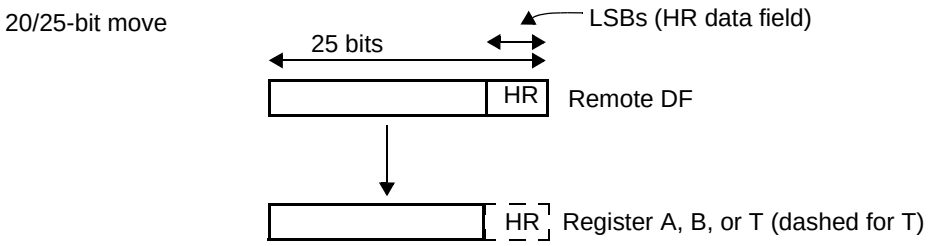


Figure 12-66. MOV32 Move Operation for REMTOREG (Case 11)



Execution

```
switch (C4:C3)
    {
    case 00:
          Selected register = Immediate Data Field;
    case 01:
          Selected register = Immediate Data Field;
          Remote Data Field = Immediate Data Field;
    case 10:
          Remote Data Field = Selected register;
    case 11:
          Selected register = Remote Data Field;
    }
If (Init Flag == 1)
    {
    ACF = 0;
    DCF = 1;
    GPF = 0;
    NAF = 0;
    }
else
    All flags remain unchanged;
Jump to Next Program Address;
```

12.6.3.14 MOV64 (Data Move 64)

Syntax

```
MOV64 {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
remote={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 4-bit unsigned integer}]
[pin= pin number]
comp_mode={ECMP | SCMP | MCMP | ACMP}
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | T | NONE}
[irq={OFF | ON}]
[data=20-bit unsigned integer]
[hr_data=5-bit unsigned integer]
}
```

- or -

Syntax

```
MOV64 {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
remote={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[cntl_val=21-bit unsigned integer]
[data=20-bit unsigned integer]
[hr_data=5-bit unsigned integer]

}
```

Opcode 1h, [P11:P8]

Table 12-85. MOV64 Program Field (P31:P0)

31 – 22	21	20	19 – 12	11 – 8	7 – 0
0	Res	BRK	Next program address	Opcode	Remote address
10	1-U	1	8	4	8

Table 12-86. MOV64 Control Field (C31:C0)

31 – 22	21	20	19 – 12	11 – 7	6 – 5	4 – 3	2 – 1	0
0	Control	Enable pin action	Conditional address	Pin select	Comp. mode	Action	Register	Int. ena.
10	1	1	8	5	2	2	2	1

Table 12-87. MOV64 Data Field (D31:D0)

31 – 25	24 – 5	4 – 0
0	Data	HR data
7	20	5

Cycles: One

Register Modified: None

Description: This instruction modifies the data field and the control field at the remote address.

MOV64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similarly to the format of the MOV64 control field. A second syntax, in which the entire 21-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar to the MOV64 control field. Either syntax may be used, but you must use one or the either but not a combination of syntaxes. See Figure 12-67.

Figure 12-67. MOV64 Move Operation

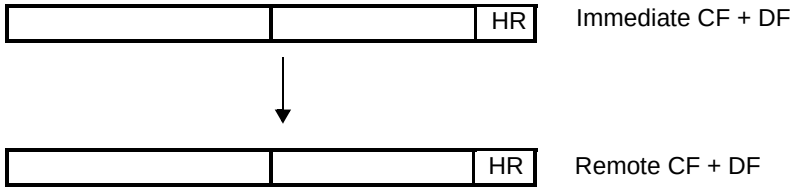


Table 12-88. MOV64 Control Field Descriptions

Control Field	Description
Control	Maintains the control field for the remote instruction.
en_pin_action	Maintains the control field for the remote instruction.
cond_addr	Maintains the control field for the remote instruction.
pin	Maintains the control field for the remote instruction.
register	Maintains the control field for the remote instruction.
action	Maintains the control field for the remote instruction.
irq	Maintains the control field for the remote instruction.
data	Specifies the 20-bit initial count value for the data field. If omitted, the field defaults to 0.
hr_data	(Optional) HR pin delay. The default value for an unspecified bit is 0.
comp_mode	Selects the comparison mode type to be used.

Table 12-89. Comparison Type Encoding Format

comp_mode	C6	C5
ECMP	0	0
SCMP	0	1
MCMP	1	0
ACMP	1	1

Execution

```
Remote Data Field = Immediate Data Field;
Remote Control Field = Immediate control Field;
Jump to Next Program Address;
```

12.6.3.15 PCNT (Period/Pulse Count)

Syntax

```
PCNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[irq={OFF | ON}]
type={RISE2FALL | FALL2RISE | FALL2FALL | RISE2RISE}
pin=pin number
[control={OFF | ON}]
[prv={OFF | ON}]
[period=20-bit unsigned integer]
[data=20-bit unsigned integer]
[hr_data=5-bit unsigned integer]
}
```

Opcode 7h, [P11:P8]

Table 12-90. PCNT Program Field (P31:P0)

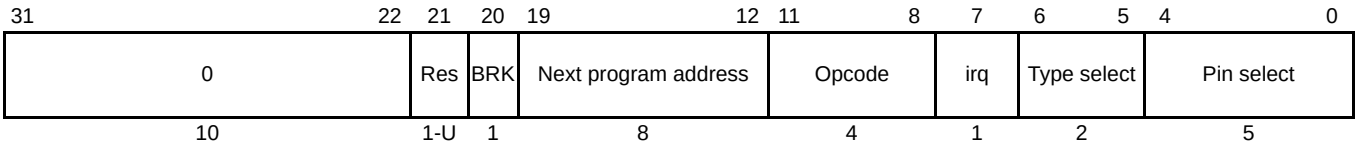


Table 12-91. PCNT Control Field (C31:C0)

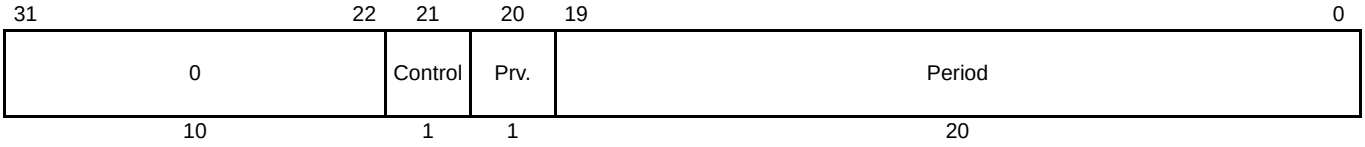
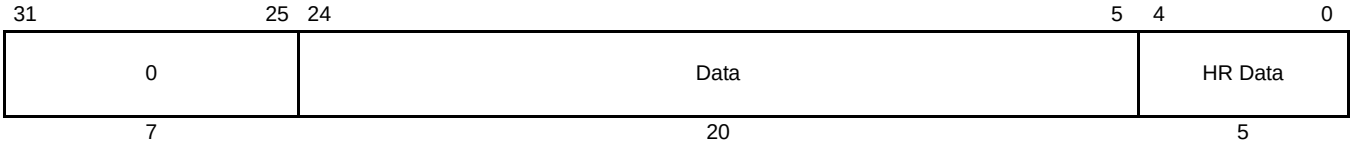


Table 12-92. PCNT Data Field (D31:D0)



Cycles One

Register Modified Register A

Description This instruction detects the edges of the external signal at loop start and measures its period or pulse duration. The counter value stored in the control field [C19:C0] and in the register A is incremented

on each timer resolution. If an HR pin is selected (HET23-HET0), then PCNT will use the HR structure on the pin to measure an HR period/pulse count value.

irq — (Optional) Specifies whether or not an interrupt is generated. A value of ON sends an interrupt if register and data field values are equivalent. If OFF is selected, no interrupt is generated.

type — (Optional) Determines the type of counter that is implemented.

Table 12-93. Counter Type Encoding Format

	P6	P5	Period/Pulse Select	Reset On	Capture On
FALL2RISE	0	0	Count low pulse duration on selected pin	Falling edge	Rising edge
RISE2FALL	0	1	Count high pulse duration on selected pin	Rising edge	Falling edge
FALL2FALL	1	0	Count period between falling edges on selected pin	Falling edge	Falling edge
RISE2RISE	1	1	Count period between rising edges on selected pin	Rising edge	Rising edge

period — Specifies the 20-bit integer value that holds the previous counter value. The previous counter value is also stored in register A.
Default: 0.

data — 20-bit integer value serving as a counter.
Default: 0.

hr_data — HR pin delay.
Default: 0.

If the control bit is set, the period value is automatically cleared after the main CPU performs a 64-bit read access.

If ***period-measure*** is selected, PCNT captures the counter value into the period/pulse data field [D24:D5] on the selected edge. The HR structure provides HR capture field [D4:D0]. The counter value [C19:C0] is reset on the same edge. The period value is a 25-bit value.

If ***pulse-measure*** is selected, PCNT captures the counter value into the period/pulse count field [D24:D5] on the selected edge. The HR structure provides HR capture field [D4:D0]. The counter value [C19:C0] is reset on the next opposite edge. The pulse value is a 25-bit value.

An edge detect is performed by comparing the current pin level, sampled at loop start, to the previous pin level stored in Prv. bit.

When the overflow count (all 1's in the counter value) is reached, PCNT stops counting until the next reset edge is detected. Table 12-33 shows the pin encoding for the pin select field [P4:P0].

Execution

```
Z = 0;
If (Period/Pulse select [P6,P5] == 1X)
     {
     If (Period value != FFFFFh)
           {
           Register A = Period value + 1;
           Period value = Period value + 1;
           }
     elseIf (specified edge not detected on selected pin)
           Register A = FFFFFh;
           HR capture value = 1Fh;
     If (specified edge detected on selected pin)
```

```
            {
            Z = 1;
            If (Period value == FFFFFh)
              {
              Register A = FFFFFh;
              Data field = FFFFFh;
              }
            else
              {
              Data field = Period value + 1;
              HR capture value = selected HR counter;
              Period value = 00000h;
              If (Interrupt Enable == 1)
                SW interrupt flag = 1;
              Jump to Next Program Address;
              }
            }
      }
else /*** Pulse mode ***/
      {
      If (Period value != FFFFFh)
            {
            Register A = Period value + 1;
            Period value = Period value + 1;
            }
      If (specified reset edge is detected on selected pin)
            Period value = 00000h;
      If (specified capture edge is detected on selected pin)
            {
            Z = 1;
            If (Period value == FFFFFh)
              {
              Register A = FFFFFh;
              Data field = FFFFFh;
              HR capture value = 11111;
              }
            else
              {
              Data field = Period value + 1;
              HR Capture Value = selected HR counter value;
              If (Interrupt Enable == 1)
                SW interrupt flag = 1;
              }
            }
      }
      Jump to Next Program Address;
```

```
Prv. bit = Selected Pin level;
```

12.6.3.16 PWCNT (Pulse Width Count)

Syntax

```
PWCNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[hr_lr={HIGH | LOW}]
[control={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}
[en_pin_action={OFF | ON}]
pin =pin number
[action={CLEAR | SET | PULSELO | PULSEHI}]
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

Opcode: Ah, [P11:P8];

Sub-Opcode: *11h, [P6-P5]*

Table 12-94. PWCNT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–5	4–0
0	Res	BRK	Next program address	Opcode	hr_lr	Sub-opcode	Reserved
10	1-U	1	8	4	1	2	5

Table 12-95. PWCNT Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4–3	2–1	0
0	Control	En. pin action	Conditional address	Pin select	Reserved	Action	Register	Int. enab.
10	1	1	8	5	2	1	1	1

Table 12-96. PWCNT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	HR data
7	20	5

Cycles: One

Register Modified: Selected register (A, B or T)

Description: This instruction defines a virtual timer used to generate variable length pulses. The counter value stored in the data field is decremented unconditionally on each timer resolution until it reaches zero, and it then stays at zero until it is reloaded with a non-zero value.

The specified pin action is performed as long as the count after count value is decremented is greater than 0. The opposite pin action is performed when the count after decrement just reaches 0.

If an HR pin is chosen (HET23–HET0) and the hr_lr bit is reset, the opposite pin action will be taken after a delay from the next resolution clock. If the hr_lr bit is set, the delay is ignored. This delay is

programmed in bits [D4:D0]. In the case of a non-HR pin, the opposite pin action is taken on the next resolution clock.

irq — ON generates an interrupt when the data field value reaches 0. No interrupt is generated for OFF
Default: OFF.

data — 20-bit integer value serving as a counter.

hr_data — HR pin delay.
Default: 0.

Execution

```
If (Data field value == 0)
    {
    Selected register = 0;
    Jump to Next Program Address;
    }
If (Data field value > 1)
    {
    Selected register = Data field value - 1;
    Data field value = Counter value - 1;
    If (Enable Pin action == 1)
          Selected Pin = Pin Action AT next loop resolution clock;
    Jump to Next Program Address;
    }
If (Data field value == 1)
    {
    Selected register = 00000h;
    Data field value = 00000h;
    If (Opposite action == 1)
          {
          If (hr_lr bit == 0)
            {
            If (Enable Pin action == 1)
              Selected Pin = Opposite level of Pin Action AT next loop
              resolution clock + HR delay;
            }
          else
            {
            If (Enable Pin action == 1)
              Selected Pin = Opposite level of Pin Action AT next loop
              resolution clock;
            }
          }
    If (Interrupt Enable == 1)
    SW interrupt flag = 1;
    Jump to Conditional Address
    }
```

12.6.3.17 RADM64 (Register Add Move)

Syntax

```
RADM64 {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
remote={label | 8-bit unsigned integer}
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
[cond_addr={label | 4-bit unsigned integer}]
[pin= pin number]
comp_mode={ECMP | SCMP | MCMP | ACMP}
[action={CLEAR | SET | PULSELO | PULSEHI}]
reg={A | B | T | NONE}
[irq={OFF | ON}]
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

or

Syntax

```
RADM64 {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
remote={label | 8-bit unsigned integer}
[cntl_val = 21-bit unsigned integer]
[control={OFF | ON}]
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

Opcode 3h, [P11:P8]

Table 12-97. RADM64 Program Field (P31:P0)

31 – 22	21	20	19 – 12	11 – 8	7 – 0
0	Res	BRK	Next program address	Opcode	Remote
10	1-U	1	8	4	8

Table 12-98. RADM64 Control Field (C31:C0)

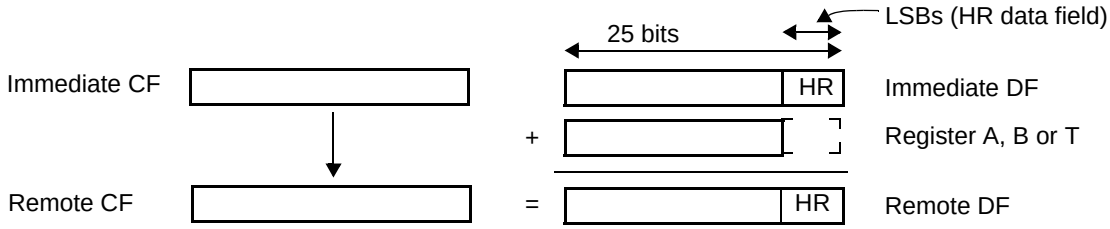
31 – 22	21	20	19 – 12	11 – 7	6 – 5	4 – 3	2 – 1	0
0	Control	Enable pin action	Conditional address	Pin select	Comp. mode	Action	Register	Int. ena.
10	1	1	8	5	2	2	2	1

Table 12-99. RADM64 Data Field (D31:D0)

31 – 25	24 – 5	4 – 0
0	Data	HR data
7	20	5

Cycles	One
Register Modified	Selected register (A, B or T)
Description	This instruction modifies the data field, the HR data field and the control field at the remote address. The advantage over DADM64 is that It executes one cycle faster. In case of the T-register selected, the addition is a 25-bit addition. The table description shows the bit encoding for determining which ALU register is selected.
	RADM64 has two distinct syntaxes. In the first syntax, bit values may be set by assigning a value to each of the control fields. This syntax is convenient for modifying control fields that are arranged similar to the format of the RADM64 control field. A second syntax, in which the entire 21-bit control field is specified by the cntl_val field, is convenient when the remote control field is dissimilar from the RADM64 control field. Either syntax may be used, but you must use one or the either but not a combination of syntaxes. See Figure 12-68.

Figure 12-68. RADM64 Add and Move Operation



comp_mode Selects the comparison mode type to be used.

Table 12-100. Comparison Type Encoding Format

comp_mode	C6	C5
ECMP	0	0
SCMP	0	1
MCMP	1	0
ACMP	1	1

Table 12-101. Control Field Description

Control Field	Description
Control	Maintains the control field for the remote instruction.
en_pin_action	Maintains the control field for the remote instruction.
cond_addr	Maintains the control field for the remote instruction.
pin	Maintains the control field for the remote instruction.
register	Maintains the control field for the remote instruction.
action	Maintains the control field for the remote instruction.
irq	Maintains the control field for the remote instruction.
data	Specifies the 20-bit initial value for the data field. If omitted, the field defaults to 0.
hr_data	Five least significant bits of the 25-bit data field. Default: 0.

Control Field	Description
cntl_val	Specifies the 21 least significant bits of the Control field.

Execution

```
Remote Data Field = Selected register + Immediate Data Field
(including HR data field);

Remote Control Field = Immediate Control Field;

Jump to Next Program Address;
```

12.6.3.18 SCMP (Sequence Compare)

Syntax

```
SCMP {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[control={OFF | ON}]
[en_pin_action={OFF | ON}]
cond_addr={label | 8-bit unsigned integer}
pin=pin number
[action={CLEAR | SET}]
[restart={OFF | ON}]
[irq={OFF | ON}]
data=20-bit unsigned integer
}
```

Opcode 0h, [P11–P8];

Sub-Opcode [C6-C5]=01

Table 12-102. SCMP Program Field (P31:P0)

31–22	21	20	19–12	11–8	7–0
0	Res	BRK	Next program address	Opcode	Reserved
10	1-U	1	8	4	8

Table 12-103. SCMP Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4	3	2	1	0
0	Control	Enable pin action	Conditional address	Pin select	Sub-opcode	Action	Res.	Cout prv	Restart enable	Int. ena.
10	1	1	8	5	2	1	1	1	1	1

Table 12-104. SCMP Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles One

Register Modified Register T (implicitly)

Description This instruction alternately performs angle- and time-based operations to generate pulse sequences, using the angle referenced time base. These pulse sequences last for a relative duration using a free running time base. Generally, register B holds the angle values and register A holds the time values.

Bit 0 of the conditional address field (C12) specifies whether the instruction is operating in angle or time operation mode.

When the compared values match in angle mode, a pin can be set or reset according to the pin action bit (C4). The pin does not change states if the enable pin action bit (C20) is reset.

The restart enable bit (C1) provides the option to unconditionally restart a sequence using the X-flag bit of ACMP.

restart	If restart is set to ON and the X flag = 1, the assembler writes a value of 1 into the immediate index field, writes the value in register A into the immediate data field, and jumps to the next program address. The X flag is set or cleared by the ECMP instruction. If restart is set to OFF, the X flag is ignored; no special action is performed. Default: OFF.
irq	ON generates an interrupt if the register and data field values are equivalent. No interrupt is generated when the field is OFF. Default: OFF.
data	Specifies the 20-bit compare value.
cond_addr	Since the LSB of the conditional address is used to select between time mode and angle mode, and since the conditional address is taken only in time mode, the destination for the conditional address must be odd.

Execution

```
If (Data field value £ Selected register value)
     Cout = 0;
else
     Cout = 1;
If (Restart Enable == 1 AND X == 1)
     {
     C12 = 1;
     Immediate Data Field = Register A;
     Jump to Next Program Address;
     }
If (Angle Mode is selected (C12 == 0) AND ((Restart En. == 1 AND X ==
0) OR Restart En. == 0))
     {
     If (Z == 0 AND (Register B value - Angle Inc. < Data field value)
     AND Cout == 0) OR

     (Z == 1 AND Cout_prv == 1 OR Cout == 0))

           {
           If (Enable Pin Action == 1)

             Selected Pin = Pin Action;
           If (Interrupt Enable == 1)

             SW interrupt flag = 1;
           Immediate Data Field = Register A;
           C12 = 1; /*** switch to Time Mode ***/
           Jump to Conditional Address;
           }

     else
           Jump to Next Program Address;
     }
```

```
If (Time Mode is selected (C12 == 1)) AND ((Restart En. == 1 AND X ==
0) OR Restart En. == 0)
    {
    Register T = Register A - Immediate Data Field;
    (Result of subtract must not exceed (2^19) - 1)
    Jump to Conditional Program Address;
    }
Cout_prv = Cout; (always executed)
```

Example 1. SCMP Example

```
  ; Time Counter in register A
A0CNT{reg=A, max=0x0fffff}
  ; Angle Counter (320 degree) in register B
A1CNT{reg=B, max=0x3C00, irq=OFF, angle_count=ON, data=0x400}
  ; Angle compare to set pulse on CC23 when match X flag is set
A2ACMP{next=A3, en_pin_action=ON, cond_addr=A3, pin=CC23, ac-
tion=SET, reg=B, irq=OFF, data=0x0220}
  ; Transfer match angle into time, when X is set transfer A into DF,
switch in time mode and jump to MCMP else next ACMP (next loop reso-
lution)
A3SCMP{next=A6, en_pin_action=ON, cond_addr=A5, pin=CC23, ac-
tion=SET, restart=ON, irq=OFF, data=0x0230}
  ; restart the sequence compare and jump to next resolution
A4MOV64{next=A6, remote=A3, en_pin_action=ON, cond_addr=A4, pin=CC23,
comp_mode=SCMP, action=SET, reg=B, data=0x0230}
  ; When T (= Reg. A - SCMP Data field) > compare value then reset
the pin and jump to MOV64 to re-initialize the SCMP Control and Data
field.
A5MCMP{next=A6, en_pin_action=ON, pin=CC23, hr_lr=LOW, action=CLEAR,
order=REG_GE_DATA, cond_addr=A4, reg=T, data=0x214, hr_data=0x10}
  ; Branch to first instruction (end of the loop)
A6BR{next=A0, cond_addr=A0, pin=CC25, event=NOCOND}
```

This program generates a pulse always at the same position of a wheel (ACMP instruction) and the pulse lasts for a certain time (value in MCMP instruction) whatever the speed of the wheel.

12.6.3.19 SCNT (Step Count)

Syntax

```
SCNT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
step={8 | 16 | 32 | 64}
[control={OFF | ON}]
gapstart=20-bit unsigned integer
[data=20-bit unsigned integer]
}
```

Opcode: Ah, [P11:P8];

Sub-Opcode: *[P6-P5]=00*

Table 12-105. SCNT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–5	4–3	2–0
0	Res	BRK	Next program address	Opcode	Res.	Sub-opcode	Step width	Res.
10	1-U	1	8	4	1	2	2	2

Table 12-106. SCNT Control Field (C31:C0)

31–22	21	20	19–0
0	Control	res.	Gap start
10	1	1	20

Table 12-107. SCNT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles: Three

Register Modified: Register A

Description: This instruction can be used only once in a program and defines a specialized virtual timer used after APCNT and before ACNT to generate an angle-referenced time base synchronized to an external signal (that is, a toothed wheel signal) as defined in APCNT and ACNT. Step width selection bits are saved in two flags, SWF0, and SWF1, to be re-used in ACNT.

SCNT multiplies the frequency of the external signal by a constant *K* defined in the step width field, [P4:P3]. The bit encoding for this field is defined in Table 12-108:

step: Specifies the step increment to be added to the counter value each program resolution. These two bits provide the values for the SWF0 and SWF1 flags. The valid values are listed in Table 12-108.

Table 12-108. Step Width Encoding for SCNT

P4	P3	Step Width (K)
0	0	8
0	1	16

P4	P3	Step Width (K)
1	0	32
1	1	64

gapstart — Defines the gap start angle, which SCNT writes to register A. The gap start value has no effect on the SCNT instruction, but if the ACNT instruction is being used, register A must contain the correct gap start value. For a typical toothed wheel gear:
GAPSTART= (stepwidth x (actual teeth on gear - 1)) + 1.

data — Specifies the 20-bit integer value serving as a counter.
Default: 0.

This instruction is incremented by the step value K on each timer resolution up to the previous period value P(n-1) measured by APCNT (stored in register T). The resulting period of SCNT is:
$P(n-1)/K$

Due to stepping, the final count of SCNT will not usually exactly match the target p(n-1). SCNT compensates for this error by starting each cycle with the remainder of the previous cycle. When SCNT reaches the target p(n-1), the zero flag is set as an increment condition for ACNT.

SCNT also specifies a gap start angle, defining the start of a range in ACNT where period measurements in APCNT are temporarily stopped to mask singularities in the external signal.

SCNT uses register A to store the gap start value. Gap start has no effect for SCNT.

Execution

```
SWF1 = P4;

SWF0 = P3;

Z = 0;

If (register T == 00000h)
    Jump to Next Program Address;

else
    {
    If (DCF == 1 OR ACF == 1)
          {
          Data Field register = 00000h;
          Counter value = 00000h;
          }
    If (DCF == 0 AND ACF == 0)
          {
          Data Field register = Data field register + Step Width;
          }
    If ((Data Field register - register T) ≥ 0)
          {
          Data field register = Data Field register - register T;
          Z = 1;
          }
    }

Register A = Gap start value;

Jump to Next Program Address;
```

12.6.3.20 SHFT (Shift)

Syntax

```
SHFT {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
smode={OR0 | OL0 | OR1 | OL1 | ORZ | OLZ | IRM | ILL | IRZ | ILZ}
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}]
cond={UNC | FALL | RISE}
pin=pin number
[reg={A | B | T | NONE}]
[irq={OFF | ON}]
data={20-bit unsigned integer}
}
```

Opcode Fh, [P11:P8]

Table 12-109. SHFT Program Field (P31:P0)

31–22	21	20	19–12	11–8	7–4	3–0
0	Res	BRK	Next program address	Opcode	Reserved	Smode
10	1-U	1	8	4	4	4

Table 12-110. SHFT Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4 3	2–1	0
0	Control	Prv.	Conditional address	Pin select	Shift condition	Res.	Register	Int. ena.
10	1	1	8	5	2	2	2	1

Table 12-111. SHFT Data Field (D31:D0)

31–25	24–5	4–0
0	Data	Reserved
7	20	5

Cycles One

Register Modified Selected register (A, B or T)

Description This instruction **may only be used on pins CC24-CC31.**

This instruction shift the data field of the Instruction. HET pins can be used for data in or data out. SHFT includes parameters to select the shift direction (in, out, left, right), shift condition (shift on a defined clock edge on HET31 or shift always), register for data storage (A, B, or T), and the data pin.

smode Shift mode.

Table 12-112. SHIFT MODE Encoding Format

smode	P3	P2	P1	P0	Operation	
OR0	0	0	0	0	Shift Out / Right	LSB 1st on HETx / 0 into MSB
OL0	0	0	0	1	Shift Out / Left	MSB 1st on HETx / 0 into LSB

smode	P3	P2	P1	P0	Operation	
OR1	0	0	1	0	Shift Out / Right	LSB 1st on HETx / 1 into MSB
OL1	0	0	1	1	Shift Out / Left	MSB 1st on HETx / 1 into LSB
ORZ	0	1	0	0	Shift Out / Right	LSB 1st on HETx / Z into MSB
OLZ	0	1	0	1	Shift Out / Left	MSB 1st on HETx / Z into LSB
IRM	1	0	0	0	Shift In / Right	HETx into MSB
ILL	1	0	0	1	Shift In / Left	HETx into LSB
IRZ	1	0	1	0	Shift In / Right	HETx in MSB / LSB into Z
ILZ	1	0	1	1	Shift In / Left	HETx in LSB / MSB into Z

cond Specifies the shift condition.

Table 12-113. SHIFT Condition Encoding

C6	C5	Shift Condition
0	X	Always
1	0	Rising edge of HET31
1	1	Falling edge of HET31

irq On generates an interrupt if the Z flag is set. A value of OFF does not generate an interrupt. Default: Off.

data Specifies the 20-bit value for the data field.

Execution

```
If (SHIFT condition == 0X)
OR (SHIFT condition == 10 AND CC31 rising edge)
OR (SHIFT condition == 11 AND CC31 falling edge)
    {
    Shift Immediate Data Field by one bit according to bits [P3:P0];
    (see Table 12-112)
    Immediate Data Field = Result of the shift;
    Selected register = Result of the shift;

    If ((Immediate Data Field == all 0's AND [P3:P0] == 000X) OR (Im-
    mediate Data Field == all 1's AND [P3:P0] == 001X))
          {
          Z = 1;
          }
    else if ([P3:P0] == 1010)
          {
          Z = LSB of the Immediate Data Field;
          }
    else if ([P3:P0] == 1011)
          {
          Z = MSB of the Immediate Data Field;
          }
    else
          {
          Z = 0;
```

```
        }

    If ((Immediate Data Field == all 0's OR
           (Immediate Data Field == all 1's)
          AND (Interrupt Enable == 1))
        {
        SW interrupt flag = 1;
        Jump to Conditional Address;
        }
    else
        {
        Jump to Next Program Address;
        }
    }
Prv. bit = CC31 Pin level; (Always executed)
Jump to Next Program Address;
```

Note:
The immediate data field evaluates all 0s or all 1s and is performed before the shift operation.

12.6.3.21 WCAP (Software Capture Word)

Syntax

```
WCAP {
[brk={OFF | ON}]
[next={label | 8-bit unsigned integer}]
[hr_lr={HIGH | LOW}]
[control={OFF | ON}]
[prv={OFF | ON}]
[cond_addr={label | 8-bit unsigned integer}]
pin=pin number
event={NOCOND | FALL | RISE | BOTH}
reg ={A | B | T | NONE}
[irq={OFF | ON}]
data=20-bit unsigned integer
[hr_data=5-bit unsigned integer]
}
```

Opcode Bh, [P11:P8]

Table 12-114. WCAP Program Field (P31:P0)

31–22	21	20	19–12	11–8	7	6–0
0	Res	BRK	Next program address	Opcode	hr_lr	Reserved
10	1-U	1	8	4	1	7

Table 12-115. WCAP Control Field (C31:C0)

31–22	21	20	19–12	11–7	6–5	4–3	2–1	0
0	Control	Prv.	Conditional address	Pin select	Capture condition	Reserved	Register	Int. ena.
10	1	1	8	5	2	2	2	1

Table 12-116. WCAP Data Field (D31:D0)

31–25	24–5	4–0
0	Data	HR data
7	20	5

Cycles One

Register Modified None

Description This instruction captures the selected register into the data field if the specified capture condition is true on the selected pin. This instruction can be used with all pins.

If an HR pin is selected (HET23-HET0) and the hr_lr bit is reset, the WCAP instruction will capture an HR time stamp into the data field on the selected edge condition. If the hr_lr bit is set, the HR capture

is ignored. Edge detection is performed by comparing the current pin level sampled at loop start to the previous pin level stored in the Prv. (Previous) bit [C20].

This instruction updates the Prv. bit each time an edge is detected regardless of whether the edge is selected.

event — Specifies the event that triggers the capture.

Table 12-117. Event Encoding Format for WCAP

C6	C5	Capture Condition
0	0	always
0	1	Capture on falling edge
1	0	Capture on rising edge
1	1	Capture on rising and falling edges

irq — ON generates an interrupt when the capture condition is met. No interrupt is generated for OFF
Default: OFF.

data — Specifies the 20-bit integer value to be written to the data field or selected register.

hr_data — HR capture value.
Default: 0.

Execution

```
If (Specified Capture Condition is true on Selected Pin
OR Unconditional capture is selected)
    {
    Immediate Data Field = Selected register value;
    If (hr_lr bit == 0)
          Capture the HR value in Immediate HR Data Field;
    If (Interrupt Enable == 1)
          SW interrupt flag = 1;
    Jump to Conditional Address;
    }
else
    Jump to Next Program Address;
Prv. bit = Selected Pin level;
```

Multi-Buffered Serial Peripheral Interface (MibSPI) Module

This reference guide provides the specifications for a 16-bit configurable synchronous multi-buffered serial peripheral interface (MibSPI). The MibSPI is a programmable-length shift register used for high-speed communication between external peripherals or other microcontrollers.

Throughout this chapter, all references to SPI also apply to MibSPI, unless otherwise noted.

Topic **Page**

13.1 Overview

The MibSPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (2 to 16 bits) to be shifted in and out of the device at a programmed bit-transfer rate. Typical applications for the SPI include interfacing to external peripherals, such as I/Os, memories, display drivers, and analog-to-digital converters.

The SPI has the following attributes:

- 16-bit shift register
- Receive buffer register
- 8-bit baud clock generator
- Serial clock (SPICLK) I/O pin
- SPISOMI/SPISIMO pins for data transfer, with programmable pin direction
- SPI enable ($\overline{\text{SPIENA}}$) I/O pin
- Up to 8 slave chip select ($\overline{\text{SPISCS}}$[7:0]) I/O pins (see the device data sheet for availability of these pins)
- SPICLK can be internally-generated (and driven) or received from an external clock source
- Each word transferred can have a unique format.
- SPI pins may be used as functional or digital I/O pins (GIOs)

13.1.1 Word Format Options

Each word transferred can have a unique format. Several format characteristics are programmable for each word transferred:

- SPICLK Frequency
- Character Length (2 to 16 bits)
- Phase (with and without delay)
- Polarity (high or low)
- Parity enabled/disabled
- CS timers for setup and hold
- Shift direction (MSB-first or LSB-first)

13.1.2 Multi-buffering (Mib) support

The MibSPI has a programmable buffer memory that enables programmed transmission to be completed without CPU intervention. The buffers are combined in different Transfer Groups (TGs) that can be triggered by external events (timers, I/O activity, etc.) or by the internal tick counter. The internal tick counter supports periodic trigger events.

13.1.2.1 Multi-buffer Mode

Multi-buffer Mode is an extension to the SPI. In multi-buffer mode many extended features are configurable:

- Number of buffers for each peripheral (or data source/destination) or group (up to 16 groupings)
- Triggers for each groups, trigger types, trigger sources for individual groups (up to 14 external trigger sources and 1 internal trigger source)

13.1.2.2 Compatibility Mode

Compatibility Mode of the MibSPI makes it behave exactly like a SPI and ensures full compatibility with other SPIs. All features in compatibility mode of the MibSPI are directly applicable to a SPI. Multi-buffer Mode features are not available in Compatibility Mode.

Note:
The SPIDAT0 register is not accessible in the multi-buffer mode of MibSPI. It is only accessible in compatibility mode.

13.1.3 Transmission Lock (Multi-Buffer Mode Master Only)

Some slave devices require transmission of a command followed by data. In this case the SPI transaction should not be interrupted by another group transfer. The LOCK bit within each buffer allows a consecutive transfer to happen without being interrupted by another higher-priority group transfer.

13.2 Operating Modes

The SPI operates as either a master or a slave. The MASTER bit (SPIGCR1[0]) selects the configuration of the SPISIMO and SPISOMI pins and the CLKMOD bit (SPIGCR1[1]) determines whether an internal or external clock source will be used.

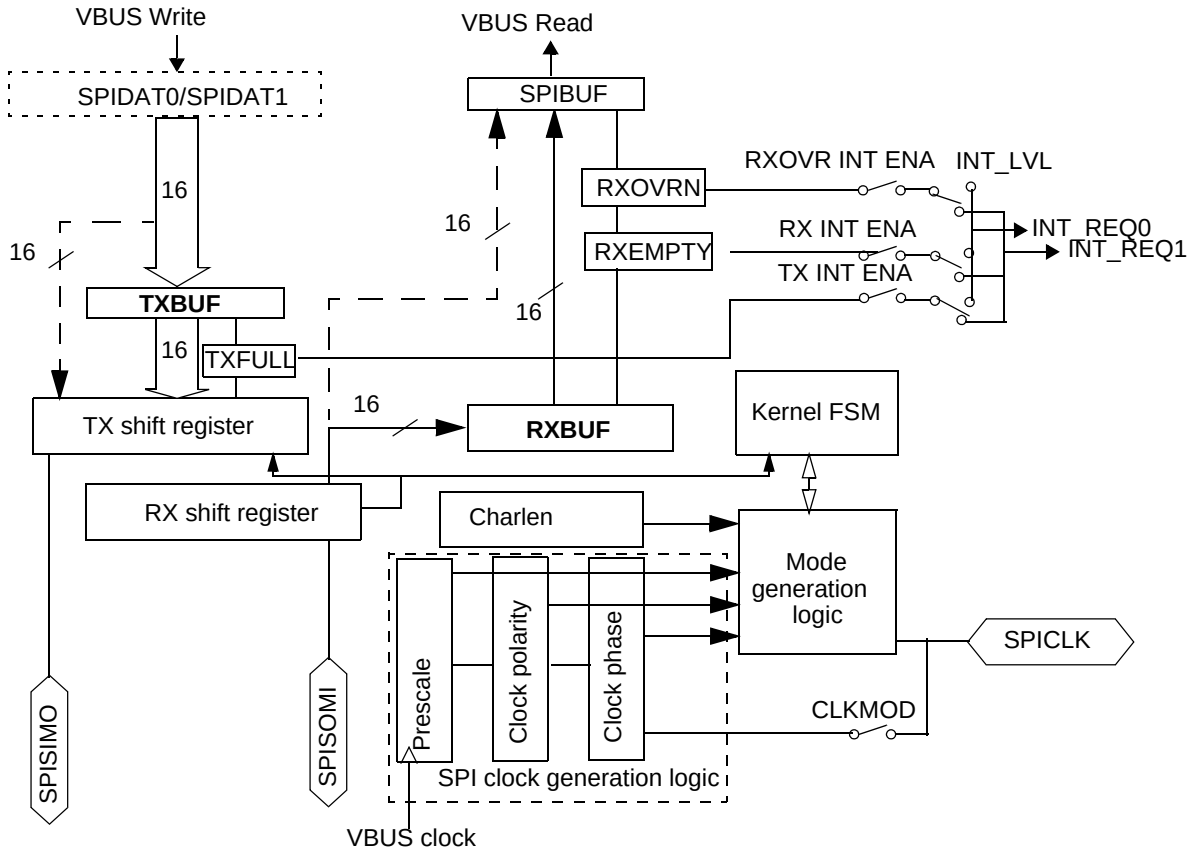
The slave chip select ($\overline{\text{SPISCS}}$[7:0]) pins, are used when communicating with multiple slave devices. When the a write occurs to SPIDAT1 in master mode, the $\overline{\text{SPISCS}}$ pins are automatically driven to select the specified slave.

A handshaking mechanism, provided by the $\overline{\text{SPIENA}}$ pin, enables a slave SPI to delay the generation of the clock signal supplied by the master if it is not prepared for the next exchange of data.

13.2.1 Data Handling

Figure 13-1 shows the SPI transaction hardware. TXBUF and RXBUF are internal buffers that are intended to improve the overall throughput of data transfer.

Figure 13-1. SPI Functional Logic Diagram



1 This is a representative diagram, which shows three-pin mode hardware.
2 TXBUF, RXBUF, and SHIFT_REGISTER are user-invisible registers.
3 SPIDAT0 and SPIDAT1 are user-visible, and are physically mapped to the contents of TXBUF.

13.2.1.1 Data Sequencing when SPIDAT0 or SPIDAT1 Is Written

- If both the TX shift register and TXBUF are empty, then the data is directly copied to the TX shift register. For devices with DMA, if DMA is enabled, a transmit DMA request (TX_DMA_REQ) is generated to cause the next word to be fetched. If transmit interrupts are enabled, a transmitter-empty interrupt is generated.
- If the TX shift register is already full or is in the process of shifting, then the data is copied to TXBUF, irrespective of whether it is already full or not and the TXFULL flag is set to 1 at the same time.
- When a shift operation is complete, data from the TXBUF (if it is full) is copied into TX shift register and the TXFULL flag is cleared to 0 to indicate that next data can be fetched. A transmit DMA request (if enabled) or a transmitter-empty interrupt (if enabled) is generated at the same time.

13.2.1.2 Data Sequencing when All Bits Shifted into RXSHIFT Register

- If both SPIBUF and RXBUF are empty, the received data in RX shift register is directly copied into SPIBUF and the receive DMA request (if enabled) is generated and the receive-interrupt (if enabled) is generated. The RXEMPTY flag in SPIBUF is cleared at the same time.
- If SPIBUF is already full at the end of receive completion, the RX shift register contents is copied to RXBUF. A receive DMA request is generated, if enabled. The receive complete interrupt line remains high.

- If SPIBUF is read by the CPU or DMA and if RXBUF is full, then the contents of RXBUF are copied to SPIBUF as soon as SPIBUF is read. RXEMPTY flag remains cleared, indicating that SPIBUF is still full.
- If both SPIBUF and RXBUF are full, then RXBUF will be overwritten and the RXOVR interrupt flag is set and an interrupt is generated, if enabled.

13.2.2 Pin Configurations

The SPI supports data connections as shown in Table 13-1.

Table 13-1. Pin Configurations

Pin	Master Mode		Slave Mode	
SPICLK	Drives the clock to external devices		Receives the clock from the external master	
SPISOMI	Receives data from the external slave		Sends data to the external master	
SPISIMO	Transmits data to the external slave		Receives data from the external master	
SPIENA	**SPIENA Disabled:** GIO	**SPIENA Enabled:** Receives ENA signal from the external slave	**SPIENA Disabled:** GIO	**SPIENA Enabled:** Receives ENA signal from the external master
SPICS[7:0]	**SPICS Disabled:** GIO	**SPICS Enabled:** Selects one or more slave devices	**SPICS Disabled:** GIO	**SPICS Enabled:** Receives the CS signal from the external master

Note:
When the SPICS[7:0] signals are disabled, the chip-select field in the transmit data is not used.

Note When the SPIENA signal is disabled, the $\overline{\text{SPIENA}}$ pin is ignored in master mode, and not driven as part of the SPI transaction in slave mode.

13.2.2.3 Three-Pin Mode

In master mode configuration (MASTER = 1 and CLKMOD = 1), the SPI provides the serial clock on the SPICLK pin. Data is transmitted on the SPISIMO pin and received on the SPISOMI pin (see Figure 13-2).

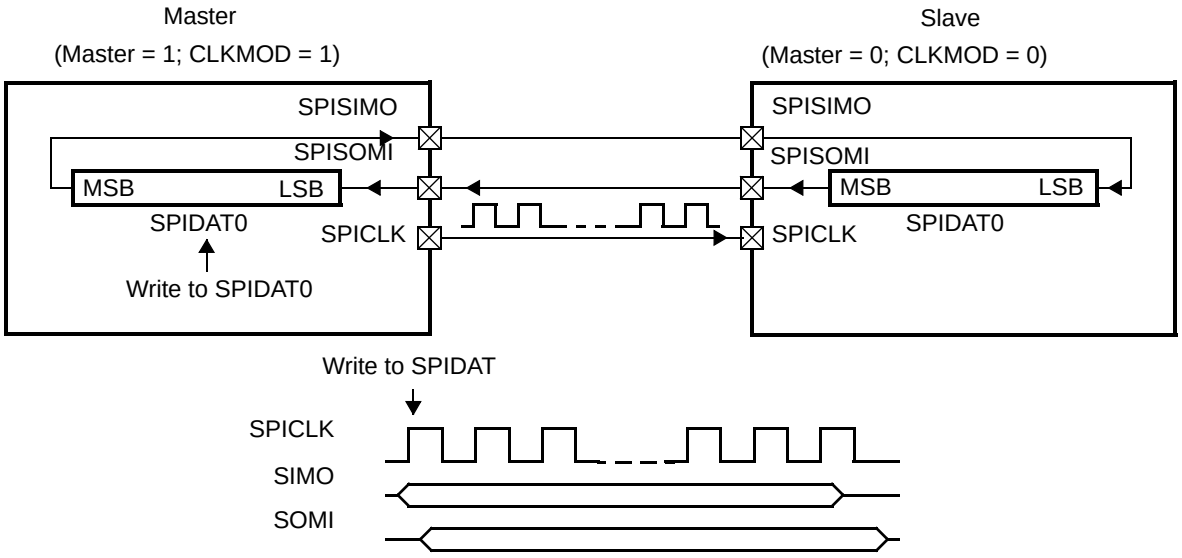
Data written to the shift register (SPIDAT0) initiates data transmission on the SPISIMO pin, MSB first. Simultaneously, received data is shifted through the SPISOMI pin into the LSB of the SPIDAT0 register. When the selected number of bits have been transmitted, the received data in the shift register is transferred to the SPIBUF register for the CPU to read. Data is stored right-justified in SPIBUF.

See Section 13.2.1.1, *Data Sequencing when SPIDAT0 or SPIDAT1 Is Written* on page 511 and Section 13.2.1.2, *Data Sequencing when All Bits Shifted into RXSHIFT Register* on page 511 for details about the data handling for transmit and receive operations.

In slave mode configuration (MASTER = 0 and CLKMOD = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock.

Data written to the SPIDAT0 register is transmitted to the network when the SPICLK signal is received from the network master. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts data on the SPISIMO pin into the RX shift register. If data is to be transmitted by the slave simultaneously, it must be written to the SPIDAT0 register before the beginning of the SPICLK signal.

Figure 13-2. SPI Three-Pin Operation



13.2.3 Operation with $\overline{\text{SPISCS}}$

In master mode, each chip select signal is used to select a specific slave. In slave mode, one chip select signal is used to enable/disable the transfer. Chip-select functionality is enabled by setting one of the $\overline{\text{SPISCS}}$ [7:0] pins as chip selects. It is disabled by setting all $\overline{\text{SPISCS}}$ [7:0] pins as GIOs.

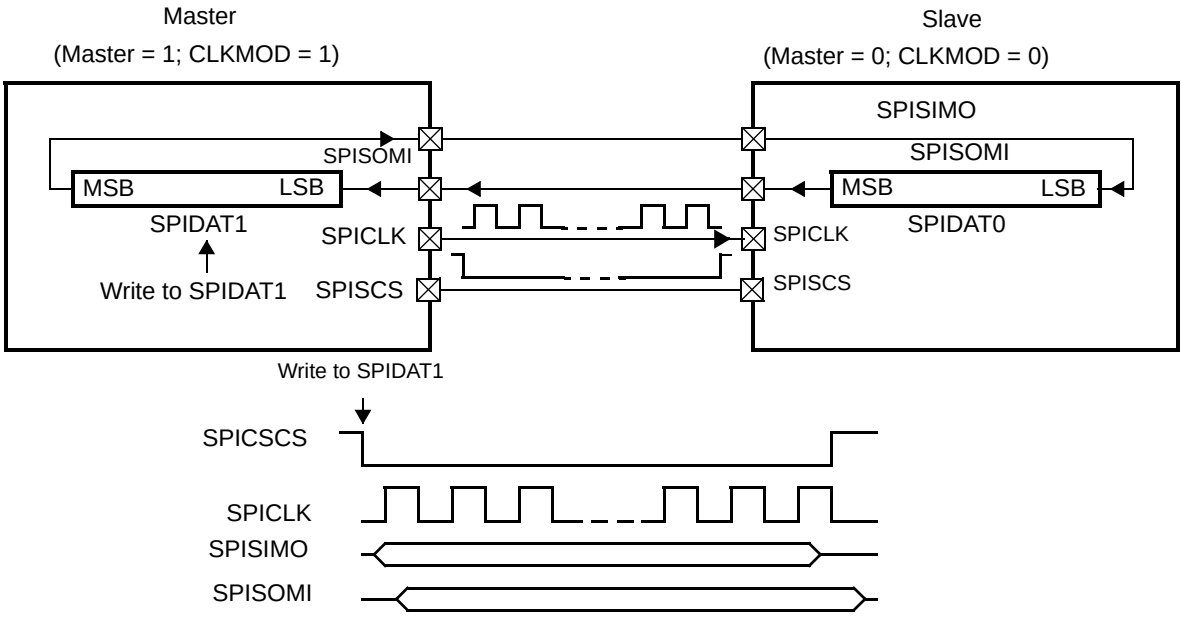
Multiple Chip Selects

The $\overline{\text{SPISCS}}$ [7:0] pins that are used must be configured as functional pins in the SPIPC0[7:0] register. The default pattern to be put on the $\overline{\text{SPISCS}}$ [7:0] when all the slaves are deactivated is set in the SPIDEF register. This pattern allows different slaves with different chip-select polarity to be activated by the SPI.

The master-mode SPI is capable of driving either 0 or 1 as the active value for any $\overline{\text{SPISCS}}$[7:0] output pin. The drive state for $\overline{\text{SPISCS}}$[7:0] pins is controlled by the CSNR field of SPIDAT1. The pattern that is driven will select the slave to which the transmission is dedicated.

In slave mode, the SPI can only be selected by an active value of 0 on any of its selected $\overline{\text{SPISCS}}$ input pin.

Figure 13-3. Operation with $\overline{\text{SPISCS}}$



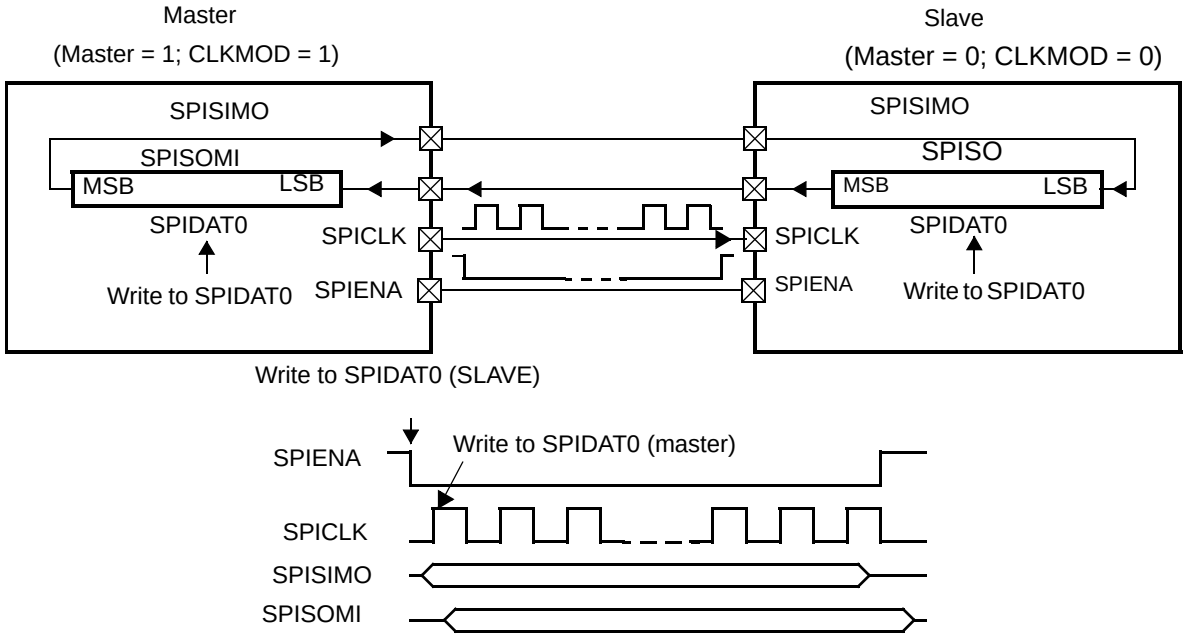
13.2.4 Operation with $\overline{\text{SPIENA}}$

The $\overline{\text{SPIENA}}$ operates as a WAIT signal pin. For both the slave and the master, the $\overline{\text{SPIENA}}$ pin must be configured to be functional (SPIPC0[8] = 1). In this mode, an active low signal from the slave on the $\overline{\text{SPIENA}}$ pin allows the master SPI to drive the clock pulse stream. A high signal tells the master to hold the clock signal (and delay SPI activity).

If the $\overline{\text{SPIENA}}$ pin is in high-z mode (ENABLE_HIGHZ = 1), the slave will put $\overline{\text{SPIENA}}$ into the high-impedance once it completes receiving a new character. If the $\overline{\text{SPIENA}}$ pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave will drive $\overline{\text{SPIENA}}$ to 1 once it completes receiving a new character. The slave will drive $\overline{\text{SPIENA}}$ low again for the next word to transfer, after new data is written to the slave TX shift register.

In master mode (CLKMOD = 1), if the $\overline{\text{SPIENA}}$ pin is configured as functional, then the pin will act as an input pin. If configured as a slave SPI and as functional, the $\overline{\text{SPIENA}}$ pin acts as an output pin.

Figure 13-4. Operation with $\overline{\text{SPIENA}}$

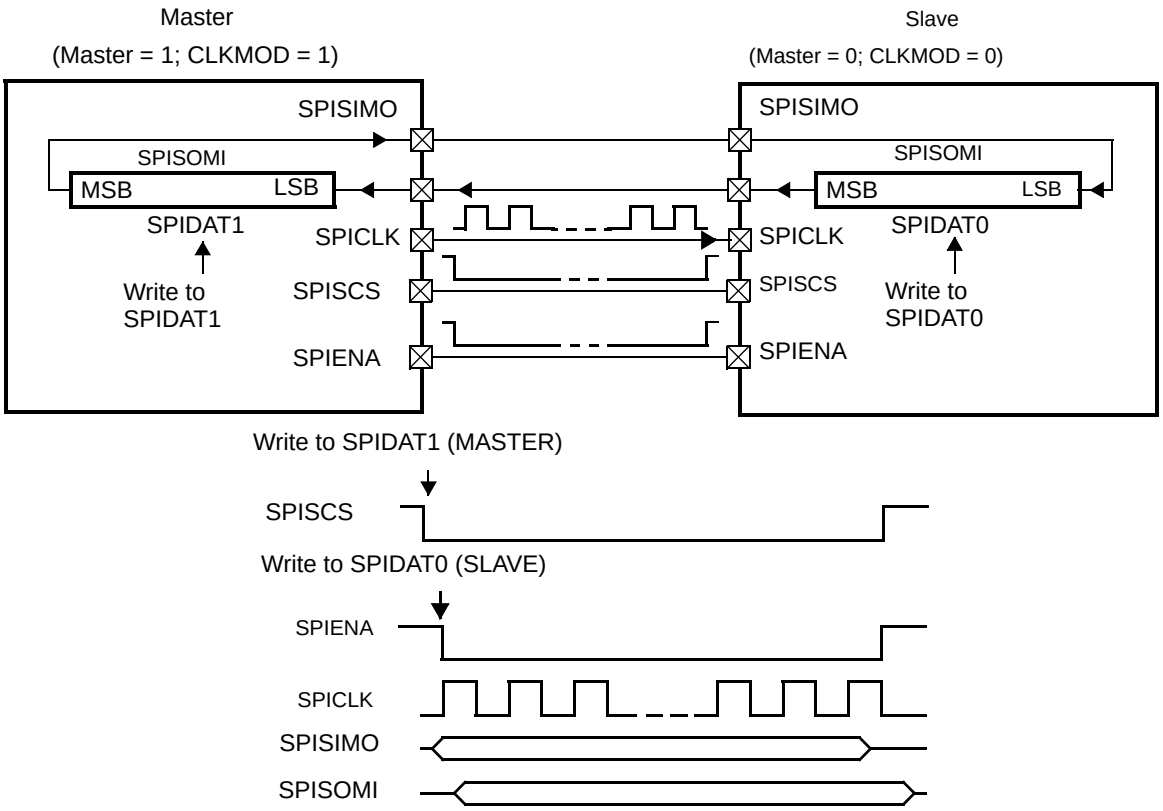


Note:
During a transfer, if a slave-mode SPI detects a deassertion of its chip select before its internal character length counter overflows, then it places SPISOMI and $\overline{\text{SPIENA}}$ (if ENABLE_HIGHZ bit is set to 1) in high-z mode. Once this condition has occurred, if a SPICLK edge is detected while the chip select is deasserted, then the SPI stops that transfer and sets an error flag DLENERR (data length) and generates an interrupt (if enabled).

13.2.5 Five-Pin Operation (Hardware Handshaking)

Five-pin operation combines the functionality of three-pin mode, plus the enable pin and one or more chip select pins. The result is full hardware handshaking. To use this mode, both the SPIENA pin and the required number of SPISCS [7:0] pins must be configured as functional pins.

Figure 13-5. SPI Five-Pin Option with SPIENA and SPISCS



If the SPIENA pin is in high-z mode (ENABLE_HIGHZ = 1), the slave SPI will put this signal into the high-impedance state by default. The slave will drive the signal SPIENA low when new data is written to the slave shift register and the slave has been selected by the master (SPISCS is low).

If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave SPI drives this pin high by default when it is in functional mode. The slave SPI will drive the SPIENA signal low when new data is written to the slave shift register (SPIDAT0/SPIDAT1) and the slave is selected by the master (SPISCS is low). If the slave is deselected by the master (SPISCS goes high), the slave SPIENA signal is driven high.

Note:
Push-pull mode of the SPIENA pin can be used only when there is a single slave in the system. When multiple SPI slave devices are connected to the common SPIENA pin, all of the slaves should configure their SPIENA pins in high-Z mode.

In master mode, if the SPISCS pins are configured as functional pins, then the pins will be in output mode. A write to the master's SPIDAT1/SPIDAT0 register will automatically drive the SPISCS signals low. The master will drive the SPISCS signals high again after completing the transfer of the bits of the data.

In slave mode (CLKMOD = 0), the SPISCS pins will act as functional inputs.

13.2.6 Data Formats

CHARLEN[4:0] specifies the number of bits (2 to 16) in the data word. The CHARLEN[4:0] value directs the state control logic to count the number of bits received or transmitted to determine when a complete word is transferred.

Data word length **must** be programmed to the same length for both the **master** and the **slave**. However, when chip selects are used, there may be multiple targets with different lengths in the system.

Note:
Data must be right-justified when it is written to the SPI for transmission irrespective of its character length or word length.

Figure 13-6 shows how a 12-bit word (0xEC9) needs to be written to the transmit buffer to be transmitted correctly.

Figure 13-6. Format for Transmitting an 8-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	x	x	x	x	1	1	0	0	1	0	0	1

Note:
The received data is always stored right-justified regardless of the character length or direction of shifting and is padded with leading 0s when the character length is less than 16.

Figure 13-7 shows how a 10-bit word (0x0A2) is stored in the buffer once it is received.

Figure 13-7. Format for Receiving an 8-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

To support multiple different types of slaves in one SPI network, four independent data word formats are implemented that allow configuration of individual data word length, polarity, phase, and bit rate. Each word transmitted can select which data format to use via the bits DFSEL[1:0] in its control field from one of the four data word formats.

Data formats 0, 1, 2, and 3 can be configured through control registers.

Each SPI data format includes the standard SPI data format with enhanced features:

- Individually-configurable shift direction can be used to select MSB first or LSB first, whereas the position of the MSB depends on the configured data word length.
- Receive data is automatically right-aligned, independent of shift direction and data word length. Transmit data has to be written right-aligned into the SPI and the internal shift register will transmit according to the selected shift direction and data word length for correct transfer.
- To increase fault detection of data transmission and reception, an odd or even parity bit can be added at the end of a data word. The parity generator can be enabled or disabled individually for each data format. If a received parity bit does not match with the locally calculated parity bit, the parity error flag (PARITYERR) is set and an interrupt is asserted (if enabled).
- Since the master-mode SPI can drive two consecutive accesses to the same slave, an 8-bit delay counter is available to satisfy the delay time for data to be refreshed in the accessed slave. The delay counter can be programmed as part of the data format.

13.2.7 Clocking Modes

SPICLK may operate in four different modes, depending on the choice of phase (delay/no delay) and the polarity (rising edge/falling edge) of the clock.

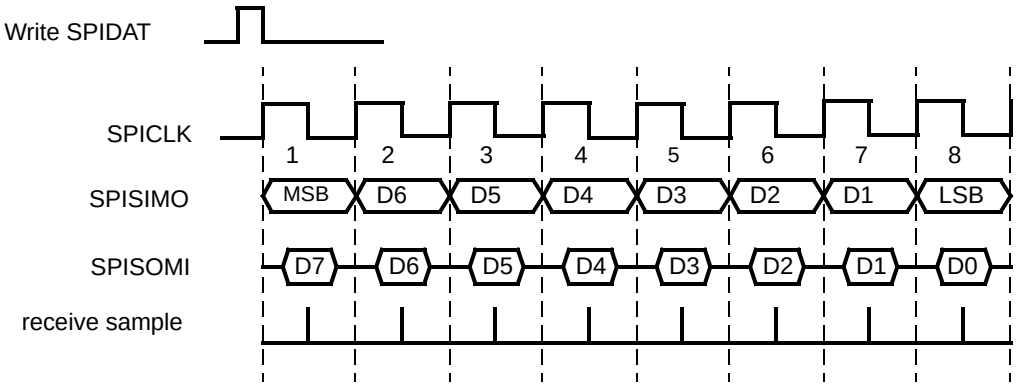
The data input and output edges depend on the values of both POLARITY and PHASE as shown in Table 13-2.

Table 13-2. Clocking Modes

POLARITY	PHASE	ACTION
0	0	Data is output on the rising edge of SPICLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK.
1	0	Data is output on the falling edge of SPICLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.

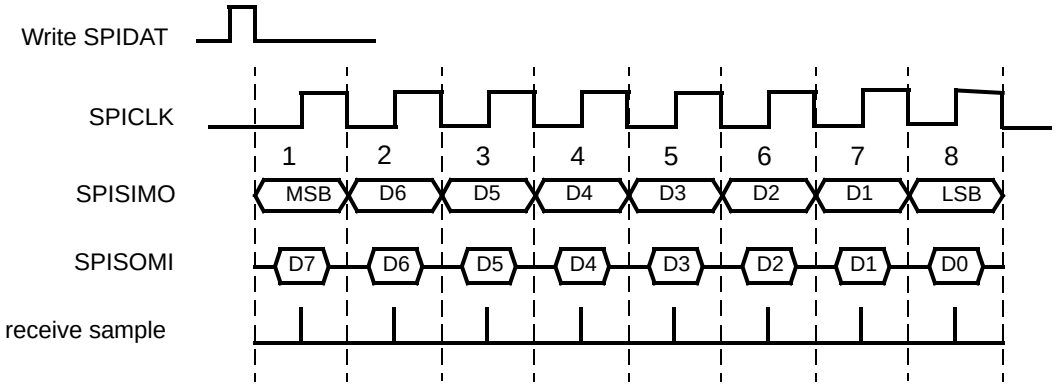
Figure 13-8 to Figure 13-11 illustrate the four possible configurations of SPICLK corresponding to each mode. Having four signal options allows the SPI to interface with many different types of serial devices.

Figure 13-8. Clock Mode with Polarity = 0 and Phase = 0



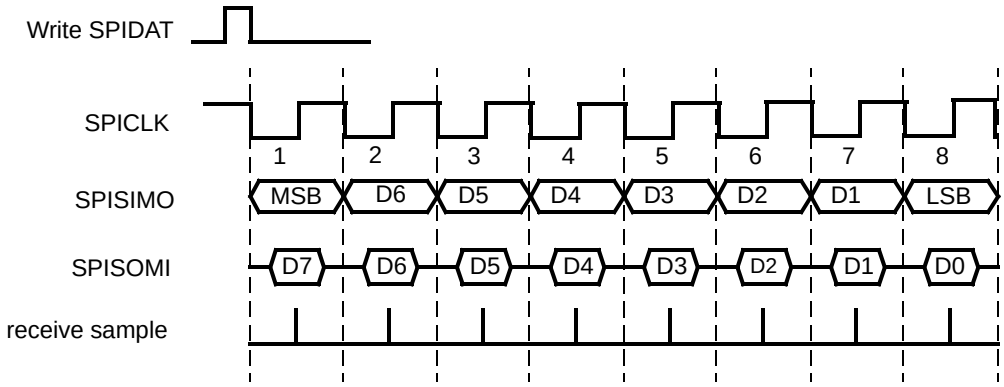
1 Data is output on the rising edge of SPICLK.
2 Input data is latched on the falling edge of SPICLK.

Figure 13-9. Clock Mode with Polarity = 0 and Phase = 1



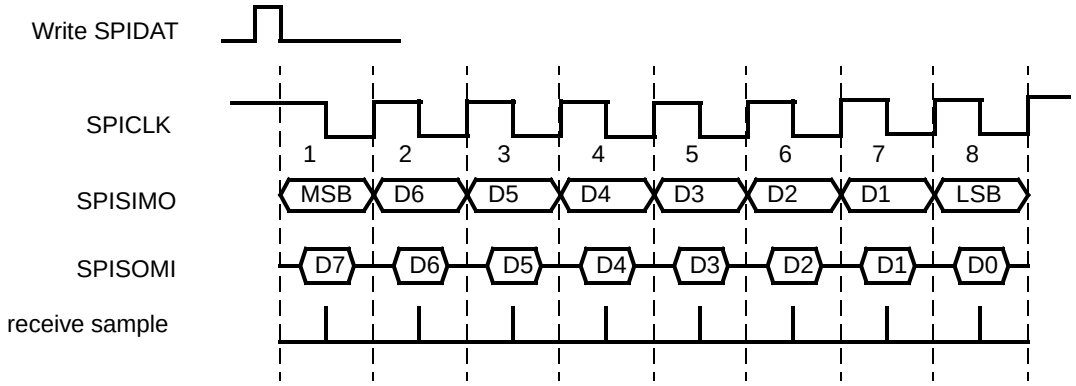
1 Data is output one-half cycle before the first rising edge of SPICLK and on subsequent falling edges of SPICLK
2 Input data is latched on the rising edge of SPICLK

Figure 13-10. Clock Mode with Polarity = 1 and Phase = 0



1 Data is output on the falling edge of SPICLK.
2 Input data is latched on the rising edge of SPICLK.

Figure 13-11. Clock Mode with Polarity = 1 and Phase = 1

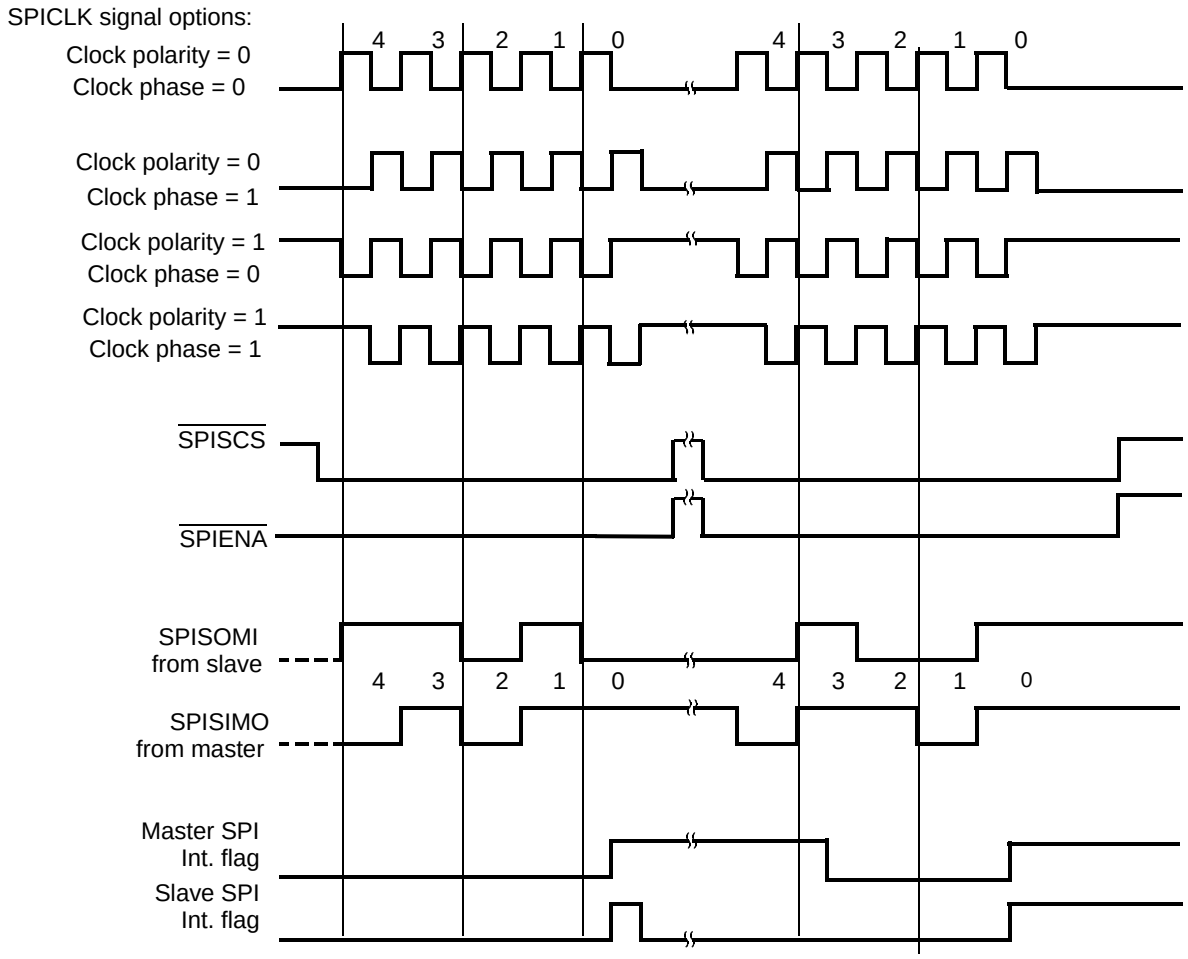


1 Data is output one-half cycle before the first falling edge of SPICLK and on the subsequent rising edges of SPICLK.
2 Input data is latched on the falling edge of SPICLK.

13.2.8 Data Transfer Example

Figure 13-12 illustrates a SPI data transfer between two devices using a character length of five bits.

Figure 13-12. Five Bits per Character (5-Pin Option)



13.2.9 Decoded and Encoded Chip Select (Master Only)

The SPI can connect to up to 8 individual slave devices using decoded chip-selects by routing one wire to each slave. For the decoded mode, the 8 chip selects in the control field are directly connected to the 8 pins. The default value of each chip select, i.e., not active, can be configured via the register CSDEF. During a transmission, the value of the chip select control field (CSNR[7:0]) of the SPIDAT1 register (SPIDAT1[23:16]) is driven on the SPISCS [7:0] pins (if available). When the transmission finishes the default chip-select value (defined by the CSDEF register) is put on the SPISCS [7:0] pins.

The SPI can support more than 8 slaves by using encoded chip selects. To connect the SPI with encoded slaves devices, the CSNR field allows multiple active SPISCS pins at the same time, which enables binary encoded chip selects from 0 to 255. To use encoded chip selects, all eight chip select lines have to be connected to each slave device and each slave needs to have a unique chip-select address. The CSDEF register is used to provide the address at which slaves devices are all de-selected.

Users can combine decoded and encoded chip selects. For example, *n* SPISCS pins can be used for encoding a *n*-bit address and the remaining pins can be connected to decoded-mode slaves.

13.2.10 Variable Chip Select Setup and Hold Timing (Master Only)

In order to support slow slave devices a delay counter can be configured to delay data transmission after the chip select is activated. A second delay counter can be configured to delay the chip select deactivation after the last data bit is transferred. Both delay counters are clocked with VCLK.

If a particular data format specifically does not require these additional set-up or hold times for the chip select pin(s), then they can be disabled in the corresponding SPIFMTx register.

13.2.11 Hold Chip-Select Active

Some slave devices require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers.

CSHOLD is programmable in both master and slave modes of the multi-buffer mode of SPI. However, the meaning of CSHOLD in master mode and slave mode are different.

Note:
If the CSHOLD bit is set within the current data control field, the programmed hold time and the following programmed set-up time will not be applied between transactions.

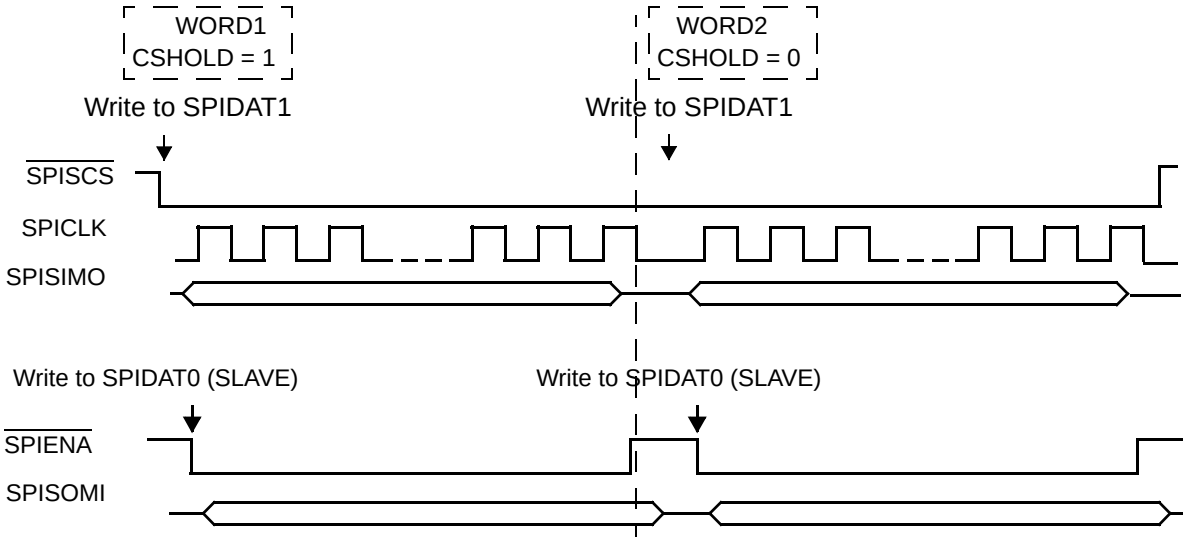
13.2.11.4 The CSHOLD Bit in Master Mode

Each word in a master-mode SPI can be individually initialized for one of the two modes via the CSHOLD bit in its control field.

If the CSHOLD bit is set in the control field of a word, the chip select signal will not be deactivated until the next control field is loaded with new chip select information. Since the chip-select is maintained active between two transfers, the chip-select hold delay (T2CDELAY) is not applied at the end of the current transaction, and the chip-select set-up time delay (C2TDELAY) is not applied as well at the beginning of the following transaction. However, the wait delay (WDELAY) will be still applied between the two transactions, if the WDEL bit is set within the control field.

Figure 13-13 shows the SPI pins when a master-mode SPI transfers a word that has its CSHOLD bit set. The chip-select pins will not be deasserted after the completion of this word. If the next word to transmit has the same chip-select number (CSNR) value, the chip select pins will be maintained until the completion of the second word, regardless of whether the CSHOLD bit is set or not.

Figure 13-13. Typical Diagram when a Buffer in Master is in CSHOLD Mode (SPI-SPI)



13.2.12 Detection of Slave Desynchronization (Master Only)

When a slave supports generation of an enable signal (ENA), de-synchronization can be detected. With the enable signal a slave indicates to the master that it is ready to exchange data. A de-synchronization can occur if one or more clock edges are missed by the slave. In this case the slave may block the SOMI line until it detects clock edges corresponding to the next data word. This would corrupt the data word of the de-synchronized slave and the consecutive data word. A configurable 8 bit time-out counter, which is clocked with SPICLK, is implemented to detect this slave malfunction. After the transmission has finished (end of last bit transferred: either last data bit or parity bit) the counter is started. If the ENA signal generated by the slave does not become inactive before the counter overflows, the DESYNC flag is set and an interrupt is asserted (if enabled).

Note: Inconsistency of Desync Flag in Compatibility Mode MibSPI.

Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is due to the fact that receive completion flag/interrupt will be generated when the buffer transfer is completed. But desync will be detected after the buffer transfer is completed. So, if VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.

This inconsistency in the desync flag is valid only in compatibility mode of MibSPI. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.

13.2.13 ENA Signal Time-Out (Master Only)

The SPI in master mode waits for the hardware handshake signal (ENA) coming from the addressed slave before performing a data transfer. To avoid stalling the SPI by a non-responsive slave device a time-out value can be configured. If the time-out counter overflows before an active ENA signal is sampled the TIMEOUT flag in the status register SPIFLG is set and the TIMEOUT flag in the status field of the corresponding buffer is set.

Note: When the chip select signal becomes active, no breaks in transmission are allowed. The next arbitration is performed while waiting for the time-out to occur.

13.2.14 Data-Length Error

A SPI can generate an error flag by detecting any mismatch in length of received or transmitted data and the programmed character length under certain conditions.

Data-Length Error in Master Mode: During a data transfer, if the SPI detects a de-assertion of the $\overline{\text{SPIENA}}$ pin (by the slave) while the character counter is not overflowed, then an error flag is set to indicate a data-length error. This can be caused by a slave receiving extra clocks (e.g. due to noise on the SPICLK line).

Note: In a master mode SPI, the data length error will be generated only if the $\overline{\text{SPIENA}}$ pin is enabled as a functional pin.

Data-Length Error in Slave Mode: During a transfer, if the SPI detects a de-assertion of the $\overline{\text{SPISCS}}$ pin before its character length counter overflows, then an error flag is set to indicate a data-length error. This situation can arise If the slave SPI misses one or more SPICLK pulses from the master. This error in slave mode implies that both the transmitted and received data were not complete.

Note: In a slave-mode SPI, the data-length error flag will be generated only if at least one of the $\overline{\text{SPISCS}}$(x) pins are configured as functional, and are being used for selecting the slave.

13.2.15 Continuous Self-Test (Master/Slave)

During data transfer, the SPI compares its own internal transmit data with its transmit data on the bus. The sample point for the compare is at one-half SPI clock after transmit point. If the data on the bus does not match the expected value, the bit-error (BITERR) flag is set and an interrupt is asserted if enabled.

Note: The compare is made from the output pin using its input buffer.

13.3 Test Features

13.3.1 Internal Loop-Back Test Mode (Master Only)

The internal loop-back self-test mode can be utilized to test the SPI transmit and receive paths, including the shift registers, the SPI buffer registers, and the parity generator. In this mode the transmit signal is internally fed back to the receiver, whereas the SIMO, SOMI, and CLK pin are disconnected, i.e., the transmitted data is internally transferred to the corresponding receive buffer while external signals remain unchanged.

This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

Note: This mode cannot be changed during transmission.

13.3.2 I/O Loopback Test Mode

I/O Loopback Test mode supports the testing of all I/Os without the aid of an external interface. Loopback can be configured as either analog-loopback (loopback through the pin-level input/output buffers) or digital loopback (internal to the SPI module). With I/O Loopback, all functional features of the SPI can be tested. Transmit data is fed back through the receive-data line(s). See Figure 13-14 for a diagram of the types of feedback available. The IOLPBKTSTCR register defines all of the available control fields.

In loopback mode, it is also possible to induce various error conditions. See Section 13.7.35, *I/O-Loopback Test Control Register (IOLPBKTSTCR)* on page 613 for details of the register field controlling these features.

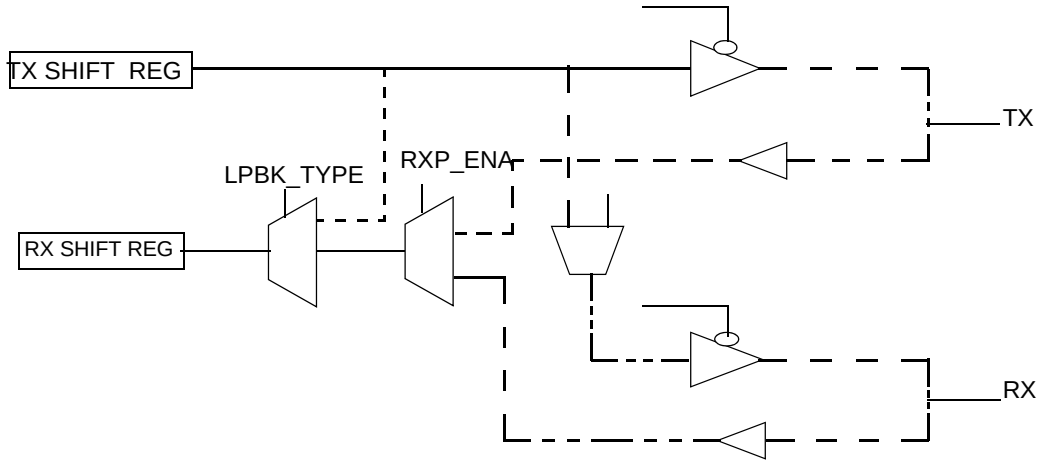
In I/O loopback test modes, even when the module is in slave mode, the SPICLK is generated internally. This SPICLK is used for all loopback-mode SPI transactions. Slave-mode features can be tested without the help of another master SPI, using the internally-generated SPICLK. Chip selects are also generated by the slave itself while it is in I/O loopback mode.

In I/O loopback test modes, if the module is in master mode, the nENA signal is also generated by internal logic so that an external interface is not required.

Note: Usage Guideline for I/O Loopback

I/O Loopback mode should be used with caution because, in some configurations, even the receive pins will be driven with transmit data. During testing, it should be ensured that none of the SPI pins are driven by any other device connected to them. Otherwise, if analog loopback is selected in I/O Loopback mode, then testing may damage the device.

Figure 13-14. I/O Paths during I/O Loopback Modes



1 This diagram is intended to illustrate loopback paths and therefore may omit some normal-mode paths.

13.3.2.1 IO Loopback Mode Operation in Slave Mode

In multi-buffer slave mode, there are some additional requirements for using I/O loopback mode (IOLPBK). In multi-buffer slave mode, the chip-select pins are the triggers for various TGs. Enabling the IOLPBK mode by writing 0xA to the IOLPBTSTENA bits of the IOLPBKTSTCR register triggers TG0 by driving SPISCS[7:0] to 0x0. The actual number of chip selects can be programmed to have any or all of the SPISCS pins as functional. All other configurations should be completed before enabling the IOLPBK mode in multi-buffer slave mode since it triggers TG0.

After the first buffer transfer is completed, the CSNR[3:0] field of the current buffer is used to trigger the next buffer. So, if multiple TGs are desired to be tested, then the CSNR field of the final buffer in each TG should hold the number of the next TG to be triggered. As long as TG boundaries are well defined and are enabled, the completion of one TG will trigger the next TG.

To stop the transfer in multi-buffer slave mode in I/O Loopback configuration, either IOLPBK mode can be disabled by writing 0x5 to the IOLPBTSTENA bits or all of the TGs can be disabled.

13.4 General-Purpose I/O

All of the SPI pins may be programmed via the SPIPC control registers to be either functional or general-purpose I/O pins.

If the SPI function is to be used, application software must ensure that at least the SPICLK pin and the SOMI and/or SIMO pins are configured as SPI functional pins, and not as GIO pins, or else the SPI state machine will be held in reset, preventing SPI transactions.

13.5 Low-Power Mode

The SPI can be put into either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SPI. During global low-power mode, all clocks to the SPI are turned off, making the module completely inactive.

Local low-power mode is asserted by setting the POWERDOWN (SPIGCR1[8]) bit; setting this bit stops the clocks to the SPI internal logic and registers. Setting the POWERDOWN bit causes the SPI to enter local low-power mode and clearing the POWERDOWN bit causes SPI to exit from local low-power mode. All registers remain accessible during local power-down mode, since the clock to the SPI registers is temporarily re-enabled for each access. RAM buffers are also accessible during low power mode.

Note: Since entering a low-power mode has the effect of suspending all state-machine activities, care must be taken when entering such modes to ensure that a valid state is entered when low-power mode is active. Application software must ensure that a low power mode is not entered during a data transfer.

13.6 Interrupts

There are two levels of vectorized interrupts supported by the SPI. These interrupts can be caused under the following circumstances:

- Transmission error
- Receive overrun
- Receive complete (receive buffer full)
- Transmit buffer empty

These interrupts may be enabled or disabled via the SPIINT0 register.

During transmission, if one of the following errors occurs: BITERR, DESYNC, DLENERR, PARITYERR, or TIMEOUT, the corresponding bit in the SPIFLG register is set. If the corresponding enable bit is set, then an interrupt is generated. The level of all the above interrupts is set by the bit fields in the SPILVL register.

The error interrupts are enabled and prioritized independently from each other, but the interrupt generated will be the same if multiple errors are enabled on the same level. The SPIFLG register should be used to determine the actual cause of an error.

Note:
Since there are two interrupt lines, one each for Level 0 and Level 1, it is possible for a programmer to separate out the interrupts for receive buffer full and transmit buffer empty. By programming one to Level 0 and the other to Level 1, it is possible to avoid a check on whether an interrupt occurred for transmit or for receive.

A programmer can also choose to group all of the error interrupts into one interrupt line and both TX-empty and RX-full interrupts into another interrupt line using the LVL control register. In this way, it is possible to separate error-checking from normal data handling.

13.6.1 Interrupts in Multi-Buffer Mode

In multi-buffer mode, the SPI can generate interrupts on two levels.

In normal multi-buffer operation, the receive and transmit are not used and therefore the enable bits of SPIINT0 are not used.

The interrupts available in multi-buffer mode are:

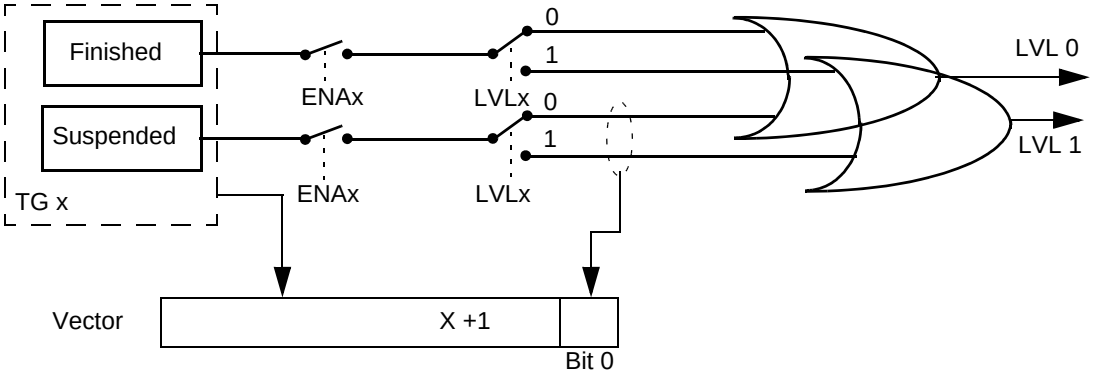
- Transmission error interrupt
- Receive overrun interrupt
- TG suspended interrupt
- TG completed interrupt

When a TG has finished and the corresponding enable bit in the TGINTENA register is set, a transfer-finished interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

When a TG is suspended by a buffer that has been set as suspend to wait until TXFULL flag or/and RXEMPTY flag are set, and if the corresponding bit in the TGINTENA register is set, an transfer-suspended interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

Figure 13-15 illustrates the TG interrupts.

Figure 13-15. TG Interrupt Structure

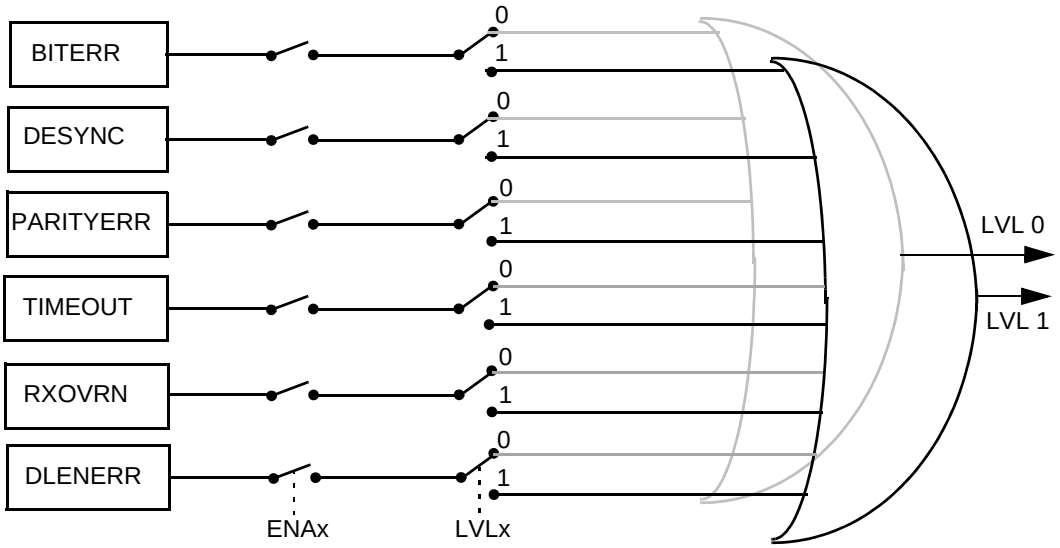


During transmission, if one of the following errors occurs, BITERR, DESYNC, PARITYERR, TIMEOUT, DLENERR, the corresponding flag in the SPIFLG register is set. If the enable bit is set, then an interrupt is generated. The level of the interrupts could be generated according to the bit field in SPILVL.

The RXOVRN interrupt is generated when a buffer in the RXRAM is overwritten by a new received word. While writing newly received data to a RXRAM location, if the RXEMPTY bit of the corresponding location is 0, then the RXOVR bit will be set to 1 during the write operation, so that the buffer starts to indicate an overrun. This RXOVR flag is also reflected in SPIFLG register as RXOVRNINTFLG and the corresponding vector number is updated in TGINTVECT0/TGINTVECT1 register. If an overrun interrupt is enabled, then an interrupt will be generated indicating an overrun condition.

The error interrupts are enabled and prioritized independently from each other, but the vector generated by the SPI will be the same if multiple errors are enabled on the same level.

Figure 13-16. SPIFLG Interrupt Structure



Since the priority of an error interrupt is lower than a completion/suspend interrupt for a TG, the interrupts can be split into two levels. By programming all the error interrupts into Level 0 and TG-complete / TG-suspend interrupts into Level 1, it is possible to get a clear indication of the source of error interrupts. However, when a vector register shows an error interrupt, the actual buffer for which the error has occurred is not readily identifiable. Since each buffer in the multi-buffer RAM is stored along with its individual status flags, each buffer should be read until a buffer with any error flag set is found.

13.7 Control Registers

This section describes the SPI control, data, and pin registers. The registers support 8-bit, 16-bit and 32-bit writes. The offset is relative to the associated base address of this module in a system. The base address for the control registers is 0xFFF7 F400.

Table 13-3. SPI Registers

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00h SPIGCR0 page 541	Reserved															
	Reserved															nRESET
0x04h SPIGCR1 page 542	Reserved							SPIEN	Reserved							LOOP BACK
	Reserved							POWER DOWN	Reserved						CLK MOD	MAS-TER
0x08h SPIINT0 page 544	Reserved							ENABLE HIGH Z	Reserved							DMA REQ EN
	Reserved						TXINT ENA	RXINT ENA	Reserved	OVRN INT ENA	Reserved	BIT ERR ENA	DESYNC ENA	PAR ERR ENA	TIME OUT ENA	DLEN ERR ENA
0x0Ch SPILVL page 547	Reserved															
	Reserved						TX INT LVL	RX INT LVL	Reserved	OVRN INT LVL	Reserved	BITERR LVL	DESYNC LVL	PAR ERR LVL	TIME OUT LVL	DLEN ERR LVL
0x10h SPIFLG page 549	Reserved							BUF INIT ACTIVE	Reserved							
	Reserved						TX INT FLG	RX INT FLG	Reserved	OVRN INT FLG	Reserved	BIT ERR FLG	DESYNC FLG	PAR ERR FLG	TIME OUT FLG	DLEN ERR FLG
0x14h SPIPC0 page 554	SOMIFUN[7:0]								SIMOFUN[7:0]							
	Reserved				SOMI FUN	SIMO FUN	CLK FUN	ENA FUN	SCSFUN[7:0]							
0x18h SPIPC1 page 556	SOMIDIR[7:0]								SIMODIR[7:0]							
	Reserved				SOMI DIR	SIMO DIR	CLK DIR	ENA DIR	SCSDIR[7:0]							

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x1Ch SPIPC2 page 558	SOMIDIN[7:0]								SIMODIN[7:0]							
	Reserved				SOMI DIN	SIMO DIN	CLK DIN	ENA DIN	SCSDIN[7:0]							
0x20h SPIPC3 page 560	SOMIDOUT[7:0]								SIMODOUT[7:0]							
	Reserved				SOMI DOUT	SIMO DOUT	CLK DOUT	ENA DOUT	SCSDOUT[7:0]							
0x24h SPIPC4 page 562	SOMISET[7:0]								SIMOSET[7:0]							
	Reserved				SOMI SET	SIMO SET	CLK SET	ENA-SET	SCSSET[7:0]							
0x28h SPIPC5 page 564	SOMICLR[7:0]								SIMOCLR[7:0]							
	Reserved				SOMI CLR	SIMO CLR	CLK CLR	ENA CLR	SCSCLR[7:0]							
0x2Ch SPIPC6 page 566	SOMIPDR[7:0]								SOMICLR[7:0]							
	Reserved				SOMI PDR	SIMO PDR	CLK PDR	ENA PDR	SCSPDR[7:0]							
0x30h SPIPC7 page 568	SOMIDIS[7:0]								SIMODIS[7:0]							
	Reserved				SOMI PDIS	SIMO PDIS	CLK PDIS	ENA PDIS	SCSPDIS[7:0]							
0x34h SPIPC8 page 570	SOMIPSEL[7:0]								SIMOPSEL[7:0]							
	Reserved				SOMI PSL	SIMO PSL	CLK PSL	ENA PSL	SCSPSL[7:0]							
0x38h SPIDAT0 page 572	Reserved															
	TXDATA(15–0)															

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x3Ch SPIDAT1 page 573	Reserved			CS HOLD	Reserved	WDEL	DFSEL		CSNR(7–0)							
	TXDATA(15–0)															
0x40h SPIBUF page 575	RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x44h SPIEMU page 579	Reserved															
	RXDATA(15–0)															
0x48h SPIDELAY page 580	C2TDELAY(7–0)								T2CDELAY(7–0)							
	T2EDELAY(7–0)								C2EDELAY(7–0)							
0x4Ch SPIDEF page 584	Reserved															
	Reserved								CSDEF[7:0]							
0x50h–5Ch SPIFMT[0:3] page 585	Reserved		WDELAY[0:3](5–0)						PARPOL[0:3]	PAR-ITY[0:3]ENA	WAITENA[0:3]	SHIFTDIR[0:3]	Reserved	DIS CS TIM-ERS	POLARITY[0:3]	PHASE[0:3]
	PRESCALE[0:3]								Reserved			CHARLEN[0:3]				
0x60h–64h TGINTVECT[0:1] page 588	Reserved															
	Reserved										INTVECT[0:1]					SUSPEND[0:1]
0x68h SPIPC9 page 592	SOMISRS[7:0]								SIMOSRS[7:0]							
	Reserved				SOMISRS0	SIMOSRS0	CLK-SRS	ENA-SRS	SCSSRS[7:0]							

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x70h MIBSPIE page 594	Reserved															RX RAM ACCESS
	Reserved															MibSPI ENA
0x74h TGITENST page 596	SET INTEN RDY[15:0]															
	SET INTEN SUS[15:0]															
0x78h TGITENCR page 597	CLR INTEN RDY[15:0]															
	CLR INTEN SUS[15:0]															
0x7Ch TGITLVST page 598	SET INTLVL RDY[15:0]															
	SET INTLVL SUS[15:0]															
0x80h TGITLVCR page 599	CLR INTLVL RDY[15:0]															
	CLR INTLVL SUS[15:0]															
0x84h TGITFLG page 600	INTFLG RDY[15:0]															
	INTFLG SUS[15:0]															
0x88–0x8Ch	Reserved															
	Reserved															
0x90h TICKCNT page 602	TICK ENA	RE LOAD	CLKCTRL(1–0)	Reserved												
	TICKVALUE(15–0)															

1 The base address of these registers can be found in the device data sheet.

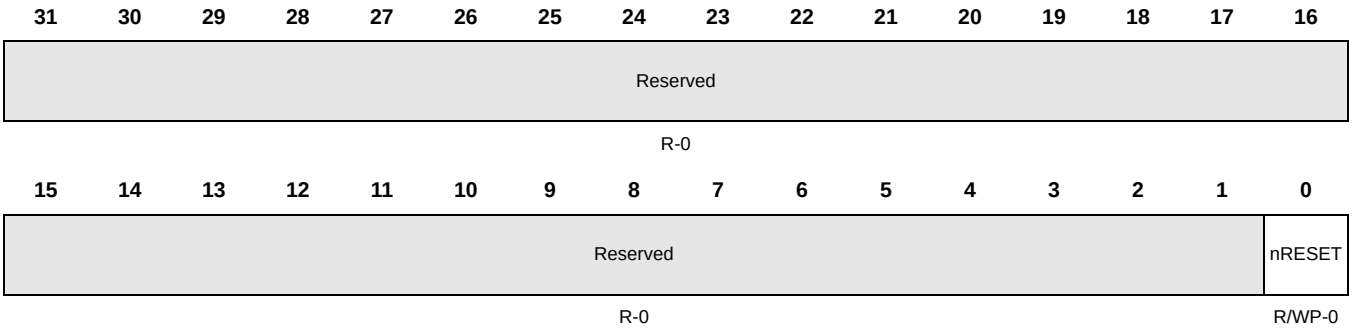
Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x94h LTGPEND page 604	Reserved				TG IN SERVICE(4–0)				Reserved							
	Reserved	LPEND(6–0)							Reserved							
0x98–D4h TG[0:15]CTRL page 606	TG ENA[0:15]	ONE SHOT [0:15]	PRST [0:15]	TGTD[0:15]	Reserved				TRGEVT[0:15](4–0)				TRIGSRC[0:15](4–0)			
	Reserved	PSTART[0:15](6–0)							Reserved	PCURRENT[0:15](6–0)						
0x11Ch	Reserved															
	Reserved															
0x20h-0x12Ch	Reserved															
	Reserved															
0x130h RXOVRN_BUF_ADDR page 612	Reserved															
	Reserved						RXOVRN_BUF_ADDR(9–0)									
0x134h IOLPBKTSTCR page 613	Reserved							SCS FAIL FLG	Reserved			CTRL_ BITERR	CTRL_ DESYNC	CTRL_ PAR ERR	CTRL_ TIME OUT	CTRL_ DLEN ERR
	Reserved			IOLPBKTSTENA(3–0)					Reserved	ERR SCS PIN(2–0				CTRL SCS PIN ERR	LPBK_ TYPE	RXP_ ENA

1 The base address of these registers can be found in the device data sheet.

Note:
TI highly recommends that write values corresponding to the reserved locations of registers be maintained as 0 consistently. This allows future enhancements to use these reserved bits as control bits without affecting the functionality of the module with any older versions of software.

13.7.1 SPI Global Control Register 0 (SPIGCR0)

Figure 13-17. SPI Global Control Register 0 (SPIGCR0) [offset = 00h]



R = Read in all modes; WP = Write in privilege mode only; -n = value after reset

Table 13-4. SPI Global Control Register 0 (SPIGCR0) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	nRESET		Reset bit for the module. This bit needs to be set to 1 before any operation on SPI can be done. After setting this bit, auto Initialization of multi-buffer RAM starts.
		0	SPI is in the reset state.
		1	SPI is out of the reset state.

13.7.2 SPI Global Control Register 1 (SPIGCR1)

Figure 13-18. SPI Global Control Register 1 (SPIGCR1) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SPIEN	Reserved							LOOP-BACK
R-0							R/W-0	R-0							R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							POWER-DOWN	Reserved						CLK-MOD	MAS-TER
R-0							R/W-0	R-0						R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; WP = Write in privilege mode only; -n = value after reset

Table 13-5. SPI Global Control Register 1 (SPIGCR1) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	SPIEN		SPI enable. This bit enables SPI transfers. This bit must be set to 1 after all other SPI configuration bits have been written. When the SPIEN bit is 0 or cleared to 0, the following SPI registers get forced to their default states: •Both TX and RX shift registers •The TXDATA fields of SPIDAT0 and SPIDAT1 registers •All the fields of the SPIFLG register •Contents of SPIBUF and the internal RXBUF registers
		0	The SPI is not activated for transfers.
		1	Activates SPI
23–17	Reserved		Reads return zero and writes have no effect.
16	LOOPBACK		Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPISIMO and SPISOMI pins are configured with SPI functionality, then the SPISIMO pin internally connected to the SPISOMI pin (transmit data is looped back as receive data). GIO mode for these pins is not supported in loopback mode. Externally, during loop-back operation, the SPICLK pin outputs an inactive value and SPISOMI remain in the high-impedance state. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result. **Note: This loopback mode can only be used in master mode. Master mode must be selected before setting LOOPBACK. When this mode is selected, the CLKMOD bit should be set to 1, meaning that SPICLK is internally generated.**
		0	Internal loop-back test mode disabled.

Table 13-5. SPI Global Control Register 1 (SPIGCR1) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Internal loop-back test mode enabled.
15–9	Reserved		Reads return zero and writes have no effect.
8	POWERDOWN		When active, the SPI state machine enters a power-down state.
		0	The SPI is in active mode.
		1	The SPI is in power-down mode.
7–2	Reserved		Reads return zero and writes have no effect.
1	CLKMOD		Clock mode. This bit selects either an internal or external clock source. This bit also determines the I/O direction of the SPIENA and SPISCS[7:0] pins in functional mode.
		0	Clock is external. •SPIENA is an output. •SPISCS[7:0] are inputs.
		1	Clock is internally-generated. •SPIENA is an output. •SPISCS[7:0] are outputs.
0	MASTER		SPISIMO/SPISOMI pin direction determination. Sets the direction of the SPISIMO and SPISOMI pins. **Note: For master-mode operation of the SPI, MASTER bit should be set to 1 and CLKMOD bit can be set either 1 or 0. The master-mode SPI can run on an external clock on SPICLK.** **For slave mode operation, both the MASTER and CLKMOD bits should be set to 0. Any other combinations may result in unpredictable behavior of the SPI. In slave mode. SPICLK will not be generated internally in slave mode.**
		0	SPISIMO pin input, SPISOMI pin output
		1	SPISOMI pin input, SPISIMO pin output

13.7.3 SPI Interrupt Register (SPIINT0)

Figure 13-19. SPI Interrupt Register (SPIINT0) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ENABLE-HIGHZ	Reserved							DMAREQEN
R-0							R/W-0	R-0							R/W-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						TXINTENA	RXINT-ENA	Reserved	RXOVRNIN-TENA	Reserved	BITER-RENA	DESYN-CENA	PAR-ERRENA	TIME-OUT-ENA	DLEN-ERRENA
R-0						R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

R = Read, W = write, P = Privilege mode; -*n* = Value after reset

Table 13-6. SPI Interrupt Register (SPIINT0) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	ENABLEHIGHZ		$\overline{\text{SPIENA}}$ pin high-impedance enable. When active, the $\overline{\text{SPIENA}}$ pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to high-impedance when not driving a low signal. If inactive, then the pin will output both a high and a low signal.
		0	$\overline{\text{SPIENA}}$ pin is pulled high when not active.
		1	$\overline{\text{SPIENA}}$ pin remains high-impedance when not active.
23–17	Reserved		Reads return zero and writes have no effect.
16	DMAREQEN		DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Enable DMA REQ only after setting the SPIEN bit to 1.
		0	DMA is not used.
		1	DMA requests will be generated.
			Note: A DMA request will be generated on the TX DMA REQ line each time a word is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1 writes.
			Note: A DMA request will be generated on the RX DMA REQ line each time a word is copied to the SPIBUF register either from RXBUF or directly from the shift register.
15–10	Reserved		Reads return zero and writes have no effect.

Table 13-6. SPI Interrupt Register (SPIINT0) Field Descriptions (Continued)

Bit	Name	Value	Description
9	TXINTENA		Causes an interrupt to be generated every time data is written to the shift register, so that the next word can be written to TXBUF. Setting this bit will generate an interrupt if the TXINTFLG bit (SPIFLG[9]) is set to 1.
		0	No interrupt will be generated upon TXINTFLG being set to 1.
		1	An interrupt will be generated upon TXINTFLG being set to 1.
			The transmitter empty interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
			Note: An interrupt request will be generated as soon as this bit is set to 1. By default it will be generated on the INT0 line. The SPILVL register can be programmed to change the interrupt line.
8	RXINTENA		Causes an interrupt to be generated when the RXINTFLAG bit (SPI-FLG[8]) is set by hardware.
		0	Interrupt will not be generated.
		1	Interrupt will be generated.
			The receiver full interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
7	Reserved		Reads return zero and writes have no effect.
6	RXOVRNINTENA		Overrun interrupt enable.
		0	Overrun interrupt will not be generated.
		1	Overrun interrupt will be generated.
5	Reserved		Reads return zero and writes have no effect.
4	BITERRENA		Enables interrupt on bit error.
		0	No interrupt asserted upon bit error.
		1	Enables an interrupt on a bit error.
3	DESYNCENA		Enables interrupt on desynchronized slave. DESYNCENA is used in master mode only.
		0	No interrupt asserted upon desynchronization error.
		1	An interrupt is asserted on desynchronization of the slave (DESYNC = 1).
2	PARERRENA		Enables interrupt-on-parity-error.
		0	No interrupt asserted on parity error.

Table 13-6. SPI Interrupt Register (SPIINT0) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	An interrupt is asserted on a parity error.
1	TIMEOUTENA		Enables interrupt on ENA signal time-out.
		0	No interrupt asserted upon ENA signal time-out.
		1	An interrupt is asserted on a time-out of the ENA signal.
0	DLENERRENA		Data length error interrupt enable. A data length error occurs under the following conditions. **Master:** When SPIENA is used, if the SPIENA pin from the slave is deasserted before the master has completed its transfer, the data length error is set. That is, if the character length counter has not overflowed while SPIENA deassertion is detected, then it means that the slave has neither received full data from the master nor has it transmitted complete data. **Slave:** When SPISCS pins are used, if the incoming valid SPISCS pin is deactivated before the character length counter overflows, then the data length error is set.
		0	No interrupt is generated upon data length error.
		1	An interrupt is asserted when a data-length error occurs.

13.7.4 SPI Interrupt Level Register (SPILVL)

Figure 13-20. SPI Interrupt Level Register (SPILVL) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						TXINTLVL	RX INTLVL	Reserved	RXOVRNIN-TLVL	Reserved	BITER-RLVL	DESYN-CLVL	PAR-ERRLVL	TIME-OUTLVL	DLEN-ERRLVL
R-0						R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

R = Read, W = Write in any mode; -*n* = Value after reset

Table 13-7. SPI Interrupt Level Register (SPILVL) Field Descriptions

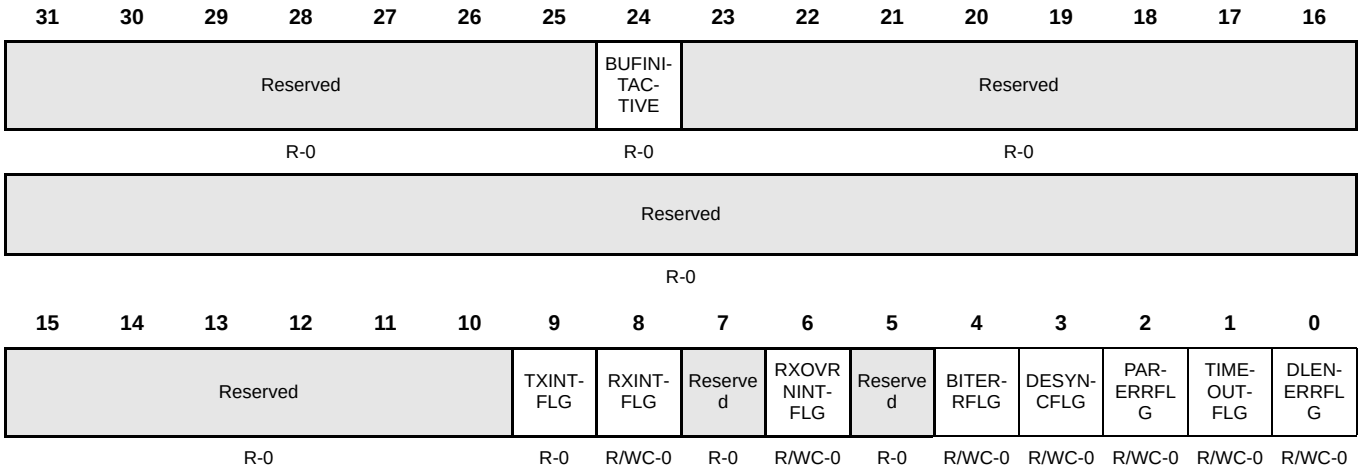
Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
9	TXINTLVL		Transmit interrupt level.
		0	Transmit interrupt is mapped to interrupt line INT0.
		1	Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL		Receive interrupt level.
		0	Receive interrupt is mapped to interrupt line INT0.
		1	Receive interrupt is mapped to interrupt line INT1.
7	Reserved		Reads return zero and writes have no effect.
6	RXOVRNINTLVL		Receive overrun interrupt level.
		0	Receive overrun interrupt is mapped to interrupt line INT0.
		1	Receive overrun interrupt is mapped to interrupt line INT1.
5	Reserved		Reads return zero and writes have no effect.
4	BITERRLVL		Bit error interrupt level.
		0	Bit error interrupt is mapped to interrupt line INT0.
		1	Bit error interrupt is mapped to interrupt line INT1.
3	DESYNCLVL		Desynchronized slave interrupt level. (master mode only).
		0	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT0.

Table 13-7. SPI Interrupt Level Register (SPILVL) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT1.
2	PARERRLVL		Parity error interrupt level.
		0	A parity error interrupt is mapped to interrupt line INT0.
		1	A parity error interrupt is mapped to interrupt line INT1.
1	TIMEOUTLVL		$\overline{\text{SPIENA}}$ pin time-out interrupt level.
		0	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT0.
		1	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT1.
0	DLEN ERR LVL		Data length error interrupt level (line) select.
		0	An interrupt on data length error is mapped to interrupt line INT0.
		1	An interrupt on data length error is mapped to interrupt line INT1.

13.7.5 SPI Flag Register (SPIFLG)

Figure 13-21. SPI Flag Register (SPIFLG) [offset = 10h]



R = Read; WC = Write/read Clear; -*n* = Value after reset

Table 13-8. SPI Flag Register (SPIFLG) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	BUFINITACTIVE		Indicates the status of multi-buffer initialization process. Software can poll for this bit to determine if it can proceed with the register configuration of multi-buffer mode registers or buffer handling. **Note: If the SPIFLG register is read while the multi-buffer RAM is being initialized, the BUF INIT ACTIVE bit will be read as 1. If SPIFLG is read after the internal automatic buffer initialization is complete, this bit will be read as 0. This bit will show a value of 1 as long as the nRESET bit is 0, but does not really indicate that buffer initialization is underway. Buffer initialization starts only when the nRESET bit is set to 1.**
		0	Multi-buffer RAM initialization is complete.
		1	Multi-buffer RAM is still being initialized. Do not attempt to write to either multi-buffer RAM or any multi-buffer mode registers.
23–10	Reserved		Reads return zero and writes have no effect.

Table 13-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
9	TXINTFLG		Transmitter-empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new word can be written to it. This flag is set when a word is copied to the shift register either directly from SPIDAT0/SPIDAT1 or from the TXBUF register. This bit is cleared by one of following methods: • Writing a new data to either SPIDAT0 or SPIDAT1 • Writing a 0 to SPIEN (SPIGCR1[24])
		0	Transmit buffer is now full. No interrupt pending for transmitter empty.
		1	Transmit buffer is empty. An interrupt is pending to fill the transmitter.
8	RXINTFLG		Receiver-full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). If RXINTEN is enabled, an interrupt is also generated. This bit is cleared under the following methods: • Reading the SPIBUF register • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive buffer full interrupt • Writing a 1 to this bit • Writing a 0 to SPIEN (SPIGCR1[24]) • System reset During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit.
		0	No new received data pending. Receive buffer is empty.
		1	A newly received data is ready to be read. Receive buffer is full. **Note: Clearing RXINTFLG bit by writing a 1 before reading the SPIBUF sets the RXEMPTY bit of the SPIBUF register too. In this way, one can ignore a received word. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF and the RXEMPTY bit will be cleared again. The SPIBUF contents should be read first if this situation needs to be avoided.**
7	Reserved		Reads return zero and writes have no effect.

Table 13-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
6	RXOVRNINTFLG		Receiver overrun flag. The SPI hardware sets this bit when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. The SPI will generate an interrupt request if this bit is set and the RXOVRN INTEN bit (SPIINT0.6) is set high. This bit is cleared under the following conditions in compatibility mode of MibSPI: • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive-buffer-overrun interrupt • Writing a 1 to RXOVRNINTFLG in the SPIFLG register itself • Writing a 0 to SPIEN • Reading the data field of the SPIBUF register **Note: Reading the SPIBUF register does not clear this RXOVRNINTFLG bit. If an RXOVRN interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF.** **Note: There is a special condition under which the RXOVRNIN-TFLG flag gets set. If both SPIBUF and RXBUF are already full and while another reception is underway, if any errors (e.g. TIMEOUT, BITERR and DLEN_ERR) occur, then RXOVR in RXBUF and RXOVRNINTFLG in SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a receive overrun.** In multi-buffer mode of MibSPI, this bit is cleared under the following conditions. • Reading the RXOVRN_BUF_ADDR register • Writing a 1 to RXOVRNINTFLG in the SPIFLG register itself In multi-buffer mode, if RXOVRNINTFLG is set, then the address of the buffer which experienced the overrun is available in RXOVRN_BUF_ADDR.
		0	Overrun condition did not occur.
		1	Overrun condition has occurred.
5	Reserved		Reads return zero and writes have no effect.

Table 13-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
4	BITERRFLG		Mismatch of internal transmit data and transmitted data. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set the SPIENA bit to 0.
		0	No bit error occurred.
		1	A bit error occurred. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag BITERRFLG is set. If BITERRENA is set an interrupt is asserted. Possible reasons for a bit error can be an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
3	DESYNCFLG		Desynchronization of slave device. Desynchronization monitor is active in master mode only.
		0	No slave desynchronization detected.
		1	A slave device is desynchronized. The master monitors the ENAble signal coming from the slave device and sets the DESYNC flag after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.
2	PARITYERRFLG		Calculated parity differs from received parity bit. If the parity generator is enabled (can be selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word the parity generator calculates the reference parity and compares it to the received parity bit. In the event of a mismatch the PARITYERR flag is set and an interrupt is asserted if PARERRENA is set.
		0	No parity error detected.
		1	A parity error occurred. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.

Table 13-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
1	TIMEOUTFLG		Time-out caused by nonactivation of ENA signal.
		0	No ENA-signal time-out occurred.
		1	An ENA signal time-out occurred. The SPI generates a time-out because the slave hasn't responded in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition the TIMEOUT flag in the status field of the corresponding buffer is set. The transmit request of the concerned buffer is cleared, i.e. the SPI doesn't re-start a data transfer from this buffer. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.
0	DLEN ERR FLG		Data-length error flag. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0. **Note: Whenever any transmission errors (TIMEOUT, BITERR, DLEN_ERR, PARITY_ERR, DESYNC) are detected and the error flags are cleared by writing to the error bit in the SPIFLG register, the corresponding error flag in SPIBUF does not get cleared. Software needs to read the SPIBUF until it becomes empty before proceeding. This ensures that all of the old status bits in SPIBUF are cleared before starting the next transfer.**
		0	No data length error has occurred.
		1	A data length error has occurred.

13.7.6 SPI Pin Control Register 0 (SPIPC0)

Figure 13-22. SPI Pin Control Register 0 (SPIPC0) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMI-FUN	SIMO-FUN	CLKFUN	ENAFUN	SCSFUN[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = write, -*n* = Value after reset

Table 13-9. SPI Pin Control (SPIPC0) Field Descriptions

Bit	Name	Value	Description
24	SOMIFUN		Slave out, master in function. Determines whether SPISOMI is to be used as a general-purpose I/O pin or as a SPI functional pin. **Note: Duplicate Control Bits for SPISOMI.** **Bit 24 is not physically implemented. It is a mirror of Bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPISOMI pin. The read value of Bit 24 always reflects the value of bit 11.**
		0	SPISOMI pin is a GIO pin.
		1	SPISOMI pin is a SPI functional pin.
16	SIMOFUN		Slave in, master out function. Determines whether SPISIMO is to be used as a general-purpose I/O pin or as a SPI functional pin. **Note: Duplicate Control Bits for SPISIMO.** **Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI pin. The read value of Bit 16 always reflects the value of bit 10.**
		0	The SPISIMO pin is a GIO pin.
		1	The SPISIMO pin is a SPI functional pin
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIFUN		Slave out, master in function. This bit determines whether the SPI-SOMI pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The SPISOMI pin is a GIO pin.
		1	The SPISOMI pin is a SPI functional pin.

Table 13-9. SPI Pin Control (SPIPC0) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMOFUN		Slave in, master out function. This bits determine whether each SPISIMO pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The SPISIMO pin is a GIO pin.
		1	The SPISIMO pin is a SPI functional pin.
9	CLKFUN		SPI clock function. This bit determines whether the SPICLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	The SPICLK pin is a GIO pin.
		1	The SPICLK pin is a SPI functional pin.
8	ENAFUN		$\overline{\text{SPIENA}}$ function. This bit determines whether the $\overline{\text{SPIENA}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The $\overline{\text{SPIENA}}$ pin is a GIO pin.
		1	The $\overline{\text{SPIENA}}$ pin is a SPI functional pin.
7-0	SCSFUN[7:0]		$\overline{\text{SPISCSx}}$ function. Determines whether each $\overline{\text{SPISCSx}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin. If the slave $\overline{\text{SPISCSx}}$ pins are in functional mode and receive an inactive high signal, the slave SPI will place its output in high-z and disable shifting.
		0	The $\overline{\text{SPISCSx}}$ pin is a GIO pin.
		1	The $\overline{\text{SPISCSx}}$ pin is a SPI functional pin.

13.7.7 SPI Pin Control Register 1 (SPIPC1)

Figure 13-23. SPI Pin Control Register 1 (SPIPC1) [offset = 18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIDIR	SIMO-DIR	CLKDIR	ENADIR	SCSDIR[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = Write; -*n* = Value after reset

Table 13-10. SPI Pin Control Register (SPIPC1) Field Descriptions

Bit	Name	Value	Description
		0	SPISOMI pin is an input.
		1	SPISOMI pin is an output.
16	SIMODIR		SPISIMO direction. Controls the direction of SPISIMO when used for general-purpose I/O. If SPISIMO pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register. **Note: Duplicate Control Bits for SPISIMO.** **Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI pin. The read value of Bit 16 always reflects the value of bit 10.**
		0	SPISIMO pin is an input.
		1	SPISIMO pin is an output.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDIR		SPISOMI direction. This bit controls the direction of the SPISOMI pin when it is used as a general-purpose I/O pin. If the SPISOMI pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISOMI pin is an input.
		1	SPISOMI pin is an output.
10	SIMODIR		SPISIMO direction. This bit controls the direction of the SPISIMO pin when it is used as a general-purpose I/O pin. If the SPISIMO pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISIMO pin is an input.
		1	SPISIMO pin is an output

Table 13-10. SPI Pin Control Register (SPIPC1) Field Descriptions (Continued)

Bit	Name	Value	Description
9	CLKDIR		SPICLK direction. This bit controls the direction of the SPICLK pin when it is used as a general-purpose I/O pin. In functional mode, the I/O direction is determined by the CLKMOD bit.
		0	SPICLK pin is an input.
		1	SPICLK pin is an output.
8	ENADIR		$\overline{\text{SPIENA}}$ direction. This bit controls the direction of the $\overline{\text{SPIENA}}$ pin when it is used as a general-purpose I/O. If the $\overline{\text{SPIENA}}$ pin is used as a functional pin, then the I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPIENA}}$ pin is an input.
		1	$\overline{\text{SPIENA}}$ pin is an output.
7–0	SCSDIR[7:0]		$\overline{\text{SPISCSx}}$ direction. These bits control the direction of each $\overline{\text{SPISCSx}}$ pin when it is used as a general-purpose I/O pin. Each pin could be configured independently from the others If the $\overline{\text{SPISCSx}}$ is used as a SPI functional pin; the I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPISCSx}}$ pin is an input.
		1	$\overline{\text{SPISCSx}}$ pin is an output.

13.7.8 SPI Pin Control Register 2 (SPIPC2)

Figure 13-24. SPI Pin Control Register 2 (SPIPC2) [offset = 1Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIDIN0	SIMODIN0	CLKDIN	ENADIN	SCSDIN[7:0]							
R-0				R-U	R-U	R-U	R-U	R-U							

R = Read; W = Write; U = Undefined; -*n* = Value after reset

Table 13-11. SPI Pin Control Register 2 (SPIPC2) Field Descriptions

Bit	Name	Value	Description
	SOMIDIN		SPISOMI data in. The value of the SPISOMI pin.
		0	The SPISOMI pin is logic 0.
		1	The SPISOMI pin is logic 1.
16	SIMODIN		SPISIMO data in. The value of the SPISIMO pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDIN		SPISOMI data in. The value of the SPISOMI pin.
		0	The SPISOMI pin is logic 0.
		1	The SPISOMI pin is logic 1.
10	SIMODIN		SPISIMO data in. The value of the SPISIMO pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.
9	CLKDIN		Clock data in. The value of the SPICLK pin.
		0	The SPICLK pin is logic 0.
		1	The SPICLK pin is logic 1
8	ENADIN		$\overline{\text{SPIENA}}$ data in. The the value of the $\overline{\text{SPIENA}}$ pin.
		0	The $\overline{\text{SPIENA}}$ pin is logic 0.
		1	The $\overline{\text{SPIENA}}$ pin is logic 1

Table 13-11. SPI Pin Control Register 2 (SPIPC2) Field Descriptions (Continued)

Bit	Name	Value	Description
7–0	SCSDIN[7:0]		$\overline{\text{SPISCSx}}$ data in. The value of the $\overline{\text{SPISCSx}}$ pins.
		0	The $\overline{\text{SPISCSx}}$ pin is logic 0.
		1	The $\overline{\text{SPISCSx}}$ pin is logic 1

13.7.9 SPI Pin Control Register 3 (SPIPC3)

Figure 13-25. SPI Pin Control Register 3 (SPIPC3) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIDOUT0	SIMODOUT0	CLKD-OUT	ENADOUT	SCSDOUT[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-U	R/W-U							

R = Read; W = Write; -*n* = Value after reset

Table 13-12. SPI Pin Control Register 3 (SPIPC3) Field Descriptions

Bit	Name	Value	Description
24	SOMIDOUT		SPISOMI data out write. This bit is only active when the SPISOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 11 or bit 24 can be used to set the direction for pin SPISOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Current value on SPISOMI pin is logic 0.
		1	Current value on SPISOMI pin is logic 1
16	SIMODOUT		SPISIMO data out write. This bit is only active when the SPISIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 10 or bit 16 can be used to set the direction for pin SPISOMI. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Current value on SPISIMO pin is logic 0.
		1	Current value on SPISIMO pin is logic 1.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDOUT		SPISOMI0 data out write. This bit is only active when the SPISOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on SPISOMI pin is logic 0.
		1	Current value on SPISOMI pin is logic 1.
10	SIMODOUT		SPISIMO data out write. This bit is only active when the SPISIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.

Table 13-12. SPI Pin Control Register 3 (SPIPC3) Field Descriptions (Continued)

Bit	Name	Value	Description
9	CLKDOUT		SPICLK data out write. This bit is only active when the SPICLK pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The SPICLK pin is logic 0.
		1	The SPICLK pin is logic 1.
8	ENADOUT		$\overline{\text{SPIENA}}$ data out write. Only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The $\overline{\text{SPIENA}}$ pin is logic 0.
		1	The $\overline{\text{SPIENA}}$ pin is logic 1.
7–0	SCSDOUT[7:0]		$\overline{\text{SPISCSx}}$ data out write. Only active when the $\overline{\text{SPISCSx}}$ pins are configured as a general-purpose I/O pins and configured as an output pins. The value of these bits indicates the value sent to the pins.
		0	The $\overline{\text{SPISCSx}}$ pin is logic 0.
		1	The $\overline{\text{SPISCSx}}$ pin is logic 1.

13.7.10 SPI Pin Control Register 4 (SPIPC4)

Figure 13-26. SPI Pin Control Register 4 (SPIPC4) [offset = 24h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMISET0	SIMOSET0	CLKSET	ENASET	SCSSET[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-U	R/W-U							

R = Read; W = Write; -*n* = Value after reset

Table 13-13. SPI Pin Control Register 4 (SPIPC4) Field Descriptions

Bit	Name	Value	Description
24	SOMISET		SPISOMI data out set. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to set the SOMI pin. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	*Read:* SPISIMO is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISOMI is logic 1. *Write:* Logic 1 is placed on SPISOMI pin if it is in general-purpose output mode.
16	SIMOSET		SPISIMO data out set. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to set the SOMI pin. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	*Read:* SPISIMI is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISIMO is logic 1. *Write:* Logic 1 is placed on SPISIMO pin if it is in general-purpose output mode.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMISET		SPISOMI data out set. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin.
		0	*Read:* SPISOMI is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISOMI is logic 1. *Write:* Logic 1 is placed on SPISOMI pin if it is in general-purpose output mode.

Table 13-13. SPI Pin Control Register 4 (SPIPC4) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMOSET		SPISIMO0 data out set. This pin is only active when the SPISIMO pin is configured as a general-purpose output pin.
		0	*Read:* SPISIMO is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISIMO is logic 1. *Write:* Logic 1 is placed on SPISIMO pin if it is in general-purpose output mode.
9	CLKSET		SPICLK data out set. This bit is only active when the SPICLK pin is configured as a general-purpose output pin.
		0	*Read:* SPICLK is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPICLK pin is logic 1. *Write:* Logic 1 is placed on the SPICLK pin if it is in general-purpose output mode.
8	ENASET		$\overline{\text{SPIENA}}$ data out set. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin.
		0	*Read:* SPIENA is logic 0. *Write:* A write to this bit has no effect.
		1	*Read:* SPIENA is logic 1. *Write:* Logic 1 is placed on $\overline{\text{SPIENA}}$ pin if it is in general-purpose O/P mode.
7–0	SCSSET[7:0]		$\overline{\text{SPISCSx}}$ data out set. This bit is only active when the $\overline{\text{SPISCSx}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit sets the corresponding $\overline{\text{SCS}}$DOUT bit to 1.
		0	*Read:* $\overline{\text{SPISCSx}}$ is logic 0. *Write:* A write to this bit has no effect.
		1	*Read:* $\overline{\text{SPISCSx}}$ is logic 1. *Write:* Logic 1 placed on $\overline{\text{SPISCSx}}$ pin if it is in general-purpose output mode.

13.7.11 SPI Pin Control Register 5 (SPIPC5)

Figure 13-27. SPI Pin Control Register 5 (SPIPC5) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMICLR0	SIMOCLR0	CLKCLR	ENACLR	SCSCLR[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-0	R/W-0							

R = Read; W = Write; -*n* = Value after reset

Table 13-14. SPI Pin Control Register 5 (SPIPC5) Field Descriptions

Bit	Name	Value	Description
24	SOMICLR		SPISOMIx data out clear. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to clear the pin SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	*Read:* The current value on SOMIDOUT is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SOMIDOUT is 1. *Write:* Logic 0 is placed on SPISOMI pin if it is in general-purpose output mode.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOCLR		SPISIMO data out clear. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to clear the pin SOMI. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	*Read:* The current value on SIMODOUT is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SIMODOUT is 1. *Write:* Logic 0 is placed on SPISIMO pin if it is in general-purpose output mode.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMICLR		SPISOMI data out clear. This bit is only active when the SPISOMI pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPISOMI is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPISOMI is 1. *Write:* Logic 0 is placed on SPISOMI pin if it is in general-purpose output mode.

Table 13-14. SPI Pin Control Register 5 (SPIPC5) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMOCLR		SPISIMO data out clear. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPISIMO is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPISIMO is 1. *Write:* Logic 0 is placed on SPISIMO pin if it is in general-purpose output mode.
9	CLKCLR		SPICLK data out clear. This bit is only active when the SPICLK pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPICLK is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPICLK is 1. *Write:* Logic 0 is placed on SPICLK pin if it is in general-purpose output mode.
8	ENACLR		$\overline{\text{SPIENA}}$ data out clear. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit clears the corresponding ENABLEDOUT bit to 0.
		0	*Read:* The current value on ENA is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on ENA is 1. *Write:* Logic 0 is placed on $\overline{\text{SPIENA}}$ pin if it is in general-purpose output mode.
7–0	SCSCLR[7:0]		$\overline{\text{SPISCSx}}$ data out clear. This bit is only active when the $\overline{\text{SPISCSx}}$ pins are configured as a general-purpose output pins.
		0	*Read:* The current value on SCSDOUTx is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SCSDOUTx is 1. *Write:* Logic 0 is placed on the $\overline{\text{SPISCS}}$ pin if it is in general-purpose output mode.

13.7.12 SPI Pin Control Register 6 (SPIPC6)

Figure 13-28. SPI Pin Control Register 6 (SPIPC6) [offset = 2Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIPDR0	SIMOPDR0	CLKPDR	ENACLR	SCSPDR[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read; W = Write; -*n* = Value after reset

Table 13-15. SPI Pin Control Register 6 (SPIPC6) Field Descriptions

Bit	Name	Value	Description
24	SOMIPDR		SPISOMI open drain enable. This bit enables open drain capability for the SPISOMI pin if the following conditions are met: • SOMIDIR = 1 (SPISOMI pin configured in GIO mode as an output) • SOMIDOUT = 1 **Bit 11 or bit 24 can both be used to enable open-drain for SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	The output value on the SPISOMIx pin is logic 1.
		1	Output pin SPISOMIx is in a high-impedance state.
16	SIMOPDR		SPISIMO open drain enable. This bit enables open drain capability for the SPISIMO pin if the following conditions are met: • SIMODIR = 1 (SPISIMO pin configured in GIO mode as an output) • SIMODOUT = 1 **Bit 10 or bit 16 can both be used to enable open-drain for SIMO. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	The output value on SPISIMOx pin is logic 1.
		1	Output pin SPISIMOx is in a high-impedance state.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIPDR		SOMI open-drain enable. This bit enables open-drain capability for SOMI if the following conditions are met. • SOMI pin configured in GIO mode as output pin • Output value on SPISOMI pin is logic 1.
		0	Output value 1 of SPISOMI pin is logic 1.
		1	Output value 1 of SPISOMI is high-impedance.

Table 13-15. SPI Pin Control Register 6 (SPIPC6) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMOPDR		SPISIMO open-drain enable. This bit enables open -drain capability for the SPISIMO pin if the following conditions are met. • SIMO pin configured in GIO mode as output pin • Output value on SPISIMO pin is logic 1.
		0	Output value 1 of SPISIMO pin is logic 1.
		1	Output value 1 of SPISIMO is high-impedance.
9	CLKPDR		CLK open drain enable. This bit enables open drain capability for the pin CLK if the following conditions are met: • SPICLK pin configured in GIO mode as an output pin • SPICLKDOUT = 1
		0	Output value on CLK pin is logic 1.
		1	Output pin CLK is in a high-impedance state.
8	ENAPDR		SPIENA pin open drain enable. This bit enables open drain capability for SPIENA if the following conditions are met: • SPIENA pin configured in GIO mode as an output pin • SPIENADOUT = 1
		0	Output value on SPIENA pin is logic 1.
		1	Output pin SPIENA is in a high-impedance state.
7–0	SCSPDR[7:0]		SPISCSx open drain enable. This bit enables open drain capability for the SPISCSx pin if the following conditions are met: • SPISCS pin configured in GIO mode as an output pin • SPISCSDOUTx = 1
		0	Output value on SCSx pin is logic 1.
		1	Output pin SCSx is in a high-impedance state.

13.7.13 SPI Pin Control Register 7(SPIPC7)

Note: Default Register Value
The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 13-29. SPI Pin Control Register 7 (SPIPC7) [offset = 30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIP DIS0	SIMOP DIS0	CLKDIS	ENAPDIS	SCSPDIS[7:0]							
R-0				R/W-n	R/W-n	R/W-n	R/W-n	R/W-n							

R = Read; W = Write; -*n* = Value after reset

Table 13-16. SPI Pin Control Register 7 (SPIPC7) Field Descriptions

Bit	Name	Value	Description
24	SOMIDIS		SOMI pull control enable/disable. This bit enables pull control capability for the SOMI pin if it is in input mode regardless of whether it is in functional or GIO mode. **Note: Bit 11 or bit 24 can be used to set pull-disable for SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Pull control on the SPISOMI pin is enabled.
		1	Pull control on the SPISOMI pin is disabled.
16	SIMODIS		SIMO pull control enable/disable. This bit enables pull control capability for the SIMO pin if it is in input mode regardless of whether it is in functional or GIO mode. **Note: Bit 10 or bit 16 can be used to set pull-disable for SIMO. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Pull control on SPISIMO pin is enabled.
		1	Pull control on SPISIMO pin is disabled.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIPDIS		SPISOMI pull control enable/disable. This bit enables pull control capability for the pin SPISOMI pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on the SPISOMI pin is enabled.
		1	Pull control on the SPISOMI pin is disabled.

Table 13-16. SPI Pin Control Register 7 (SPIPC7) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMOPDIS		SPISIMO pull control enable/disable. This bit enables pull control capability for the pin SPISIMO pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on SPISIMO pin is enabled.
		1	Pull control on SPISIMO pin is disabled.
9	CLKPDIS		CLK pull control enable/disable. This bit enables pull control capability for the pin SPICLK pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on CLK pin is enabled.
		1	Pull control on CLK pin is disabled.
8	ENAPDIS		ENABLE pull control enable/disable. This bit enables pull control capability for the pin SPIENA pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on ENABLE pin is enabled.
		1	Pull control on ENABLE pin is disabled.
7–0	SCSPDIS[7:0]		SCSx pull control enable/disable. This bit enables pull control capability for the pin SPISCSx pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on SCSx pin is enabled.
		1	Pull control on SCSx pin is disabled.

13.7.14 SPI Pin Control Register 8(SPIPC8)

Note: Default Register Value
The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 13-30. SPI Pin Control Register 8 (SPIPC8) [offset = 34h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMI-PSEL[0]	Reserved							SIMO-PSEL[0]
R-0							R/W-n	R-0							R/W-n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMI-PSEL	SIMO-PSEL	CLKPSEL	ENAPSEL	SCSPSEL[7:0]							
R-0				R/W-n	R/W-n	R/W-n	R/W-n	R/W-n							

R = Read; W = Write; -*n* = Value after reset

Table 13-17. SPI Pin Control Register 8 (SPIPC8) Field Descriptions

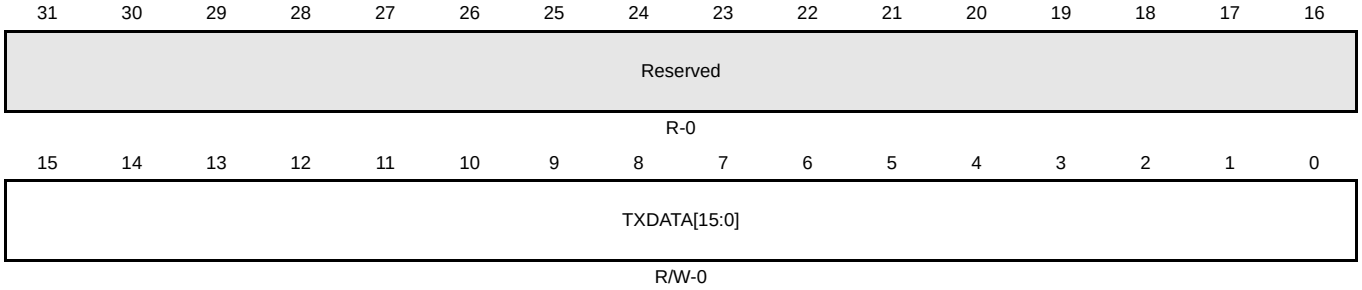
Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIPSEL		SPISOMI pull select.
		0	Input buffer for SPISOMI is disabled if PULLDIS = 1
		1	Input buffer for SPISOMI is enabled if PULLDIS = 1
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOPSEL		SPISIMO pull select.
		0	Input buffer for SPISIMO is disabled if PULLDIS = 1
		1	Input buffer for SPISIMO is enabled if PULLDIS = 1
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIPSEL		SOMI pull select.
		0	Input buffer for SOMI is disabled if PULLDIS = 1
		1	Input buffer for SOMI is enabled if PULLDIS = 1
10	SIMOPSEL		SPISIMO pull select.
		0	Input buffer for SPISIMO is disabled if PULLDIS = 1
		1	Input buffer for SPISIMO is enabled if PULLDIS = 1

Table 13-17. SPI Pin Control Register 8 (SPIPC8) Field Descriptions

Bit	Name	Value	Description
9	CLKPSEL		CLK pull select.
		0	Input buffer for CLK is disabled if PULLDIS = 1
		1	Input buffer for CLK is enabled if PULLDIS = 1
8	ENAPSEL		ENABLE pull select.
		0	Input buffer for ENABLE is disabled if PULLDIS = 1
		1	Input buffer for ENABLE is enabled if PULLDIS = 1
7–0	SCSPSEL[7:0]		SCSx pull select.
		0	Input buffer for SCSx is disabled if PULLDIS = 1
		1	Input buffer for SCSx is enabled if PULLDIS = 1

13.7.15 SPI Transmit Data Register 0 (SPIDAT0)

Figure 13-31. SPI Transmit Data Register 0 (SPIDAT0) [offset = 38h]



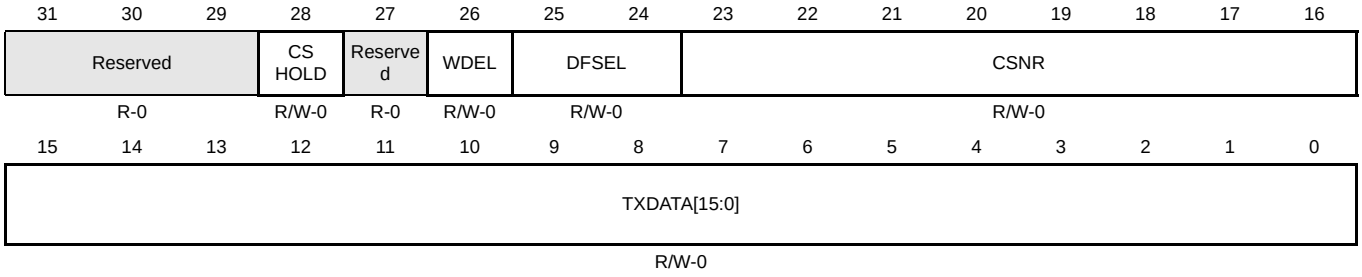
R = Read, W = write, -*n* = Value after reset

Table 13-18. SPI Transmit Data Register 0 (SPIDAT0) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–0	TXDATA(15–0)	0–FFFFh	SPI transmit data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, TXBUF holds the written data. SPIEN (SPICGR1[24]) must be set to 1 before this register can be written to. Writing a 0 to the SPIEN register forces the lower 16 bits of the SPIDAT0 register to 0x00. **Note: When this register is read, the contents TXBUF, which holds the latest written data, will be returned.** **Note: Regardless of character length, the transmit word should be right-justified before writing to the SPIDAT0 register.** **Note: The default data format control register for SPIDAT0 is SPIFMT0. However, it is possible to reprogram the DFSEL[1:0] fields of SPIDAT1 before using SPIDAT0, to select a different SPIFMTx register.**

13.7.16 SPI Transmit Data Register 1 (SPIDAT1)

Figure 13-32. SPI Transmit Data Register 1 (SPIDAT1) [offset = 3Ch]



R = Read, W = write, -*n* = Value after reset

Table 13-19. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions

Bit	Name	Value	Description
31–29	Reserved		Reads return zero and writes have no effect.
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only in compatibility-mode of SPI, (it is ignored in slave mode). CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.
27	Reserved		Reads return zero and writes have no effect.

Table 13-19. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions (Continued)

Bit	Name	Value	Description
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, $\overline{\text{SPISCS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the SPISCS pin remains deactivated depends upon the time taken to supply a new word after completing the shift operation. If TXBUF is already full, then the SPISCS will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPISCS}}$ pins will be de-activated for at least (WDELAY + 2) * VCLK_Period duration.
25–24	DFSEL		Data word format select
		00	Data word format 0 is selected
		01	Data word format 1 is selected
		10	Data word format 2 is selected
		11	Data word format 3 is selected
23–16	CSNR	0–FFh	Chip select number. CSNR defines the chip-select that will be activated during the data transfer. **Note: Writing to only the control field does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL fields in the control field to select the required phase/polarity combination.**
15–0	TXDATA(15–0)	0–FFFFh	**Transfer data.** When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of SPIDAT1 to 0x0000. Write to this register ONLY when using the automatic slave chip-select feature (see Section 13.2, *Operating Modes* on page 510 for more information). A write to this register will drive the contents of CSNR[7:0] on the $\overline{\text{SPISCS}}$[7:0] pins, if they are configured as functional pins. When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

13.7.17 SPI Receive Buffer Register (SPIBUF)

Figure 13-33. SPI Receive Buffer Register (SPIBUF) [offset = 40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLEN-ERR	LCSNR							
R-1	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA[15:0)]															
R-0															

R = Read, W = write, C = Clear; S = Set; -*n* = Value after reset

Table 13-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions

Bit	Name	Value	Description
31	RXEMPTY		**Receive data buffer empty.** When the host reads the SPIBUF field or the entire SPIBUF register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, and the RXEMPTY flag is cleared.
		0	New data has been received and copied into the SPIBUF field.
		1	No data has been received since the last read of SPIBUF. This flag gets set to 1 under the following conditions: • Reading the RXDATA portion of the SPIBUF register. • Writing a 1 to clear the RXINTFLG bit in the SPIFLG register. Write-clearing the RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA portion of SPIBUF (or the entire register).

Table 13-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

Bit	Name	Value	Description
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to SPIBUF; the contents of SPIBUF are overwritten only after it is read by the VBUSP master (e.g. CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPIFLG or SPIVECTx shows the RXOVRN condition. Two read operations from the SPIBUF register are required to reach the overwritten buffer word (one to read SPIBUF, which then transfers RXDATA into SPIBUF for the second read). This flag is cleared to 0 when the RXDATA is read. **Note: A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BIT-ERR and DLEN_ERR occur, then RXOVR in RXBUF and SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		**Transmit data buffer full.** This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		**Bit error.** There was a mismatch of internal transmit data and transmitted data.
		0	No bit error occurred. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**

Table 13-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

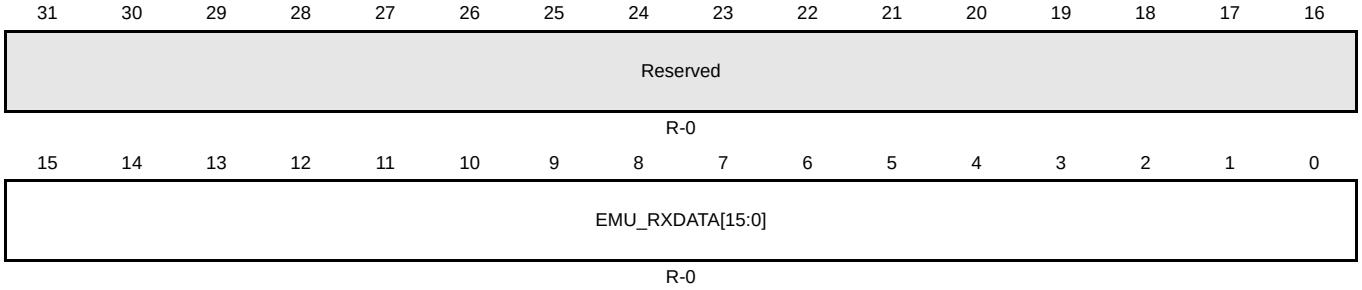
Bit	Name	Value	Description
		1	A bit error occurred. The SPI samples the signal of the transmit pins (master: SIMOx, slave: SOMIx) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		**Desynchronization of slave device.** This bit is valid in master mode only. The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. **Note: In the Compatibility Mode MibSPI, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.**
		0	No slave desynchronization detected. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		1	A slave device is desynchronized.
26	PARITYERR		**Parity error.** The calculated parity differs from the received parity bit. If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No parity error detected.
		1	A parity error occurred.

Table 13-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

Bit	Name	Value	Description
25	TIMEOUT		**Time-out because of non-activation of ENA pin.** The SPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPIFLG register is set. **This bit is valid only in master mode.** **This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.**
		0	No ENA-pin time-out occurred.
		1	An ENA signal time-out occurred.
24	DLENERR		**Data length error flag.** **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No data-length error has occurred.
		1	A data length error has occurred.
23–16	LCSNR	0–FFh	**Last chip select number.** LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer.
15–0	RXDATA[15:0]	0–FFFFh	**SPI receive data.** This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

13.7.18 SPI Emulation Register (SPIEMU)

Figure 13-34. SPI Emulation Register (SPIEMU) [offset = 44h]



R = Read, -*n* = Value after reset

Table 13-21. SPI Emulation Register (SPIEMU) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–0	EMU_RXDATA [15:0]	0–FFFFh	**SPI receive data.** The SPI emulation register is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF is that a read from SPIEMU does not clear any of the status flags.

13.7.19 SPI Delay Register (SPIDELAY)

Figure 13-35. SPI Delay Register (SPIDELAY) [offset = 48h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C2TDELAY[7:0]								T2CDELAY[7:0]							
R/W-0								R/W-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2EDELAY[7:0]								C2EDELAY[7:0]							
R/W-0								R/W-0							

R = Read, -*n* = Value after reset

Table 13-22. SPI Delay Register (SPIDELAY) Field Descriptions

Bit	Name	Value	Description
31–24	C2TDELAY[7:0]	0-FF	**Chip-select-active to transmit-start delay.** See Figure 13-36 for an example. C2TDELAY is used only in master mode. It defines a setup time (for the slave device) that delays the data transmission from the chip select active edge by a multiple of VCLK cycles. The setup time value is calculated as follows. $t_{C2TDELAY}$ = (C2TDELAY + 2) ***** VCLK Period **Note: If C2TDELAY = 0, then $t_{C2TDELAY}$ = 0.** Example: VCLK = 25 MHz -> VCLK Period = 40ns; C2TDELAY = 07h; > $t_{C2TDELAY}$ = 360 ns When the chip select signal becomes active, the slave has to prepare data transfer within 360 ns. **Note: If phase = 1, the delay between SPICS falling edge to the first edge of SPICLK will have an additional 0.5 SPICLK period delay. This delay is as per the SPI protocol.**

Table 13-22. SPI Delay Register (SPIDELAY) Field Descriptions (Continued)

Bit	Name	Value	Description
23–16	T2CDELAY[7:0]	0–FF	**Transmit-end-to-chip-select-inactive-delay.** See Figure 13-37 for an example. T2CDELAY is used only in master mode. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of VCLK cycles after the last bit is transferred. The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY +1)* VCLK Period **Note: If T2CDELAY = 0, then $t_{T2CDELAY}$ = 0** Example: VCLK = 25 MHz -> VCLK Period = 40ns; T2CDELAY = 03h; > $t_{T2CDELAY}$ = 160 ns; After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. **Note: If phase = 0, then between the last edge of SPICLK and rise-edge of SPICS there will be an additional delay of 0.5 SPICLK period. This is as per the SPI protocol.** Both C2TDELAY and T2CDELAY counters do not have any dependency on the SPIENA pin value. Even if the SPIENA pin is asserted by the slave, the master will continue to delay the start of SPICLK until the C2TDELAY counter overflows. Similarly, even if the SPIENA pin is deasserted by the slave, the master will continue to hold the SPISCS pins active until the T2CDELAY counter overflows. In this way, it is guaranteed that the setup and hold times of the SPISCS pins are determined by the delay timers alone. To achieve better throughput, it should be ensured that these two timers are kept at the minimum possible values.

Table 13-22. SPI Delay Register (SPIDELAY) Field Descriptions (Continued)

Bit	Name	Value	Description
15–8	T2EDELAY[7:0]	0–FFh	**Transmit-data-finished to ENA-pin-inactive time-out.** T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before SPIENA signal has to become inactive and after SPISCS becomes inactive. SPICLK depends on which data format is selected. If the slave device is missing one or more clock edges, it becomes de-synchronized. In this case, although the master has finished the data transfer, the slave is still waiting for the missed clock pulses and the ENA signal isn't disabled. The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the SPIENA signal isn't deactivated in time. The DESYNC flag is set to indicate that the slave device did not de-assert its SPIENA pin in time to acknowledge that it received all bits of the sent word. See Figure 13-38 for an example of this condition. **Note: DESYNC is also set if the SPI detects a de-assertion of SPIENA before the end of the transmission.** The time-out value is calculated as follows: $t_{T2EDELAY}$ = T2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; T2EDELAY = 10h; > $t_{T2EDELAY}$=2 μs; The slave device has to disable the ENA signal within 2 μs, otherwise DESYNC is set and an interrupt is asserted (if enabled).
7–0	C2EDELAY[7:0]	0–FFh	**Chip-select-active to ENA-signal-active time-out.** C2EDELAY is used only in master mode and it applies only if the addressed slave generates an ENA signal as a hardware handshake response. C2EDELAY defines the maximum time between when the SPI activates the chip-select signal and the addressed slave has to respond by activating the ENA signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. The SPI clock depends on whether data format 0 or data format 1 is selected. See Figure 13-39 for an example of this condition. **Note: If the slave device does not respond with the ENA signal before the time-out value is reached, the TIMEOUT flag in the SPIFLG register is set and a interrupt is asserted (if enabled).** If a time-out occurs, the SPI clears the transmit request of the timed-out buffer, sets the TIMEOUT flag for the current buffer, and continues with the transfer of the next buffer in the sequence that is enabled. The timeout value is calculated as follows: $t_{C2EDELAY}$ = C2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; C2EDELAY = 30 h; > $t_{C2EDELAY}$=6 ms; The slave device has to activate the ENA signal within 6 ms after the SPI has activated the chip select signal (SPISCS), otherwise the TIMEOUT flag is set and an interrupt is asserted (if enabled).

Figure 13-36. Example: $t_{C2TDELAY}$ = 8 VCLK Cycles

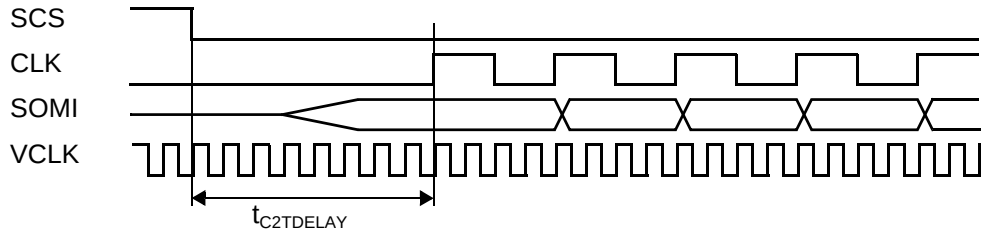


Figure 13-37. Example: $t_{T2CDELAY}$ = 4 VCLK Cycles

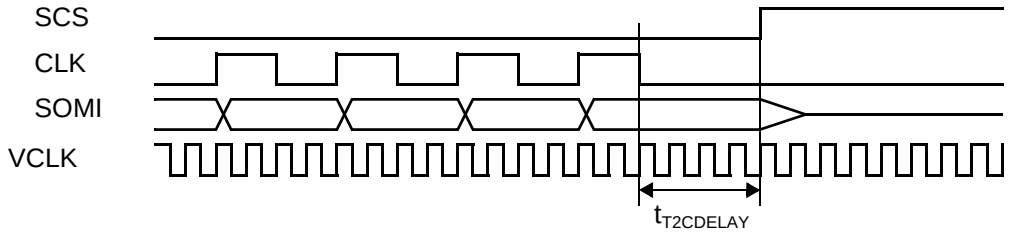


Figure 13-38. Transmit-Data-Finished-to-ENA-Inactive-Timeout

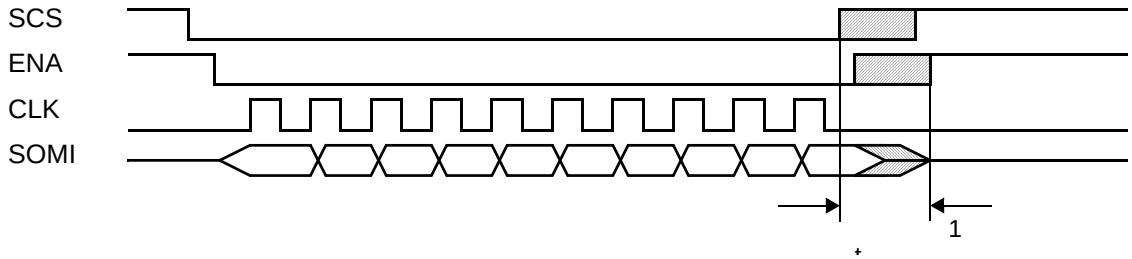
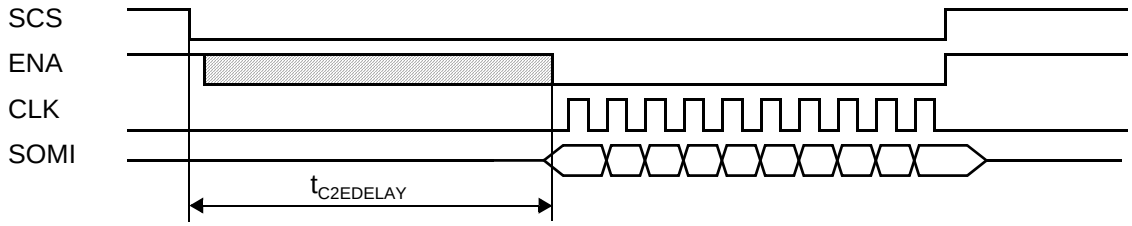
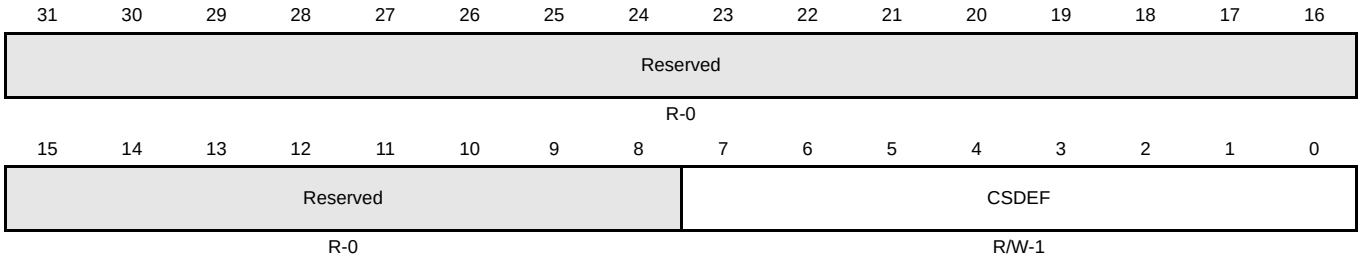


Figure 13-39. Chip-Select-Active-to-ENA-Signal-Active-Timeout



13.7.20 SPI Default Chip Select Register (SPIDEF)

Figure 13-40. SPI Default Chip Select Register (SPIDEF) [offset = 4Ch]



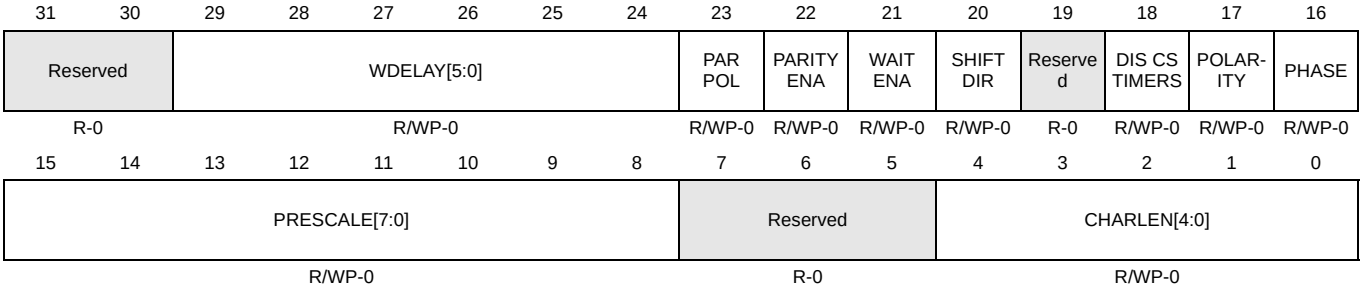
R = Read, W = Write; -*n* = Value after reset

Table 13-23. SPI Default Chip Select Register (SPIDEF) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	CSDEF	0-FFh	**Chip select default pattern.** Master-mode only. The CSDEFx bits are output to the SPISCS pins when no transmission is being performed. It allows the user to set a programmable chip-select pattern that deselects all of the SPI slaves.
		0	SPISCSx is set to 0 when no transfer is active.
		1	SPISCSx is set to 1 when no transfer is active.

13.7.21 SPI Data Format Registers (SPIFMT[3:0])

Figure 13-41. SPI Data Format Registers (SPIFMT[3:0]) [offset = 5Ch–50h]



R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 13-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions

Bit	Name	Value	Description
31–30	Reserved		Reads return zero and writes have no effect.
29–24	WDELAY[5:0]	0–3F	Delay in between transmissions for data format x (x= 0,1,2,3). Idle time that will be applied at the end of the current transmission if the bit WDEL is set in the current buffer. The delay to be applied is equal to: WDELAY * P_{VCLK} + 2 * P_{VCLK} P_{VCLK} -> Period of VCLK.
23	PARPOL		Parity polarity: even or odd. PARPOLx can be modified in privilege mode only. It can be used for data format x (x= 0,1,2,3).
		0	An even parity flag is added at the end of the transmit data stream.
		1	An odd parity flag is added at the end of the transmit data stream.
22	PARITYENA		Parity enable for data format x.
		0	No parity generation/ verification is performed for this data format.
		1	A parity bit is transmitted at the end of each transmitted word. At the end of a transfer the parity generator compares the received parity bit with the locally-calculated parity flag. If the parity bits do not match the RXERR flag is set in the corresponding control field. The parity type (even or odd) can be selected via the PARPOL bit.
21	WAITENA		The master waits for the ENA signal from slave for data format x. WAITENA is valid in master mode only. WAITENA enables a flexible SPI network where slaves with ENA signal and slaves without ENA signal can be mixed. WAITENA defines, for each transferred word, whether the addressed slave generates the ENA signal or not.
		0	The SPI does not wait for the ENA signal from the slave and directly starts the transfer.

Table 13-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Before the SPI starts the data transfer it waits for the ENA signal to become low. If the ENA signal is not pulled down by the addressed slave before the internal time-out counter (C2EDELAY) overflows, then the master aborts the transfer and sets the TIMEOUT error flag.
20	SHIFTDIR		**Shift direction for data format x.** With bit SHIFTDIRx, the shift direction for data format x (x=0,1,2,3) can be selected.
		0	MSB is shifted out first.
		1	LSB is shifted out first.
19	Reserved		Reads return zero and writes have no effect.
18	DIS CS TIMERS		**Disable chip-select timers for this format.** The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format, if they are not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip-select delay timers for any slaves.
		0	Both C2TDELAY and T2CDELAY counts are inserted for the chip selects.
		1	No C2TDELAY or T2CDELAY is inserted in the chip select timings.
17	POLARITY		**SPI data format x clock polarity.** POLARITYx defines the clock polarity of data format x. The following restrictions apply when switching clock phase and/or polarity: 1. In 3-pin/4-pin with nENA pin configuration of a slave SPI, the clock phase and polarity cannot be changed on-the-fly between two transfers. The slave should be reset and reconfigured if clock phase/polarity needs to be switched. In summary, SPI format switching is not fully supported in slave mode. 2. Even while using chip select pins, the polarity of SPICLK can be switched only while the slave is not selected by a valid chip select. The master SPI should ensure that while switching SPICLK polarity, it has deselected all of its slaves. Otherwise, the switching of SPICLK polarity may be incorrectly treated as a clock edge by some slaves.
		0	If POLARITYx is set to 0 the SPI clock signal is low-inactive, i.e., before and after data transfer the clock signal is low.
		1	If POLARITYx is set to 1 the SPI clock signal is high-inactive, i.e., before and after data transfer the clock signal is high.

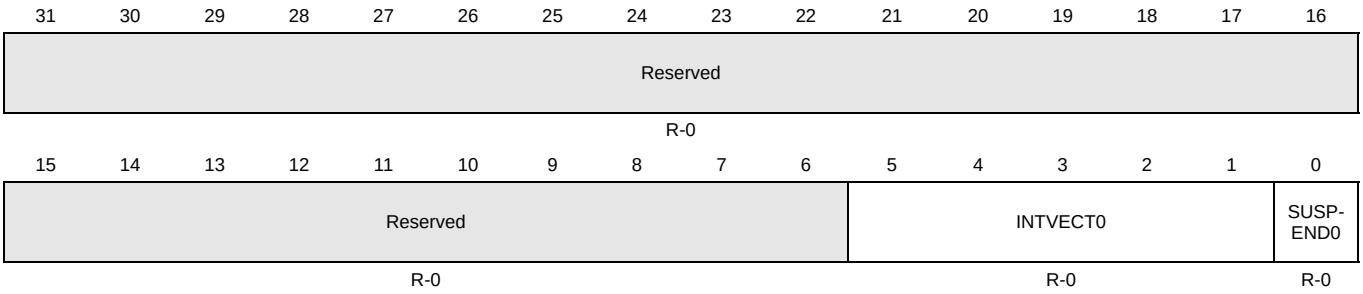
Table 13-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions (Continued)

Bit	Name	Value	Description
16	PHASE		**SPI data format x clock delay.** PHASEx defines the clock delay of data format x.
		0	If PHASEx is set to 0 the SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge.
		1	If PHASEx is set to 1 the SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15–8	PRESCALE[7:0]		SPI data format x prescaler. PRESCALEx determines the bit transfer rate of data format x if the SPI is the network master. PRESCALEx is use to derive SPICLK from VCLK. If the SPI is configured as slave, PRESCALEx **does not need** to be configured. The clock rate for data format x can be calculated as: $BR_{Formatx} = \frac{VBUSPCLCK}{(PRESCALEx + 1)}$ **Note: When PRESCALEx is set to 0, the SPI clock rate defaults to VCLK/2.**
7–5	Reserved		Reads return zero and writes have no effect.
4–0	CHARLEN[4:0]	0–1F	SPI data format x data-word length. CHARLENx defines the word length of data format x. Legal values are 0x02 (data word length = 2 bit) to 0x10 (data word length = 16). Illegal values, such as 0x00 or 0x1F are not allowed; their effect is indeterminate.

13.7.22 Interrupt Vector 0 (INTVECT0)

Note: The TG interrupt is not available in MibSPI in compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 13-42. *Interrupt Vector 0* (*INTVECT0*) [offset = 60h]



R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 13-25. *Transfer Group Interrupt Vector 0* (*INTVECT0*)]

Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–1	INTVECT0		**INTVECT0. Interrupt vector for interrupt line INT0.** Returns the vector of the pending interrupt at interrupt line INT0. If more than one interrupt is pending, INTVECT0 always references the highest prior interrupt source first. **Note:** This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.
		00000	There is no pending interrupt.
		00001 + x	Transfer group x (x=0,..,15) has a pending interrupt. SUSPEND0 reflects the type of interrupt (*suspended* or *finished*).
		10001	Error Interrupt pending. The lower half of SPIINT0 contains more details about the type of error.
		10011	The pending interrupt is a “Receive Buffer Overrun” interrupt.
		10010	**SPI mode:** The pending interrupt is a “Receive Buffer Full” interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		10100	**SPI mode:** The pending interrupt is a “Transmit Buffer Empty” interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		all other combinations	**SPI mode:** Reserved. These bit combinations should not occur.

Table 13-25. *Transfer Group Interrupt Vector 0* (*INTVECT0*)] (Continued)

Bit	Name	Value	Description
0	SUSPEND0		**Transfer suspended / Transfer finished interrupt flag.** Every time INTVECT0 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT0 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a “transfer finished” interrupt. In other words, the buffer array referenced by INTVECT0 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a “transfer suspended” interrupt. In other words, the transfer group referenced by INTVECT0 has asserted an interrupt because the buffer to be transferred next is in “suspend-to-wait” mode.

Note: Reading from the INTVECT0 register when “Transmit Empty” is indicated does not clear the TXINTFLG flag in the SPIFLG register. Writing a new word to the SPIDATx register clears the “Transmit Empty” interrupt.

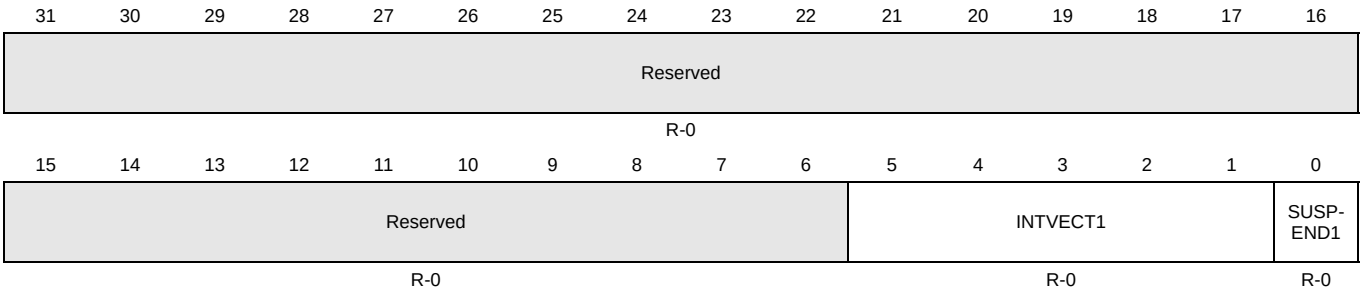
Note: In multi-buffer mode, INTVECT0 contains the interrupt for the highest priority transfer group. A read from INTVECT0 automatically causes the next-highest priority transfer group’s interrupt status to get loaded into INTVECT0 and its corresponding SUSPEND flag to get loaded into SUSPEND0. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT0 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPIFLG register or by reading the RXOVERN_BUF_ADDR register.

13.7.23 Interrupt Vector 1 (INTVECT1)

Note: The TG interrupt is not available in SPI in compatibility mode compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 13-43. *Interrupt Vector 1* (*INTVECT1*) [offset = 64h]



R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 13-26. *Transfer Group Interrupt Vector 1* (*INTVECT1*)]

Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–1	INTVECT1		**INTVECT1. Interrupt vector for interrupt line INT1.** Returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, INTVECT1 always references the highest prior interrupt source first. **Note:** This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.
		00000	There is no pending interrupt. **SPI mode only.**
		10001	Error Interrupt pending. The lower half of SPIINT1 contains more details about the type of error. **SPI mode only.**
		10011	The pending interrupt is a “Receive Buffer Overrun” interrupt. **SPI mode only.**
		10010	The pending interrupt is a “Receive Buffer Full” interrupt. **SPI mode only.**
		10100	The pending interrupt is a “Transmit Buffer Empty” interrupt. **SPI mode only.**
		all other combinations	Reserved. These bit combinations should not occur. **SPI mode only.**

Table 13-26. *Transfer Group Interrupt Vector 1* (*INTVECT1*)] (Continued)

Bit	Name	Value	Description
0	SUSPEND1		**Transfer suspended / Transfer finished interrupt flag.** Every time INTVECT1 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT1 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a “transfer finished” interrupt. In other words, the buffer array referenced by INTVECT1 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a “transfer suspended” interrupt. In other words, the transfer group referenced by INTVECT1 has asserted an interrupt because the buffer to be transferred next is in “suspend-to-wait” mode.

Note: Reading from the INTVECT1 register when “Transmit Empty” is indicated does not clear the TXINTFLG flag in the SPIFLG register. Writing a new word to the SPIDATx register clears the “Transmit Empty” interrupt.

Note: In multi-buffer mode, INTVECT1 contains the interrupt for the highest priority transfer group. A read from INTVECT1 automatically causes the next-highest priority transfer group’s interrupt status to get loaded into INTVECT1 and its corresponding SUSPEND flag to get loaded into SUSPEND1. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT1 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPIFLG register or by reading the RXOVERN_BUF_ADDR register.

13.7.24 SPI Pin Control Register 9 (SPIPC9)

Note: Register bits vary by device
Register bits 31:24 and 23:16 of this register reflect the number of SPISIMO/SPISOMI pins per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 13-44. SPI Pin Control Register 9 (SPIPC9) [offset = 68h[

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOMISRS[7:0]								SIMOSRS[7:0]							
R/W-0								R/W-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMISRS0	SIMOSRS0	CLK-SRS		SCSSRS[7:0]							
0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = write, P = Privilege mode, -*n* = Value after reset

Table 13-27. SPI Pin Control Register 9 (SPIPC9) Field Descriptions

Bit	Name	Value	Description
31–24	SOMISRS[7:0]		SPISOMIx slew rate control. Each bit controls the slew rate for the corresponding SPISOMIx pin. **Note: Bit 11 or bit 24 can be used to control the slew rate for SPISOMI0. If a 32-bit write is performed, bit 11 will have priority over bit 24.** **Note: For devices with less than 8 SPISOMI pins, the remaining bit fields are reserved.**
		0	Normal buffer select.
		1	Slow buffer select.
23–16	SIMOSRS[7:0]		SPISIMOx slew rate control. Each bit controls the slew rate for the corresponding SPISIMOx pin. **Note: Bit 10 or bit 16 can be used to control the slew rate for SPISIMO0. If a 32-bit write is performed, bit 10 will have priority over bit 16.** **Note: For devices with less than 8 SPISIMO pins, the remaining bit fields are reserved.**
		0	Normal buffer select.
		1	Slow buffer select.

Table 13-27. SPI Pin Control Register 9 (SPIPC9) Field Descriptions (Continued)

Bit	Name	Value	Description
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMISRS0		SPISOMI0 slew rate control. This bit controls the slew rate for the SPISOMI0 pin.
		0	Normal buffer select.
		1	Slow buffer select.
10	SPISIMOSRS0		SPISIMO0 slew rate control. This bit controls the slew rate for the SPISIMO0 pin.
		0	Normal buffer select.
		1	Slow buffer select.
9	SPICLKSRS		SPICLK pin slew rate control. This bit controls the slew rate for the SPICLK pin.
		0	Normal buffer select.
		1	Slow buffer select.
8	SPIENASRS		$\overline{\text{SPIENA}}$ pin slew rate control. This bit controls the slew rate for the $\overline{\text{SPIENA}}$ pin.
		0	Fast buffer Select.
		1	Slow buffer Select.
7–0	SPISCSSRS[7:0]		$\overline{\text{SPISCS}}$x pin slew rate control. Each bit controls the slew rate for the corresponding $\overline{\text{SPISCS}}$x pin.
		0	Normal buffer select.
		1	Slow buffer select

13.7.25 Multi-buffer Mode Enable Register (MIBSPIE)

Note: Accessibility of Multi-Buffer RAM
The multi-buffer RAM is not accessible unless the MSPIENA bit set to 1. The only exception to this is in test mode, where, by setting RXRAMACCESS to 1, the multi-buffer RAM can be fully accessed for both read and write.

Figure 13-45. Multi-buffer Mode Enable Register (MIBSPIE) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RXRA-MAC-CESS
R-0															R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MSPI-ENA
R-0															R/WP-0

R = Read, W = write, P = Privilege mode, -*n* = Value after reset

Table 13-28. Multi-buffer Mode Enable Register (MIBSPIE) Field Descriptions

Bit	Name	Value	Description
31–17	Reserved		Reads return zeros and writes have no effect.
16	RXRAMACCESS		**Receive-RAM access control.** During normal operating mode of SPI, the receive data/status portion of multi-buffer RAM is read-only. To enable testing of receive RAM, direct read/write access is enabled by setting this bit.
		0	The RX portion of multi-buffer RAM is not writable by the CPU.
		1	The whole of multi-buffer RAM is fully accessible for read/write by the CPU.
			Note: The RX RAM ACCESS bit remains 0 after reset and it should remain set to 0 at all times, except when testing the RAM. SPI should be given a local reset by using the nRESET (SPIGCR0[0]) bit after RAM testing is performed so that the multi-buffer RAM gets re-initialized.
15–1	Reserved		Reads return zeros and writes have no effect.

Table 13-28. Multi-buffer Mode Enable Register (MIBSPIE) Field Descriptions (Continued)

Bit	Name	Value	Description
0	MSPIENA		Multi-buffer mode enable. After power-up or reset, MSPIENA remains cleared, which means that the SPI runs in compatibility mode by default. If multi-buffer mode is desired, this register should be configured first after configuring the SPIGCR0 register. If MSPIENA is not set to 1, the multi-buffer mode registers are not writable.
		0	The SPI runs in compatibility mode, i.e., in this mode the MibSPI is fully code-compliant to the standard TMS470Px SPI. No multi-buffered-mode features are supported.
		1	The SPI is configured to run in multi-buffer mode.

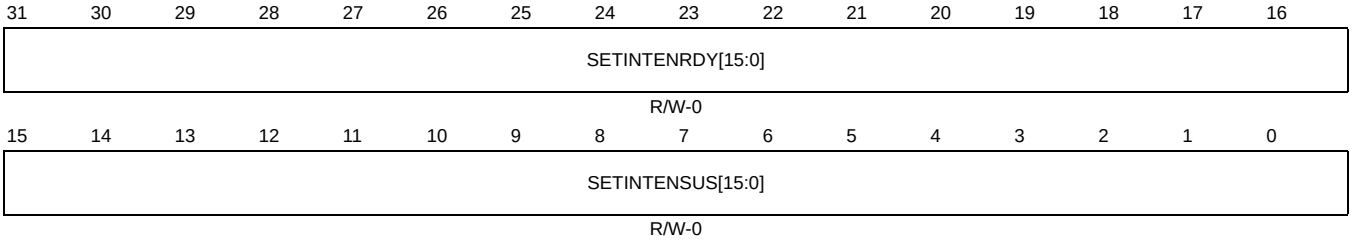
Note: Accessibility of Registers
Registers from this offset address onwards are not accessible in SPI compatibility mode. They are accessible only in the multi-buffer mode.

13.7.26 TG Interrupt Enable Set Register (TGITENST)

The register TGITENST contains the TG interrupt enable flags for transfer-finished and for transfer-suspended events. Each of the enable bits in the higher half-word and the lower half-word of TGITENST belongs to one TG.

The register map shown in Figure 13-46 and Table 13-29 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 13-46. TG Interrupt Enable Set Register (TGITENST) [offset = 74h]



R = Read; W = Write; -*n* = Value after reset

Table 13-29. TG Interrupt Enable Set Register (TGITENST) Field Descriptions

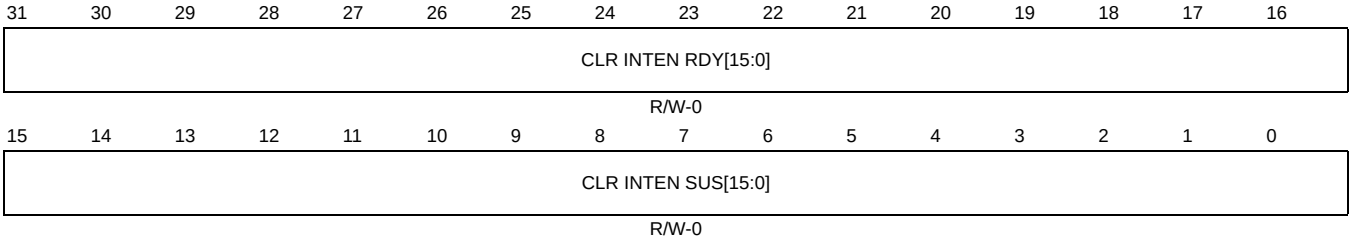
Bit	Name	Value	Description
31-16	SET INTENRDY[15:0]		TG interrupt set (enable) when transfer finished.
		0	*Read:* The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx completes. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. Write: Enable the TGx-completed interrupt. The interrupt gets generated when TGx completes.
15–0	SET INTEN SUS[15:0]		TG interrupt set (enabled) when transfer suspended
		0	*Read:* The TGx-suspended interrupt is disabled. This interrupt does not get generated when TGx is suspended. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is enabled. The interrupt gets generated when TGx is suspended. *Write:* Enable the TGx-suspended interrupt. The interrupt gets generated when TGx is suspended.

13.7.27 MibSPI TG Interrupt Enable Clear Register (TGITENCR)

The register TGITENCR is used to clear the interrupt enables for the TG-completed interrupt and the TG-suspended interrupts.

The register map shown in Figure 13-47 and Table 13-30 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 13-47. TG Interrupt Enable Clear Register (TGITENCR) [offset = 78h]



R = Read; W = Write; -*n* = Value after reset

Table 13-30. TG Interrupt Enable Clear Register (TGITENCR) Field Descriptions

Bit	Name	Value	Description
31-16	CLR INTEN RDY[15:0]		TG interrupt clear (disabled) when transfer finished.
		0	*Read:* The TGx-completed interrupt is disabled. The interrupt does not get generated when TGx completes. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. *Write:* Disable the TGx-completed interrupt.
15–0	CLR INTEN SUS[15:0]		TG interrupt clear (disabled) when transfer suspended.
		0	*Read:* The TGx-suspended interrupt is disabled. This interrupt does not get generated when TGx is suspended. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is enabled. The interrupt gets generated when TGx completes. *Write:* Disables the TGx-suspended interrupt.

13.7.28 Transfer Group Interrupt Level Set Register (TGITLVST)

The register TGITLVST sets the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 1.

The register map shown in Figure 13-48 and Table 13-31 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 13-48. Transfer Group Interrupt Level Set Register (TGITLVST) [offset = 7Ch]

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

R = Read; W = Write; -*n* = Value after reset

Table 13-31. Transfer Group Interrupt Level Set Register (TGITLVST) Field Descriptions

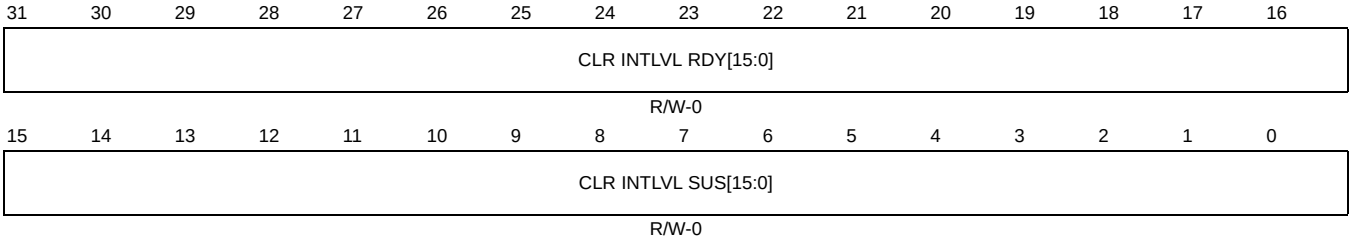
Bit	Name	Value	Description
31-16	SET INTLVL RDY[15:0]		Transfer-group completed interrupt level set.
		0	*Read:* The TGx-completed interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is set to INT1. *Write:* Set the TGx-completed interrupt to INT1.
15–0	SET INTLVL SUS[15:0]		Transfer-group suspended interrupt level set.
		0	*Read:* TGx-suspended interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is set to INT1. *Write:* Set the TG-x suspended interrupt INT1.

13.7.29 Transfer Group Interrupt Level Clear Register (TGITLVCR)

The register TGITLVCR clears the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 0.

The register map shown in Figure 13-49 and Table 13-32 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 13-49. Transfer Group Interrupt Level Clear Register (TGITLVCR) [offset = 80h]



R = Read; W = Write; -*n* = Value after reset

Table 13-32. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

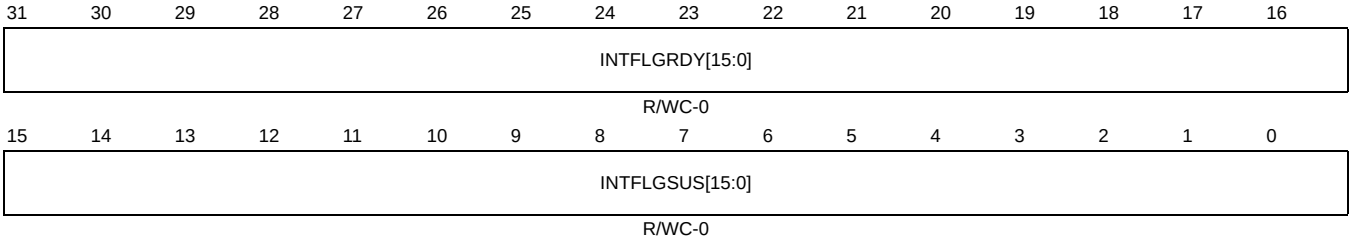
Bit	Name	Value	Description
31-16	CLR INTLVL RDY[15:0]		Transfer-group completed interrupt level clear.
		0	*Read:* The TGx-completed interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is set to INT1. *Write:* Sets the TG-x completed interrupt to INT0.
15–0	CLR INTLVL SUS[15:0]		Transfer group suspended interrupt level clear.
		0	*Read:* The TG-x suspended interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is set to INT1. *Write:* Sets the TGx-suspended interrupt to INT0.

13.7.30 Transfer Group Interrupt Flag Register (TGINTFLAG)

The TGINTFLAG register comprises the transfer group interrupt flags for transfer-completed interrupts (INTFLGRDYx) and for transfer-suspended interrupts (INTFLGSUSx). Each of the interrupt flags in the higher half-word and the lower half-word of TGINTFLAG belongs to one TG.

The register map shown in Figure 13-50 and Table 13-33 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 13-50. Transfer Group Interrupt Flag Register (TGINTFLAG) [offset = 84h]



R = Read; W = Write; C = Clear; *-n* = Value after reset

Table 13-33. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

Bit	Name	Value	Description
31-16	INTFLGRDY[15:0]		Transfer-group interrupt flag for a transfer-completed interrupt. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGRDYx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the vector registers is 0.**
		0	*Read:* No transfer-completed interrupt occurred since last clearing of the INTFLGRDYx flag. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* A transfer finished interrupt from transfer group x occurred. No matter whether the interrupt is enabled or disabled (INTENRDYx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGRDYx is set right after the transfer from TGx is finished. *Write:* The corresponding bit flag is cleared.

Table 13-33. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions (Continued)

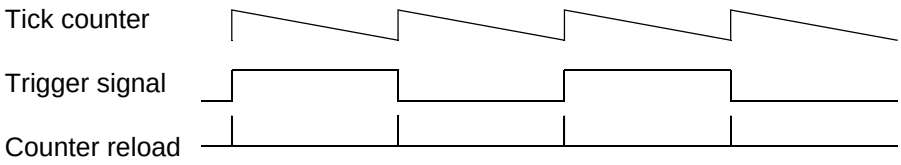
Bit	Name	Value	Description
15–0	INTFLG SUS[15:0]		Transfer-group interrupt flag for a transfer-suspend interrupt. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGSUSx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the corresponding vector registers is 1.**
		0	*Read:* No transfer-suspended interrupt occurred since the last clearing of the INTFLGSUSx flag. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* A transfer-suspended interrupt from TGx occurred. No matter whether the interrupt is enabled or disabled (INTENSUSx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGSUSx is set right after the transfer from transfer group x is suspended. *Write:* The corresponding bit flag is cleared.

13.7.31 Tick Count Register (TICKCNT)

One of the trigger sources for TGs is an internal periodic time trigger. This time trigger is called a tick counter and is basically a down-counter with a preload/reload value. Every time the tick counter detects an underflow it reloads the initial value and toggles the trigger signal provided to the TGs.

The trigger signal, shown in Figure 13-51 as a square wave, illustrates the different trigger event types for the TGs (e.g., rising edge, falling edge, and both edges).

Figure 13-51. Tick Counter Operation



This register is shown in Figure 13-52 and described in Table 13-34.

Figure 13-52. Tick Count Register (TICKCNT) [offset = 90h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TICK ENA	RE LOAD	CLKCTRL(1–0)		Reserved											
R/W-0	R0/S-0	R/W-0		R-0											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TICKVALUE(15–0)															
R/W-0															

R = Read; W = Write; S = Set; C = Clear; -*n* = Value after reset;

Table 13-34. Tick Count Register (TICKCNT) Field Descriptions

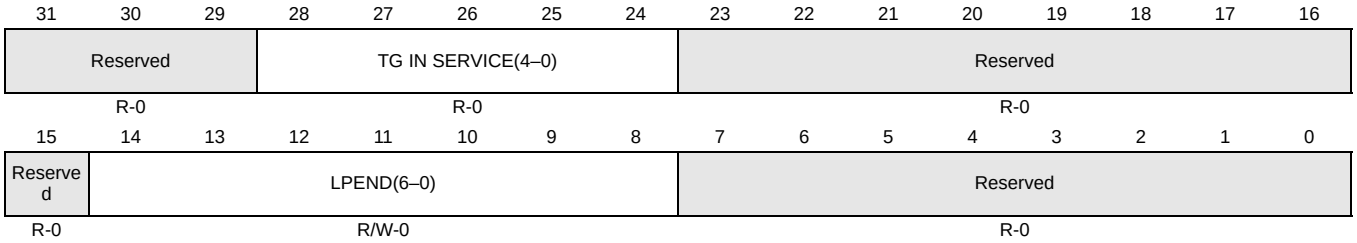
Bit	Name	Value	Description
31	TICK ENA		Tick counter enable.
		0	The internal tick counter is disabled. The counter value remains unchanged. **Note: When the tick counter is disabled, the trigger signal is forced low.**
		1	The internal tick counter is enabled and is clocked by the clock source selected by CLKCTRL[1:0]. When the tick counter is enabled it starts down-counting from its current value. When TICK-ENA goes from 0 to 1, the tick counter is automatically loaded with the contents of TICKVALUE.

Table 13-34. Tick Count Register (TICKCNT) Field Descriptions (Continued)

Bit	Name	Value	Description
30	RELOAD		**Pre-load the tick counter.** RELOAD is a set-only bit; writing a 1 to it reloads the tick counter with the value stored in TICKVALUE. Reading RELOAD always returns a 0. **Note: When the tick counter is reloaded by the RELOAD bit, the trigger signal is not toggled.**
29–28	CLKCTRL(1–0)		**Tick counter clock source control.** CLKCTRL defines the clock source that is used to clock the internal tick counter.
		00	SPICLK of data word format 0 is selected as the clock source of the tick counter
		01	SPICLK of data word format 1 is selected as the clock source of the tick counter
		10	SPICLK of data word format 2 is selected as the clock source of the tick counter
		11	SPICLK of data word format 3 is selected as the clock source of the tick counter
27–16	Reserved		Reads return 0 and writes have no effect.
15–0	TICKVALUE(15–0)	0–FFFF	**Initial value for the tick counter.** TICKVALUE stores the initial value for the tick counter. The tick counter is loaded with the contents of TICKVALUE every time an underflow condition occurs and every time the RELOAD flag is set by the host.

13.7.32 Last TG End Pointer (LTGPEND)

Figure 13-53. Last TG End Pointer (LTGPEND) [offset = 94h]



R = Read, W = Write, C = Clear, -*n* = Value after reset, x = indeterminate

Table 13-35. Last TG End Pointer (LTGPEND) Field Descriptions

Bit	Name	Value	Description
31–29	Reserved		Reads return 0 and writes have no effect.
28–24	TG IN SERVICE(4–0)		**The TG number currently being serviced by the sequencer.** These bits indicate the current TG that is being serviced. This field can generally be used for code debugging.
		00000	No TG is being serviced by the sequencer.
		00001	TG0 is being serviced by the sequencer.
		. . .	...
		10000	TG15 is being serviced by the sequencer.
			Note: The number of transfer groups varies by device.
		10001–11111	Invalid values

Table 13-35. Last TG End Pointer (LTGPEND) Field Descriptions (Continued)

Bit	Name	Value	Description
23–15	Reserved		Reads return 0 and writes have no effect.
14–8	LPEND(6–0)	0–7Fh	**Last TG end pointer.** Usually the TG end address (PEND) is inherently defined by the start value of the starting pointer of the subsequent TG (PSTART). The TG ends one word before the next TG starts (PEND[x]=PSTART[x+1] - 1). For a full configuration of MibSPI, the last TG has no subsequent TG, i.e., no end address is defined. Therefore LPEND has to be programmed to specify explicitly the end address of the last TG. **Note: LPEND allows SW compatibility for MibSPI variants Because of software compatibility reasons, the last TG end address has to be configurable; otherwise a super-set MibSPI cannot emulate a MibSPI with less TGs without software changes.** **Note: The number of transfer groups varies by device, thus the last TG also varies by device.**
7–0	Reserved		Reads return 0 and writes have no effect.

13.7.33 TGx Control Registers (TGxCTRL)

Each TG can be configured via one dedicated control register. The register description below shows one control register(x) which is identical for all TGs. For example, the control register for TG 2 is named TG2CTRL and is located at address *base0+98h+4*2*. The actual number of available control registers varies by device.

Figure 13-54. MibSPI TG Control Registers (TGxCTRL) [offsets = 98h–D4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TG ENA[0:15]	ONE SHOT[0:15]	PRST[0:15]	TGTD[0:15]	Reserved				TRGEVT[0:15](3–0)				TRIGSRC[0:15](3–0)			
R/W-0	R/W-0	R/W-0	R-0	R-0				R/W-0				R/W-0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PSTART[0:15](6–0)							Reserved	PCURRENT[0:15](6–0)						
R-0	R/W-0							R-0	R-0						

R = Read; W = Write; *-n* = Value after reset;

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions

Bit	Name	Value	Description
31	TGENA		TGx enable. If the correct event (TRIGEVTx) occurs at the selected source (TRIGSRCx) a group transfer is initiated if no higher priority TG is in active transfer mode or if one or more higher-priority TGs are in transfer-suspend mode. Disabling a TG while a transfer is ongoing will finish the ongoing word transfer but not the whole group transfer.
		0	TGx is disabled.
		1	TGx is enabled.
30	ONESHOTx		Single transfer for TGx.
		0	TGx initiates a transfer every time a trigger event occurs and TGENA is set.
		1	A transfer from TGx will be performed only once (one shot) after a valid trigger event at the selected trigger source. After the transfer is finished the TGENAx control bit will be cleared and therefore no additional transfer can be triggered before the host enables the TG again. This one shot mode ensures that after one group transfer the host has enough time to read the received data and to provide new transmit data.

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
29	PRSTx		TGx pointer reset mode. Configures the way to resolve trigger events during an ongoing transfer. This bit is meaningful only for level-triggered TGs. Edge-triggered TGs cannot be restarted before their completion by another edge. The PRST bit will have no effect on this behavior. **Note: When the PRST bit is set, if the buffer being transferred at the time of a new trigger event is a LOCK, CSHOLD or NOBRK buffer, then only after finishing those transfers, the TG will be restarted. This means that even if the TG is retriggered, the TG will only be restarted after finishing the transfer of the first non-LOCK or non-CSHOLD buffer. In the case of the NOBRK buffer, after completing the ICOUNT number of transfers, the TG will be restarted from its PSTART.** This means that TX control fields such as LOCK and CSHOLD, and DMA control fields such as NOBRK have higher priority over anything else. They have the capability to delay the restart of the TG even if it is retriggered when PRST is 1.
		0	If a trigger event occurs during a transfer from TGx, the event is ignored and is not stored internally. The TGx transfer has priority over additional trigger events.
		1	The TGx pointer (PCURRENTx) will be reset to the start address (PSTARTx) when a valid trigger event occurs at the selected trigger source while a transfer from the same TG is ongoing. Every trigger event resets PCURRENTx no matter whether the concerned TG is in transfer mode or not. The trigger events have priority over the ongoing transfer.
28	TGTDx		TG triggered.
		0	TGx has not been triggered or is no longer waiting for service.
		1	TGx has been triggered and is either currently being serviced or waiting for servicing.
27–24	Reserved		Reads return 0 and writes have no effect.

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
23–20	TRIGEVTx(3-0)		**Type of trigger event.** A level-triggered TG can be stopped by de-activating the level trigger. However, the following restrictions apply. 1. Deactivating the level trigger for a TG during a NOBRK transfer does not stop the transfers until all of the ICOUNT number of buffers are transferred for the NOBRK buffer. Once a NOBRK buffer is prefetched, the trigger event loses control over the TG until the NOBRK buffer transfer is completed. 2. Once the transfer of a buffer with CSHOLD or LOCK bit set starts, deactivating the trigger level does not stop the transfer until the sequencer completes the transfer of the next non-CSHOLD or non-LOCK buffer in the same TG. 3. Once the last buffer in a TG is pre-fetched, de-activating the trigger level does not stop the transfer group until the last buffer transfer is completed. This means even if the trigger level is deactivated at the beginning of the penultimate (one-before-last) buffer transfer, the sequencer continues with the same TG until it is completed.
		0000	never — Never trigger TGx. This is the default value after reset.
		0001	rising edge — A rising edge (0 to 1) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0010	falling edge — A falling edge (1 to 0) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0011	both edges — Rising and falling edges at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0100	Reserved
		0101	high-active — While the selected trigger source (TRIGSRCx) is at a logic high level (1) the group transfer is continued and at the end of one group transfer restarted at the beginning. If the logic level changes to low (0) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**
		0110	low-active — While the selected trigger source (TRIGSRCx) is at a logic low level (0) the group transfer is continued and at the end of one restarted at the beginning. If the logic level changes to high (1) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description	
		0111	always	A repetitive group transfer will be performed. **Note: By setting the TRIGSRC to 0000b, the TRIGEVT to 0111b (ALWAYS), and the ONESHOTx bit to 1, software can trigger this TG. Upon setting the TGENA bit, the TG is immediately triggered.** **Note: If ONESHOTx is set the transfer is performed only once.**
		1xxx	Reserved	

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
19–16	TRIGSRCx(3-0)		**Trigger source.** After reset, the trigger sources of all TGs are disabled.
		0000	disabled
		0001	EXT0 External trigger source 0. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0010	EXT1 External trigger source 1. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0011	EXT2 External trigger source 2. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0100	EXT3 External trigger source 3. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0101	EXT4 External trigger source 4. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0110	EXT5 External trigger source 5. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0111	EXT6 External trigger source 6. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1000	EXT7 External trigger source 7. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1001	EXT8 External trigger source 8. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1010	EXT9 External trigger source 9. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1011	EXT10 External trigger source 10. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1100	EXT11 External trigger source 11. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1101	EXT12 External trigger source 12. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1110	EXT13 External trigger source 13. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1111	TICK Internal periodic event trigger. The tick counter can initiate periodic group transfers.
15	Reserved		Reads return 0 and writes have no effect.

Table 13-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
14–8	PSTARTx(6-0)	0–7Fh	**TG start address.** PSTARTx stores the start address of the corresponding TG. The corresponding end address is inherently defined by the subsequent TG's start address minus one (PENDx[TGx] = PSTARTx[TGx+1]-1). PSTARTx is copied into PCURRENTx when: • The TG is enabled. • The end of the TG is reached during a transfer. • A trigger event occurs while PRST is set to 1.
7	Reserved		Reads return 0 and writes have no effect.
6–0	PCURRENTx(6-0)	0–7Fh	Pointer to current buffer. PCURRENT is read-only. PCURRENTx stores the address (0...127) of the buffer that corresponds to this TG. If the TG switches from active transfer mode to suspend to wait, PCURRENTx contains the address of the currently suspended word. After the TG resumes from suspend to wait mode, the next buffer will be transferred; i.e. no buffer data is transferred because of suspend to wait mode.

Note: TG0 has the highest priority and TG15 has the lowest priority.

Under the following conditions a lower priority TG cannot be interrupted by a higher priority TG.

1. When there is a CSHOLD or LOCK buffer, until the completion of the next buffer transfer which is a non-CSHOLD or non-LOCK buffer.

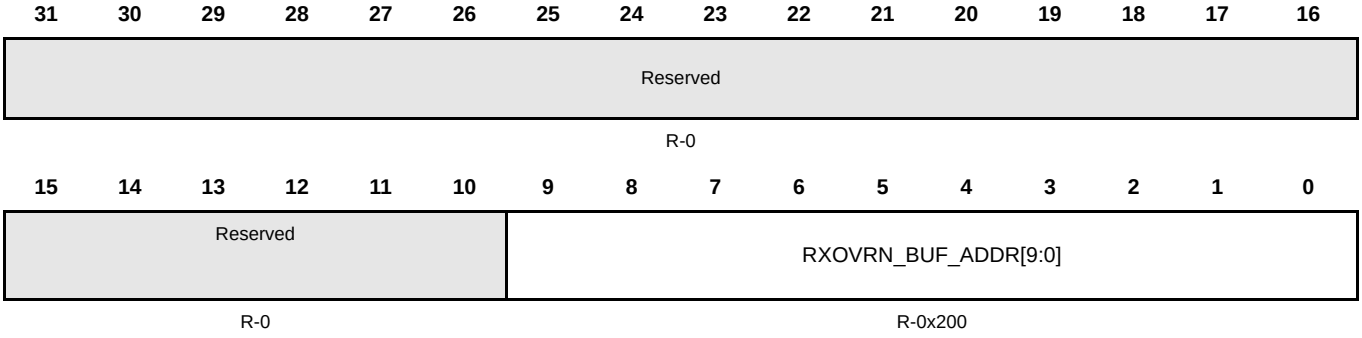
2. An entire sequence of words transferred for a NOBRK DMA buffer.

3. Once the last word in a TG is pre-fetched.

13.7.34 RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR)

In multi-buffer mode, if a particular RXRAM location is written by the MibSPI sequencer logic after the completion of a new transfer when that location already contains valid data, the RX_OVR bit will be set to 1 while the data is being written. The RXOVRN_BUF_ADDR register captures the address of the RXRAM location for which a receiver overrun condition occurred.

Figure 13-55. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) [offset = 130h]



R = Read, -*n* = Value after power-on reset

Table 13-37. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
9–0	RXOVRN_BUF_ADDR[9:0]	200–3FCh	**Address in RXRAM at which an overwrite occurred.** This address value will show only the offset address of the RAM location in the multi-buffer RAM address space. Refer to the device-specific data sheet for the actual absolute address of RXRAM. This word-aligned address can vary from 0x200 - 0x3FC. Contents of this register are valid only when any of the TGINTVECT0 or TGINTVECT1 and SPIFLG registers show an RXOVRN error vector while in multi-buffer mode. If there are multiple overrun errors, then this register holds the address of first overrun address until it is read. **Note: Reading this register clears the RXOVRN interrupt flag in the SPIFLG register and the TGINTVECTx.** **Note: Receiver overrun errors in multi-buffer mode can be completely avoided by using the *SUSPEND until RXEMPTY* feature, which can be programmed into each buffer of any TG. However, using the *SUSPEND until RXEMPTY* feature will make the sequencer wait until the current RXRAM location is read by the VBUS master before it can start the transfer for the same buffer location again. This may affect the overall throughput of the SPI transfer. By enabling the interrupt on RXOVRN in multi-buffer mode, the user can rely on interrupts to know if a receiver overrun has occurred. The address of the overrun in RXRAM is indicated in this RXOVRN_BUF_ADDR register.**

13.7.35 I/O-Loopback Test Control Register (IOLPBKTSTCR)

This register controls test mode for I/O pins. It also controls whether loop-back should be digital or analog. In addition, it contains control bits to induce error conditions into the module. These are to be used only for module testing.

All of the control/status bits in this register are valid only when the IO LPBK TST ENA field is set to 1010.

Figure 13-56. I/O-Loopback Test Control Register (IOLPBKTSTCR) [offset = 134h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SCS FAIL FLG	Reserved			CTRL BIT ERR	CTRLDES YNC	CTRL PAR ERR	CTRL TIME OUT	CTRL DLEN ERR
R-0							R/C-0	R-0			R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				IOLPBKTSTENA[3:0]				Reserved		ERR SCS PIN			CTRL SCS PIN ERR	LPBK TYPE	RXP ENA
R-0				R/WP-0101				R-0		R/WP-000			R/WP-0	R/WP-0	R/WP-0

R = Read, WP = Write in Privilege mode only, C = Clear; -*n* = Value after power-on reset

Table 13-38. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	SCS FAIL FLAG		Bit indicating a failure on $\overline{\text{SPISCS}}$ pin compare during analog loopback.
		0	*Read:* No miscompares occurred on any of the eight chip select pins (vs. the internal chip select number CSNR during transfers). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A comparison between the internal CSNR field and the analog looped-back value of one or more of the $\overline{\text{SPISCS}}$[7:0] pins failed. A stuck-at fault is detected on one of the $\overline{\text{SPISCS}}$[7:0]. Comparison is done only on the pins that are configured as functional and during transfer operation. *Write:* This flag bit is cleared.
23–221	Reserved		Reads return zero and writes have no effect.
20	CTRL BITERR		Controls inducing of BITERR during I/O loopback test mode.
		0	Do not interfere with looped-back data.
		1	Introduces bit errors by inverting the value of the incoming data during loopback.

Table 13-38. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions (Continued)

Bit	Name	Value	Description
19	CTRL DESYNC		Controls inducing of the desync error during I/O loopback test mode.
		0	Do not cause a desync error.
		1	Induce a desync error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 0 even after the transfer is complete. This forcing will be retained until the kernel reaches the idle state.
18	CTRL PARERR		Controls inducing of parity errors during I/O loopback test mode.
		0	Do not cause a parity error.
		1	Induce a parity error by inverting the polarity of the parity bit.
17	CTRL TIMEOUT		Controls inducing of the timeout error during I/O loopbacK test mode.
		0	Do not cause a timeout error.
		1	Induce a timeout error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 1 when transmission is initiated. The forcing will be retained until the kernel reaches the idle state.
16	CTRL DLENERR		Controls inducing of the data length error during I/O loopback test mode.
		0	Do not cause a data-length error.
		1	Induce a data-length error.
			Master mode: the $\overline{\text{SPIENA}}$ pin (if functional) is forced to 1 when the module starts shifting data.
			Slave mode: the incoming $\overline{\text{SPISCS}}$ pin (if functional) is forced to 1 when the module starts shifting data.
15–12	Reserved		Reads return zero and writes have no effect.
11–8	IOLPBKTSTENA[3:0]		Module I/O loopback test enable key.
		1010	Enable I/O loopback test mode.
		All other values	Disable I/O loopback test mode.
7–6	Reserved		Reads return zero and writes have no effect.

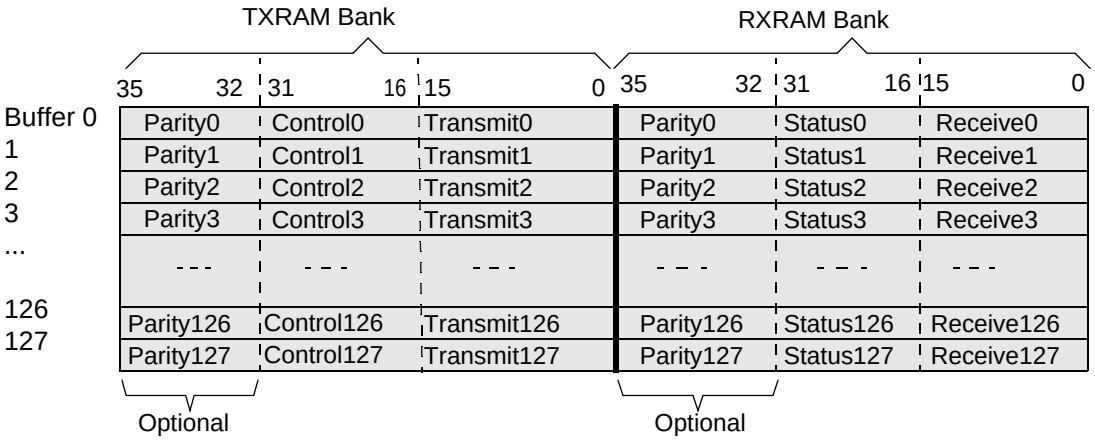
Table 13-38. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions (Continued)

Bit	Name	Value	Description
5–3	ERR SCS PIN[2:0]		Inject error on chip-select pin number x. The value in this field is decoded as the number of the chip select pin on which to inject an error. During analog loopback, if CTRL SCS PIN ERR bit is set to 1, then the chipselect pin selected by this field is forced to the opposite of its value in the CSNR.
		000	Select $\overline{\text{SPISCS}}$[0] for injecting error
		001	Select $\overline{\text{SPISCS}}$[1] for injecting error
		. . .	. . .
		111	Select $\overline{\text{SPISCS}}$[7] for injecting error.
2	CTRL SCS PIN ERR		Enable/disable the injection of an error on the $\overline{\text{SPISCS}}$[7:0] pins. The individual $\overline{\text{SPISCS}}$[7:0] pins can be chosen using the ERR SCS PIN field.
		0	Disable the $\overline{\text{SPISCS}}$[7:0] error-inducing logic.
		1	Enable the $\overline{\text{SPISCS}}$[7:0] error-inducing logic.
1	LPBK TYPE		**Module I/O loopback type (analog/digital).** See Figure 13-14 for the different types of loopback modes.
		0	Enable Digital loopback when IOLPBKTSTENA = 1010.
		1	Enable Analog loopback when IOLPBKTSTENA = 1010.
0	RXP ENA		Enable analog loopback through the receive pin. Note: This bit is valid only when LPBK TYPE = 1, which chooses analog loopback mode.
		0	Analog loopback is through the transmit pin.
		1	Analog loopback is through the receive pin.

13.8 Multi-Buffer RAM

The multi-buffer RAM is used for holding transit & received data, control and status information. The multi-buffer RAM contains two banks of up to 128 32-bit words for a maximum configuration. One bank (TXRAM) contains entries for transmit data (replicating the SPIDAT1 register). The other bank (RXRAM) contains received data (replicating the SPIBUF register). The buffers can be partitioned into multiple TGs, each containing a programmable number of buffers. Each of the buffers can be subdivided into an up to 16-bit transmit field, an up to 16-bit receive field, an up to 16-bit control field, and an up to 16-bit status field, as displayed in Figure 13-57.

Figure 13-57. Multi-Buffer RAM Configuration



All fields can be read and written with 8-bit, 16-bit, or 32-bit accesses.

The transmit fields can be written and read in the address range 000h to 1FFh. The transmit words contain data and control fields.

The receive RAM fields are read-only and can be accessed through the address range 200h to 3FCh. The receive words contain data and status fields.

The chip select number bit field CSNR[7:0] of the control field for a given word is mirrored into the corresponding receive-buffer status field after transmission.

13.8.1 Multi-buffer RAM Register Summary

This section describes the Multi-buffer RAM control and transmit-data fields of each word of TXRAM, and the status and receive-data fields of each word of RXRAM. The offset of the multi-buffer RAM is 0xFF0E 0000.

Figure 13-58. MibSPI RAM Register Summary

Offset Address† Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
Transmit Buffers																
0x00h–1FCh Buffer[0:127]	BUFMODE(2–0)			CS HOLD	LOCK	WDEL	DFSEL(1–0)		CSNR(7–0)							
	TXDATA(15–0)															
Receive Buffers:																
0x200–3FCh Buffer[0:127]	RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x204h Buffer 1	RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x3F8h Buffer 126	RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x3FCh Buffer 127	RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARITYERR	TIMEOUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															

13.8.2 Multi-buffer RAM Transmit Data Register

Each word of TXRAM is a transmit-buffer register.

Figure 13-59. Multi-buffer RAM Transmit Data Register [offset = Base + 000–1FFh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUFMODE			CS HOLD	LOCK	WDEL	DFSEL[1:0]		CSNR[7:0]							
R/W-0			R/W-0	R/W-0	R/W-0	R/W-0		R/W-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDATA[15:0]															
R/W-0															

R = Read, W = write, -*n* = Value after reset

Table 13-39. Multi-buffer RAM Transmit Data Register Field Descriptions

Bit	Name	Value	Description
31–29	BUFMODE		Specify conditions that are recognized by the sequencer to initiate transfers of each buffer word. When one of the “skip” modes is selected, the sequencer checks the buffer status every time it reads from this buffer. If the current buffer status (TXFULL, RXEMPTY) does not match, the buffer is skipped without a data transfer. When one of the “suspend” modes is selected, the sequencer checks the buffer status when it reads from this buffer. If TXFULL and/or RXEMPTY do not match, the sequencer waits until a match occurs. No data transfer is initiated until the status condition of this buffer changes.
		000	**disabled.** The buffer is disabled
		001	**skip single-transfer mode.** Skip this buffer until the corresponding TXFULL flag is set (i.e. new transmit data is available).
		010	**skip overwrite-protect mode.** Skip this buffer until the corresponding RXEMPTY flag is set (i.e. new receive data can be stored in RXDATA without data loss).
		011	**skip single-transfer overwrite-protect mode**. Skip this buffer until both of the corresponding TXFULL and RXEMPTY flags are set. (i.e. new transmit data available and previous data received by the host).
		100	**continuous mode.** Initiate a transfer each time the sequencer checks this buffer. Data words are retransmitted if the buffer has not been updated. Receive data is overwritten, even if it has not been read.

Table 13-39. Multi-buffer RAM Transmit Data Register Field Descriptions (Continued)

Bit	Name	Value	Description
		101	**suspend single-transfer mode.** Suspend-to-wait until the corresponding TXFULL flag is set (i.e. the sequencer stops at the current buffer until new transmit data is written in the TXDATA field).
		110	**suspend overwrite-protect mode.** Suspend-to-wait until the corresponding RXEMPTY flag is set (i.e. the sequencer stops at the current buffer until the previously-received data is read by the host.
		111	**suspend single-transfer overwrite-protect mode.** Suspend-to-wait until the corresponding TXFULL and RXEMPTY flags are set (i.e. the sequencer stops at the current buffer until new transmit data is written into the TXDATA field and the previously-received data is read by the host).
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only in compatibility-mode of MibSPI, (it is ignored in slave mode). CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.
27	LOCK		Lock two consecutive buffer words. Do not allow interruption by TG's with higher priority.
		0	Any higher-priority TG can begin at the end of the current transaction.
		1	A higher-priority TG cannot occur until after the next unlocked buffer word is transferred.

Table 13-39. Multi-buffer RAM Transmit Data Register Field Descriptions (Continued)

Bit	Name	Value	Description
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, $\overline{\text{SPISCS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the SPISCS pin remains deactivated also depends upon the time taken to supply a new word after completing the shift operation (in compatibility mode). If TXBUF is already full, then the SPISCS will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPISCS}}$ pins will be de-activated for at least (WDELAY + 2) * VCLK_Period duration.
25–24	DFSEL[2:0]		Data word format select
		00	Data word format 0 is selected
		01	Data word format 1 is selected
		10	Data word format 2 is selected
		11	Data word format 3 is selected
23–16	CSNR[7:0]	0–FF	Chip select number. CSNR defines the chip-select that will be activated during the data transfer. **Note: Writing to only the control field (using byte writes) does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL fields in the control field to select the required phase/polarity combination.**
15–0	TXDATA[15:0]	0–7FFFh	**Transfer data.** When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of SPIDAT1 to 0x0000. Write to this register ONLY when using the automatic slave chip-select feature (see Section 13.2, *Operating Modes* on page 510 for more information). A write to this register will drive the contents of CSNR[7:0] on the $\overline{\text{SPISCS}}$[7:0] pins, if they are configured as functional pins. When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

13.8.3 Multi-buffer RAM Receive Buffer Register

Each word of RXRAM is a receive-buffer register.

Figure 13-60. Multi-buffer RAM Receive Buffer Register [offset = RAM Base + 200–3FFh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLEN-ERR	LCSNR[7:0]							
RS-1	RC-0	R-0	RC-0	RC-0	RC-0	RC-0	RC-0	R-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA[15:0)]															
R/W-0															

R = Read, W = write, C = Clear; S = Set; -*n* = Value after reset

Table 13-40. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
31	RXEMPTY		**Receive data buffer empty.** When the host reads the SPIBUF field or the entire SPIBUF register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, and the RXEMPTY flag is cleared.
		0	New data has been received and copied into the SPIBUF field.
		1	No data has been received since the last read of SPIBUF. This flag gets set to 1 under the following conditions: • Reading the RXDATA portion of the SPIBUF register. • Writing a 1 to clear the RXINTFLG bit in the SPIFLG register. Write-clearing the RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA portion of SPIBUF (or the entire register).

Table 13-40. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to SPIBUF; the contents of SPIBUF are overwritten only after it is read by the VBUSP master (e.g. CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPIFLG or SPIVECTx shows the RXOVRN condition. Two read operations from the SPIBUF register are required to reach the overwritten buffer word (one to read SPIBUF, which then transfers RXDATA into SPIBUF for the second read). This flag is cleared to 0 when the RXDATA is read. **Note: A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BIT-ERR and DLEN_ERR occur, then RXOVR in RXBUF and SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		**Transmit data buffer full.** This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		11	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		**Bit error.** There was a mismatch of internal transmit data and transmitted data.
		0	No bit error occurred. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**

Table 13-40. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
		1	A bit error occurred. The MibSPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		**Desynchronization of slave device.** This bit is valid in master mode only. The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. **Note: There is a possible inconsistency of DESYNC in the Compatibility Mode MibSPI. Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.** **This inconsistency in the desync flag is valid only in the compatibility mode of MibSPI. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.**
		0	No slave desynchronization detected. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		1	A slave device is desynchronized.
26	PARITYERR		**Parity error.** The calculated parity differs from the received parity bit. If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No parity error detected.
		1	A parity error occurred.

Table 13-40. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
25	TIMEOUT		**Time-out because of non-activation of ENA pin.** The MibSPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPIFLG register is set. **This bit is valid only in master mode.** **This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.**
		0	No ENA-pin time-out occurred.
		1	An ENA signal time-out occurred.
24	DLENERR		**Data length error flag.** **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No data-length error has occurred.
		1	A data length error has occurred.
23–16	LCSNR[7:0]	0–FFh	**Last chip select number.** LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer. The updated after transmission during the write-back of received data.
15–0	RXDATA[15:0]	0–FFFFh	**SPI receive data.** This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

Serial Peripheral Interface (SPI) Module

This reference guide provides the specifications for a 16-bit configurable synchronous multi-buffered serial peripheral interface (MibSPI). The MibSPI is a programmable-length shift register used for high-speed communication between external peripherals or other microcontrollers.

Throughout this chapter, all references to SPI also apply to MibSPI, unless otherwise noted.

Topic **Page**

14.1 Overview

The MibSPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (2 to 16 bits) to be shifted in and out of the device at a programmed bit-transfer rate. Typical applications for the SPI include interfacing to external peripherals, such as I/Os, memories, display drivers, and analog-to-digital converters.

The SPI has the following attributes:

- 16-bit shift register
- Receive buffer register
- 8-bit baud clock generator
- Serial clock (SPICLK) I/O pin
- SPISOMI/SPISIMO pins for data transfer, with programmable pin direction
- SPI enable ($\overline{\text{SPIENA}}$) I/O pin
- Up to 8 slave chip select ($\overline{\text{SPISCS}}$[7:0]) I/O pins (see the device data sheet for availability of these pins)
- SPICLK can be internally-generated (and driven) or received from an external clock source
- Each word transferred can have a unique format.
- SPI pins may be used as functional or digital I/O pins (GIOs)

14.1.1 Word Format Options

Each word transferred can have a unique format. Several format characteristics are programmable for each word transferred:

- SPICLK Frequency
- Character Length (2 to 16 bits)
- Phase (with and without delay)
- Polarity (high or low)
- Parity enabled/disabled
- CS timers for setup and hold
- Shift direction (MSB-first or LSB-first)

14.1.2 Multi-buffering (Mib) support

The MibSPI has a programmable buffer memory that enables programmed transmission to be completed without CPU intervention. The buffers are combined in different Transfer Groups (TGs) that can be triggered by external events (timers, I/O activity, etc.) or by the internal tick counter. The internal tick counter supports periodic trigger events.

14.1.2.1 Multi-buffer Mode

Multi-buffer Mode is an extension to the SPI. In multi-buffer mode many extended features are configurable:

- Number of buffers for each peripheral (or data source/destination) or group (up to 16 groupings)
- Triggers for each groups, trigger types, trigger sources for individual groups (up to 14 external trigger sources and 1 internal trigger source)

14.1.2.2 Compatibility Mode

Compatibility Mode of the MibSPI makes it behave exactly like a SPI and ensures full compatibility with other SPIs. All features in compatibility mode of the MibSPI are directly applicable to a SPI. Multi-buffer Mode features are not available in Compatibility Mode.

Note:
The SPIDAT0 register is not accessible in the multi-buffer mode of MibSPI. It is only accessible in compatibility mode.

14.1.3 Transmission Lock (Multi-Buffer Mode Master Only)

Some slave devices require transmission of a command followed by data. In this case the SPI transaction should not be interrupted by another group transfer. The LOCK bit within each buffer allows a consecutive transfer to happen without being interrupted by another higher-priority group transfer.

14.2 Operating Modes

The SPI operates as either a master or a slave. The MASTER bit (SPIGCR1[0]) selects the configuration of the SPISIMO and SPISOMI pins and the CLKMOD bit (SPIGCR1[1]) determines whether an internal or external clock source will be used.

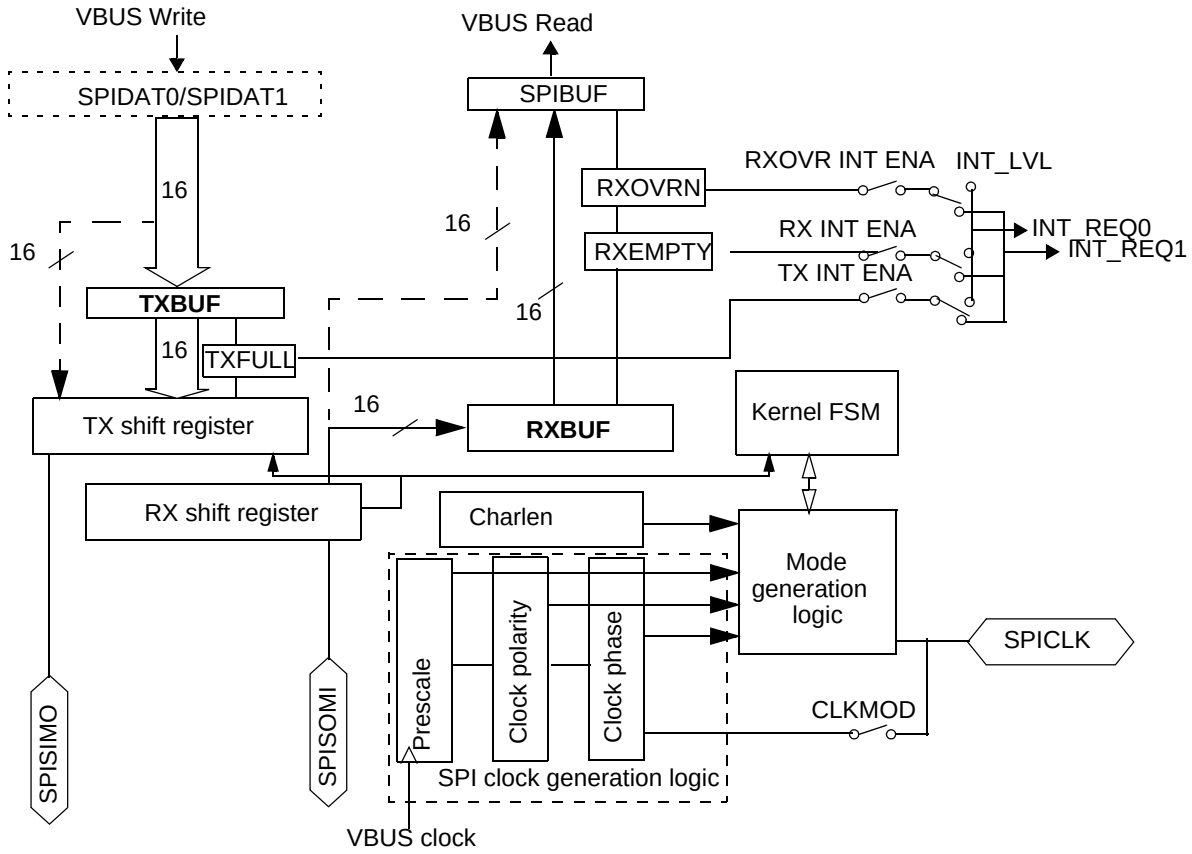
The slave chip select ($\overline{\text{SPISCS}}$[7:0]) pins, are used when communicating with multiple slave devices. When the a write occurs to SPIDAT1 in master mode, the $\overline{\text{SPISCS}}$ pins are automatically driven to select the specified slave.

A handshaking mechanism, provided by the $\overline{\text{SPIENA}}$ pin, enables a slave SPI to delay the generation of the clock signal supplied by the master if it is not prepared for the next exchange of data.

14.2.1 Data Handling

Figure 14-1 shows the SPI transaction hardware. TXBUF and RXBUF are internal buffers that are intended to improve the overall throughput of data transfer.

Figure 14-1. SPI Functional Logic Diagram



1 This is a representative diagram, which shows three-pin mode hardware.
2 TXBUF, RXBUF, and SHIFT_REGISTER are user-invisible registers.
3 SPIDAT0 and SPIDAT1 are user-visible, and are physically mapped to the contents of TXBUF.

14.2.1.1 Data Sequencing when SPIDAT0 or SPIDAT1 Is Written

- If both the TX shift register and TXBUF are empty, then the data is directly copied to the TX shift register. For devices with DMA, if DMA is enabled, a transmit DMA request (TX_DMA_REQ) is generated to cause the next word to be fetched. If transmit interrupts are enabled, a transmitter-empty interrupt is generated.
- If the TX shift register is already full or is in the process of shifting, then the data is copied to TXBUF, irrespective of whether it is already full or not and the TXFULL flag is set to 1 at the same time.
- When a shift operation is complete, data from the TXBUF (if it is full) is copied into TX shift register and the TXFULL flag is cleared to 0 to indicate that next data can be fetched. A transmit DMA request (if enabled) or a transmitter-empty interrupt (if enabled) is generated at the same time.

14.2.1.2 Data Sequencing when All Bits Shifted into RXSHIFT Register

- If both SPIBUF and RXBUF are empty, the received data in RX shift register is directly copied into SPIBUF and the receive DMA request (if enabled) is generated and the receive-interrupt (if enabled) is generated. The RXEMPTY flag in SPIBUF is cleared at the same time.
- If SPIBUF is already full at the end of receive completion, the RX shift register contents is copied to RXBUF. A receive DMA request is generated, if enabled. The receive complete interrupt line remains high.

- If SPIBUF is read by the CPU or DMA and if RXBUF is full, then the contents of RXBUF are copied to SPIBUF as soon as SPIBUF is read. RXEMPTY flag remains cleared, indicating that SPIBUF is still full.
- If both SPIBUF and RXBUF are full, then RXBUF will be overwritten and the RXOVR interrupt flag is set and an interrupt is generated, if enabled.

14.2.2 Pin Configurations

The SPI supports data connections as shown in Table 14-1.

Table 14-1. Pin Configurations

Pin	Master Mode		Slave Mode	
SPICLK	Drives the clock to external devices		Receives the clock from the external master	
SPISOMI	Receives data from the external slave		Sends data to the external master	
SPISIMO	Transmits data to the external slave		Receives data from the external master	
SPIENA	**SPIENA Disabled:** GIO	**SPIENA Enabled:** Receives ENA signal from the external slave	**SPIENA Disabled:** GIO	**SPIENA Enabled:** Receives ENA signal from the external master
SPICS[7:0]	**SPICS Disabled:** GIO	**SPICS Enabled:** Selects one or more slave devices	**SPICS Disabled:** GIO	**SPICS Enabled:** Receives the CS signal from the external master

Note:
When the SPICS[7:0] signals are disabled, the chip-select field in the transmit data is not used.

Note When the SPIENA signal is disabled, the $\overline{\text{SPIENA}}$ pin is ignored in master mode, and not driven as part of the SPI transaction in slave mode.

14.2.2.3 Three-Pin Mode

In master mode configuration (MASTER = 1 and CLKMOD = 1), the SPI provides the serial clock on the SPICLK pin. Data is transmitted on the SPISIMO pin and received on the SPISOMI pin (see Figure 14-2).

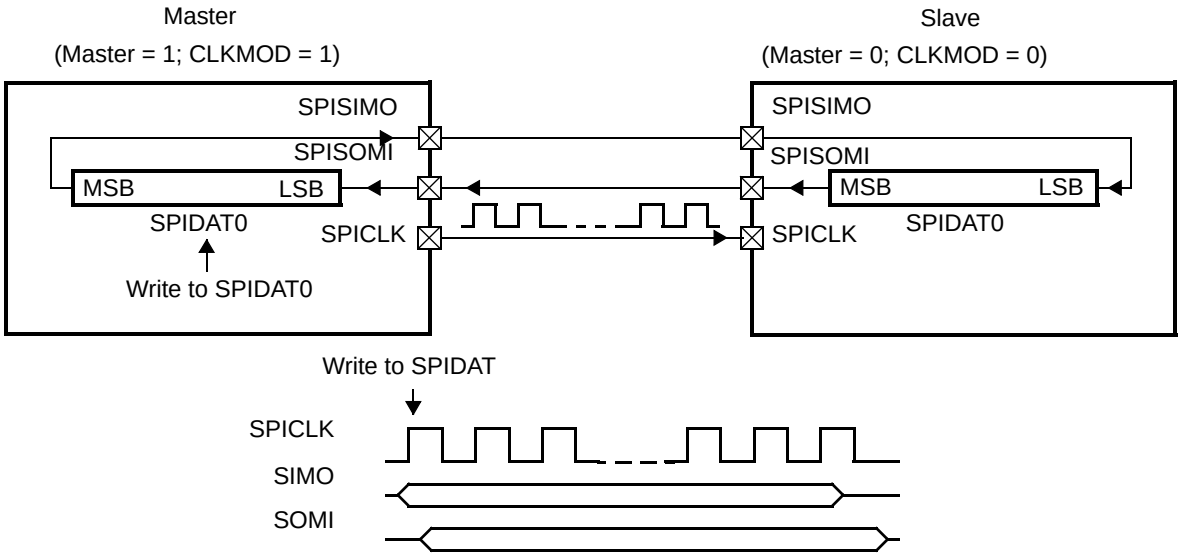
Data written to the shift register (SPIDAT0) initiates data transmission on the SPISIMO pin, MSB first. Simultaneously, received data is shifted through the SPISOMI pin into the LSB of the SPIDAT0 register. When the selected number of bits have been transmitted, the received data in the shift register is transferred to the SPIBUF register for the CPU to read. Data is stored right-justified in SPIBUF.

See Section 14.2.1.1, *Data Sequencing when SPIDAT0 or SPIDAT1 Is Written* on page 631 and Section 14.2.1.2, *Data Sequencing when All Bits Shifted into RXSHIFT Register* on page 631 for details about the data handling for transmit and receive operations.

In slave mode configuration (MASTER = 0 and CLKMOD = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock.

Data written to the SPIDAT0 register is transmitted to the network when the SPICLK signal is received from the network master. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts data on the SPISIMO pin into the RX shift register. If data is to be transmitted by the slave simultaneously, it must be written to the SPIDAT0 register before the beginning of the SPICLK signal.

Figure 14-2. SPI Three-Pin Operation



14.2.3 Operation with $\overline{\text{SPISCS}}$

In master mode, each chip select signal is used to select a specific slave. In slave mode, one chip select signal is used to enable/disable the transfer. Chip-select functionality is enabled by setting one of the $\overline{\text{SPISCS}}$ [7:0] pins as chip selects. It is disabled by setting all $\overline{\text{SPISCS}}$ [7:0] pins as GIOs.

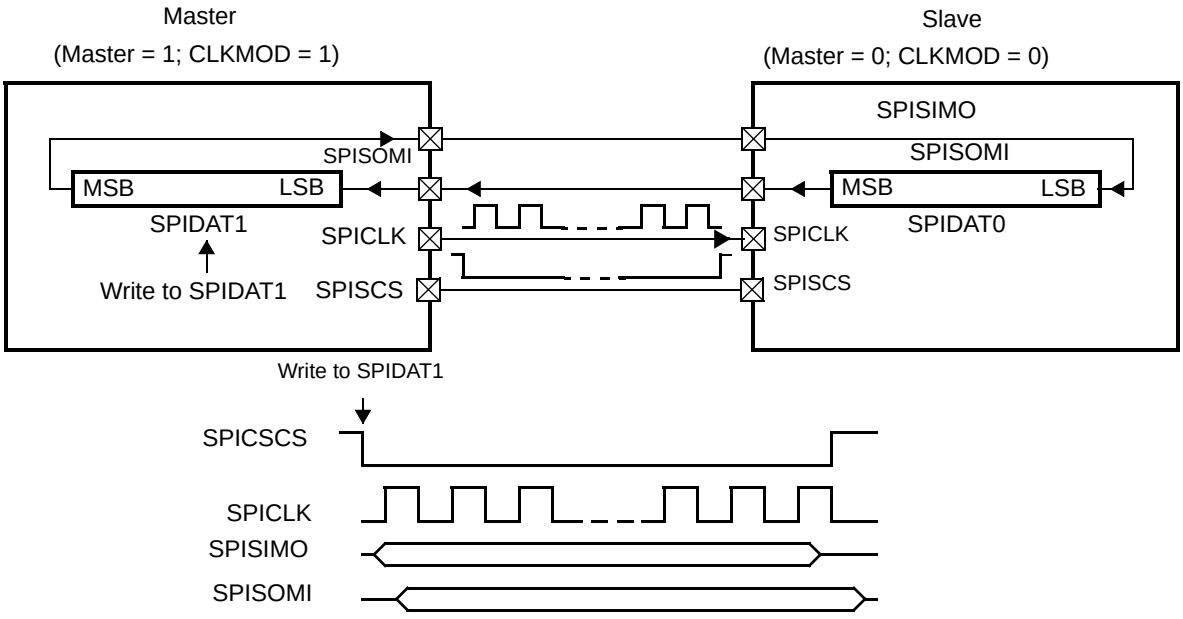
Multiple Chip Selects

The $\overline{\text{SPISCS}}$ [7:0] pins that are used must be configured as functional pins in the SPIPC0[7:0] register. The default pattern to be put on the $\overline{\text{SPISCS}}$ [7:0] when all the slaves are deactivated is set in the SPIDEF register. This pattern allows different slaves with different chip-select polarity to be activated by the SPI.

The master-mode SPI is capable of driving either 0 or 1 as the active value for any $\overline{\text{SPISCS}}$[7:0] output pin. The drive state for $\overline{\text{SPISCS}}$[7:0] pins is controlled by the CSNR field of SPIDAT1. The pattern that is driven will select the slave to which the transmission is dedicated.

In slave mode, the SPI can only be selected by an active value of 0 on any of its selected $\overline{\text{SPISCS}}$ input pin.

Figure 14-3. Operation with $\overline{\text{SPISCS}}$



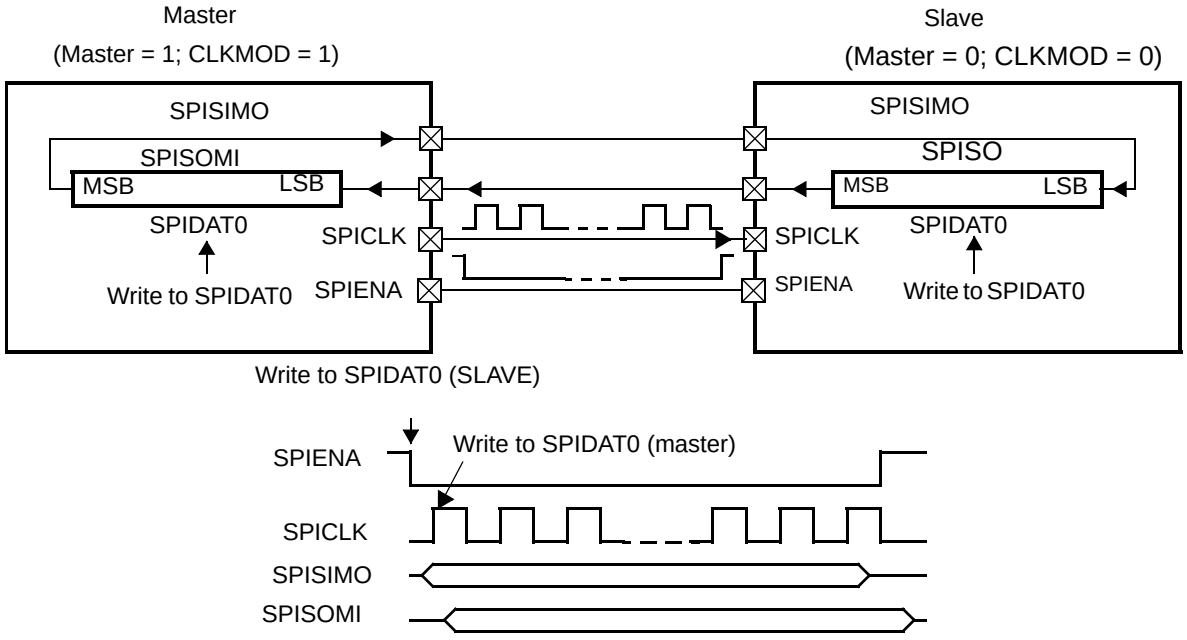
14.2.4 Operation with $\overline{\text{SPIENA}}$

The $\overline{\text{SPIENA}}$ operates as a WAIT signal pin. For both the slave and the master, the $\overline{\text{SPIENA}}$ pin must be configured to be functional (SPIPC0[8] = 1). In this mode, an active low signal from the slave on the $\overline{\text{SPIENA}}$ pin allows the master SPI to drive the clock pulse stream. A high signal tells the master to hold the clock signal (and delay SPI activity).

If the $\overline{\text{SPIENA}}$ pin is in high-z mode (ENABLE_HIGHZ = 1), the slave will put $\overline{\text{SPIENA}}$ into the high-impedance once it completes receiving a new character. If the $\overline{\text{SPIENA}}$ pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave will drive $\overline{\text{SPIENA}}$ to 1 once it completes receiving a new character. The slave will drive $\overline{\text{SPIENA}}$ low again for the next word to transfer, after new data is written to the slave TX shift register.

In master mode (CLKMOD = 1), if the $\overline{\text{SPIENA}}$ pin is configured as functional, then the pin will act as an input pin. If configured as a slave SPI and as functional, the $\overline{\text{SPIENA}}$ pin acts as an output pin.

Figure 14-4. Operation with $\overline{\text{SPIENA}}$

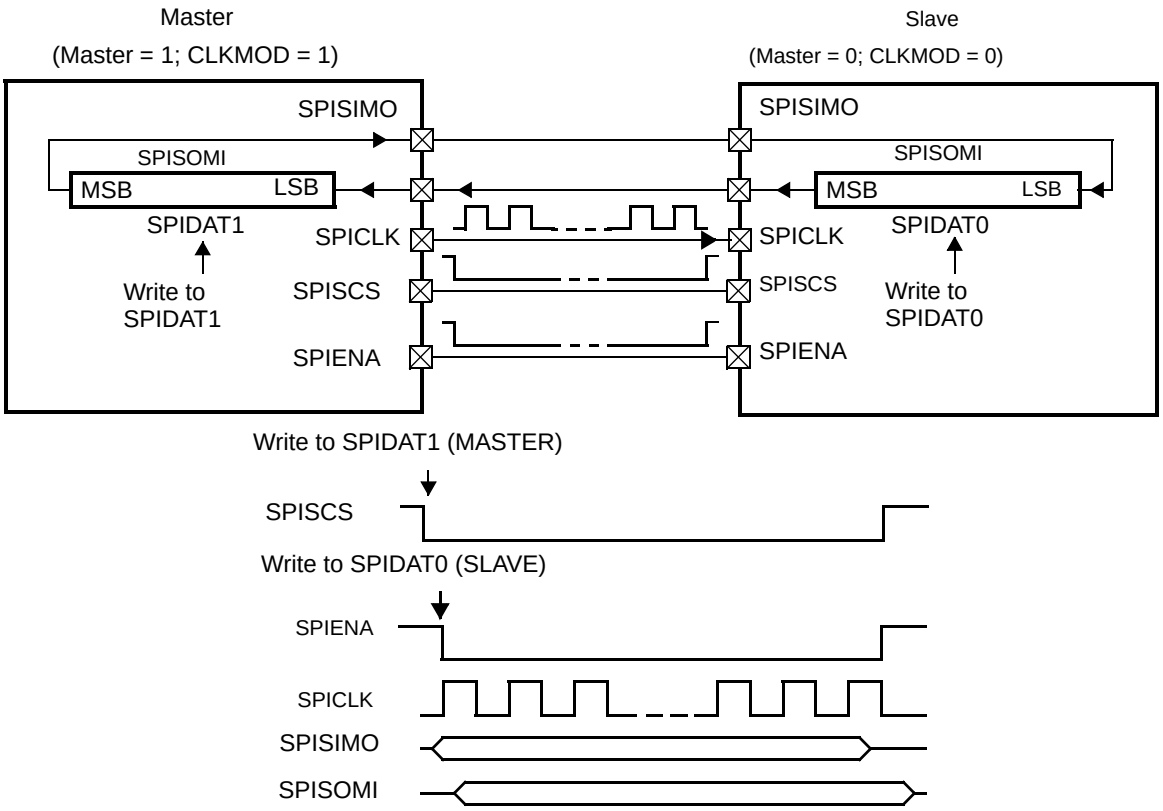


Note:
During a transfer, if a slave-mode SPI detects a deassertion of its chip select before its internal character length counter overflows, then it places SPISOMI and $\overline{\text{SPIENA}}$ (if ENABLE_HIGHZ bit is set to 1) in high-z mode. Once this condition has occurred, if a SPICLK edge is detected while the chip select is deasserted, then the SPI stops that transfer and sets an error flag DLENERR (data length) and generates an interrupt (if enabled).

14.2.5 Five-Pin Operation (Hardware Handshaking)

Five-pin operation combines the functionality of three-pin mode, plus the enable pin and one or more chip select pins. The result is full hardware handshaking. To use this mode, both the SPIENA pin and the required number of SPISCS [7:0] pins must be configured as functional pins.

Figure 14-5. SPI Five-Pin Option with SPIENA and SPISCS



If the SPIENA pin is in high-z mode (ENABLE_HIGHZ = 1), the slave SPI will put this signal into the high-impedance state by default. The slave will drive the signal SPIENA low when new data is written to the slave shift register and the slave has been selected by the master (SPISCS is low).

If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave SPI drives this pin high by default when it is in functional mode. The slave SPI will drive the SPIENA signal low when new data is written to the slave shift register (SPIDAT0/SPIDAT1) and the slave is selected by the master (SPISCS is low). If the slave is deselected by the master (SPISCS goes high), the slave SPIENA signal is driven high.

Note:
Push-pull mode of the SPIENA pin can be used only when there is a single slave in the system. When multiple SPI slave devices are connected to the common SPIENA pin, all of the slaves should configure their SPIENA pins in high-Z mode.

In master mode, if the SPISCS pins are configured as functional pins, then the pins will be in output mode. A write to the master's SPIDAT1/SPIDAT0 register will automatically drive the SPISCS signals low. The master will drive the SPISCS signals high again after completing the transfer of the bits of the data.

In slave mode (CLKMOD = 0), the SPISCS pins will act as functional inputs.

14.2.6 Data Formats

CHARLEN[4:0] specifies the number of bits (2 to 16) in the data word. The CHARLEN[4:0] value directs the state control logic to count the number of bits received or transmitted to determine when a complete word is transferred.

Data word length **must** be programmed to the same length for both the **master** and the **slave**. However, when chip selects are used, there may be multiple targets with different lengths in the system.

Note:
Data must be right-justified when it is written to the SPI for transmission irrespective of its character length or word length.

Figure 14-6 shows how a 12-bit word (0xEC9) needs to be written to the transmit buffer to be transmitted correctly.

Figure 14-6. Format for Transmitting an 8-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	x	x	x	x	1	1	0	0	1	0	0	1

Note:
The received data is always stored right-justified regardless of the character length or direction of shifting and is padded with leading 0s when the character length is less than 16.

Figure 14-7 shows how a 10-bit word (0x0A2) is stored in the buffer once it is received.

Figure 14-7. Format for Receiving an 8-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

To support multiple different types of slaves in one SPI network, four independent data word formats are implemented that allow configuration of individual data word length, polarity, phase, and bit rate. Each word transmitted can select which data format to use via the bits DFSEL[1:0] in its control field from one of the four data word formats.

Data formats 0, 1, 2, and 3 can be configured through control registers.

Each SPI data format includes the standard SPI data format with enhanced features:

- Individually-configurable shift direction can be used to select MSB first or LSB first, whereas the position of the MSB depends on the configured data word length.
- Receive data is automatically right-aligned, independent of shift direction and data word length. Transmit data has to be written right-aligned into the SPI and the internal shift register will transmit according to the selected shift direction and data word length for correct transfer.
- To increase fault detection of data transmission and reception, an odd or even parity bit can be added at the end of a data word. The parity generator can be enabled or disabled individually for each data format. If a received parity bit does not match with the locally calculated parity bit, the parity error flag (PARITYERR) is set and an interrupt is asserted (if enabled).
- Since the master-mode SPI can drive two consecutive accesses to the same slave, an 8-bit delay counter is available to satisfy the delay time for data to be refreshed in the accessed slave. The delay counter can be programmed as part of the data format.

14.2.7 Clocking Modes

SPICLK may operate in four different modes, depending on the choice of phase (delay/no delay) and the polarity (rising edge/falling edge) of the clock.

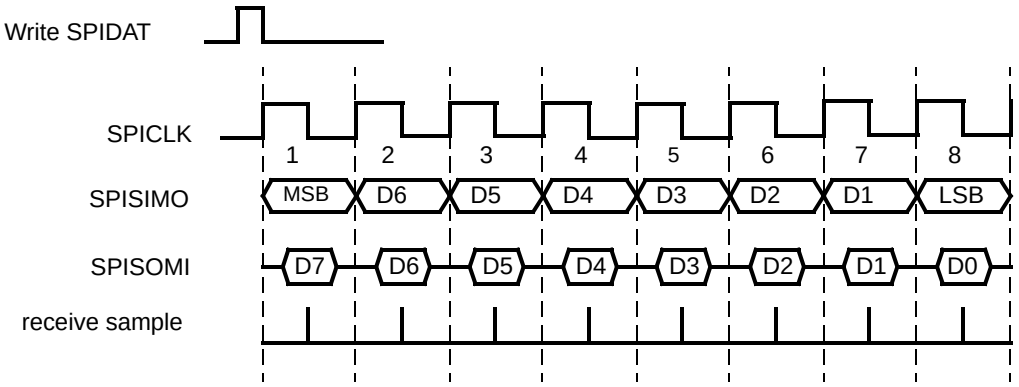
The data input and output edges depend on the values of both POLARITY and PHASE as shown in Table 14-2.

Table 14-2. Clocking Modes

POLARITY	PHASE	ACTION
0	0	Data is output on the rising edge of SPICLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK.
1	0	Data is output on the falling edge of SPICLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.

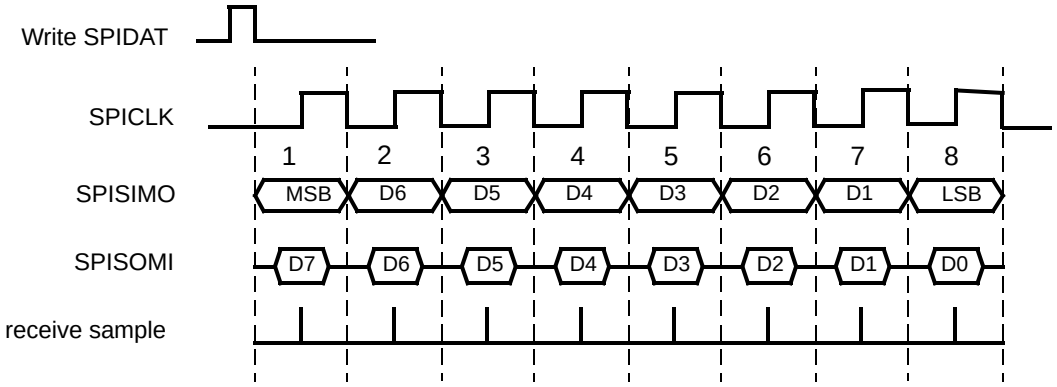
Figure 14-8 to Figure 14-11 illustrate the four possible configurations of SPICLK corresponding to each mode. Having four signal options allows the SPI to interface with many different types of serial devices.

Figure 14-8. Clock Mode with Polarity = 0 and Phase = 0



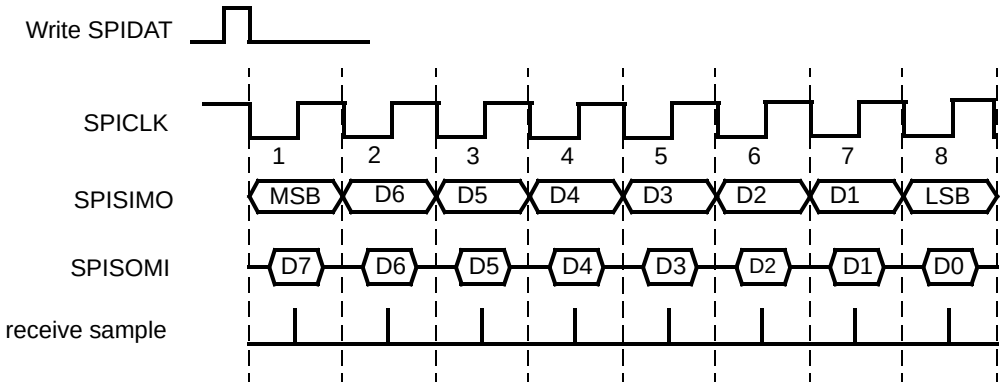
1 Data is output on the rising edge of SPICLK.
2 Input data is latched on the falling edge of SPICLK.

Figure 14-9. Clock Mode with Polarity = 0 and Phase = 1



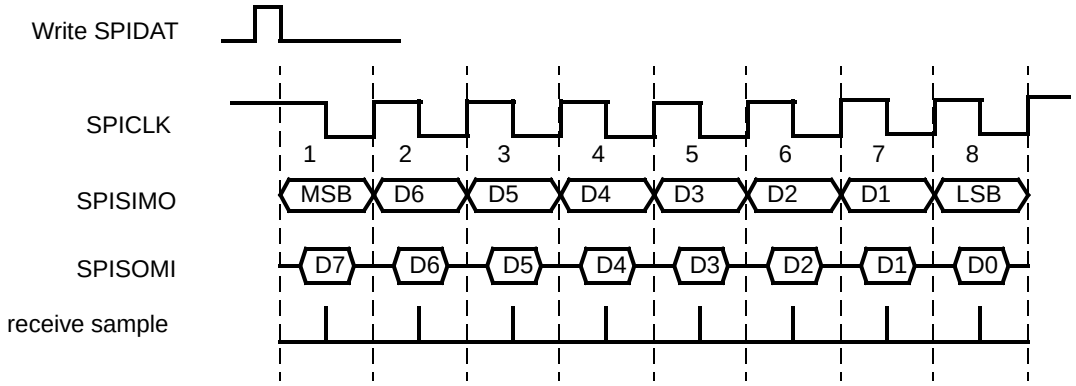
1 Data is output one-half cycle before the first rising edge of SPICLK and on subsequent falling edges of SPICLK
2 Input data is latched on the rising edge of SPICLK

Figure 14-10. Clock Mode with Polarity = 1 and Phase = 0



1 Data is output on the falling edge of SPICLK.
2 Input data is latched on the rising edge of SPICLK.

Figure 14-11. Clock Mode with Polarity = 1 and Phase = 1

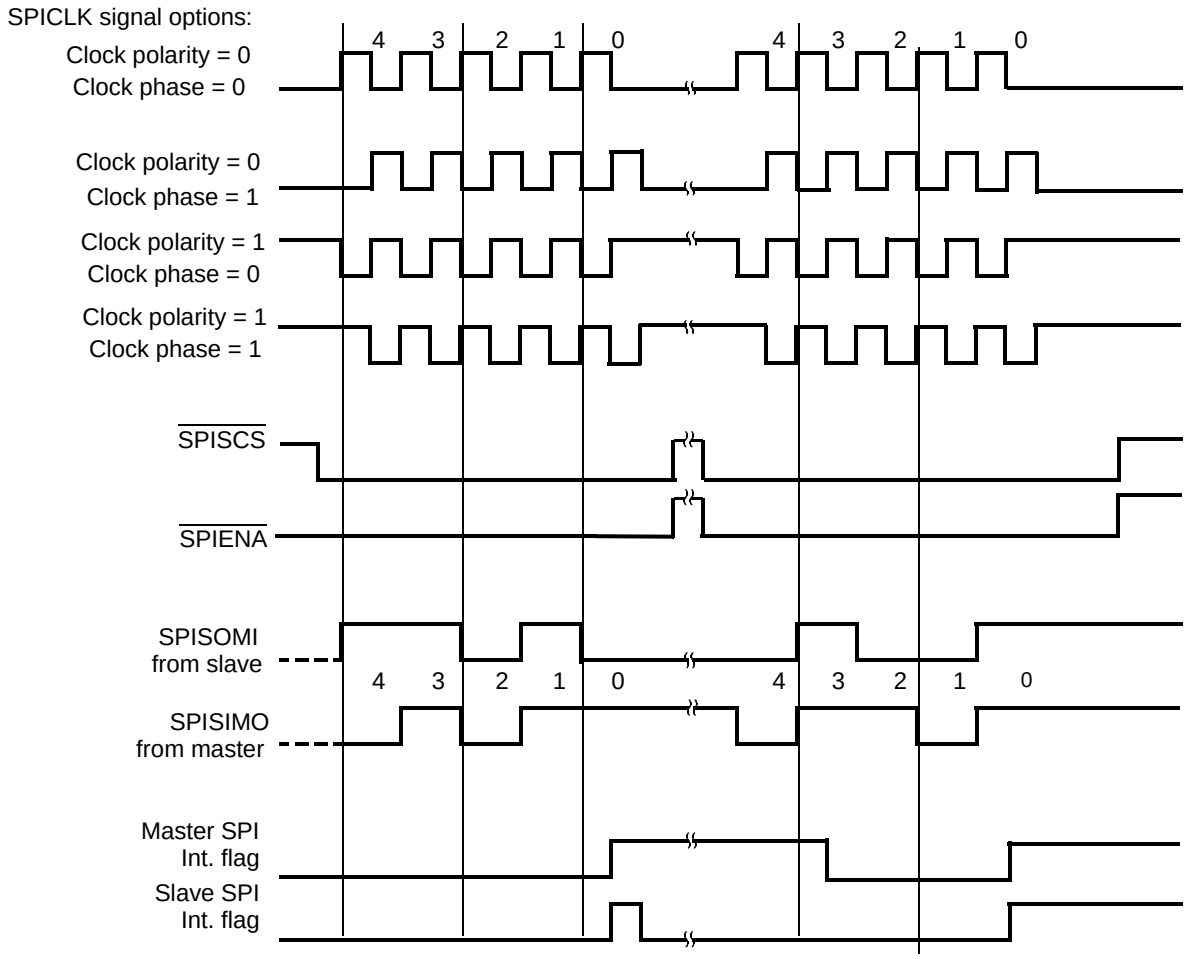


1 Data is output one-half cycle before the first falling edge of SPICLK and on the subsequent rising edges of SPICLK.
2 Input data is latched on the falling edge of SPICLK.

14.2.8 Data Transfer Example

Figure 14-12 illustrates a SPI data transfer between two devices using a character length of five bits.

Figure 14-12. Five Bits per Character (5-Pin Option)



14.2.9 Decoded and Encoded Chip Select (Master Only)

The SPI can connect to up to 8 individual slave devices using decoded chip-selects by routing one wire to each slave. For the decoded mode, the 8 chip selects in the control field are directly connected to the 8 pins. The default value of each chip select, i.e., not active, can be configured via the register CSDEF. During a transmission, the value of the chip select control field (CSNR[7:0]) of the SPIDAT1 register (SPIDAT1[23:16]) is driven on the SPISCS [7:0] pins (if available). When the transmission finishes the default chip-select value (defined by the CSDEF register) is put on the SPISCS [7:0] pins.

The SPI can support more than 8 slaves by using encoded chip selects. To connect the SPI with encoded slaves devices, the CSNR field allows multiple active SPISCS pins at the same time, which enables binary encoded chip selects from 0 to 255. To use encoded chip selects, all eight chip select lines have to be connected to each slave device and each slave needs to have a unique chip-select address. The CSDEF register is used to provide the address at which slaves devices are all de-selected.

Users can combine decoded and encoded chip selects. For example, *n* SPISCS pins can be used for encoding a *n*-bit address and the remaining pins can be connected to decoded-mode slaves.

14.2.10 Variable Chip Select Setup and Hold Timing (Master Only)

In order to support slow slave devices a delay counter can be configured to delay data transmission after the chip select is activated. A second delay counter can be configured to delay the chip select deactivation after the last data bit is transferred. Both delay counters are clocked with VCLK.

If a particular data format specifically does not require these additional set-up or hold times for the chip select pin(s), then they can be disabled in the corresponding SPIFMTx register.

14.2.11 Hold Chip-Select Active

Some slave devices require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers.

CSHOLD is programmable in both master and slave modes of the multi-buffer mode of SPI. However, the meaning of CSHOLD in master mode and slave mode are different.

Note:
If the CSHOLD bit is set within the current data control field, the programmed hold time and the following programmed set-up time will not be applied between transactions.

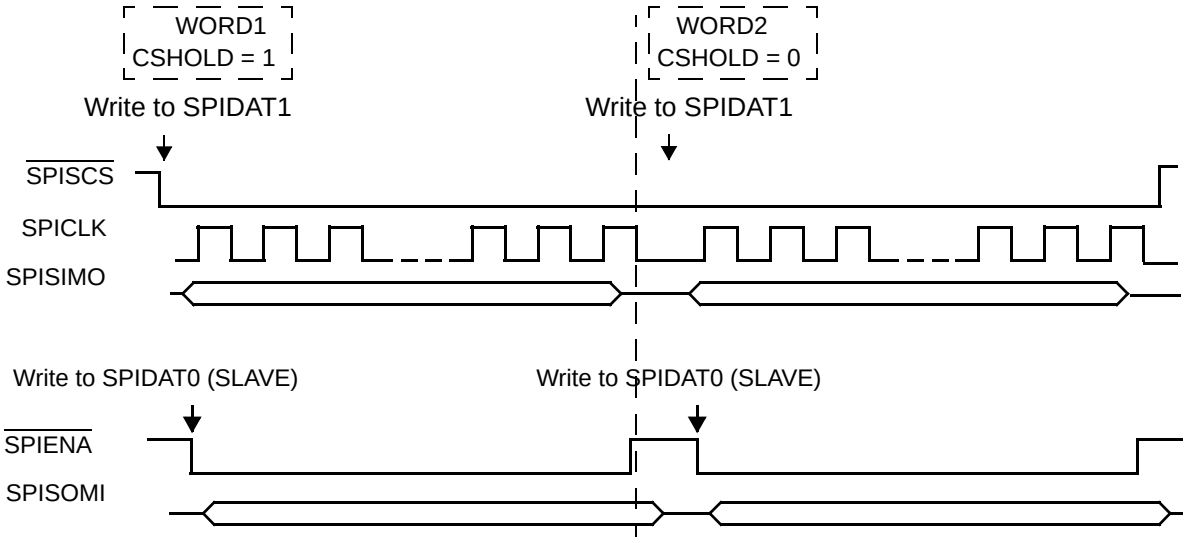
14.2.11.4 The CSHOLD Bit in Master Mode

Each word in a master-mode SPI can be individually initialized for one of the two modes via the CSHOLD bit in its control field.

If the CSHOLD bit is set in the control field of a word, the chip select signal will not be deactivated until the next control field is loaded with new chip select information. Since the chip-select is maintained active between two transfers, the chip-select hold delay (T2CDELAY) is not applied at the end of the current transaction, and the chip-select set-up time delay (C2TDELAY) is not applied as well at the beginning of the following transaction. However, the wait delay (WDELAY) will be still applied between the two transactions, if the WDEL bit is set within the control field.

Figure 14-13 shows the SPI pins when a master-mode SPI transfers a word that has its CSHOLD bit set. The chip-select pins will not be deasserted after the completion of this word. If the next word to transmit has the same chip-select number (CSNR) value, the chip select pins will be maintained until the completion of the second word, regardless of whether the CSHOLD bit is set or not.

Figure 14-13. Typical Diagram when a Buffer in Master is in CSHOLD Mode (SPI-SPI)



14.2.12 Detection of Slave Desynchronization (Master Only)

When a slave supports generation of an enable signal (ENA), de-synchronization can be detected. With the enable signal a slave indicates to the master that it is ready to exchange data. A de-synchronization can occur if one or more clock edges are missed by the slave. In this case the slave may block the SOMI line until it detects clock edges corresponding to the next data word. This would corrupt the data word of the de-synchronized slave and the consecutive data word. A configurable 8 bit time-out counter, which is clocked with SPICLK, is implemented to detect this slave malfunction. After the transmission has finished (end of last bit transferred: either last data bit or parity bit) the counter is started. If the ENA signal generated by the slave does not become inactive before the counter overflows, the DESYNC flag is set and an interrupt is asserted (if enabled).

Note: Inconsistency of Desync Flag in Compatibility Mode MibSPI.

Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is due to the fact that receive completion flag/interrupt will be generated when the buffer transfer is completed. But desync will be detected after the buffer transfer is completed. So, if VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.

This inconsistency in the desync flag is valid only in compatibility mode of MibSPI. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.

14.2.13 ENA Signal Time-Out (Master Only)

The SPI in master mode waits for the hardware handshake signal (ENA) coming from the addressed slave before performing a data transfer. To avoid stalling the SPI by a non-responsive slave device a time-out value can be configured. If the time-out counter overflows before an active ENA signal is sampled the TIMEOUT flag in the status register SPIFLG is set and the TIMEOUT flag in the status field of the corresponding buffer is set.

Note: When the chip select signal becomes active, no breaks in transmission are allowed. The next arbitration is performed while waiting for the time-out to occur.

14.2.14 Data-Length Error

A SPI can generate an error flag by detecting any mismatch in length of received or transmitted data and the programmed character length under certain conditions.

Data-Length Error in Master Mode: During a data transfer, if the SPI detects a de-assertion of the $\overline{\text{SPIENA}}$ pin (by the slave) while the character counter is not overflowed, then an error flag is set to indicate a data-length error. This can be caused by a slave receiving extra clocks (e.g. due to noise on the SPICLK line).

Note: In a master mode SPI, the data length error will be generated only if the $\overline{\text{SPIENA}}$ pin is enabled as a functional pin.

Data-Length Error in Slave Mode: During a transfer, if the SPI detects a de-assertion of the $\overline{\text{SPISCS}}$ pin before its character length counter overflows, then an error flag is set to indicate a data-length error. This situation can arise If the slave SPI misses one or more SPICLK pulses from the master. This error in slave mode implies that both the transmitted and received data were not complete.

Note: In a slave-mode SPI, the data-length error flag will be generated only if at least one of the $\overline{\text{SPISCS}}$(x) pins are configured as functional, and are being used for selecting the slave.

14.2.15 Continuous Self-Test (Master/Slave)

During data transfer, the SPI compares its own internal transmit data with its transmit data on the bus. The sample point for the compare is at one-half SPI clock after transmit point. If the data on the bus does not match the expected value, the bit-error (BITERR) flag is set and an interrupt is asserted if enabled.

Note: The compare is made from the output pin using its input buffer.

14.3 Test Features

14.3.1 Internal Loop-Back Test Mode (Master Only)

The internal loop-back self-test mode can be utilized to test the SPI transmit and receive paths, including the shift registers, the SPI buffer registers, and the parity generator. In this mode the transmit signal is internally fed back to the receiver, whereas the SIMO, SOMI, and CLK pin are disconnected, i.e., the transmitted data is internally transferred to the corresponding receive buffer while external signals remain unchanged.

This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

Note: This mode cannot be changed during transmission.

14.3.2 I/O Loopback Test Mode

I/O Loopback Test mode supports the testing of all I/Os without the aid of an external interface. Loopback can be configured as either analog-loopback (loopback through the pin-level input/output buffers) or digital loopback (internal to the SPI module). With I/O Loopback, all functional features of the SPI can be tested. Transmit data is fed back through the receive-data line(s). See Figure 14-14 for a diagram of the types of feedback available. The IOLPBKTSTCR register defines all of the available control fields.

In loopback mode, it is also possible to induce various error conditions. See Section 14.7.39, *I/O-Loopback Test Control Register (IOLPBKTSTCR)* on page 738 for details of the register field controlling these features.

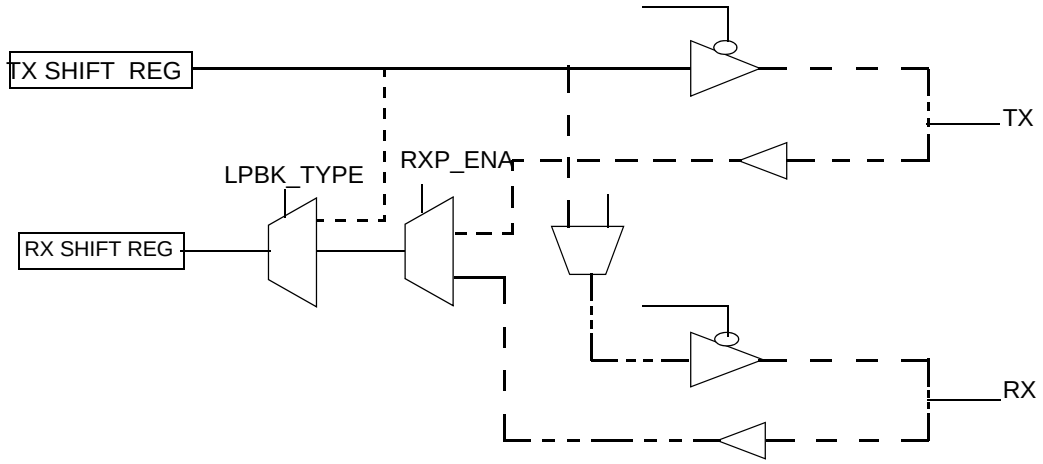
In I/O loopback test modes, even when the module is in slave mode, the SPICLK is generated internally. This SPICLK is used for all loopback-mode SPI transactions. Slave-mode features can be tested without the help of another master SPI, using the internally-generated SPICLK. Chip selects are also generated by the slave itself while it is in I/O loopback mode.

In I/O loopback test modes, if the module is in master mode, the nENA signal is also generated by internal logic so that an external interface is not required.

Note: Usage Guideline for I/O Loopback

I/O Loopback mode should be used with caution because, in some configurations, even the receive pins will be driven with transmit data. During testing, it should be ensured that none of the SPI pins are driven by any other device connected to them. Otherwise, if analog loopback is selected in I/O Loopback mode, then testing may damage the device.

Figure 14-14. I/O Paths during I/O Loopback Modes



Checks the analog loopback path through the receive buffer

Checks the analog loopback path through the transmit buffer

Digital loopback path

1 This diagram is intended to illustrate loopback paths and therefore may omit some normal-mode paths.

14.3.2.1 IO Loopback Mode Operation in Slave Mode

In multi-buffer slave mode, there are some additional requirements for using I/O loopback mode (IOLPBK). In multi-buffer slave mode, the chip-select pins are the triggers for various TGs. Enabling the IOLPBK mode by writing 0xA to the IOLPBTSTENA bits of the IOLPBKTSTCR register triggers TG0 by driving SPISCS[7:0] to 0x0. The actual number of chip selects can be programmed to have any or all of the SPISCS pins as functional. All other configurations should be completed before enabling the IOLPBK mode in multi-buffer slave mode since it triggers TG0.

After the first buffer transfer is completed, the CSNR[3:0] field of the current buffer is used to trigger the next buffer. So, if multiple TGs are desired to be tested, then the CSNR field of the final buffer in each TG should hold the number of the next TG to be triggered. As long as TG boundaries are well defined and are enabled, the completion of one TG will trigger the next TG.

To stop the transfer in multi-buffer slave mode in I/O Loopback configuration, either IOLPBK mode can be disabled by writing 0x5 to the IOLPBTSTENA bits or all of the TGs can be disabled.

14.4 General-Purpose I/O

All of the SPI pins may be programmed via the SPIPC control registers to be either functional or general-purpose I/O pins.

If the SPI function is to be used, application software must ensure that at least the SPICLK pin and the SOMI and/or SIMO pins are configured as SPI functional pins, and not as GIO pins, or else the SPI state machine will be held in reset, preventing SPI transactions.

14.5 Low-Power Mode

The SPI can be put into either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SPI. During global low-power mode, all clocks to the SPI are turned off, making the module completely inactive.

Local low-power mode is asserted by setting the POWERDOWN (SPIGCR1[8]) bit; setting this bit stops the clocks to the SPI internal logic and registers. Setting the POWERDOWN bit causes the SPI to enter local low-power mode and clearing the POWERDOWN bit causes SPI to exit from local low-power mode. All registers remain accessible during local power-down mode, since the clock to the SPI registers is temporarily re-enabled for each access. RAM buffers are also accessible during low power mode.

Note: Since entering a low-power mode has the effect of suspending all state-machine activities, care must be taken when entering such modes to ensure that a valid state is entered when low-power mode is active. Application software must ensure that a low power mode is not entered during a data transfer.

14.6 Interrupts

There are two levels of vectorized interrupts supported by the SPI. These interrupts can be caused under the following circumstances:

- Transmission error
- Receive overrun
- Receive complete (receive buffer full)
- Transmit buffer empty

These interrupts may be enabled or disabled via the SPIINT0 register.

During transmission, if one of the following errors occurs: BITERR, DESYNC, DLENERR, PARITYERR, or TIMEOUT, the corresponding bit in the SPIFLG register is set. If the corresponding enable bit is set, then an interrupt is generated. The level of all the above interrupts is set by the bit fields in the SPILVL register.

The error interrupts are enabled and prioritized independently from each other, but the interrupt generated will be the same if multiple errors are enabled on the same level. The SPIFLG register should be used to determine the actual cause of an error.

Note:
Since there are two interrupt lines, one each for Level 0 and Level 1, it is possible for a programmer to separate out the interrupts for receive buffer full and transmit buffer empty. By programming one to Level 0 and the other to Level 1, it is possible to avoid a check on whether an interrupt occurred for transmit or for receive.

A programmer can also choose to group all of the error interrupts into one interrupt line and both TX-empty and RX-full interrupts into another interrupt line using the LVL control register. In this way, it is possible to separate error-checking from normal data handling.

14.6.1 Interrupts in Multi-Buffer Mode

In multi-buffer mode, the SPI can generate interrupts on two levels.

In normal multi-buffer operation, the receive and transmit are not used and therefore the enable bits of SPIINT0 are not used.

The interrupts available in multi-buffer mode are:

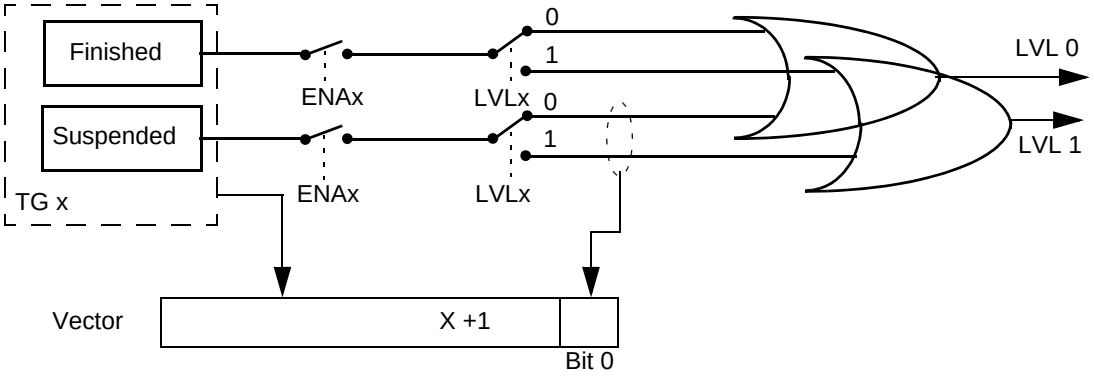
- Transmission error interrupt
- Receive overrun interrupt
- TG suspended interrupt
- TG completed interrupt

When a TG has finished and the corresponding enable bit in the TGINTENA register is set, a transfer-finished interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

When a TG is suspended by a buffer that has been set as suspend to wait until TXFULL flag or/and RXEMPTY flag are set, and if the corresponding bit in the TGINTENA register is set, an transfer-suspended interrupt is generated. The level of priority of the interrupt is determined by the corresponding bit in the TGINTLVL register.

Figure 14-15 illustrates the TG interrupts.

Figure 14-15. TG Interrupt Structure

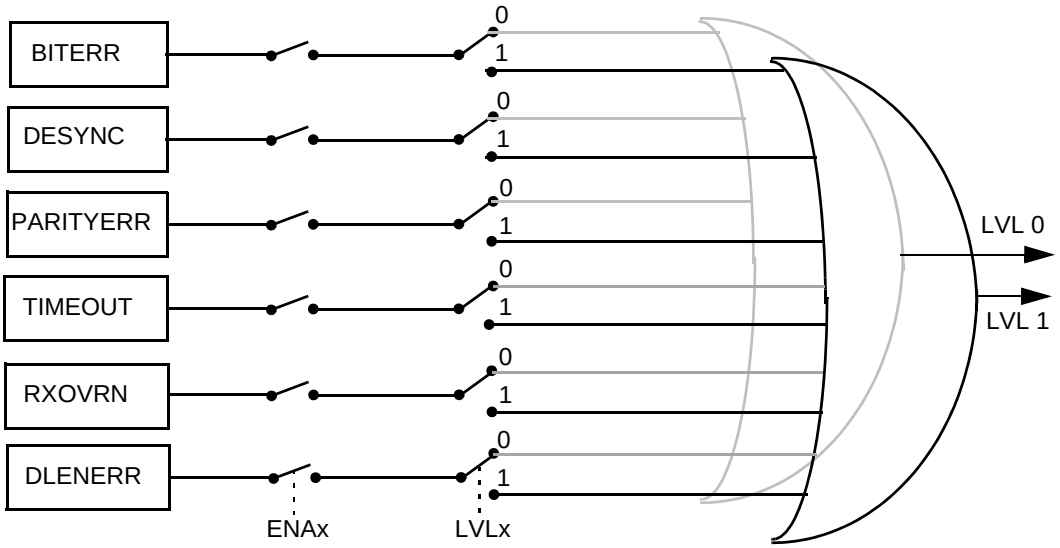


During transmission, if one of the following errors occurs, BITERR, DESYNC, PARITYERR, TIMEOUT, DLENERR, the corresponding flag in the SPIFLG register is set. If the enable bit is set, then an interrupt is generated. The level of the interrupts could be generated according to the bit field in SPILVL.

The RXOVRN interrupt is generated when a buffer in the RXRAM is overwritten by a new received word. While writing newly received data to a RXRAM location, if the RXEMPTY bit of the corresponding location is 0, then the RXOVR bit will be set to 1 during the write operation, so that the buffer starts to indicate an overrun. This RXOVR flag is also reflected in SPIFLG register as RXOVRNINTFLG and the corresponding vector number is updated in TGINTVECT0/TGINTVECT1 register. If an overrun interrupt is enabled, then an interrupt will be generated indicating an overrun condition.

The error interrupts are enabled and prioritized independently from each other, but the vector generated by the SPI will be the same if multiple errors are enabled on the same level.

Figure 14-16. SPIFLG Interrupt Structure



Since the priority of an error interrupt is lower than a completion/suspend interrupt for a TG, the interrupts can be split into two levels. By programming all the error interrupts into Level 0 and TG-complete / TG-suspend interrupts into Level 1, it is possible to get a clear indication of the source of error interrupts. However, when a vector register shows an error interrupt, the actual buffer for which the error has occurred is not readily identifiable. Since each buffer in the multi-buffer RAM is stored along with its individual status flags, each buffer should be read until a buffer with any error flag set is found.

14.7 Control Registers

This section describes the SPI control, data, and pin registers. The registers support 8-bit, 16-bit and 32-bit writes. The offset is relative to the associated base address of this module in a system. The base address for the control registers is 0xFFF7 F400.

Table 14-3. SPI Registers

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00h SPIGCR0 page 662	Reserved															
	Reserved															nRESET
0x04h SPIGCR1 page 663	Reserved							SPIEN	Reserved							LOOP BACK
	Reserved							POWER DOWN	Reserved						CLK MOD	MAS-TER
0x08h SPIINT0 page 665	Reserved							ENABLE HIGH Z	Reserved							DMA REQ EN
	Reserved						TXINT ENA	RXINT ENA	Reserved	OVRN INT ENA	Reserved	BIT ERR ENA	DESYNC ENA	PAR ERR ENA	TIME OUT ENA	DLEN ERR ENA
0x0Ch SPILVL page 668	Reserved															
	Reserved						TX INT LVL	RX INT LVL	Reserved	OVRN INT LVL	Reserved	BITERR LVL	DESYNC LVL	PAR ERR LVL	TIME OUT LVL	DLEN ERR LVL
0x10h SPIFLG page 670	Reserved							BUF INIT ACTIVE	Reserved							
	Reserved						TX INT FLG	RX INT FLG	Reserved	OVRN INT FLG	Reserved	BIT ERR FLG	DESYNC FLG	PAR ERR FLG	TIME OUT FLG	DLEN ERR FLG
0x14h SPIPC0 page 675	SOMIFUN[7:0]								SIMOFUN[7:0]							
	Reserved					SOMI FUN	SIMO FUN	CLK FUN	ENA FUN	SCSFUN[7:0]						
0x18h SPIPC1 page 677	SOMIDIR[7:0]								SIMODIR[7:0]							
	Reserved					SOMI DIR	SIMO DIR	CLK DIR	ENA DIR	SCSDIR[7:0]						

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x1Ch SPIPC2 page 679	SOMIDIN[7:0]								SIMODIN[7:0]							
	Reserved				SOMI DIN	SIMO DIN	CLK DIN	ENA DIN	SCSDIN[7:0]							
0x20h SPIPC3 page 681	SOMIDOUT[7:0]								SIMODOUT[7:0]							
	Reserved				SOMI DOUT	SIMO DOUT	CLK DOUT	ENA DOUT	SCSDOUT[7:0]							
0x24h SPIPC4 page 683	SOMISET[7:0]								SIMOSET[7:0]							
	Reserved				SOMI SET	SIMO SET	CLK SET	ENA-SET	SCSSET[7:0]							
0x28h SPIPC5 page 685	SOMICLR[7:0]								SIMOCLR[7:0]							
	Reserved				SOMI CLR	SIMO CLR	CLK CLR	ENA CLR	SCSCLR[7:0]							
0x2Ch SPIPC6 page 687	SOMIPDR[7:0]								SOMICLR[7:0]							
	Reserved				SOMI PDR	SIMO PDR	CLK PDR	ENA PDR	SCSPDR[7:0]							
0x30h SPIPC7 page 689	SOMIDIS[7:0]								SIMODIS[7:0]							
	Reserved				SOMI PDIS	SIMO PDIS	CLK PDIS	ENA PDIS	SCSPDIS[7:0]							
0x34h SPIPC8 page 691	SOMIPSEL[7:0]								SIMOPSEL[7:0]							
	Reserved				SOMI PSL	SIMO PSL	CLK PSL	ENA PSL	SCSPSL[7:0]							
0x38h SPIDAT0 page 693	Reserved															
	TXDATA(15–0)															

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x3Ch SPIDAT1 page 694	Reserved			CS HOLD	Reserved	WDEL	DFSEL		CSNR(7–0)							
	TXDATA(15–0)															
0x40h SPIBUF page 696	RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLENERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x44h SPIEMU page 700	Reserved															
	RXDATA(15–0)															
0x48h SPIDELAY page 701	C2TDELAY(7–0)								T2CDELAY(7–0)							
	T2EDELAY(7–0)								C2EDELAY(7–0)							
0x4Ch SPIDEF page 705	Reserved															
	Reserved								CSDEF[7:0]							
0x50h–5Ch SPIFMT[0:3] page 706	Reserved		WDELAY[0:3](5–0)						PARPOL[0:3]	PARITY[0:3]ENA	WAITENA[0:3]	SHIFTDIR[0:3]	Reserved	DIS CS TIMERS	POLARITY[0:3]	PHASE[0:3]
	PRESCALE[0:3]								Reserved			CHARLEN[0:3]				
0x60h–64h TGINTVECT[0:1] page 709	Reserved															
	Reserved										INTVECT[0:1]					SUSPEND[0:1]
0x68h SPIPC9 page 713	SOMISRS[7:0]								SIMOSRS[7:0]							
	Reserved				SOMISRS0	SIMOSRS0	CLK-SRS	ENA-SRS	SCSSRS[7:0]							

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x70h MIBSPIE page 715	Reserved															RX RAM ACCESS
	Reserved															MibSPI ENA
0x74h TGITENST page 717	SET INTEN RDY[15:0]															
	SET INTEN SUS[15:0]															
0x78h TGITENCR page 718	CLR INTEN RDY[15:0]															
	CLR INTEN SUS[15:0]															
0x7Ch TGITLVST page 719	SET INTLVL RDY[15:0]															
	SET INTLVL SUS[15:0]															
0x80h TGITLVCR page 720	CLR INTLVL RDY[15:0]															
	CLR INTLVL SUS[15:0]															
0x84h TGITFLG page 721	INTFLG RDY[15:0]															
	INTFLG SUS[15:0]															
0x88–0x8Ch	Reserved															
	Reserved															
0x90h TICKCNT page 723	TICK ENA	RE LOAD	CLKCTRL(1–0)	Reserved												
	TICKVALUE(15–0)															

1 The base address of these registers can be found in the device data sheet.

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x94h LTGPEND page 725	Reserved			TG IN SERVICE(4–0)					Reserved							
	Reserved	LPEND(6–0)							Reserved							
0x98–D4h TG[0:15]CTRL page 727	TG ENA[0:15]	ONE SHOT [0:15]	PRST [0:15]	TGTD[0:15]	Reserved				TRGEVT[0:15](4–0)				TRIGSRC[0:15](4–0)			
	Reserved	PSTART[0:15](6–0)							Reserved	PCURRENT[0:15](6–0)						
0x11Ch	Reserved															
	Reserved															
0x120h UERRCTRL page 733	Reserved															
	Reserved							PTEST -EN	Reserved				EDEN(3–0)			
0x124h UERRSTAT page 734	Reserved															
	Reserved														EDFLG1	EDFLG0
0x128h UERRADDR1 page 735	Reserved															
	Reserved							UERRADDR1(9–0)								
0x12Ch UERRADDR0 page 736	Reserved															
	Reserved							UERRADDR0(8–0)								
0x130h RXOVRN_BUF_ADDR page 737	Reserved															
	Reserved							RXOVRN_BUF_ADDR(9–0)								

1 The base address of these registers can be found in the device data sheet.

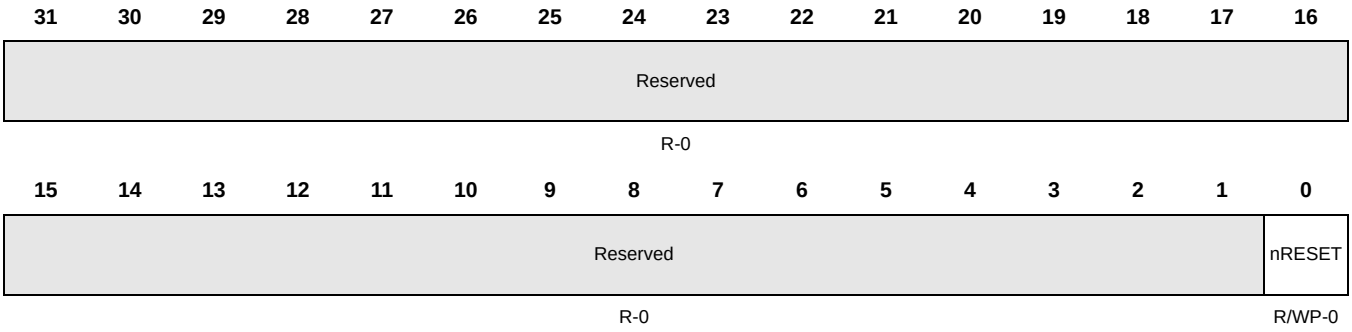
Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x134h IOLPBKTSTCR page 738	Reserved							SCS FAIL FLG	Reserved			CTRL_ BITERR	CTRL_ DESYNC	CTRL_ PAR ERR	CTRL_ TIME OUT	CTRL_ DLEN ERR
	Reserved				IOLPBKTSTENA(3–0)			Reserved	ERR SCS PIN(2–0					CTRL SCS PIN ERR	LPBK_ TYPE	RXP_ ENA

1 The base address of these registers can be found in the device data sheet.

Note:
TI highly recommends that write values corresponding to the reserved locations of registers be maintained as 0 consistently. This allows future enhancements to use these reserved bits as control bits without affecting the functionality of the module with any older versions of software.

14.7.1 SPI Global Control Register 0 (SPIGCR0)

Figure 14-17. SPI Global Control Register 0 (SPIGCR0) [offset = 00h]



R = Read in all modes; WP = Write in privilege mode only; -n = value after reset

Table 14-4. SPI Global Control Register 0 (SPIGCR0) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved		Reads return zero and writes have no effect.
0	nRESET		Reset bit for the module. This bit needs to be set to 1 before any operation on SPI can be done. After setting this bit, auto Initialization of multi-buffer RAM starts.
		0	SPI is in the reset state.
		1	SPI is out of the reset state.

14.7.2 SPI Global Control Register 1 (SPIGCR1)

Figure 14-18. SPI Global Control Register 1 (SPIGCR1) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SPIEN	Reserved							LOOP-BACK
R-0							R/W-0	R-0							R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							POWER-DOWN	Reserved						CLK-MOD	MAS-TER
R-0							R/W-0	R-0						R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; WP = Write in privilege mode only; -n = value after reset

Table 14-5. SPI Global Control Register 1 (SPIGCR1) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	SPIEN		SPI enable. This bit enables SPI transfers. This bit must be set to 1 after all other SPI configuration bits have been written. When the SPIEN bit is 0 or cleared to 0, the following SPI registers get forced to their default states: •Both TX and RX shift registers •The TXDATA fields of SPIDAT0 and SPIDAT1 registers •All the fields of the SPIFLG register •Contents of SPIBUF and the internal RXBUF registers
		0	The SPI is not activated for transfers.
		1	Activates SPI
23–17	Reserved		Reads return zero and writes have no effect.
16	LOOPBACK		Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPISIMO and SPISOMI pins are configured with SPI functionality, then the SPISIMO pin is internally connected to the SPISOMI pin (transmit data is looped back as receive data). GIO mode for these pins is not supported in loopback mode. Externally, during loop-back operation, the SPICLK pin outputs an inactive value and SPISOMI remains in the high-impedance state. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result. **Note: This loopback mode can only be used in master mode. Master mode must be selected before setting LOOPBACK. When this mode is selected, the CLKMOD bit should be set to 1, meaning that SPICLK is internally generated.**
		0	Internal loop-back test mode disabled.

Table 14-5. SPI Global Control Register 1 (SPIGCR1) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Internal loop-back test mode enabled.
15–9	Reserved		Reads return zero and writes have no effect.
8	POWERDOWN		When active, the SPI state machine enters a power-down state.
		0	The SPI is in active mode.
		1	The SPI is in power-down mode.
7–2	Reserved		Reads return zero and writes have no effect.
1	CLKMOD		Clock mode. This bit selects either an internal or external clock source. This bit also determines the I/O direction of the SPIENA and SPISCS[7:0] pins in functional mode.
		0	Clock is external. •SPIENA is an output. •SPISCS[7:0] are inputs.
		1	Clock is internally-generated. •SPIENA is an output. •SPISCS[7:0] are outputs.
0	MASTER		SPISIMO/SPISOMI pin direction determination. Sets the direction of the SPISIMO and SPISOMI pins. **Note: For master-mode operation of the SPI, MASTER bit should be set to 1 and CLKMOD bit can be set either 1 or 0. The master-mode SPI can run on an external clock on SPICLK.** **For slave mode operation, both the MASTER and CLKMOD bits should be set to 0. Any other combinations may result in unpredictable behavior of the SPI. In slave mode. SPICLK will not be generated internally in slave mode.**
		0	SPISIMO pin is an input, SPISOMI pin is an output
		1	SPISOMI pin is an input, SPISIMO pin is an output

14.7.3 SPI Interrupt Register (SPIINT0)

Figure 14-19. SPI Interrupt Register (SPIINT0) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ENABLE-HIGHZ	Reserved							DMAREQEN
R-0							R/W-0	R-0							R/W-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						TXINTENA	RXINT-ENA	Reserved	RXOVRNIN-TENA	Reserved	BITER-RENA	DESYN-CENA	PAR-ERRENA	TIME-OUT-ENA	DLEN-ERRENA
R-0						R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

R = Read, W = write, P = Privilege mode; -*n* = Value after reset

Table 14-6. SPI Interrupt Register (SPIINT0) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	ENABLEHIGHZ		$\overline{\text{SPIENA}}$ pin high-impedance enable. When active, the $\overline{\text{SPIENA}}$ pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to high-impedance when not driving a low signal. If inactive, then the pin will output both a high and a low signal.
		0	$\overline{\text{SPIENA}}$ pin is pulled high when not active.
		1	$\overline{\text{SPIENA}}$ pin remains high-impedance when not active.
23–17	Reserved		Reads return zero and writes have no effect.
16	DMAREQEN		DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Enable DMA REQ only after setting the SPIEN bit to 1.
		0	DMA is not used.
		1	DMA requests will be generated.
			Note: A DMA request will be generated on the TX DMA REQ line each time a word is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1 writes.
			Note: A DMA request will be generated on the RX DMA REQ line each time a word is copied to the SPIBUF register either from RXBUF or directly from the shift register.
15–10	Reserved		Reads return zero and writes have no effect.

Table 14-6. SPI Interrupt Register (SPIINT0) Field Descriptions (Continued)

Bit	Name	Value	Description
9	TXINTENA		Causes an interrupt to be generated every time data is written to the shift register, so that the next word can be written to TXBUF. Setting this bit will generate an interrupt if the TXINTFLG bit (SPIFLG[9]) is set to 1.
		0	No interrupt will be generated upon TXINTFLG being set to 1.
		1	An interrupt will be generated upon TXINTFLG being set to 1.
			The transmitter empty interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
			Note: An interrupt request will be generated as soon as this bit is set to 1. By default it will be generated on the INT0 line. The SPILVL register can be programmed to change the interrupt line.
8	RXINTENA		Causes an interrupt to be generated when the RXINTFLAG bit (SPI-FLG[8]) is set by hardware.
		0	Interrupt will not be generated.
		1	Interrupt will be generated.
			The receiver full interrupt is valid in compatibility mode of SPI only. In multi-buffered mode, this interrupts will not be generated, even if it is enabled.
7	Reserved		Reads return zero and writes have no effect.
6	RXOVRNINTENA		Overrun interrupt enable.
		0	Overrun interrupt will not be generated.
		1	Overrun interrupt will be generated.
5	Reserved		Reads return zero and writes have no effect.
4	BITERRENA		Enables interrupt on bit error.
		0	No interrupt asserted upon bit error.
		1	Enables an interrupt on a bit error.
3	DESYNCENA		Enables interrupt on desynchronized slave. DESYNCENA is used in master mode only.
		0	No interrupt asserted upon desynchronization error.
		1	An interrupt is asserted on desynchronization of the slave (DESYNC = 1).
2	PARERRENA		Enables interrupt-on-parity-error.
		0	No interrupt asserted on parity error.

Table 14-6. SPI Interrupt Register (SPIINT0) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	An interrupt is asserted on a parity error.
1	TIMEOUTENA		Enables interrupt on ENA signal time-out.
		0	No interrupt asserted upon ENA signal time-out.
		1	An interrupt is asserted on a time-out of the ENA signal.
0	DLENERRENA		Data length error interrupt enable. A data length error occurs under the following conditions. **Master:** When SPIENA is used, if the SPIENA pin from the slave is deasserted before the master has completed its transfer, the data length error is set. That is, if the character length counter has not overflowed while SPIENA deassertion is detected, then it means that the slave has neither received full data from the master nor has it transmitted complete data. **Slave:** When SPISCS pins are used, if the incoming valid SPISCS pin is deactivated before the character length counter overflows, then the data length error is set.
		0	No interrupt is generated upon data length error.
		1	An interrupt is asserted when a data-length error occurs.

14.7.4 SPI Interrupt Level Register (SPILVL)

Figure 14-20. SPI Interrupt Level Register (SPILVL) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						TXINTLVL	RX INTLVL	Reserved	RXOVRNIN-TLVL	Reserved	BITER-RLVL	DESYN-CLVL	PAR-ERRLVL	TIME-OUTLVL	DLEN-ERRLVL
R-0						R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

R = Read, W = Write in any mode; *-n* = Value after reset

Table 14-7. SPI Interrupt Level Register (SPILVL) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
9	TXINTLVL		Transmit interrupt level.
		0	Transmit interrupt is mapped to interrupt line INT0.
		1	Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL		Receive interrupt level.
		0	Receive interrupt is mapped to interrupt line INT0.
		1	Receive interrupt is mapped to interrupt line INT1.
7	Reserved		Reads return zero and writes have no effect.
6	RXOVRNINTLVL		Receive overrun interrupt level.
		0	Receive overrun interrupt is mapped to interrupt line INT0.
		1	Receive overrun interrupt is mapped to interrupt line INT1.
5	Reserved		Reads return zero and writes have no effect.
4	BITERRLVL		Bit error interrupt level.
		0	Bit error interrupt is mapped to interrupt line INT0.
		1	Bit error interrupt is mapped to interrupt line INT1.
3	DESYNCLVL		Desynchronized slave interrupt level. (master mode only).
		0	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT0.

Table 14-7. SPI Interrupt Level Register (SPILVL) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	An interrupt caused by desynchronization of the slave is mapped to interrupt line INT1.
2	PARERRLVL		Parity error interrupt level.
		0	A parity error interrupt is mapped to interrupt line INT0.
		1	A parity error interrupt is mapped to interrupt line INT1.
1	TIMEOUTLVL		$\overline{\text{SPIENA}}$ pin time-out interrupt level.
		0	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT0.
		1	An interrupt on a time-out of the ENA signal (TIMEOUT = 1) is mapped to interrupt line INT1.
0	DLEN ERR LVL		Data length error interrupt level (line) select.
		0	An interrupt on data length error is mapped to interrupt line INT0.
		1	An interrupt on data length error is mapped to interrupt line INT1.

14.7.5 SPI Flag Register (SPIFLG)

Figure 14-21. SPI Flag Register (SPIFLG) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							BUFINI-TAC-TIVE	Reserved							
R-0							R-0	R-0							
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TXINT-FLG	RXINT-FLG	Reserved	RXOVRNINT-FLG	Reserved	BITER-RFLG	DESYN-CFLG	PAR-ERRFLG	TIME-OUT-FLG	DLEN-ERRFLG
R-0						R-0	R/WC-0	R-0	R/WC-0	R-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0

R = Read; WC = Write/read Clear; -*n* = Value after reset

Table 14-8. SPI Flag Register (SPIFLG) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	BUFINITACTIVE		Indicates the status of multi-buffer initialization process. Software can poll for this bit to determine if it can proceed with the register configuration of multi-buffer mode registers or buffer handling. **Note: If the SPIFLG register is read while the multi-buffer RAM is being initialized, the BUF INIT ACTIVE bit will be read as 1. If SPIFLG is read after the internal automatic buffer initialization is complete, this bit will be read as 0. This bit will show a value of 1 as long as the nRESET bit is 0, but does not really indicate that buffer initialization is underway. Buffer initialization starts only when the nRESET bit is set to 1.**
		0	Multi-buffer RAM initialization is complete.
		1	Multi-buffer RAM is still being initialized. Do not attempt to write to either multi-buffer RAM or any multi-buffer mode registers.
23–10	Reserved		Reads return zero and writes have no effect.

Table 14-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
9	TXINTFLG		Transmitter-empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new word can be written to it. This flag is set when a word is copied to the shift register either directly from SPIDAT0/SPIDAT1 or from the TXBUF register. This bit is cleared by one of following methods: • Writing a new data to either SPIDAT0 or SPIDAT1 • Writing a 0 to SPIEN (SPIGCR1[24])
		0	Transmit buffer is now full. No interrupt pending for transmitter empty.
		1	Transmit buffer is empty. An interrupt is pending to fill the transmitter.
8	RXINTFLG		Receiver-full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). If RXINTEN is enabled, an interrupt is also generated. This bit is cleared under the following methods: • Reading the SPIBUF register • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive buffer full interrupt • Writing a 1 to this bit • Writing a 0 to SPIEN (SPIGCR1[24]) • System reset During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit.
		0	No new received data pending. Receive buffer is empty.
		1	A newly received data is ready to be read. Receive buffer is full. **Note: Clearing RXINTFLG bit by writing a 1 before reading the SPIBUF sets the RXEMPTY bit of the SPIBUF register too. In this way, one can ignore a received word. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF and the RXEMPTY bit will be cleared again. The SPIBUF contents should be read first if this situation needs to be avoided.**
7	Reserved		Reads return zero and writes have no effect.

Table 14-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
6	RXOVRNINTFLG		Receiver overrun flag. The SPI hardware sets this bit when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. The SPI will generate an interrupt request if this bit is set and the RXOVRN INTEN bit (SPIINT0.6) is set high. This bit is cleared under the following conditions in compatibility mode of MibSPI: • Reading TGINTVECT0 or TGINTVECT1 register when there is a receive-buffer-overrun interrupt • Writing a 1 to RXOVRNINTFLG in the SPIFLG register itself • Writing a 0 to SPIEN • Reading the data field of the SPIBUF register **Note: Reading the SPIBUF register does not clear this RXOVRNINTFLG bit. If an RXOVRN interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF.** **Note: There is a special condition under which the RXOVRNIN-TFLG flag gets set. If both SPIBUF and RXBUF are already full and while another reception is underway, if any errors (e.g. TIMEOUT, BITERR and DLEN_ERR) occur, then RXOVR in RXBUF and RXOVRNINTFLG in SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a receive overrun.** In multi-buffer mode of MibSPI, this bit is cleared under the following conditions. • Reading the RXOVRN_BUF_ADDR register • Writing a 1 to RXOVRNINTFLG in the SPIFLG register itself In multi-buffer mode, if RXOVRNINTFLG is set, then the address of the buffer which experienced the overrun is available in RXOVRN_BUF_ADDR.
		0	Overrun condition did not occur.
		1	Overrun condition has occurred.
5	Reserved		Reads return zero and writes have no effect.

Table 14-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
4	BITERRFLG		Mismatch of internal transmit data and transmitted data. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set the SPIENA bit to 0.
		0	No bit error occurred.
		1	A bit error occurred. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag BITERRFLG is set. If BITERRENA is set an interrupt is asserted. Possible reasons for a bit error can be an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
3	DESYNCFLG		Desynchronization of slave device. Desynchronization monitor is active in master mode only.
		0	No slave desynchronization detected.
		1	A slave device is desynchronized. The master monitors the ENAble signal coming from the slave device and sets the DESYNC flag after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.
2	PARITYERRFLG		Calculated parity differs from received parity bit. If the parity generator is enabled (can be selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word the parity generator calculates the reference parity and compares it to the received parity bit. In the event of a mismatch the PARITYERR flag is set and an interrupt is asserted if PARERRENA is set.
		0	No parity error detected.
		1	A parity error occurred. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.

Table 14-8. SPI Flag Register (SPIFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
1	TIMEOUTFLG		Time-out caused by nonactivation of ENA signal.
		0	No ENA-signal time-out occurred.
		1	An ENA signal time-out occurred. The SPI generates a time-out because the slave hasn't responded in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition the TIMEOUT flag in the status field of the corresponding buffer is set. The transmit request of the concerned buffer is cleared, i.e. the SPI doesn't re-start a data transfer from this buffer. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0.
0	DLEN ERR FLG		Data-length error flag. This flag can be cleared by one of the following methods. • Write a 1 to this bit. • Set SPIENA bit to 0. **Note: Whenever any transmission errors (TIMEOUT, BITERR, DLEN_ERR, PARITY_ERR, DESYNC) are detected and the error flags are cleared by writing to the error bit in the SPIFLG register, the corresponding error flag in SPIBUF does not get cleared. Software needs to read the SPIBUF until it becomes empty before proceeding. This ensures that all of the old status bits in SPIBUF are cleared before starting the next transfer.**
		0	No data length error has occurred.
		1	A data length error has occurred.

14.7.6 SPI Pin Control Register 0 (SPIPC0)

Figure 14-22. SPI Pin Control Register 0 (SPIPC0) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMIFUN	Reserved							SIMO-FUN
R-0							R/W-0	R-0							R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMI-FUN	SIMO-FUN	CLKFUN	ENAFUN	SCSFUN[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = write, -*n* = Value after reset

Table 14-9. SPI Pin Control (SPIPC0) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIFUN		Slave out, master in function. Determines whether SPISOMI is to be used as a general-purpose I/O pin or as a SPI functional pin. **Note: Duplicate Control Bits for SPISOMI.** **Bit 24 is not physically implemented. It is a mirror of Bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPISOMI pin. The read value of Bit 24 always reflects the value of bit 11.**
		0	SPISOMI pin is a GIO pin.
		1	SPISOMI pin is a SPI functional pin.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOFUN		Slave in, master out function. Determines whether SPISIMO is to be used as a general-purpose I/O pin or as a SPI functional pin. **Note: Duplicate Control Bits for SPISIMO.** **Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI pin. The read value of Bit 16 always reflects the value of bit 10.**
		0	The SPISIMO pin is a GIO pin.
		1	The SPISIMO pin is a SPI functional pin
15–12	Reserved		Reads return zero and writes have no effect.

Table 14-9. SPI Pin Control (SPIPC0) Field Descriptions (Continued)

Bit	Name	Value	Description
11	SOMIFUN		Slave out, master in function. This bit determines whether the SPI-SOMI pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The SPISOMI pin is a GIO pin.
		1	The SPISOMI pin is a SPI functional pin.
10	SIMOFUN		Slave in, master out function. This bits determine whether each SPISIMO pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The SPISIMO pin is a GIO pin.
		1	The SPISIMO pin is a SPI functional pin.
9	CLKFUN		SPI clock function. This bit determines whether the SPICLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	The SPICLK pin is a GIO pin.
		1	The SPICLK pin is a SPI functional pin.
8	ENAFUN		$\overline{\text{SPIENA}}$ function. This bit determines whether the $\overline{\text{SPIENA}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	The $\overline{\text{SPIENA}}$ pin is a GIO pin.
		1	The $\overline{\text{SPIENA}}$ pin is a SPI functional pin.
7-0	SCSFUN[7:0]		$\overline{\text{SPISCSx}}$ function. Determines whether each $\overline{\text{SPISCSx}}$ pin is to be used as a general-purpose I/O pin or as a SPI functional pin. If the slave $\overline{\text{SPISCSx}}$ pins are in functional mode and receive an inactive high signal, the slave SPI will place its output in high-z and disable shifting.
		0	The $\overline{\text{SPISCSx}}$ pin is a GIO pin.
		1	The $\overline{\text{SPISCSx}}$ pin is a SPI functional pin.

14.7.7 SPI Pin Control Register 1 (SPIPC1)

Figure 14-23. SPI Pin Control Register 1 (SPIPC1) [offset = 18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMIDIR	Reserved							SIMO-DIR
R-0							R/W-0	R-0							R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIDIR	SIMO-DIR	CLKDIR	ENADIR	SCSDIR[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = Write; -*n* = Value after reset

Table 14-10. SPI Pin Control Register (SPIPC1) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIDIR		SPISOMI direction. Controls the direction of SPISOMI when used for general-purpose I/O. If SPISOMI pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register. **Note: Duplicate Control Bits for SPISOMI.** **Bit 24 is not physically implemented. It is a mirror of Bit 11. Any write to bit 24 will be reflected on bit 11. When bit 24 and bit 11 are simultaneously written, the value of bit 11 will control the SPISOMI pin. The read value of Bit 24 always reflects the value of bit 11.**
31-25	Reserved		Reads return zero and writes have no effect.
		0	SPISOMI pin is an input.
		1	SPISOMI pin is an output.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMODIR		SPISIMO direction. Controls the direction of SPISIMO when used for general-purpose I/O. If SPISIMO pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register. **Note: Duplicate Control Bits for SPISIMO.** **Bit 16 is not physically implemented. It is a mirror of Bit 10. Any write to bit 16 will be reflected on bit 10. When bit 16 and bit 10 are simultaneously written, the value of bit 10 will control the SPISOMI pin. The read value of Bit 16 always reflects the value of bit 10.**
		0	SPISIMO pin is an input.

Table 14-10. SPI Pin Control Register (SPIPC1) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	SPISIMO pin is an output.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDIR		SPISOMI direction. This bit controls the direction of the SPISOMI pin when it is used as a general-purpose I/O pin. If the SPISOMI pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISOMI pin is an input.
		1	SPISOMI pin is an output.
10	SIMODIR		SPISIMO direction. This bit controls the direction of the SPISIMO pin when it is used as a general-purpose I/O pin. If the SPISIMO pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit in the SPIGCR1 register.
		0	SPISIMO pin is an input.
		1	SPISIMO pin is an output
9	CLKDIR		SPICLK direction. This bit controls the direction of the SPICLK pin when it is used as a general-purpose I/O pin. In functional mode, the I/O direction is determined by the CLKMOD bit.
		0	SPICLK pin is an input.
		1	SPICLK pin is an output.
8	ENADIR		$\overline{\text{SPIENA}}$ direction. This bit controls the direction of the $\overline{\text{SPIENA}}$ pin when it is used as a general-purpose I/O. If the $\overline{\text{SPIENA}}$ pin is used as a functional pin, then the I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPIENA}}$ pin is an input.
		1	$\overline{\text{SPIENA}}$ pin is an output.
7–0	SCSDIR[7:0]		$\overline{\text{SPISCSx}}$ direction. These bits control the direction of each $\overline{\text{SPISCSx}}$ pin when it is used as a general-purpose I/O pin. Each pin could be configured independently from the others If the $\overline{\text{SPISCSx}}$ is used as a SPI functional pin; the I/O direction is determined by the CLKMOD bit (SPIGCR1[1]).
		0	$\overline{\text{SPISCSx}}$ pin is an input.
		1	$\overline{\text{SPISCSx}}$ pin is an output.

14.7.8 SPI Pin Control Register 2 (SPIPC2)

Figure 14-24. SPI Pin Control Register 2 (SPIPC2) [offset = 1Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMIDIN	Reserved							SIMO-DIN
R-0							R-U	R-0							R-U

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIDIN0	SIMODIN0	CLKDIN	ENADIN	SCSDIN[7:0]							
R-0				R-U	R-U	R-U	R-U	R-U							

R = Read; W = Write; U = Undefined; -*n* = Value after reset

Table 14-11. SPI Pin Control Register 2 (SPIPC2) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
	SOMIDIN		SPISOMI data in. The value of the SPISOMI pin.
		0	The SPISOMI pin is logic 0.
		1	The SPISOMI pin is logic 1.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMODIN		SPISIMO data in. The value of the SPISIMO pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDIN		SPISOMI data in. The value of the SPISOMI pin.
		0	The SPISOMI pin is logic 0.
		1	The SPISOMI pin is logic 1.
10	SIMODIN		SPISIMO data in. The value of the SPISIMO pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.
9	CLKDIN		Clock data in. The value of the SPICLK pin.
		0	The SPICLK pin is logic 0.
		1	The SPICLK pin is logic 1

Table 14-11. SPI Pin Control Register 2 (SPIPC2) Field Descriptions (Continued)

Bit	Name	Value	Description
8	ENADIN		$\overline{\text{SPIENA}}$ data in. The the value of the $\overline{\text{SPIENA}}$ pin.
		0	The $\overline{\text{SPIENA}}$ pin is logic 0.
		1	The $\overline{\text{SPIENA}}$ pin is logic 1
7–0	SCSDIN[7:0]		$\overline{\text{SPISCSx}}$ data in. The value of the $\overline{\text{SPISCSx}}$ pins.
		0	The $\overline{\text{SPISCSx}}$ pin is logic 0.
		1	The $\overline{\text{SPISCSx}}$ pin is logic 1

14.7.9 SPI Pin Control Register 3 (SPIPC3)

Figure 14-25. SPI Pin Control Register 3 (SPIPC3) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMID-OUT	Reserved							SIMOD-OUT
R/W-0							R/W-U	R/W-0							R/W-U

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMID OUT0	SIMOD OUT0	CLKD-OUT	ENADOUT	SCSDOUT[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-U	R/W-U							

R = Read; W = Write; -*n* = Value after reset

Table 14-12. SPI Pin Control Register 3 (SPIPC3) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIDOUT		SPISOMI data out write. This bit is only active when the SPISOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 11 or bit 24 can be used to set the direction for pin SPISOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Current value on SPISOMI pin is logic 0.
		1	Current value on SPISOMI pin is logic 1
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMODOUT		SPISIMO data out write. This bit is only active when the SPISIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin. **Bit 10 or bit 16 can be used to set the direction for pin SPISOMI. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Current value on SPISIMO pin is logic 0.
		1	Current value on SPISIMO pin is logic 1.
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIDOUT		SPISOMI0 data out write. This bit is only active when the SPISOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value on SPISOMI pin is logic 0.
		1	Current value on SPISOMI pin is logic 1.

Table 14-12. SPI Pin Control Register 3 (SPIPC3) Field Descriptions (Continued)

Bit	Name	Value	Description
10	SIMODOUT		SPISIMO data out write. This bit is only active when the SPISIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The SPISIMO pin is logic 0.
		1	The SPISIMO pin is logic 1.
9	CLKDOUT		SPICLK data out write. This bit is only active when the SPICLK pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The SPICLK pin is logic 0.
		1	The SPICLK pin is logic 1.
8	ENADOUT		$\overline{\text{SPIENA}}$ data out write. Only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	The $\overline{\text{SPIENA}}$ pin is logic 0.
		1	The $\overline{\text{SPIENA}}$ pin is logic 1.
7–0	SCSDOUT[7:0]		$\overline{\text{SPISCSx}}$ data out write. Only active when the $\overline{\text{SPISCSx}}$ pins are configured as a general-purpose I/O pins and configured as an output pins. The value of these bits indicates the value sent to the pins.
		0	The $\overline{\text{SPISCSx}}$ pin is logic 0.
		1	The $\overline{\text{SPISCSx}}$ pin is logic 1.

14.7.10 SPI Pin Control Register 4 (SPIPC4)

Figure 14-26. SPI Pin Control Register 4 (SPIPC4) [offset = 24h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMISET	Reserved							SOMISET
R-0							R/W-U	R-0							R/W-U

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMISET0	SIMOSET0	CLKSET	ENASET	SCSSET[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-U	R/W-U							

R = Read; W = Write; -*n* = Value after reset

Table 14-13. SPI Pin Control Register 4 (SPIPC4) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMISET		SPISOMI data out set. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to set the SOMI pin. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	*Read:* SPISIMO is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISOMI is logic 1. *Write:* Logic 1 is placed on SPISOMI pin if it is in general-purpose output mode.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOSET		SPISIMO data out set. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to set the SOMI pin. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	*Read:* SPISIMI is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISIMO is logic 1. *Write:* Logic 1 is placed on SPISIMO pin if it is in general-purpose output mode.
15–12	Reserved		Reads return zero and writes have no effect.

Table 14-13. SPI Pin Control Register 4 (SPIPC4) Field Descriptions (Continued)

Bit	Name	Value	Description
11	SOMISET		SPISOMI data out set. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin.
		0	*Read:* SPISOMI is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISOMI is logic 1. *Write:* Logic 1 is placed on SPISOMI pin if it is in general-purpose output mode.
10	SIMOSET		SPISIMO0 data out set. This pin is only active when the SPISIMO pin is configured as a general-purpose output pin.
		0	*Read:* SPISIMO is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPISIMO is logic 1. *Write:* Logic 1 is placed on SPISIMO pin if it is in general-purpose output mode.
9	CLKSET		SPICLK data out set. This bit is only active when the SPICLK pin is configured as a general-purpose output pin.
		0	*Read:* SPICLK is logic 0. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* SPICLK pin is logic 1. *Write:* Logic 1 is placed on the SPICLK pin if it is in general-purpose output mode.
8	ENASET		$\overline{\text{SPIENA}}$ data out set. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin.
		0	*Read:* SPIENA is logic 0. *Write:* A write to this bit has no effect.
		1	*Read:* SPIENA is logic 1. *Write:* Logic 1 is placed on $\overline{\text{SPIENA}}$ pin if it is in general-purpose O/P mode.
7–0	SCSSET[7:0]		$\overline{\text{SPISCSx}}$ data out set. This bit is only active when the $\overline{\text{SPISCSx}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit sets the corresponding $\overline{\text{SCS}}$DOUT bit to 1.
		0	*Read:* $\overline{\text{SPISCSx}}$ is logic 0. *Write:* A write to this bit has no effect.
		1	*Read:* $\overline{\text{SPISCSx}}$ is logic 1. *Write:* Logic 1 placed on $\overline{\text{SPISCSx}}$ pin if it is in general-purpose output mode.

14.7.11 SPI Pin Control Register 5 (SPIPC5)

Figure 14-27. SPI Pin Control Register 5 (SPIPC5) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMICLR	Reserved							SIMOCLR
R-0							R/W-U	R-0							R/W-U

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMICLR0	SIMOCLR0	CLKCLR	ENACLR	SCSCLR[7:0]							
R-0				R/W-U	R/W-U	R/W-U	R/W-0	R/W-0							

R = Read; W = Write; -*n* = Value after reset

Table 14-14. SPI Pin Control Register 5 (SPIPC5) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMICLR		SPISOMIx data out clear. This pin is only active when the SPISOMI pin is configured as a general-purpose output pin. **Bit 11 or bit 24 can be used to clear the pin SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	*Read:* The current value on SOMIDOUT is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SOMIDOUT is 1. *Write:* Logic 0 is placed on SPISOMI pin if it is in general-purpose output mode.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOCLR		SPISIMO data out clear. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin. **Bit 10 or bit 16 can be used to clear the pin SOMI. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	*Read:* The current value on SIMODOUT is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SIMODOUT is 1. *Write:* Logic 0 is placed on SPISIMO pin if it is in general-purpose output mode.
15–12	Reserved		Reads return zero and writes have no effect.

Table 14-14. SPI Pin Control Register 5 (SPIPC5) Field Descriptions (Continued)

Bit	Name	Value	Description
11	SOMICLR		SPISOMI data out clear. This bit is only active when the SPISOMI pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPISOMI is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPISOMI is 1. *Write:* Logic 0 is placed on SPISOMI pin if it is in general-purpose output mode.
10	SIMOCLR		SPISIMO data out clear. This bit is only active when the SPISIMO pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPISIMO is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPISIMO is 1. *Write:* Logic 0 is placed on SPISIMO pin if it is in general-purpose output mode.
9	CLKCLR		SPICLK data out clear. This bit is only active when the SPICLK pin is configured as a general-purpose output pin.
		0	*Read:* The current value on SPICLK is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on SPICLK is 1. *Write:* Logic 0 is placed on SPICLK pin if it is in general-purpose output mode.
8	ENACLR		$\overline{\text{SPIENA}}$ data out clear. This bit is only active when the $\overline{\text{SPIENA}}$ pin is configured as a general-purpose output pin. A value of 1 written to this bit clears the corresponding ENABLEDOUT bit to 0.
		0	*Read:* The current value on ENA is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on $\overline{\text{ENA}}$ is 1. *Write:* Logic 0 is placed on $\overline{\text{SPIENA}}$ pin if it is in general-purpose output mode.
7–0	SCSCLR[7:0]		$\overline{\text{SPISCSx}}$ data out clear. This bit is only active when the $\overline{\text{SPISCSx}}$ pins are configured as a general-purpose output pins.
		0	*Read:* The current value on SCSDOUTx is 0. Write: A write of 0 to this bit has no effect.
		1	*Read:* The current value on $\overline{\text{SCSDOUT}}$x is 1. *Write:* Logic 0 is placed on the $\overline{\text{SPISCS}}$ pin if it is in general-purpose output mode.

14.7.12 SPI Pin Control Register 6 (SPIPC6)

Figure 14-28. SPI Pin Control Register 6 (SPIPC6) [offset = 2Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMI-PDR[0]	Reserved							SIMO-PDR[0]
R-0							R/W-0	R-0							R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIP DR0	SIMOP DR0	CLKPDR	ENACLR	SCSPDR[7:0]							
R-0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read; W = Write; -*n* = Value after reset

Table 14-15. SPI Pin Control Register 6 (SPIPC6) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIPDR		SPISOMI open drain enable. This bit enables open drain capability for the SPISOMI pin if the following conditions are met: • SOMIDIR = 1 (SPISOMI pin configured in GIO mode as an output) • SOMIDOUT = 1 **Bit 11 or bit 24 can both be used to enable open-drain for SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	The output value on the SPISOMIx pin is logic 1.
		1	Output pin SPISOMIx is in a high-impedance state.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOPDR		SPISIMO open drain enable. This bit enables open drain capability for the SPISIMO pin if the following conditions are met: • SIMODIR = 1 (SPISIMO pin configured in GIO mode as an output) • SIMODOUT = 1 **Bit 10 or bit 16 can both be used to enable open-drain for SIMO. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	The output value on SPISIMOx pin is logic 1.
		1	Output pin SPISIMOx is in a high-impedance state.
15–12	Reserved		Reads return zero and writes have no effect.

Table 14-15. SPI Pin Control Register 6 (SPIPC6) Field Descriptions (Continued)

Bit	Name	Value	Description
11	SOMIPDR		SOMI open-drain enable. This bit enables open-drain capability for SOMI if the following conditions are met. • SOMI pin configured in GIO mode as output pin • Output value on SPISOMI pin is logic 1.
		0	Output value 1 of SPISOMI pin is logic 1.
		1	Output value 1 of SPISOMI is high-impedance.
10	SIMOPDR		SPISIMO open-drain enable. This bit enables open -drain capability for the SPISIMO pin if the following conditions are met. • SIMO pin configured in GIO mode as output pin • Output value on SPISIMO pin is logic 1.
		0	Output value 1 of SPISIMO pin is logic 1.
		1	Output value 1 of SPISIMO is high-impedance.
9	CLKPDR		CLK open drain enable. This bit enables open drain capability for the pin CLK if the following conditions are met: • SPICLK pin configured in GIO mode as an output pin • SPICLKDOUT = 1
		0	Output value on CLK pin is logic 1.
		1	Output pin CLK is in a high-impedance state.
8	ENAPDR		SPIENA pin open drain enable. This bit enables open drain capability for SPIENA if the following conditions are met: • SPIENA pin configured in GIO mode as an output pin • SPIENADOUT = 1
		0	Output value on SPIENA pin is logic 1.
		1	Output pin SPIENA is in a high-impedance state.
7–0	SCSPDR[7:0]		SPISCSx open drain enable. This bit enables open drain capability for the SPISCSx pin if the following conditions are met: • SPISCS pin configured in GIO mode as an output pin • SPISCSDOUTx = 1
		0	Output value on SCSx pin is logic 1.
		1	Output pin SCSx is in a high-impedance state.

14.7.13 SPI Pin Control Register 7(SPIPC7)

Note: Default Register Value
The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 14-29. SPI Pin Control Register 7 (SPIPC7) [offset = 30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMI-DIS[0]	Reserved							SIMO-DIS[0]
R-0							R/W-n	R-0							R/W-n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMIP DIS0	SIMOP DIS0	CLKDIS	ENAPDIS	SCSPDIS[7:0]							
R-0				R/W-n	R/W-n	R/W-n	R/W-n	R/W-n							

R = Read; W = Write; -*n* = Value after reset

Table 14-16. SPI Pin Control Register 7 (SPIPC7) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIDIS		SOMI pull control enable/disable. This bit enables pull control capability for the SOMI pin if it is in input mode regardless of whether it is in functional or GIO mode. **Note: Bit 11 or bit 24 can be used to set pull-disable for SOMI. If a 32-bit write is performed, bit 11 will have priority over bit 24.**
		0	Pull control on the SPISOMI pin is enabled.
		1	Pull control on the SPISOMI pin is disabled.
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMODIS		SIMO pull control enable/disable. This bit enables pull control capability for the SIMO pin if it is in input mode regardless of whether it is in functional or GIO mode. **Note: Bit 10 or bit 16 can be used to set pull-disable for SIMO. If a 32-bit write is performed, bit 10 will have priority over bit 16.**
		0	Pull control on SPISIMO pin is enabled.
		1	Pull control on SPISIMO pin is disabled.
15–12	Reserved		Reads return zero and writes have no effect.

Table 14-16. SPI Pin Control Register 7 (SPIPC7) Field Descriptions (Continued)

Bit	Name	Value	Description
11	SOMIPDIS		SPISOMI pull control enable/disable. This bit enables pull control capability for the pin SPISOMI pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on the SPISOMI pin is enabled.
		1	Pull control on the SPISOMI pin is disabled.
10	SIMOPDIS		SPISIMO pull control enable/disable. This bit enables pull control capability for the pin SPISIMO pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on SPISIMO pin is enabled.
		1	Pull control on SPISIMO pin is disabled.
9	CLKPDIS		CLK pull control enable/disable. This bit enables pull control capability for the pin SPICLK pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on CLK pin is enabled.
		1	Pull control on CLK pin is disabled.
8	ENAPDIS		ENABLE pull control enable/disable. This bit enables pull control capability for the pin SPIENA pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on ENABLE pin is enabled.
		1	Pull control on ENABLE pin is disabled.
7–0	SCSPDIS[7:0]		SCSx pull control enable/disable. This bit enables pull control capability for the pin SPISCSx pin if it is in input mode regardless of whether it is in functional or GIO mode.
		0	Pull control on SCSx pin is enabled.
		1	Pull control on SCSx pin is disabled.

14.7.14 SPI Pin Control Register 8(SPIPC8)

Note: Default Register Value
The default values of these register bits vary by device. See your device datasheet for information about default pin states, which correspond to the register reset values (see the pin-list table).

Figure 14-30. SPI Pin Control Register 8 (SPIPC8) [offset = 34h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SOMI-PSEL[0]	Reserved							SIMO-PSEL[0]
R-0							R/W-n	R-0							R/W-n

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMI-PSEL	SIMO-PSEL	CLKPSEL	ENAPSEL	SCSPSEL[7:0]							
R-0				R/W-n	R/W-n	R/W-n	R/W-n	R/W-n							

R = Read; W = Write; -*n* = Value after reset

Table 14-17. SPI Pin Control Register 8 (SPIPC8) Field Descriptions

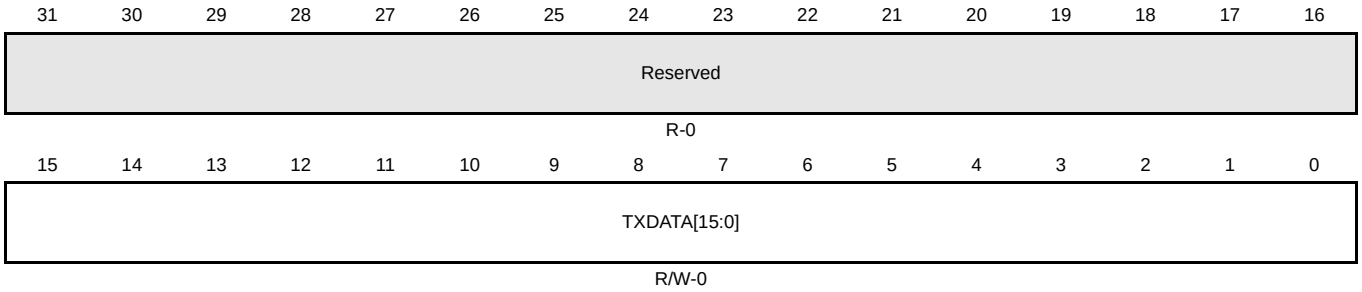
Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	SOMIPSEL		SPISOMI pull select.
		0	Input buffer for SPISOMI is disabled if PULLDIS = 1
		1	Input buffer for SPISOMI is enabled if PULLDIS = 1
23-17	Reserved		Reads return zero and writes have no effect.
16	SIMOPSEL		SPISIMO pull select.
		0	Input buffer for SPISIMO is disabled if PULLDIS = 1
		1	Input buffer for SPISIMO is enabled if PULLDIS = 1
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMIPSEL		SOMI pull select.
		0	Input buffer for SOMI is disabled if PULLDIS = 1
		1	Input buffer for SOMI is enabled if PULLDIS = 1
10	SIMOPSEL		SPISIMO pull select.
		0	Input buffer for SPISIMO is disabled if PULLDIS = 1
		1	Input buffer for SPISIMO is enabled if PULLDIS = 1

Table 14-17. SPI Pin Control Register 8 (SPIPC8) Field Descriptions

Bit	Name	Value	Description
9	CLKPSEL		CLK pull select.
		0	Input buffer for CLK is disabled if PULLDIS = 1
		1	Input buffer for CLK is enabled if PULLDIS = 1
8	ENAPSEL		ENABLE pull select.
		0	Input buffer for ENABLE is disabled if PULLDIS = 1
		1	Input buffer for ENABLE is enabled if PULLDIS = 1
7–0	SCSPSEL[7:0]		SCSx pull select.
		0	Input buffer for SCSx is disabled if PULLDIS = 1
		1	Input buffer for SCSx is enabled if PULLDIS = 1

14.7.15 SPI Transmit Data Register 0 (SPIDAT0)

Figure 14-31. SPI Transmit Data Register 0 (SPIDAT0) [offset = 38h]



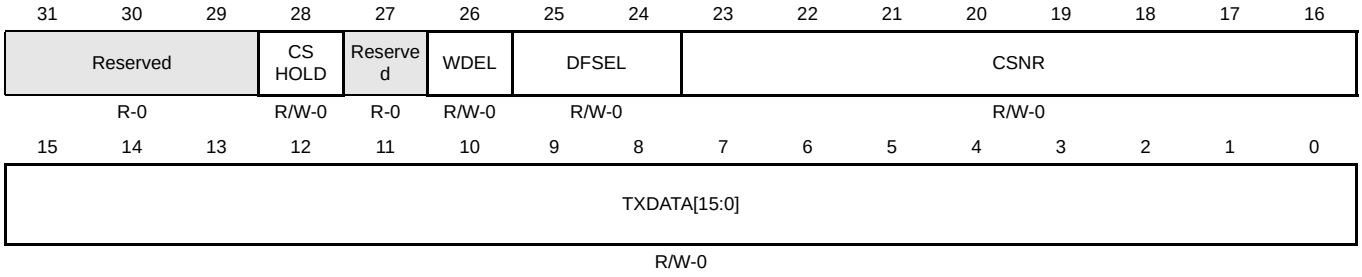
R = Read, W = write, -*n* = Value after reset

Table 14-18. SPI Transmit Data Register 0 (SPIDAT0) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–0	TXDATA(15–0)	0–FFFFh	SPI transmit data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, TXBUF holds the written data. SPIEN (SPICGR1[24]) must be set to 1 before this register can be written to. Writing a 0 to the SPIEN register forces the lower 16 bits of the SPIDAT0 register to 0x00. **Note: When this register is read, the contents TXBUF, which holds the latest written data, will be returned.** **Note: Regardless of character length, the transmit word should be right-justified before writing to the SPIDAT0 register.** **Note: The default data format control register for SPIDAT0 is SPIFMT0. However, it is possible to reprogram the DFSEL[1:0] fields of SPIDAT1 before using SPIDAT0, to select a different SPIFMTx register.**

14.7.16 SPI Transmit Data Register 1 (SPIDAT1)

Figure 14-32. SPI Transmit Data Register 1 (SPIDAT1) [offset = 3Ch]



R = Read, W = write, -*n* = Value after reset

Table 14-19. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions

Bit	Name	Value	Description
31–29	Reserved		Reads return zero and writes have no effect.
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only in compatibility-mode of SPI, (it is ignored in slave mode). CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.
27	Reserved		Reads return zero and writes have no effect.

Table 14-19. SPI Transmit Data Register 1 (SPIDAT1) Field Descriptions (Continued)

Bit	Name	Value	Description
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, $\overline{\text{SPISCS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the SPISCS pin remains deactivated depends upon the time taken to supply a new word after completing the shift operation. If TXBUF is already full, then the SPISCS will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPISCS}}$ pins will be de-activated for at least (WDELAY + 2) * VCLK_Period duration.
25–24	DFSEL		Data word format select
		00	Data word format 0 is selected
		01	Data word format 1 is selected
		10	Data word format 2 is selected
		11	Data word format 3 is selected
23–16	CSNR	0–FFh	Chip select number. CSNR defines the chip-select that will be activated during the data transfer. **Note: Writing to only the control field does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL fields in the control field to select the required phase/polarity combination.**
15–0	TXDATA(15–0)	0–FFFFh	**Transfer data.** When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of SPIDAT1 to 0x0000. Write to this register ONLY when using the automatic slave chip-select feature (see Section 14.2, *Operating Modes* on page 630 for more information). A write to this register will drive the contents of CSNR[7:0] on the $\overline{\text{SPISCS}}$[7:0] pins, if they are configured as functional pins. When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

14.7.17 SPI Receive Buffer Register (SPIBUF)

Figure 14-33. SPI Receive Buffer Register (SPIBUF) [offset = 40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLEN-ERR	LCSNR							
R-1	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA[15:0)]															
R-0															

R = Read, W = write, C = Clear; S = Set; -*n* = Value after reset

Table 14-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions

Bit	Name	Value	Description
31	RXEMPTY		**Receive data buffer empty.** When the host reads the SPIBUF field or the entire SPIBUF register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, and the RXEMPTY flag is cleared.
		0	New data has been received and copied into the SPIBUF field.
		1	No data has been received since the last read of SPIBUF. This flag gets set to 1 under the following conditions: • Reading the RXDATA portion of the SPIBUF register. • Writing a 1 to clear the RXINTFLG bit in the SPIFLG register. Write-clearing the RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA portion of SPIBUF (or the entire register).

Table 14-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

Bit	Name	Value	Description
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to SPIBUF; the contents of SPIBUF are overwritten only after it is read by the VBUSP master (e.g. CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPIFLG or SPIVECTx shows the RXOVRN condition. Two read operations from the SPIBUF register are required to reach the overwritten buffer word (one to read SPIBUF, which then transfers RXDATA into SPIBUF for the second read). This flag is cleared to 0 when the RXDATA is read. **Note: A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BIT-ERR and DLEN_ERR occur, then RXOVR in RXBUF and SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		**Transmit data buffer full.** This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		**Bit error.** There was a mismatch of internal transmit data and transmitted data.
		0	No bit error occurred. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**

Table 14-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

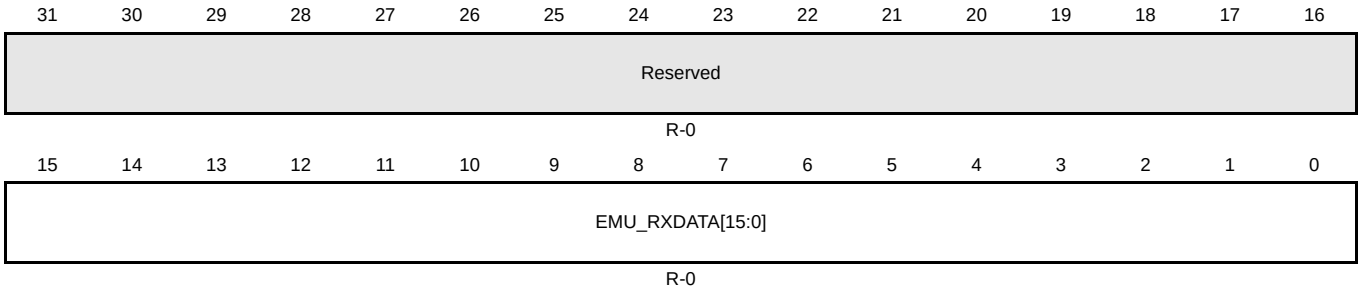
Bit	Name	Value	Description
		1	A bit error occurred. The SPI samples the signal of the transmit pins (master: SIMOx, slave: SOMIx) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		**Desynchronization of slave device.** This bit is valid in master mode only. The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. **Note: In the Compatibility Mode MibSPI, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.**
		0	No slave desynchronization detected. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		1	A slave device is desynchronized.
26	PARITYERR		**Parity error.** The calculated parity differs from the received parity bit. If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No parity error detected.
		1	A parity error occurred.

Table 14-20. SPI Receive Buffer Register (SPIBUF) Field Descriptions (Continued)

Bit	Name	Value	Description
25	TIMEOUT		**Time-out because of non-activation of ENA pin.** The SPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPIFLG register is set. **This bit is valid only in master mode.** **This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.**
		0	No ENA-pin time-out occurred.
		1	An ENA signal time-out occurred.
24	DLENERR		**Data length error flag.** **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No data-length error has occurred.
		1	A data length error has occurred.
23–16	LCSNR	0–FFh	**Last chip select number.** LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer.
15–0	RXDATA[15:0]	0–FFFFh	**SPI receive data.** This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

14.7.18 SPI Emulation Register (SPIEMU)

Figure 14-34. SPI Emulation Register (SPIEMU) [offset = 44h]



R = Read, -*n* = Value after reset

Table 14-21. SPI Emulation Register (SPIEMU) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return zero and writes have no effect.
15–0	EMU_RXDATA [15:0]	0–FFFFh	**SPI receive data.** The SPI emulation register is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF is that a read from SPIEMU does not clear any of the status flags.

14.7.19 SPI Delay Register (SPIDELAY)

Figure 14-35. SPI Delay Register (SPIDELAY) [offset = 48h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C2TDELAY[7:0]								T2CDELAY[7:0]							
R/W-0								R/W-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2EDELAY[7:0]								C2EDELAY[7:0]							
R/W-0								R/W-0							

R = Read, -*n* = Value after reset

Table 14-22. SPI Delay Register (SPIDELAY) Field Descriptions

Bit	Name	Value	Description
31–24	C2TDELAY[7:0]	0-FF	**Chip-select-active to transmit-start delay.** See Figure 14-36 for an example. C2TDELAY is used only in master mode. It defines a setup time (for the slave device) that delays the data transmission from the chip select active edge by a multiple of VCLK cycles. The setup time value is calculated as follows. $t_{C2TDELAY}$ = (C2TDELAY + 2) ***** VCLK Period **Note: If C2TDELAY = 0, then $t_{C2TDELAY}$ = 0.** Example: VCLK = 25 MHz -> VCLK Period = 40ns; C2TDELAY = 07h; > $t_{C2TDELAY}$ = 360 ns When the chip select signal becomes active, the slave has to prepare data transfer within 360 ns. **Note: If phase = 1, the delay between SPICS falling edge to the first edge of SPICLK will have an additional 0.5 SPICLK period delay. This delay is as per the SPI protocol.**

Table 14-22. SPI Delay Register (SPIDELAY) Field Descriptions (Continued)

Bit	Name	Value	Description
23–16	T2CDELAY[7:0]	0–FF	**Transmit-end-to-chip-select-inactive-delay.** See Figure 14-37 for an example. T2CDELAY is used only in master mode. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of VCLK cycles after the last bit is transferred. The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY +1)* VCLK Period **Note: If T2CDELAY = 0, then $t_{T2CDELAY}$ = 0** Example: VCLK = 25 MHz -> VCLK Period = 40ns; T2CDELAY = 03h; > $t_{T2CDELAY}$ = 160 ns; After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. **Note: If phase = 0, then between the last edge of SPICLK and rise-edge of SPICS there will be an additional delay of 0.5 SPICLK period. This is as per the SPI protocol.** Both C2TDELAY and T2CDELAY counters do not have any dependency on the SPIENA pin value. Even if the SPIENA pin is asserted by the slave, the master will continue to delay the start of SPICLK until the C2TDELAY counter overflows. Similarly, even if the SPIENA pin is deasserted by the slave, the master will continue to hold the SPISCS pins active until the T2CDELAY counter overflows. In this way, it is guaranteed that the setup and hold times of the SPISCS pins are determined by the delay timers alone. To achieve better throughput, it should be ensured that these two timers are kept at the minimum possible values.

Table 14-22. SPI Delay Register (SPIDELAY) Field Descriptions (Continued)

Bit	Name	Value	Description
15–8	T2EDELAY[7:0]	0–FFh	**Transmit-data-finished to ENA-pin-inactive time-out.** T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before SPIENA signal has to become inactive and after SPISCS becomes inactive. SPICLK depends on which data format is selected. If the slave device is missing one or more clock edges, it becomes de-synchronized. In this case, although the master has finished the data transfer, the slave is still waiting for the missed clock pulses and the ENA signal isn't disabled. The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the SPIENA signal isn't deactivated in time. The DESYNC flag is set to indicate that the slave device did not de-assert its SPIENA pin in time to acknowledge that it received all bits of the sent word. See Figure 14-38 for an example of this condition. **Note: DESYNC is also set if the SPI detects a de-assertion of SPIENA before the end of the transmission.** The time-out value is calculated as follows: $t_{T2EDELAY}$ = T2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; T2EDELAY = 10h; > $t_{T2EDELAY}$=2 μs; The slave device has to disable the ENA signal within 2 μs, otherwise DESYNC is set and an interrupt is asserted (if enabled).
7–0	C2EDELAY[7:0]	0–FFh	**Chip-select-active to ENA-signal-active time-out.** C2EDELAY is used only in master mode and it applies only if the addressed slave generates an ENA signal as a hardware handshake response. C2EDELAY defines the maximum time between when the SPI activates the chip-select signal and the addressed slave has to respond by activating the ENA signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. The SPI clock depends on whether data format 0 or data format 1 is selected. See Figure 14-39 for an example of this condition. **Note: If the slave device does not respond with the ENA signal before the time-out value is reached, the TIMEOUT flag in the SPIFLG register is set and a interrupt is asserted (if enabled).** If a time-out occurs, the SPI clears the transmit request of the timed-out buffer, sets the TIMEOUT flag for the current buffer, and continues with the transfer of the next buffer in the sequence that is enabled. The timeout value is calculated as follows: $t_{C2EDELAY}$ = C2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; C2EDELAY = 30 h; > $t_{C2EDELAY}$=6 ms; The slave device has to activate the ENA signal within 6 ms after the SPI has activated the chip select signal (SPISCS), otherwise the TIMEOUT flag is set and an interrupt is asserted (if enabled).

Figure 14-36. Example: $t_{C2TDELAY}$ = 8 VCLK Cycles

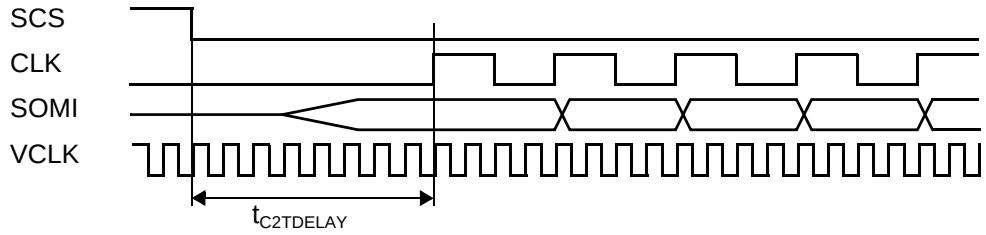


Figure 14-37. Example: $t_{T2CDELAY}$ = 4 VCLK Cycles

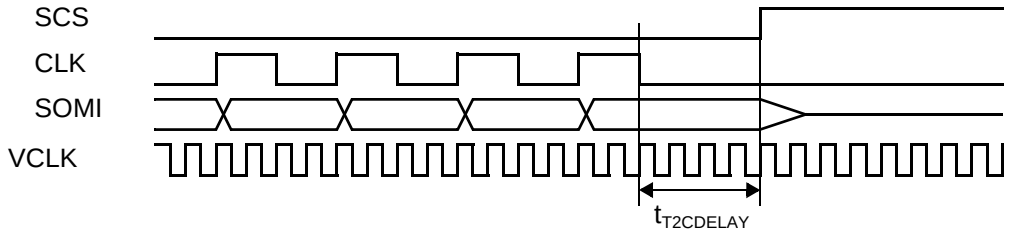


Figure 14-38. Transmit-Data-Finished-to-ENA-Inactive-Timeout

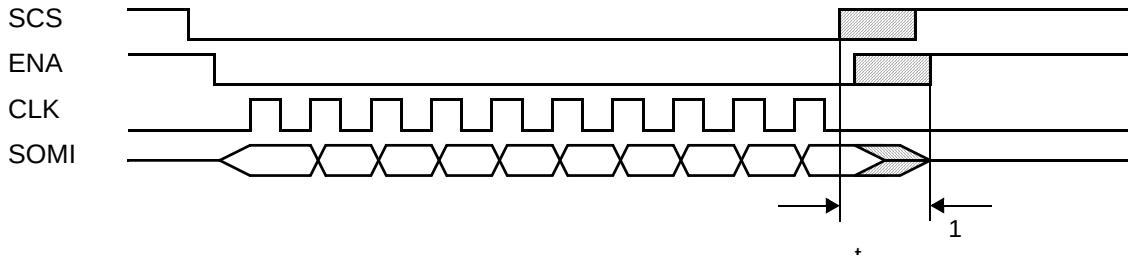
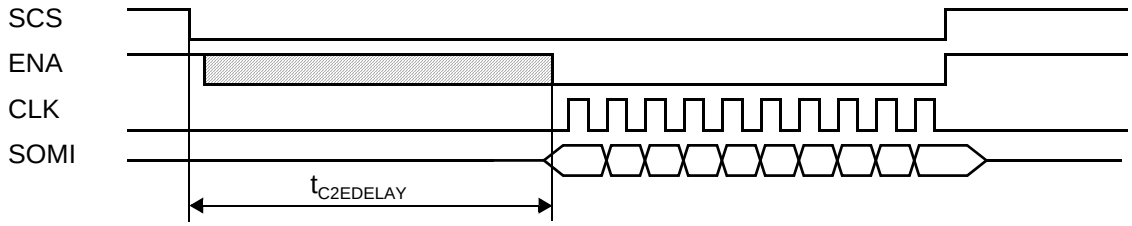
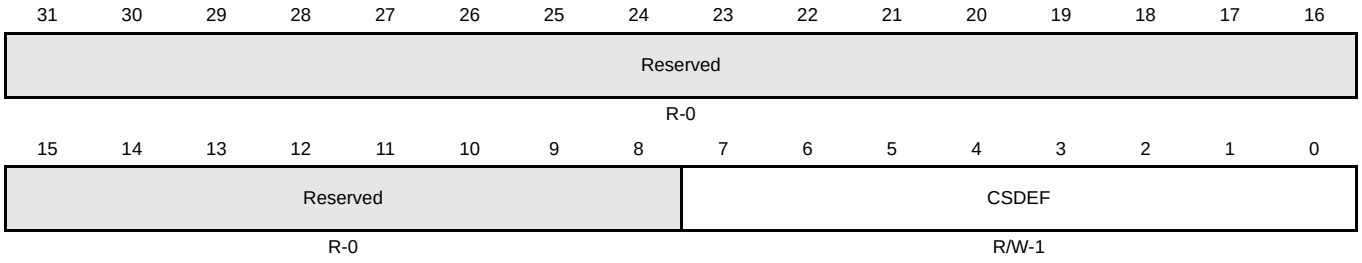


Figure 14-39. Chip-Select-Active-to-ENA-Signal-Active-Timeout



14.7.20 SPI Default Chip Select Register (SPIDEF)

Figure 14-40. SPI Default Chip Select Register (SPIDEF) [offset = 4Ch]



R = Read, W = Write; -*n* = Value after reset

Table 14-23. SPI Default Chip Select Register (SPIDEF) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		Reads return zero and writes have no effect.
7–0	CSDEF	0-FFh	**Chip select default pattern.** Master-mode only. The CSDEFx bits are output to the SPISCS pins when no transmission is being performed. It allows the user to set a programmable chip-select pattern that deselects all of the SPI slaves.
		0	SPISCSx is set to 0 when no transfer is active.
		1	SPISCSx is set to 1 when no transfer is active.

14.7.21 SPI Data Format Registers (SPIFMT[3:0])

Figure 14-41. SPI Data Format Registers (SPIFMT[3:0]) [offset = 5Ch–50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		WDELAY[5:0]						PAR POL	PARITY ENA	WAIT ENA	SHIFT DIR	Reserved	DIS CS TIMERS	POLAR-ITY	PHASE
R-0		R/WP-0						R/WP-0	R/WP-0	R/WP-0	R/WP-0	R-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRESCALE[7:0]								Reserved			CHARLEN[4:0]				
R/WP-0								R-0			R/WP-0				

R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 14-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions

Bit	Name	Value	Description
31–30	Reserved		Reads return zero and writes have no effect.
29–24	WDELAY[5:0]	0–3F	Delay in between transmissions for data format x (x= 0,1,2,3). Idle time that will be applied at the end of the current transmission if the bit WDEL is set in the current buffer. The delay to be applied is equal to: WDELAY * P_{VCLK} + 2 * P_{VCLK} P_{VCLK} -> Period of VCLK.
23	PARPOL		Parity polarity: even or odd. PARPOLx can be modified in privilege mode only. It can be used for data format x (x= 0,1,2,3).
		0	An even parity flag is added at the end of the transmit data stream.
		1	An odd parity flag is added at the end of the transmit data stream.
22	PARITYENA		Parity enable for data format x.
		0	No parity generation/ verification is performed for this data format.
		1	A parity bit is transmitted at the end of each transmitted word. At the end of a transfer the parity generator compares the received parity bit with the locally-calculated parity flag. If the parity bits do not match the RXERR flag is set in the corresponding control field. The parity type (even or odd) can be selected via the PARPOL bit.
21	WAITENA		The master waits for the ENA signal from slave for data format x. WAITENA is valid in master mode only. WAITENA enables a flexible SPI network where slaves with ENA signal and slaves without ENA signal can be mixed. WAITENA defines, for each transferred word, whether the addressed slave generates the ENA signal or not.
		0	The SPI does not wait for the ENA signal from the slave and directly starts the transfer.

Table 14-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Before the SPI starts the data transfer it waits for the ENA signal to become low. If the ENA signal is not pulled down by the addressed slave before the internal time-out counter (C2EDELAY) overflows, then the master aborts the transfer and sets the TIMEOUT error flag.
20	SHIFTDIR		**Shift direction for data format x.** With bit SHIFTDIRx, the shift direction for data format x (x=0,1,2,3) can be selected.
		0	MSB is shifted out first.
		1	LSB is shifted out first.
19	Reserved		Reads return zero and writes have no effect.
18	DIS CS TIMERS		**Disable chip-select timers for this format.** The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format, if they are not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip-select delay timers for any slaves.
		0	Both C2TDELAY and T2CDELAY counts are inserted for the chip selects.
		1	No C2TDELAY or T2CDELAY is inserted in the chip select timings.
17	POLARITY		**SPI data format x clock polarity.** POLARITYx defines the clock polarity of data format x. The following restrictions apply when switching clock phase and/or polarity: 1. In 3-pin/4-pin with nENA pin configuration of a slave SPI, the clock phase and polarity cannot be changed on-the-fly between two transfers. The slave should be reset and reconfigured if clock phase/polarity needs to be switched. In summary, SPI format switching is not fully supported in slave mode. 2. Even while using chip select pins, the polarity of SPICLK can be switched only while the slave is not selected by a valid chip select. The master SPI should ensure that while switching SPICLK polarity, it has deselected all of its slaves. Otherwise, the switching of SPICLK polarity may be incorrectly treated as a clock edge by some slaves.
		0	If POLARITYx is set to 0 the SPI clock signal is low-inactive, i.e., before and after data transfer the clock signal is low.
		1	If POLARITYx is set to 1 the SPI clock signal is high-inactive, i.e., before and after data transfer the clock signal is high.

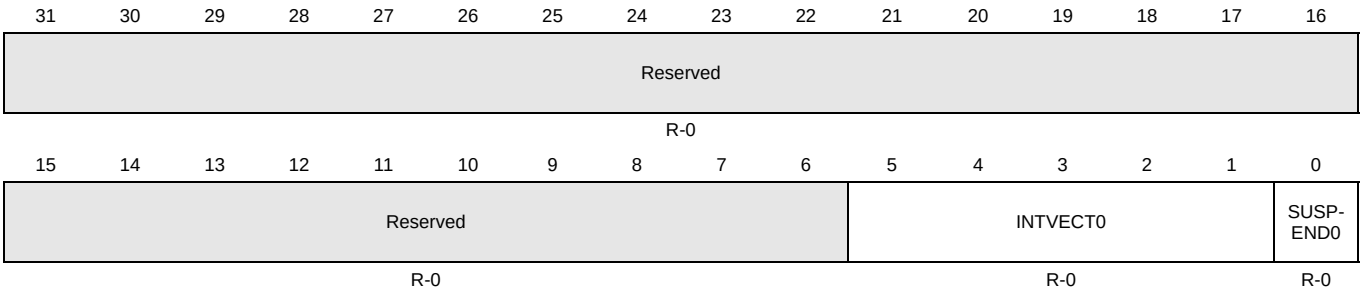
Table 14-24. SPI Data Format Registers (SPIFMT[3:0]) Field Descriptions (Continued)

Bit	Name	Value	Description
16	PHASE		**SPI data format x clock delay.** PHASEx defines the clock delay of data format x.
		0	If PHASEx is set to 0 the SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge.
		1	If PHASEx is set to 1 the SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15–8	PRESCALE[7:0]		SPI data format x prescaler. PRESCALEx determines the bit transfer rate of data format x if the SPI is the network master. PRESCALEx is use to derive SPICLK from VCLK. If the SPI is configured as slave, PRESCALEx **does not need** to be configured. The clock rate for data format x can be calculated as: $BR_{Formatx} = \frac{VBUSPCLCK}{(PRESCALEx + 1)}$ **Note: When PRESCALEx is set to 0, the SPI clock rate defaults to VCLK/2.**
7–5	Reserved		Reads return zero and writes have no effect.
4–0	CHARLEN[4:0]	0–1F	SPI data format x data-word length. CHARLENx defines the word length of data format x. Legal values are 0x02 (data word length = 2 bit) to 0x10 (data word length = 16). Illegal values, such as 0x00 or 0x1F are not allowed; their effect is indeterminate.

14.7.22 Interrupt Vector 0 (INTVECT0)

Note: The TG interrupt is not available in MibSPI in compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 14-42. *Interrupt Vector 0* (*INTVECT0*) [offset = 60h]



R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 14-25. *Transfer Group Interrupt Vector 0* (*INTVECT0*)]

Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–1	INTVECT0		**INTVECT0. Interrupt vector for interrupt line INT0.** Returns the vector of the pending interrupt at interrupt line INT0. If more than one interrupt is pending, INTVECT0 always references the highest prior interrupt source first. **Note:** This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.
		00000	There is no pending interrupt.
		00001 + x	Transfer group x (x=0,..,15) has a pending interrupt. SUSPEND0 reflects the type of interrupt (*suspended* or *finished*).
		10001	Error Interrupt pending. The lower half of SPIINT0 contains more details about the type of error.
		10011	The pending interrupt is a “Receive Buffer Overrun” interrupt.
		10010	**SPI mode:** The pending interrupt is a “Receive Buffer Full” interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		10100	**SPI mode:** The pending interrupt is a “Transmit Buffer Empty” interrupt. **Mib mode:** Reserved. This bit combination should not occur.
		all other combinations	**SPI mode:** Reserved. These bit combinations should not occur.

Table 14-25. *Transfer Group Interrupt Vector 0* (*INTVECT0*)] (Continued)

Bit	Name	Value	Description
0	SUSPEND0		**Transfer suspended / Transfer finished interrupt flag.** Every time INTVECT0 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT0 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a “transfer finished” interrupt. In other words, the buffer array referenced by INTVECT0 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a “transfer suspended” interrupt. In other words, the transfer group referenced by INTVECT0 has asserted an interrupt because the buffer to be transferred next is in “suspend-to-wait” mode.

Note: Reading from the INTVECT0 register when “Transmit Empty” is indicated does not clear the TXINTFLG flag in the SPIFLG register. Writing a new word to the SPIDATx register clears the “Transmit Empty” interrupt.

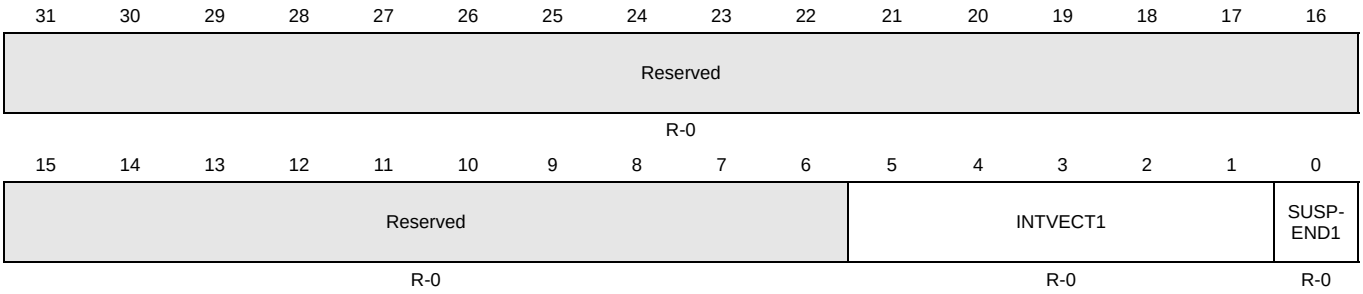
Note: In multi-buffer mode, INTVECT0 contains the interrupt for the highest priority transfer group. A read from INTVECT0 automatically causes the next-highest priority transfer group’s interrupt status to get loaded into INTVECT0 and its corresponding SUSPEND flag to get loaded into SUSPEND0. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT0 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPIFLG register or by reading the RXOVERN_BUF_ADDR register.

14.7.23 Interrupt Vector 1 (INTVECT1)

Note: The TG interrupt is not available in SPI in compatibility mode compatibility mode. Therefore, there is no possibility to access this register in compatibility mode.

Figure 14-43. *Interrupt Vector 1* (*INTVECT1*) [offset = 64h]



R = Read, WP = Write in privilege mode only; -*n* = Value after reset

Table 14-26. *Transfer Group Interrupt Vector 1* (*INTVECT1*)]

Bit	Name	Value	Description
31–6	Reserved		Reads return zero and writes have no effect.
5–1	INTVECT1		**INTVECT1. Interrupt vector for interrupt line INT1.** Returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, INTVECT1 always references the highest prior interrupt source first. **Note:** This field reflects the status of the SPIFLG register in vector format. Any updates to the SPIFLG register will automatically cause updates to this field.
		00000	There is no pending interrupt. **SPI mode only.**
		10001	Error Interrupt pending. The lower half of SPIINT1 contains more details about the type of error. **SPI mode only.**
		10011	The pending interrupt is a “Receive Buffer Overrun” interrupt. **SPI mode only.**
		10010	The pending interrupt is a “Receive Buffer Full” interrupt. **SPI mode only.**
		10100	The pending interrupt is a “Transmit Buffer Empty” interrupt. **SPI mode only.**
		all other combinations	Reserved. These bit combinations should not occur. **SPI mode only.**

Table 14-26. *Transfer Group Interrupt Vector 1* (*INTVECT1*)] (Continued)

Bit	Name	Value	Description
0	SUSPEND1		**Transfer suspended / Transfer finished interrupt flag.** Every time INTVECT1 is read by the host, the corresponding interrupt flag of the referenced transfer group is cleared and INTVECT1 is updated with the vector coming next in the priority chain.
		0	The interrupt type is a “transfer finished” interrupt. In other words, the buffer array referenced by INTVECT1 has asserted an interrupt because all of data from the transfer group has been transferred.
		1	The interrupt type is a “transfer suspended” interrupt. In other words, the transfer group referenced by INTVECT1 has asserted an interrupt because the buffer to be transferred next is in “suspend-to-wait” mode.

Note: Reading from the INTVECT1 register when “Transmit Empty” is indicated does not clear the TXINTFLG flag in the SPIFLG register. Writing a new word to the SPIDATx register clears the “Transmit Empty” interrupt.

Note: In multi-buffer mode, INTVECT1 contains the interrupt for the highest priority transfer group. A read from INTVECT1 automatically causes the next-highest priority transfer group’s interrupt status to get loaded into INTVECT1 and its corresponding SUSPEND flag to get loaded into SUSPEND1. The transfer group with the lowest number has the highest priority, and the transfer group with the highest number has the lowest priority.

Reading the INTVECT1 register when the RXOVRN interrupt is indicated in multi-buffer mode does not clear the RXOVRN flag and hence does not clear the vector. The RXOVRN interrupt vector may be cleared in multi-buffer mode either by write-clearing the RXOVRN flag in the SPIFLG register or by reading the RXOVERN_BUF_ADDR register.

14.7.24 SPI Pin Control Register 9 (SPIPC9)

Note: Register bits vary by device
Register bits 31:24 and 23:16 of this register reflect the number of SPISIMO/SPISOMI pins per device. On devices with 8 data-line support, all of bits 31 to 16 are implemented. On devices with less than 8 data lines, only a subset of these bits are available. Unimplemented bits return 0 upon read and are not writable.

Figure 14-44. SPI Pin Control Register 9 (SPIPC9) [offset = 68h[

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SOMISRS[7:0]								SIMOSRS[7:0]							
R/W-0								R/W-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SOMISRS0	SIMOSRS0	CLK-SRS		SCSSRS[7:0]							
0				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0							

R = Read, W = write, P = Privilege mode, -*n* = Value after reset

Table 14-27. SPI Pin Control Register 9 (SPIPC9) Field Descriptions

Bit	Name	Value	Description
31–24	SOMISRS[7:0]		SPISOMIx slew rate control. Each bit controls the slew rate for the corresponding SPISOMIx pin. **Note: Bit 11 or bit 24 can be used to control the slew rate for SPISOMI0. If a 32-bit write is performed, bit 11 will have priority over bit 24.** **Note: For devices with less than 8 SPISOMI pins, the remaining bit fields are reserved.**
		0	Normal buffer select.
		1	Slow buffer select.
23–16	SIMOSRS[7:0]		SPISIMOx slew rate control. Each bit controls the slew rate for the corresponding SPISIMOx pin. **Note: Bit 10 or bit 16 can be used to control the slew rate for SPISIMO0. If a 32-bit write is performed, bit 10 will have priority over bit 16.** **Note: For devices with less than 8 SPISIMO pins, the remaining bit fields are reserved.**
		0	Normal buffer select.
		1	Slow buffer select.

Table 14-27. SPI Pin Control Register 9 (SPIPC9) Field Descriptions (Continued)

Bit	Name	Value	Description
15–12	Reserved		Reads return zero and writes have no effect.
11	SOMISRS0		SPISOMI0 slew rate control. This bit controls the slew rate for the SPISOMI0 pin.
		0	Normal buffer select.
		1	Slow buffer select.
10	SPISIMOSRS0		SPISIMO0 slew rate control. This bit controls the slew rate for the SPISIMO0 pin.
		0	Normal buffer select.
		1	Slow buffer select.
9	SPICLKSRS		SPICLK pin slew rate control. This bit controls the slew rate for the SPICLK pin.
		0	Normal buffer select.
		1	Slow buffer select.
8	SPIENASRS		$\overline{\text{SPIENA}}$ pin slew rate control. This bit controls the slew rate for the $\overline{\text{SPIENA}}$ pin.
		0	Fast buffer Select.
		1	Slow buffer Select.
7–0	SPISCSSRS[7:0]		$\overline{\text{SPISCS}}$x pin slew rate control. Each bit controls the slew rate for the corresponding $\overline{\text{SPISCS}}$x pin.
		0	Normal buffer select.
		1	Slow buffer select

14.7.25 Multi-buffer Mode Enable Register (MIBSPIE)

Note: Accessibility of Multi-Buffer RAM
The multi-buffer RAM is not accessible unless the MSPIENA bit set to 1. The only exception to this is in test mode, where, by setting RXRAMACCESS to 1, the multi-buffer RAM can be fully accessed for both read and write.

Figure 14-45. Multi-buffer Mode Enable Register (MIBSPIE) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RXRA-MAC-CESS
R-0															R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MSPI-ENA
R-0															R/WP-0

R = Read, W = write, P = Privilege mode, -*n* = Value after reset

Table 14-28. Multi-buffer Mode Enable Register (MIBSPIE) Field Descriptions

Bit	Name	Value	Description
31–17	Reserved		Reads return zeros and writes have no effect.
16	RXRAMACCESS		**Receive-RAM access control.** During normal operating mode of SPI, the receive data/status portion of multi-buffer RAM is read-only. To enable testing of receive RAM, direct read/write access is enabled by setting this bit.
		0	The RX portion of multi-buffer RAM is not writable by the CPU.
		1	The whole of multi-buffer RAM is fully accessible for read/write by the CPU.
			Note: The RX RAM ACCESS bit remains 0 after reset and it should remain set to 0 at all times, except when testing the RAM. SPI should be given a local reset by using the nRESET (SPIGCR0[0]) bit after RAM testing is performed so that the multi-buffer RAM gets re-initialized.
15–1	Reserved		Reads return zeros and writes have no effect.

Table 14-28. Multi-buffer Mode Enable Register (MIBSPIE) Field Descriptions (Continued)

Bit	Name	Value	Description
0	MSPIENA		Multi-buffer mode enable. After power-up or reset, MSPIENA remains cleared, which means that the SPI runs in compatibility mode by default. If multi-buffer mode is desired, this register should be configured first after configuring the SPIGCR0 register. If MSPIENA is not set to 1, the multi-buffer mode registers are not writable.
		0	The SPI runs in compatibility mode, i.e., in this mode the MibSPI is fully code-compliant to the standard TMS470Px SPI. No multi-buffered-mode features are supported.
		1	The SPI is configured to run in multi-buffer mode.

Note: Accessibility of Registers
Registers from this offset address onwards are not accessible in SPI compatibility mode. They are accessible only in the multi-buffer mode.

14.7.26 TG Interrupt Enable Set Register (TGITENST)

The register TGITENST contains the TG interrupt enable flags for transfer-finished and for transfer-suspended events. Each of the enable bits in the higher half-word and the lower half-word of TGITENST belongs to one TG.

The register map shown in Figure 14-46 and Table 14-29 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 14-46. TG Interrupt Enable Set Register (TGITENST) [offset = 74h]

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

R = Read; W = Write; -*n* = Value after reset

Table 14-29. TG Interrupt Enable Set Register (TGITENST) Field Descriptions

Bit	Name	Value	Description
31-16	SET INTENRDY[15:0]		TG interrupt set (enable) when transfer finished.
		0	*Read:* The TGx-completed interrupt is disabled. This interrupt does not get generated when TGx completes. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. Write: Enable the TGx-completed interrupt. The interrupt gets generated when TGx completes.
15–0	SET INTEN SUS[15:0]		TG interrupt set (enabled) when transfer suspended
		0	*Read:* The TGx-suspended interrupt is disabled. This interrupt does not get generated when TGx is suspended. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is enabled. The interrupt gets generated when TGx is suspended. *Write:* Enable the TGx-suspended interrupt. The interrupt gets generated when TGx is suspended.

14.7.27 MibSPI TG Interrupt Enable Clear Register (TGITENCR)

The register TGITENCR is used to clear the interrupt enables for the TG-completed interrupt and the TG-suspended interrupts.

The register map shown in Figure 14-47 and Table 14-30 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 14-47. TG Interrupt Enable Clear Register (TGITENCR) [offset = 78h]

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

R = Read; W = Write; -*n* = Value after reset

Table 14-30. TG Interrupt Enable Clear Register (TGITENCR) Field Descriptions

Bit	Name	Value	Description
31-16	CLR INTEN RDY[15:0]		TG interrupt clear (disabled) when transfer finished.
		0	*Read:* The TGx-completed interrupt is disabled. The interrupt does not get generated when TGx completes. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is enabled. The interrupt gets generated when TGx completes. *Write:* Disable the TGx-completed interrupt.
15–0	CLR INTEN SUS[15:0]		TG interrupt clear (disabled) when transfer suspended.
		0	*Read:* The TGx-suspended interrupt is disabled. This interrupt does not get generated when TGx is suspended. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is enabled. The interrupt gets generated when TGx completes. *Write:* Disables the TGx-suspended interrupt.

14.7.28 Transfer Group Interrupt Level Set Register (TGITLVST)

The register TGITLVST sets the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 1.

The register map shown in Figure 14-48 and Table 14-31 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 14-48. Transfer Group Interrupt Level Set Register (TGITLVST) [offset = 7Ch]

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

R = Read; W = Write; -*n* = Value after reset

Table 14-31. Transfer Group Interrupt Level Set Register (TGITLVST) Field Descriptions

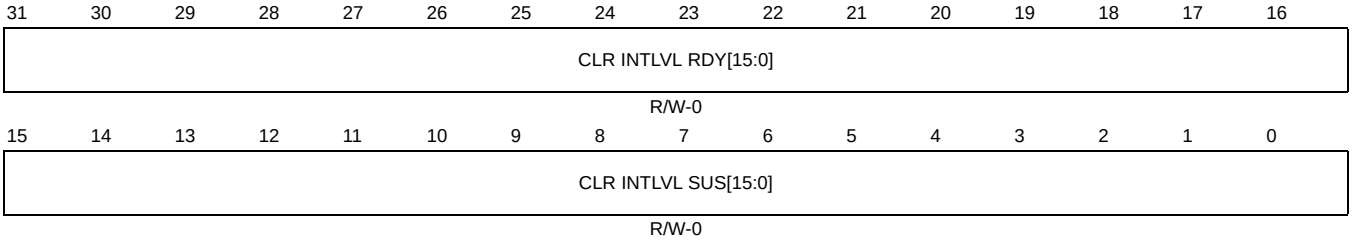
Bit	Name	Value	Description
31-16	SET INTLVL RDY[15:0]		Transfer-group completed interrupt level set.
		0	*Read:* The TGx-completed interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is set to INT1. *Write:* Set the TGx-completed interrupt to INT1.
15–0	SET INTLVL SUS[15:0]		Transfer-group suspended interrupt level set.
		0	*Read:* TGx-suspended interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is set to INT1. *Write:* Set the TG-x suspended interrupt INT1.

14.7.29 Transfer Group Interrupt Level Clear Register (TGITLVCR)

The register TGITLVCR clears the level of interrupts for transfer completed interrupt and for transfer suspended interrupt to level 0.

The register map shown in Figure 14-49 and Table 14-32 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 14-49. Transfer Group Interrupt Level Clear Register (TGITLVCR) [offset = 80h]



R = Read; W = Write; -*n* = Value after reset

Table 14-32. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

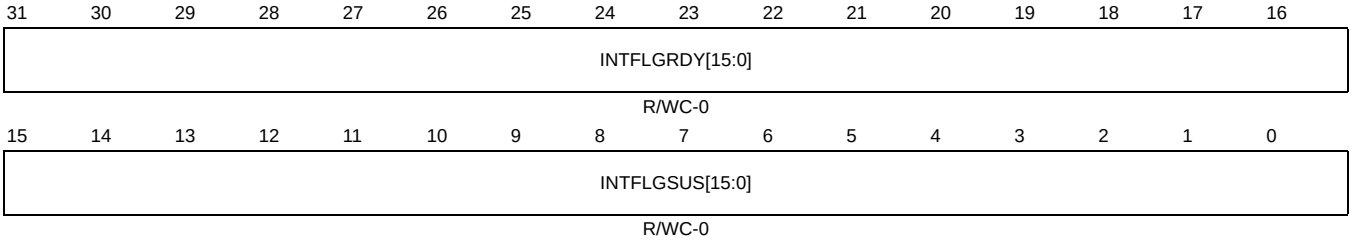
Bit	Name	Value	Description
31-16	CLR INTLVL RDY[15:0]		Transfer-group completed interrupt level clear.
		0	*Read:* The TGx-completed interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-completed interrupt is set to INT1. *Write:* Sets the TG-x completed interrupt to INT0.
15–0	CLR INTLVL SUS[15:0]		Transfer group suspended interrupt level clear.
		0	*Read:* The TG-x suspended interrupt is set to INT0. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* The TGx-suspended interrupt is set to INT1. *Write:* Sets the TGx-suspended interrupt to INT0.

14.7.30 Transfer Group Interrupt Flag Register (TGINTFLAG)

The TGINTFLAG register comprises the transfer group interrupt flags for transfer-completed interrupts (INTFLGRDYx) and for transfer-suspended interrupts (INTFLGSUSx). Each of the interrupt flags in the higher half-word and the lower half-word of TGINTFLAG belongs to one TG.

The register map shown in Figure 14-50 and Table 14-33 represents a super-set device with the maximum number of TGs (16) assumed. The actual number of bits available varies per device.

Figure 14-50. Transfer Group Interrupt Flag Register (TGINTFLAG) [offset = 84h]



R = Read; W = Write; C = Clear; *-n* = Value after reset

Table 14-33. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions

Bit	Name	Value	Description
31-16	INTFLGRDY[15:0]		Transfer-group interrupt flag for a transfer-completed interrupt. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGRDYx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the vector registers is 0.**
		0	*Read:* No transfer-completed interrupt occurred since last clearing of the INTFLGRDYx flag. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* A transfer finished interrupt from transfer group x occurred. No matter whether the interrupt is enabled or disabled (INTENRDYx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGRDYx is set right after the transfer from TGx is finished. *Write:* The corresponding bit flag is cleared.

Table 14-33. Transfer Group Interrupt Level Clear Register (TGITLVCR) Field Descriptions (Continued)

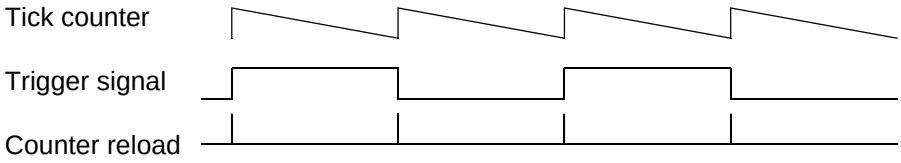
Bit	Name	Value	Description
15–0	INTFLG SUS[15:0]		Transfer-group interrupt flag for a transfer-suspend interrupt. **Note: Read Clear Behavior. Reading the interrupt vector registers TGINTVECT0 or TGINTVECT1 automatically clears the interrupt flag bit INTFLGSUSx referenced by the vector number given by INTVECT0/INTVECT1 bits, if the SUSPEND[0:1] bit in the corresponding vector registers is 1.**
		0	*Read:* No transfer-suspended interrupt occurred since the last clearing of the INTFLGSUSx flag. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* A transfer-suspended interrupt from TGx occurred. No matter whether the interrupt is enabled or disabled (INTENSUSx = don't care) or whether the interrupt is mapped to INT0 or INT1, INTFLGSUSx is set right after the transfer from transfer group x is suspended. *Write:* The corresponding bit flag is cleared.

14.7.31 Tick Count Register (TICKCNT)

One of the trigger sources for TGs is an internal periodic time trigger. This time trigger is called a tick counter and is basically a down-counter with a preload/reload value. Every time the tick counter detects an underflow it reloads the initial value and toggles the trigger signal provided to the TGs.

The trigger signal, shown in Figure 14-51 as a square wave, illustrates the different trigger event types for the TGs (e.g., rising edge, falling edge, and both edges).

Figure 14-51. Tick Counter Operation



This register is shown in Figure 14-52 and described in Table 14-34.

Figure 14-52. Tick Count Register (TICKCNT) [offset = 90h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TICK ENA	RE LOAD	CLKCTRL(1–0)		Reserved											
R/W-0	R0/S-0	R/W-0		R-0											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TICKVALUE(15–0)															
R/W-0															

R = Read; W = Write; S = Set; C = Clear; -*n* = Value after reset;

Table 14-34. Tick Count Register (TICKCNT) Field Descriptions

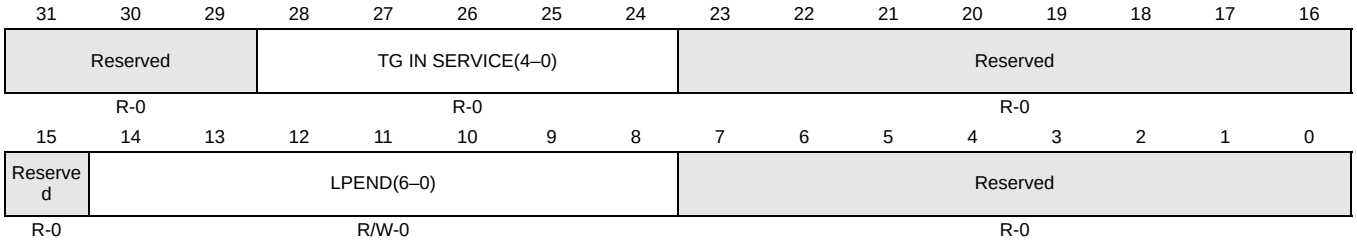
Bit	Name	Value	Description
31	TICK ENA		Tick counter enable.
		0	The internal tick counter is disabled. The counter value remains unchanged. **Note: When the tick counter is disabled, the trigger signal is forced low.**
		1	The internal tick counter is enabled and is clocked by the clock source selected by CLKCTRL[1:0]. When the tick counter is enabled it starts down-counting from its current value. When TICK-ENA goes from 0 to 1, the tick counter is automatically loaded with the contents of TICKVALUE.

Table 14-34. Tick Count Register (TICKCNT) Field Descriptions (Continued)

Bit	Name	Value	Description
30	RELOAD		**Pre-load the tick counter.** RELOAD is a set-only bit; writing a 1 to it reloads the tick counter with the value stored in TICKVALUE. Reading RELOAD always returns a 0. **Note: When the tick counter is reloaded by the RELOAD bit, the trigger signal is not toggled.**
29–28	CLKCTRL(1–0)		**Tick counter clock source control.** CLKCTRL defines the clock source that is used to clock the internal tick counter.
		00	SPICLK of data word format 0 is selected as the clock source of the tick counter
		01	SPICLK of data word format 1 is selected as the clock source of the tick counter
		10	SPICLK of data word format 2 is selected as the clock source of the tick counter
		11	SPICLK of data word format 3 is selected as the clock source of the tick counter
27–16	Reserved		Reads return 0 and writes have no effect.
15–0	TICKVALUE(15–0)	0–FFFF	**Initial value for the tick counter.** TICKVALUE stores the initial value for the tick counter. The tick counter is loaded with the contents of TICKVALUE every time an underflow condition occurs and every time the RELOAD flag is set by the host.

14.7.32 Last TG End Pointer (LTGPEND)

Figure 14-53. Last TG End Pointer (LTGPEND) [offset = 94h]



R = Read, W = Write, C = Clear, -*n* = Value after reset, x = indeterminate

Table 14-35. Last TG End Pointer (LTGPEND) Field Descriptions

Bit	Name	Value	Description
31–29	Reserved		Reads return 0 and writes have no effect.
28–24	TG IN SERVICE(4–0)		**The TG number currently being serviced by the sequencer.** These bits indicate the current TG that is being serviced. This field can generally be used for code debugging.
		00000	No TG is being serviced by the sequencer.
		00001	TG0 is being serviced by the sequencer.
		. . .	...
		10000	TG15 is being serviced by the sequencer.
			Note: The number of transfer groups varies by device.
		10001–11111	Invalid values

Table 14-35. Last TG End Pointer (LTGPEND) Field Descriptions (Continued)

Bit	Name	Value	Description
23–15	Reserved		Reads return 0 and writes have no effect.
14–8	LPEND(6–0)	0–7Fh	**Last TG end pointer.** Usually the TG end address (PEND) is inherently defined by the start value of the starting pointer of the subsequent TG (PSTART). The TG ends one word before the next TG starts (PEND[x]=PSTART[x+1] - 1). For a full configuration of MibSPI, the last TG has no subsequent TG, i.e., no end address is defined. Therefore LPEND has to be programmed to specify explicitly the end address of the last TG. **Note: LPEND allows SW compatibility for MibSPI variants Because of software compatibility reasons, the last TG end address has to be configurable; otherwise a super-set MibSPI cannot emulate a MibSPI with less TGs without software changes.** **Note: The number of transfer groups varies by device, thus the last TG also varies by device.**
7–0	Reserved		Reads return 0 and writes have no effect.

14.7.33 TGx Control Registers (TGxCTRL)

Each TG can be configured via one dedicated control register. The register description below shows one control register(x) which is identical for all TGs. For example, the control register for TG 2 is named TG2CTRL and is located at address *base0+98h+4*2*. The actual number of available control registers varies by device.

Figure 14-54. MibSPI TG Control Registers (TGxCTRL) [offsets = 98h–D4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TG ENA[0:15]	ONE SHOT[0:15]	PRST[0:15]	TGTD[0:15]	Reserved				TRGEVT[0:15](3–0)				TRIGSRC[0:15](3–0)			
R/W-0	R/W-0	R/W-0	R-0	R-0				R/W-0				R/W-0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PSTART[0:15](6–0)							Reserved	PCURRENT[0:15](6–0)						
R-0	R/W-0							R-0	R-0						

R = Read; W = Write; *-n* = Value after reset;

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions

Bit	Name	Value	Description
31	TGENA		TGx enable. If the correct event (TRIGEVTx) occurs at the selected source (TRIGSRCx) a group transfer is initiated if no higher priority TG is in active transfer mode or if one or more higher-priority TGs are in transfer-suspend mode. Disabling a TG while a transfer is ongoing will finish the ongoing word transfer but not the whole group transfer.
		0	TGx is disabled.
		1	TGx is enabled.
30	ONESHOTx		Single transfer for TGx.
		0	TGx initiates a transfer every time a trigger event occurs and TGENA is set.
		1	A transfer from TGx will be performed only once (one shot) after a valid trigger event at the selected trigger source. After the transfer is finished the TGENAx control bit will be cleared and therefore no additional transfer can be triggered before the host enables the TG again. This one shot mode ensures that after one group transfer the host has enough time to read the received data and to provide new transmit data.

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
29	PRSTx		TGx pointer reset mode. Configures the way to resolve trigger events during an ongoing transfer. This bit is meaningful only for level-triggered TGs. Edge-triggered TGs cannot be restarted before their completion by another edge. The PRST bit will have no effect on this behavior. **Note: When the PRST bit is set, if the buffer being transferred at the time of a new trigger event is a LOCK, CSHOLD or NOBRK buffer, then only after finishing those transfers, the TG will be restarted. This means that even if the TG is retriggered, the TG will only be restarted after finishing the transfer of the first non-LOCK or non-CSHOLD buffer. In the case of the NOBRK buffer, after completing the ICOUNT number of transfers, the TG will be restarted from its PSTART.** This means that TX control fields such as LOCK and CSHOLD, and DMA control fields such as NOBRK have higher priority over anything else. They have the capability to delay the restart of the TG even if it is retriggered when PRST is 1.
		0	If a trigger event occurs during a transfer from TGx, the event is ignored and is not stored internally. The TGx transfer has priority over additional trigger events.
		1	The TGx pointer (PCURRENTx) will be reset to the start address (PSTARTx) when a valid trigger event occurs at the selected trigger source while a transfer from the same TG is ongoing. Every trigger event resets PCURRENTx no matter whether the concerned TG is in transfer mode or not. The trigger events have priority over the ongoing transfer.
28	TGTDx		TG triggered.
		0	TGx has not been triggered or is no longer waiting for service.
		1	TGx has been triggered and is either currently being serviced or waiting for servicing.
27–24	Reserved		Reads return 0 and writes have no effect.

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
23–20	TRIGEVTx(3-0)		**Type of trigger event.** A level-triggered TG can be stopped by de-activating the level trigger. However, the following restrictions apply. 1. Deactivating the level trigger for a TG during a NOBRK transfer does not stop the transfers until all of the ICOUNT number of buffers are transferred for the NOBRK buffer. Once a NOBRK buffer is prefetched, the trigger event loses control over the TG until the NOBRK buffer transfer is completed. 2. Once the transfer of a buffer with CSHOLD or LOCK bit set starts, deactivating the trigger level does not stop the transfer until the sequencer completes the transfer of the next non-CSHOLD or non-LOCK buffer in the same TG. 3. Once the last buffer in a TG is pre-fetched, de-activating the trigger level does not stop the transfer group until the last buffer transfer is completed. This means even if the trigger level is deactivated at the beginning of the penultimate (one-before-last) buffer transfer, the sequencer continues with the same TG until it is completed.
		0000	never — Never trigger TGx. This is the default value after reset.
		0001	rising edge — A rising edge (0 to 1) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0010	falling edge — A falling edge (1 to 0) at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0011	both edges — Rising and falling edges at the selected trigger source (TRIGSRCx) initiates a transfer for TGx
		0100	Reserved
		0101	high-active — While the selected trigger source (TRIGSRCx) is at a logic high level (1) the group transfer is continued and at the end of one group transfer restarted at the beginning. If the logic level changes to low (0) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**
		0110	low-active — While the selected trigger source (TRIGSRCx) is at a logic low level (0) the group transfer is continued and at the end of one restarted at the beginning. If the logic level changes to high (1) during an ongoing group transfer, the whole group transfer will be stopped. **Note: If ONESHOTx is set the transfer is performed only once.**

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description	
		0111	always	A repetitive group transfer will be performed. **Note: By setting the TRIGSRC to 0000b, the TRIGEVT to 0111b (ALWAYS), and the ONESHOTx bit to 1, software can trigger this TG. Upon setting the TGENA bit, the TG is immediately triggered.** **Note: If ONESHOTx is set the transfer is performed only once.**
		1xxx	Reserved	

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
19–16	TRIGSRCx(3-0)		**Trigger source.** After reset, the trigger sources of all TGs are disabled.
		0000	disabled
		0001	EXT0 External trigger source 0. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0010	EXT1 External trigger source 1. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0011	EXT2 External trigger source 2. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0100	EXT3 External trigger source 3. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0101	EXT4 External trigger source 4. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0110	EXT5 External trigger source 5. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		0111	EXT6 External trigger source 6. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1000	EXT7 External trigger source 7. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1001	EXT8 External trigger source 8. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1010	EXT9 External trigger source 9. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1011	EXT10 External trigger source 10. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1100	EXT11 External trigger source 11. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1101	EXT12 External trigger source 12. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1110	EXT13 External trigger source 13. The actual source varies per device (e.g. HET I/O channel, event pin, etc.).
		1111	TICK Internal periodic event trigger. The tick counter can initiate periodic group transfers.
15	Reserved		Reads return 0 and writes have no effect.

Table 14-36. TG Control Registers (TGxCTRL) Field Descriptions (Continued)

Bit	Name	Value	Description
14–8	PSTARTx(6-0)	0–7Fh	**TG start address.** PSTARTx stores the start address of the corresponding TG. The corresponding end address is inherently defined by the subsequent TG's start address minus one (PENDx[TGx] = PSTARTx[TGx+1]-1). PSTARTx is copied into PCURRENTx when: • The TG is enabled. • The end of the TG is reached during a transfer. • A trigger event occurs while PRST is set to 1.
7	Reserved		Reads return 0 and writes have no effect.
6–0	PCURRENTx(6-0)	0–7Fh	Pointer to current buffer. PCURRENT is read-only. PCURRENTx stores the address (0...127) of the buffer that corresponds to this TG. If the TG switches from active transfer mode to suspend to wait, PCURRENTx contains the address of the currently suspended word. After the TG resumes from suspend to wait mode, the next buffer will be transferred; i.e. no buffer data is transferred because of suspend to wait mode.

Note: TG0 has the highest priority and TG15 has the lowest priority.

Under the following conditions a lower priority TG cannot be interrupted by a higher priority TG.

1. When there is a CSHOLD or LOCK buffer, until the completion of the next buffer transfer which is a non-CSHOLD or non-LOCK buffer.

2. An entire sequence of words transferred for a NOBRK DMA buffer.

3. Once the last word in a TG is pre-fetched.

14.7.34 Multi-buffer RAM *Uncorrectable Parity Error Control Register (UERRCTRL)*

Figure 14-55. Multi-buffer RAM Uncorrectable Parity Error Control Register (UERRCTRL) [offset = 120h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							PTESTEN	Reserved				EDEN[3:0]			
R-0							R/WP-0	R-0				R/WP-0101			

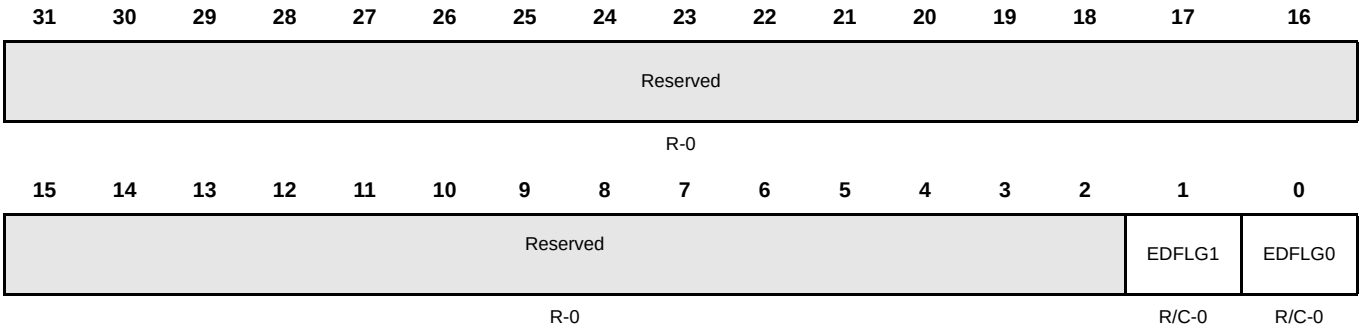
R = Read, WP = Write in Privilege mode only, -*n* = Value after reset

Table 14-37. Multi-buffer RAM Uncorrectable Parity Error Control Register (UERRCTRL) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved		Reads return zero and writes have no effect.
8–4	Reserved		Reads return zero and writes have no effect.
3–0	EDEN[3:0]		Error detection enable. These bits enable parity error detection.
		0101	Parity error detection logic (default) is disabled.
		All other values	Parity error detection logic is enabled.
			Note: It is recommended to write a 1010 to enable error detection, to guard against a soft error from disabling parity error detection.

14.7.35 Multi-buffer RAM Uncorrectable Parity Error Status Register (UERRSTAT)

Figure 14-56. Multi-buffer RAM Uncorrectable Parity Error Status Register (UERRSTAT) [offset = 124h]



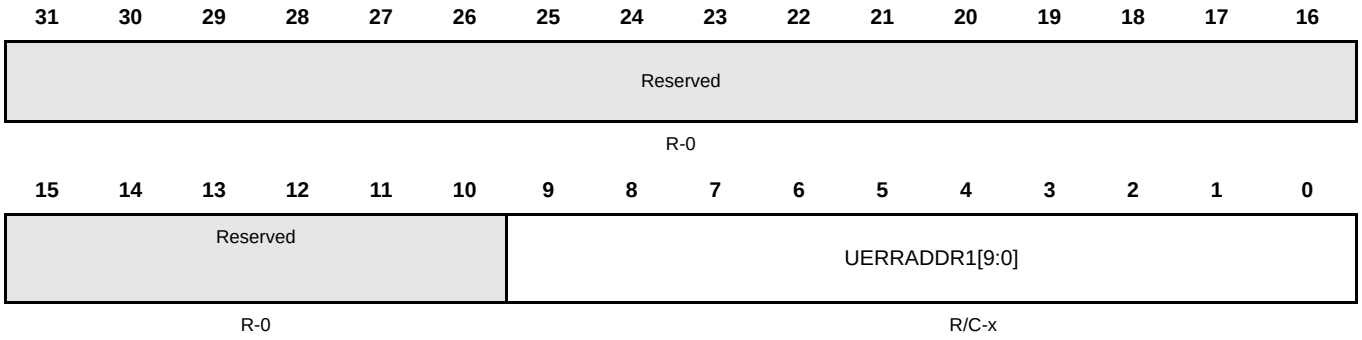
R = Read; C = Clear; -*n* = Value after reset

Table 14-38. Multi-buffer RAM Uncorrectable Parity Error Status Register (UERRSTAT) Field Descriptions

Bit	Name	Value	Description
31–2	Reserved		Reads return zero and writes have no effect.
1	EDFLG1		Uncorrectable parity error detection flag. This flag indicates if a parity error occurred in the RXRAM. **Note: Reading the UERRADDR1 register clears the EDFLG1 bit.**
		0	*Read:* No error has occurred. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* An error was detected and the address is captured in the UERRADDR1 register. *Write:* The bit is cleared to 0.
0	EDFLG0		Uncorrectable parity error detection flag. This flag indicates if a parity error occurred in the TXRAM. **Note: Reading the UERRADDR0 register clears the EDFLG0 bit.**
		0	*Read:* No error has occurred. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* An error was detected and the address is captured in the UERRADDR0 register. *Write:* The bit was cleared.

14.7.36 RXRAM Uncorrectable Parity Error Address Register (UERRADDR1)

Figure 14-57. RXRAM Uncorrectable Parity Error Address Register (UERRADDR1) [offset = 128h]



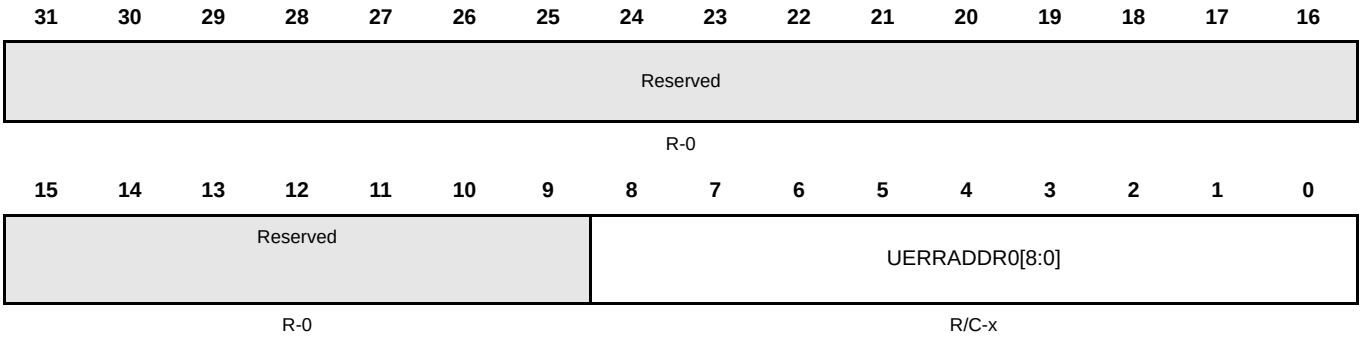
R = Read, C = Clear; -*n* = Value after power-on reset; -*x* = Interdeterminate

Table 14-39. RXRAM Uncorrectable Parity Error Address Register (UERRADDR1) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
9–0	OVERRADDR1 [9:0]	200–3FFh	**Uncorrectable parity error address for RXRAM.** This register holds the address where a parity error is generated while reading RXRAM. Only the CPU or DMA can read from RXRAM locations. The address captured is byte-aligned. This error address is frozen from being updated until it is read by the CPU. The offset address of RXRAM varies from 0x200 - 0x3FF. The register does not clear its contents during or after module-level reset, system-level reset or even power-up reset. A read operation to this register clears its contents to the default value 0x200. After a power-up reset the contents will be unpredictable. A read operation can be performed after power-up to keep the register at its default value, if required. However, the contents of this register are meaningful only when EDFLG1 is set to 1. **Note: A read of the UERRADDR1 register will clear EDFLG1 in the UERRSTAT register. However, in emulation mode when the SUSPEND signal is high, a read from the UERRADDR1 register does not clear EDFLG1.**

14.7.37 TXRAM Uncorrectable Parity Error Address Register (UERRADDR0)

Figure 14-58. TXRAM Uncorrectable Parity Error Address Register (UERRADDR0) [offset = 12Ch]



R = Read; C-Clear; *-n* = Value after power-on reset; *-x* = Indeterminate

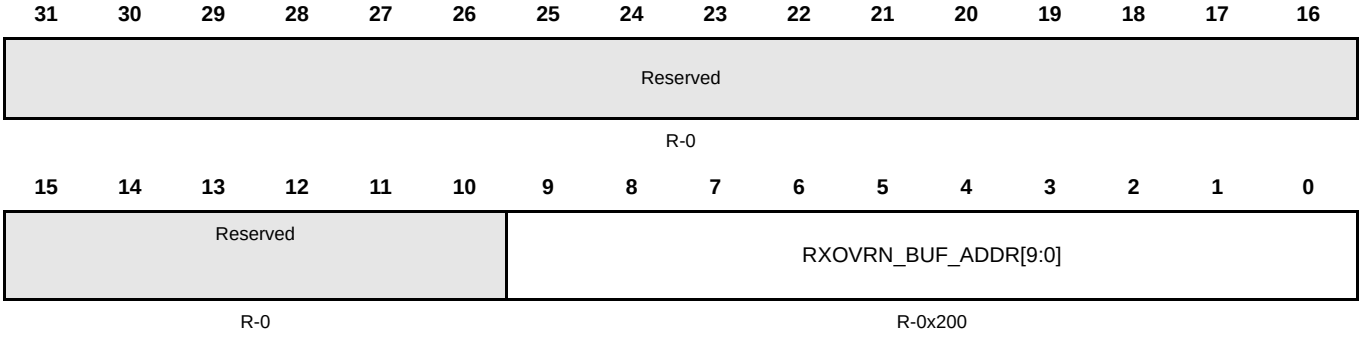
Table 14-40. TXRAM Uncorrectable Parity Error Address Register (UERRADDR0) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
8–0	UERRADDR1 [8:0]	0–1FFh	**Uncorrectable parity error address for TXRAM.** This register holds the address where a parity error is generated while reading from TXRAM. The TXRAM can be read either by CPU or by the MibSPI sequencer logic for transmission. The address captured is byte- aligned. This error address is frozen from being updated until it is read by the CPU. The offset address of TXRAM varies from 0x000 - 0x1FF. The register does not clear its contents during or after module-level reset, system-level reset, or even power-up reset. A read operation to this register clears its contents to all 0s. After a power-up reset, the contents of this register will be unpredictable. A read operation can be performed after power-up to clear the this register's contents, if required. However, the contents of this register are meaningful only when EDFLG0 is set to 1. **Note: A read from the UERRADDR0 register will clear EDFLG0 in the UERRSTAT register. However, in emulation mode when the SUSPEND signal is high, a read from the UERRADDR0 register does not clear EDFLG0.**

14.7.38 RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR)

In multi-buffer mode, if a particular RXRAM location is written by the MibSPI sequencer logic after the completion of a new transfer when that location already contains valid data, the RX_OVR bit will be set to 1 while the data is being written. The RXOVRN_BUF_ADDR register captures the address of the RXRAM location for which a receiver overrun condition occurred.

Figure 14-59. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) [offset = 130h]



R = Read, -*n* = Value after power-on reset

Table 14-41. RXRAM Overrun Buffer Address Register (RXOVRN_BUF_ADDR) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved		Reads return zero and writes have no effect.
9–0	RXOVRN_BUF_ADDR[9:0]	200–3FCh	**Address in RXRAM at which an overwrite occurred.** This address value will show only the offset address of the RAM location in the multi-buffer RAM address space. Refer to the device-specific data sheet for the actual absolute address of RXRAM. This word-aligned address can vary from 0x200 - 0x3FC. Contents of this register are valid only when any of the TGINTVECT0 or TGINTVECT1 and SPIFLG registers show an RXOVRN error vector while in multi-buffer mode. If there are multiple overrun errors, then this register holds the address of first overrun address until it is read. **Note: Reading this register clears the RXOVRN interrupt flag in the SPIFLG register and the TGINTVECTx.** **Note: Receiver overrun errors in multi-buffer mode can be completely avoided by using the *SUSPEND until RXEMPTY* feature, which can be programmed into each buffer of any TG. However, using the *SUSPEND until RXEMPTY* feature will make the sequencer wait until the current RXRAM location is read by the VBUS master before it can start the transfer for the same buffer location again. This may affect the overall throughput of the SPI transfer. By enabling the interrupt on RXOVRN in multi-buffer mode, the user can rely on interrupts to know if a receiver overrun has occurred. The address of the overrun in RXRAM is indicated in this RXOVRN_BUF_ADDR register.**

14.7.39 I/O-Loopback Test Control Register (IOLPBKTSTCR)

This register controls test mode for I/O pins. It also controls whether loop-back should be digital or analog. In addition, it contains control bits to induce error conditions into the module. These are to be used only for module testing.

All of the control/status bits in this register are valid only when the IO LPBK TST ENA field is set to 1010.

Figure 14-60. I/O-Loopback Test Control Register (IOLPBKTSTCR) [offset = 134h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SCS FAIL FLG	Reserved			CTRL BIT ERR	CTRLDES YNC	CTRL PAR ERR	CTRL TIME OUT	CTRL DLEN ERR
R-0							R/C-0	R-0			R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				IOLPBKTSTENA[3:0]				Reserved		ERR SCS PIN			CTRL SCS PIN ERR	LPBK TYPE	RXP ENA
R-0				R/WP-0101				R-0		R/WP-000			R/WP-0	R/WP-0	R/WP-0

R = Read, WP = Write in Privilege mode only, C = Clear; -*n* = Value after power-on reset

Table 14-42. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zero and writes have no effect.
24	SCS FAIL FLAG		Bit indicating a failure on $\overline{\text{SPISCS}}$ pin compare during analog loopback.
		0	*Read:* No miscompares occurred on any of the eight chip select pins (vs. the internal chip select number CSNR during transfers). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A comparison between the internal CSNR field and the analog looped-back value of one or more of the $\overline{\text{SPISCS}}$[7:0] pins failed. A stuck-at fault is detected on one of the $\overline{\text{SPISCS}}$[7:0]. Comparison is done only on the pins that are configured as functional and during transfer operation. *Write:* This flag bit is cleared.
23–221	Reserved		Reads return zero and writes have no effect.
20	CTRL BITERR		Controls inducing of BITERR during I/O loopback test mode.
		0	Do not interfere with looped-back data.
		1	Introduces bit errors by inverting the value of the incoming data during loopback.

Table 14-42. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions (Continued)

Bit	Name	Value	Description
19	CTRL DESYNC		Controls inducing of the desync error during I/O loopback test mode.
		0	Do not cause a desync error.
		1	Induce a desync error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 0 even after the transfer is complete. This forcing will be retained until the kernel reaches the idle state.
18	CTRL PARERR		Controls inducing of parity errors during I/O loopback test mode.
		0	Do not cause a parity error.
		1	Induce a parity error by inverting the polarity of the parity bit.
17	CTRL TIMEOUT		Controls inducing of the timeout error during I/O loopbacK test mode.
		0	Do not cause a timeout error.
		1	Induce a timeout error by forcing the incoming $\overline{\text{SPIENA}}$ pin (if functional) to remain 1 when transmission is initiated. The forcing will be retained until the kernel reaches the idle state.
16	CTRL DLENERR		Controls inducing of the data length error during I/O loopback test mode.
		0	Do not cause a data-length error.
		1	Induce a data-length error.
			Master mode: the $\overline{\text{SPIENA}}$ pin (if functional) is forced to 1 when the module starts shifting data.
			Slave mode: the incoming $\overline{\text{SPISCS}}$ pin (if functional) is forced to 1 when the module starts shifting data.
15–12	Reserved		Reads return zero and writes have no effect.
11–8	IOLPBKTSTENA[3:0]		Module I/O loopback test enable key.
		1010	Enable I/O loopback test mode.
		All other values	Disable I/O loopback test mode.
7–6	Reserved		Reads return zero and writes have no effect.

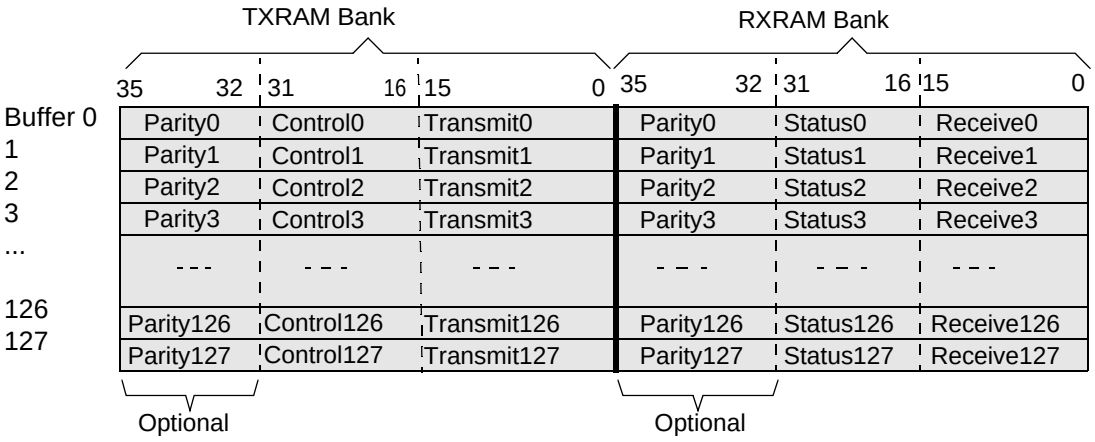
Table 14-42. I/O-Loopback Test Control Register (IOLPBKTSTCR) Field Descriptions (Continued)

Bit	Name	Value	Description
5–3	ERR SCS PIN[2:0]		Inject error on chip-select pin number x. The value in this field is decoded as the number of the chip select pin on which to inject an error. During analog loopback, if CTRL SCS PIN ERR bit is set to 1, then the chipselect pin selected by this field is forced to the opposite of its value in the CSNR.
		000	Select $\overline{\text{SPISCS}}$[0] for injecting error
		001	Select $\overline{\text{SPISCS}}$[1] for injecting error
		. . .	. . .
		111	Select $\overline{\text{SPISCS}}$[7] for injecting error.
2	CTRL SCS PIN ERR		Enable/disable the injection of an error on the $\overline{\text{SPISCS}}$[7:0] pins. The individual $\overline{\text{SPISCS}}$[7:0] pins can be chosen using the ERR SCS PIN field.
		0	Disable the $\overline{\text{SPISCS}}$[7:0] error-inducing logic.
		1	Enable the $\overline{\text{SPISCS}}$[7:0] error-inducing logic.
1	LPBK TYPE		**Module I/O loopback type (analog/digital).** See Figure 14-14 for the different types of loopback modes.
		0	Enable Digital loopback when IOLPBKTSTENA = 1010.
		1	Enable Analog loopback when IOLPBKTSTENA = 1010.
0	RXP ENA		Enable analog loopback through the receive pin. Note: This bit is valid only when LPBK TYPE = 1, which chooses analog loopback mode.
		0	Analog loopback is through the transmit pin.
		1	Analog loopback is through the receive pin.

14.8 Multi-Buffer RAM

The multi-buffer RAM is used for holding transit & received data, control and status information. The multi-buffer RAM contains two banks of up to 128 32-bit words for a maximum configuration. One bank (TXRAM) contains entries for transmit data (replicating the SPIDAT1 register). The other bank (RXRAM) contains received data (replicating the SPIBUF register). The buffers can be partitioned into multiple TGs, each containing a programmable number of buffers. Each of the buffers can be subdivided into an up to 16-bit transmit field, an up to 16-bit receive field, an up to 16-bit control field, and an up to 16-bit status field, as displayed in Figure 14-61.

Figure 14-61. Multi-Buffer RAM Configuration



All fields can be read and written with 8-bit, 16-bit, or 32-bit accesses.

The transmit fields can be written and read in the address range 000h to 1FFh. The transmit words contain data and control fields.

The receive RAM fields are read-only and can be accessed through the address range 200h to 3FCh. The receive words contain data and status fields.

The chip select number bit field CSNR[7:0] of the control field for a given word is mirrored into the corresponding receive-buffer status field after transmission.

14.8.1 Multi-buffer RAM Register Summary

This section describes the Multi-buffer RAM control and transmit-data fields of each word of TXRAM, and the status and receive-data fields of each word of RXRAM. The offset of the multi-buffer RAM is 0xFF0E 0000.

Figure 14-62. MibSPI RAM Register Summary

Offset Address† Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
Transmit Buffers																
0x00h–1FCh Buffer[0:127]	BUFMODE(2–0)			CS HOLD	LOCK	WDEL	DFSEL(1–0)		CSNR(7–0)							
	TXDATA(15–0)															
Receive Buffers:																
0x200–3FCh Buffer[0:127]	RXEM PTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARIT YERR	TIME OUT	DLEN ERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x204h Buffer 1	RXEM PTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARIT YERR	TIME OUT	DLEN ERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x3F8h Buffer 126	RXEM PTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARIT YERR	TIME OUT	DLEN ERR	LCSNR(7–0)							
	RXDATA(15–0)															
0x3FCh Buffer 127	RXEM PTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARIT YERR	TIME OUT	DLEN ERR	LCSNR(7–0)							
	RXDATA(15–0)															

14.8.2 Multi-buffer RAM Transmit Data Register

Each word of TXRAM is a transmit-buffer register.

Figure 14-63. Multi-buffer RAM Transmit Data Register [offset = Base + 000–1FFh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUFMODE			CS HOLD	LOCK	WDEL	DFSEL[1:0]		CSNR[7:0]							
R/W-0			R/W-0	R/W-0	R/W-0	R/W-0		R/W-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDATA[15:0]															
R/W-0															

R = Read, W = write, -*n* = Value after reset

Table 14-43. Multi-buffer RAM Transmit Data Register Field Descriptions

Bit	Name	Value	Description
31–29	BUFMODE		Specify conditions that are recognized by the sequencer to initiate transfers of each buffer word. When one of the “skip” modes is selected, the sequencer checks the buffer status every time it reads from this buffer. If the current buffer status (TXFULL, RXEMPTY) does not match, the buffer is skipped without a data transfer. When one of the “suspend” modes is selected, the sequencer checks the buffer status when it reads from this buffer. If TXFULL and/or RXEMPTY do not match, the sequencer waits until a match occurs. No data transfer is initiated until the status condition of this buffer changes.
		000	**disabled.** The buffer is disabled
		001	**skip single-transfer mode.** Skip this buffer until the corresponding TXFULL flag is set (i.e. new transmit data is available).
		010	**skip overwrite-protect mode.** Skip this buffer until the corresponding RXEMPTY flag is set (i.e. new receive data can be stored in RXDATA without data loss).
		011	**skip single-transfer overwrite-protect mode**. Skip this buffer until both of the corresponding TXFULL and RXEMPTY flags are set. (i.e. new transmit data available and previous data received by the host).
		100	**continuous mode.** Initiate a transfer each time the sequencer checks this buffer. Data words are retransmitted if the buffer has not been updated. Receive data is overwritten, even if it has not been read.

Table 14-43. Multi-buffer RAM Transmit Data Register Field Descriptions (Continued)

Bit	Name	Value	Description
		101	**suspend single-transfer mode.** Suspend-to-wait until the corresponding TXFULL flag is set (i.e. the sequencer stops at the current buffer until new transmit data is written in the TXDATA field).
		110	**suspend overwrite-protect mode.** Suspend-to-wait until the corresponding RXEMPTY flag is set (i.e. the sequencer stops at the current buffer until the previously-received data is read by the host.
		111	**suspend single-transfer overwrite-protect mode.** Suspend-to-wait until the corresponding TXFULL and RXEMPTY flags are set (i.e. the sequencer stops at the current buffer until new transmit data is written into the TXDATA field and the previously-received data is read by the host).
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only in compatibility-mode of MibSPI, (it is ignored in slave mode). CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed. If two consecutive transfers are dedicated to the same chip select this chip select signal will be deactivated for at least 2VCLK cycles before it is activated again.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select number equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared, or until the chip-select number changes.
27	LOCK		Lock two consecutive buffer words. Do not allow interruption by TG's with higher priority.
		0	Any higher-priority TG can begin at the end of the current transaction.
		1	A higher-priority TG cannot occur until after the next unlocked buffer word is transferred.

Table 14-43. Multi-buffer RAM Transmit Data Register Field Descriptions (Continued)

Bit	Name	Value	Description
26	WDEL		Enable the delay counter at the end of the current transaction. **Note: The WDEL bit is supported in master mode only. In slave mode, this bit will be ignored.**
		0	No delay will be inserted. However, $\overline{\text{SPISCS}}$ pins will still be de-activated for at least for 2VCLK cycles if CSHOLD = 0. **Note: The duration for which the SPISCS pin remains deactivated also depends upon the time taken to supply a new word after completing the shift operation (in compatibility mode). If TXBUF is already full, then the SPISCS will be deasserted for at least two VCLK cycles (if WDEL = 0).**
		1	After a transaction, WDELAY of the corresponding data format will be loaded into the delay counter. No transaction will be performed until the WDELAY counter overflows. The $\overline{\text{SPISCS}}$ pins will be de-activated for at least (WDELAY + 2) * VCLK_Period duration.
25–24	DFSEL[2:0]		Data word format select
		00	Data word format 0 is selected
		01	Data word format 1 is selected
		10	Data word format 2 is selected
		11	Data word format 3 is selected
23–16	CSNR[7:0]	0–FF	Chip select number. CSNR defines the chip-select that will be activated during the data transfer. **Note: Writing to only the control field (using byte writes) does not initiate any SPI transfer in master mode. This feature can be used to set up SPICLK phase or polarity before actually starting the transfer by only updating the DFSEL fields in the control field to select the required phase/polarity combination.**
15–0	TXDATA[15:0]	0–7FFFh	**Transfer data.** When written, these bits are copied to the shift register if it is empty. If the shift register is not empty, then they are held in TXBUF. SPIEN must be set to 1 before this register can be written to. Writing a 0 to SPIEN forces the lower 16 bits of SPIDAT1 to 0x0000. Write to this register ONLY when using the automatic slave chip-select feature (see Section 14.2, *Operating Modes* on page 630 for more information). A write to this register will drive the contents of CSNR[7:0] on the $\overline{\text{SPISCS}}$[7:0] pins, if they are configured as functional pins. When this register is read, the contents of TXBUF, which holds the latest data written, will be returned. **Note: Regardless of the character length, the transmit data should be right-justified before writing to the SPIDAT1 register.**

14.8.3 Multi-buffer RAM Receive Buffer Register

Each word of RXRAM is a receive-buffer register.

Figure 14-64. Multi-buffer RAM Receive Buffer Register [offset = RAM Base + 200–3FFh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXEMPTY	RX OVR	TX FULL	BIT ERR	DE SYNC	PARITYERR	TIME OUT	DLEN-ERR	LCSNR[7:0]							
RS-1	RC-0	R-0	RC-0	RC-0	RC-0	RC-0	RC-0	R-0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA[15:0)]															
R/W-0															

R = Read, W = write, C = Clear; S = Set; -*n* = Value after reset

Table 14-44. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
31	RXEMPTY		**Receive data buffer empty.** When the host reads the SPIBUF field or the entire SPIBUF register, it automatically sets the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, and the RXEMPTY flag is cleared.
		0	New data has been received and copied into the SPIBUF field.
		1	No data has been received since the last read of SPIBUF. This flag gets set to 1 under the following conditions: • Reading the RXDATA portion of the SPIBUF register. • Writing a 1 to clear the RXINTFLG bit in the SPIFLG register. Write-clearing the RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, RXINTFLG can be cleared by reading the RXDATA portion of SPIBUF (or the entire register).

Table 14-44. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into RXBUF while it is already full, RXOVR is set. Overruns always occur to RXBUF, not to SPIBUF; the contents of SPIBUF are overwritten only after it is read by the VBUSP master (e.g. CPU, DMA, or other host processor). If enabled, the RXOVRN interrupt is generated when RXBUF is overwritten, and reading either SPIFLG or SPIVECTx shows the RXOVRN condition. Two read operations from the SPIBUF register are required to reach the overwritten buffer word (one to read SPIBUF, which then transfers RXDATA into SPIBUF for the second read). This flag is cleared to 0 when the RXDATA is read. **Note: A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors such as TIMEOUT, BIT-ERR and DLEN_ERR occur, then RXOVR in RXBUF and SPIFLG registers will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receive overrun.**
		0	No receive data overrun condition occurred since last read of the data field.
		1	A receive data overrun condition occurred since last read of the data field.
29	TXFULL		**Transmit data buffer full.** This flag is a read-only flag. Writing into the SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the word is copied to the shift register, the TXFULL flag will be cleared. Writing to SPIDAT0 or SPIDAT1 when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		11	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		**Bit error.** There was a mismatch of internal transmit data and transmitted data.
		0	No bit error occurred. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**

Table 14-44. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
		1	A bit error occurred. The MibSPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (one-half clock cycle after the transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the BITERR flag is set. Possible reasons for a bit error include noise, an excessively high bit rate, capacitive load, or another master/slave trying to transmit at the same time.
27	DESYNC		**Desynchronization of slave device.** This bit is valid in master mode only. The master monitors the ENA signal coming from the slave device and sets the DESYNC flag if ENA is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master. **Note: There is a possible inconsistency of DESYNC in the Compatibility Mode MibSPI. Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt is generated when the buffer transfer is completed. But desynchronization is detected after the buffer transfer is completed. So, if the VBUS master reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.** **This inconsistency in the desync flag is valid only in the compatibility mode of MibSPI. In multi-buffer mode, the desync flag is always guaranteed to be for the current buffer.**
		0	No slave desynchronization detected. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		1	A slave device is desynchronized.
26	PARITYERR		**Parity error.** The calculated parity differs from the received parity bit. If the parity generator is enabled (selected individually for each buffer) an even or odd parity bit is added at the end of a data word. During reception of the data word, the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARITYERR flag is set. **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No parity error detected.
		1	A parity error occurred.

Table 14-44. Multi-buffer Receive Buffer Register Field Descriptions

Bit	Name	Value	Description
25	TIMEOUT		**Time-out because of non-activation of ENA pin.** The MibSPI generates a time-out when the slave does not respond in time by activating the ENA signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set. In addition, the TIMEOUT flag in the status field of the corresponding buffer and in the SPIFLG register is set. **This bit is valid only in master mode.** **This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.**
		0	No ENA-pin time-out occurred.
		1	An ENA signal time-out occurred.
24	DLENERR		**Data length error flag.** **Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.**
		0	No data-length error has occurred.
		1	A data length error has occurred.
23–16	LCSNR[7:0]	0–FFh	**Last chip select number.** LCSNR in the status field is a copy of CSNR in the corresponding control field. It contains the chip select number that was activated during the last word transfer. The updated after transmission during the write-back of received data.
15–0	RXDATA[15:0]	0–FFFFh	**SPI receive data.** This is the received word, transferred from the receive shift-register at the end of a transfer. Regardless of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

Analog To Digital Converter (ADC) Module

15.1 Overview

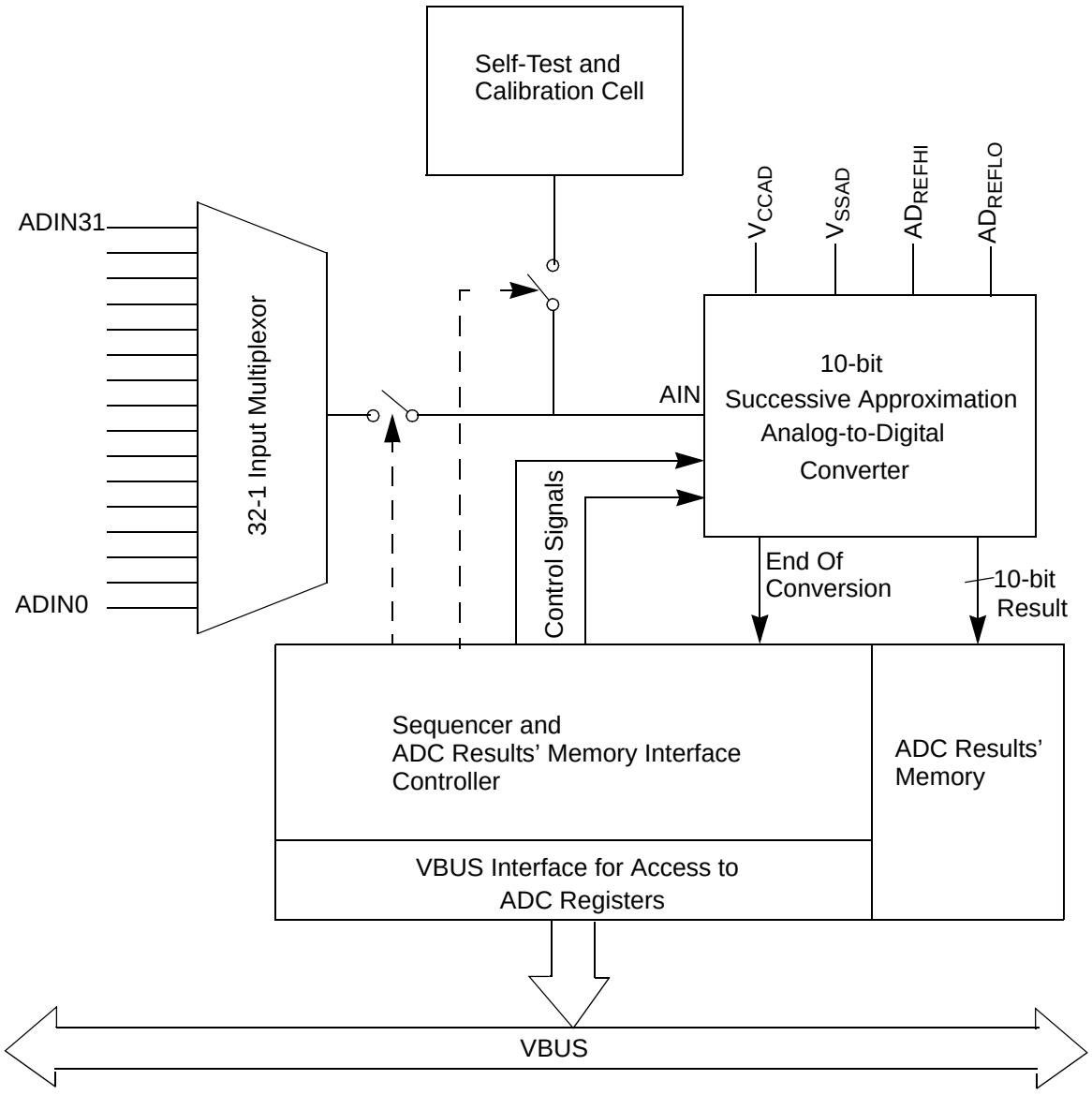
The main features of the ADC include:

- 10-bit resolution
- AD_{REFHI} and AD_{REFLO} pins (high and low reference voltages)
- Total Sample/Hold/Convert time: 1.55 μs Typical Minimum (see specific device datasheet)
- Three conversion groups: Group1 and Group2 software triggered by default, Event Group hardware triggered by default
- All three conversion groups can be configured to be hardware-triggered
- Programmer's model is ADC with three memory regions, one for each group
- Size of memory regions for each conversion group is programmable
- Memory regions are serviced by interrupt and/or by DMA.
- Conversion results can be read out of a FIFO structure for each group, or the ADC conversion results' memory can be read directly.
- Programmable interrupt threshold counter is available for each group.
- Programmable magnitude threshold interrupts
- Option to store channel ID along with data for enhanced robustness.
- Option to read either 8-bit or 10-bit values from memory regions - saves shift operations for software using only the eight MSBs of the conversion result.
- Multichannel conversions performed in ascending order, one channel at a time
- Single or continuous conversion modes
- One or two software-controlled conversions: up to 32 channels converted on a software request
- From one to three event-initiated conversions: up to 32 channels converted on an external event or internal timer event
- Programmable 5-bit ADC clock prescaler (ADCLK) to optimize conversion rate
- Programmable acquisition time for each conversion group
- Embedded self-test logic
- Embedded calibration logic
- Power-down mode
- External event pin (ADEVT) programmable as general-purpose I/O
- Up to eight device-specific hardware events to trigger conversions (refer device datasheet)

15.2 Introduction

This section presents a brief functional description of the analog-to-digital converter (ADC) module. Figure 15-1 illustrates the components of the ADC module.

Figure 15-1. ADC Block Diagram



15.2.1 Input Multiplexor

The input multiplexor (MUX) connects the selected input channel to the AIN input of the ADC core. The ADC module supports up to 32 inputs as shown in Figure 15-1. The sequencer selects the channel to be converted.

15.2.2 Self-Test and Calibration Cell

The ADC includes specific hardware for detecting open/short on an ADC analog input pin. It also allows the application program to calibrate the ADC. The self-test mode and the calibration modes are controlled by the sequencer. For more details, see the Section 15.8.1 and Section 15.8.2.

15.2.3 Analog-to-Digital Converter Core

The ADC core is a combination voltage scaling, charge redistribution Successive Approximation Register (SAR) based analog-to-digital converter. The 10-bit output code is determined 1 bit at a time, starting with the MSB (D9). The upper 4 bits are converted using a switched capacitor charge redistribution scheme. The lower 6 bits are derived from a voltage scaled reference (resistor string between AD_{REFHI} and AD_{REFLO}). By combining the two techniques an area-efficient low-power ADC is realized.

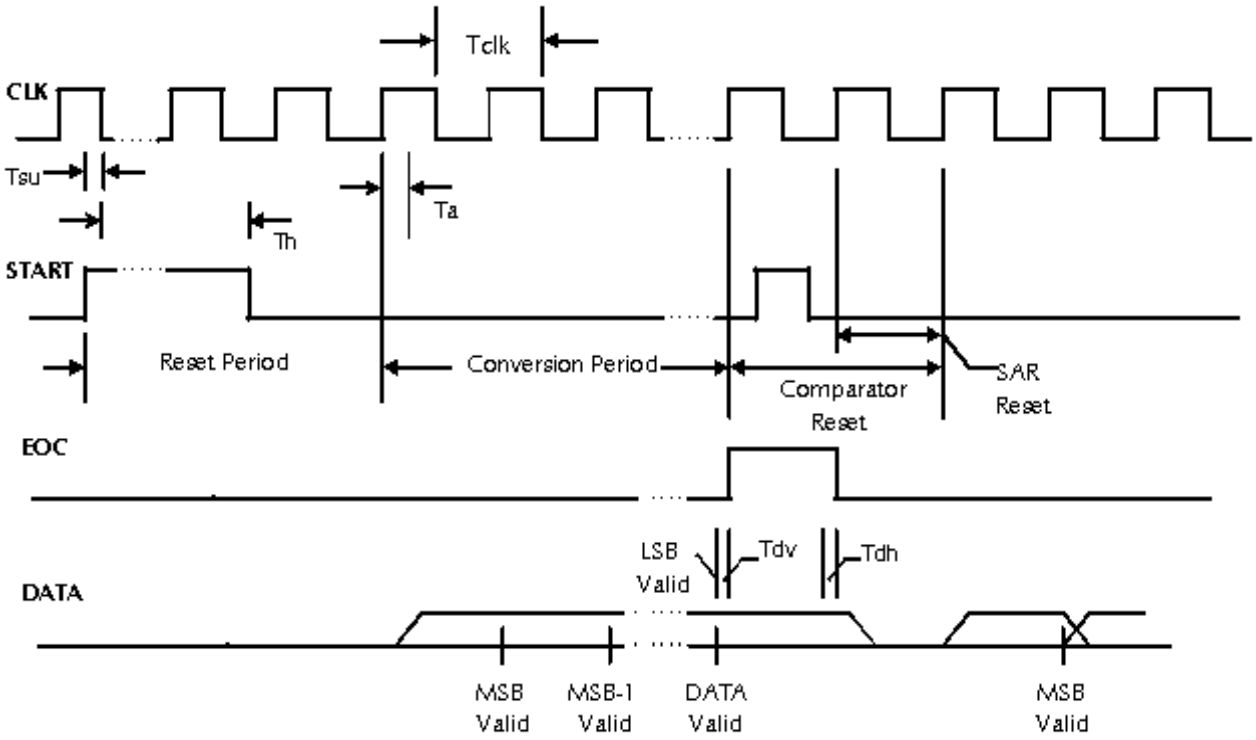
The analog conversion range is determined by the reference voltages: AD_{REFHI} and AD_{REFLO}. AD_{REFHI} is the top reference voltage and is the maximum analog voltage that can be converted. An analog input voltage equal to AD_{REFHI} results in an output code of $3FF_H$. AD_{REFLO} is the bottom reference voltage and is the minimum analog voltage that can be converted. Applying an input equal to AD_{REFLO} results in an output code of 000H. Both AD_{REFHI} and AD_{REFLO} must be chosen not to exceed the analog power supplies: V_{CCAD} and V_{SSAD}. Input voltages between AD_{REFHI} and AD_{REFLO} produce a conversion result that is proportional to the difference of [AD_{REFHI} – AD_{REFLO}]. The digital result of the conversion process is given by (EQ 1):

(EQ 1)

$$DigitalResult = \frac{1024x(InputVoltage - ADREFLO - 0.5LSB)}{(ADREFHI - ADREFLO)}$$

The Figure 15-2 shows the timing of the interface signals between the ADC core and the sequencer.

Figure 15-2. ADC Core Timing



The ADC sequencer logic generates the clock for the ADC core. This is the ADCLK signal, which can be scaled down from the VCLK to the desired value by configuring registers in the sequencer. The sequencer also provides the ADC core with a signal (START) to start the conversion. The duration of this signal is again controlled by register settings in the sequencer, and defines the Reset period as shown in Figure 15-2.

The analog input signal is sampled directly onto the switched capacitor array during the Reset period, providing an inherent Sample/Hold function. The duration of this Reset period is controlled by register settings in the sequencer. At the end of the last CLK period in the Conversion period (LSB clock period) the final (LSB) bit is stable and EOC (End-of-Conversion) is asserted by the ADC core.

Please refer to the device datasheet for ADC timing parameters applicable to your device.

15.2.4 Sequencer

The sequencer coordinates the operations of the ADC, including the input multiplexer, the ADC core, and the result memory. In addition, the logic of the sequencer sets the status register flags when the conversion is ongoing, stopped, or finished.

The logic handles CPU read/write operations, interrupts, halts, resets, memory allocation and handling. The logic in this block also supports self-test mode and calibration operations. This logic block generates the internal ADC clock that controls ADC timing.

All the features of the sequencer are discussed in detail in the following sections of this document.

15.3 Basic Features and Usage of the ADC

This section describes the usage of the basic features of the ADC module.

15.3.1 Conversion groups

The ADC module can service one of three conversion groups at any time: Group1, Group2, and the Event Group. Each conversion group can be configured in single-pass or repeated conversion modes. The Group1 and Group2 are software-triggered by default while the Event Group is hardware triggered by default.

15.3.2 How to setup the ADCLK speed and the acquisition time?

The ADC sequencer generates the clock for the ADC core. This is the ADCLK. The ADCLK frequency must not exceed 20MHz. Refer to the specific device datasheet to find the actual maximum ADCLK frequency specification.

The ADCLK is generated by dividing down the input clock to the ADC module, which is the VBUSP interface clock, VCLK. A 5-bit field in the ADCLOCKCR, bits [4:0], is used to divide down the VCLK by 1 up to 32.

The signal acquisition time for each group is separately configurable using the ADG1SAMP[11:0], ADG2SAMP[11:0], and ADEVSAMP[11:0] registers. The acquisition time is specified in terms of ADCLK cycles and ranges from a minimum of 2 ADCLK cycles to a maximum of 4098 ADCLK cycles.

For example, Group1 acquisition time, t_{ACQG1} = ADG1SAMP[11:0] + 2, in ADCLK cycles

15.3.3 How to select an input channel for conversion?

The ADC module needs to be enabled first before selecting an input channel for conversion. The ADC module can be enabled by setting the ADC EN bit in the ADC Operating Mode Control Register (ADOPMODECR). Multiple input channels can be selected for conversion in each group. Only one input channel is converted at a time. The channels to be converted are configured in one or more of the three conversion groups' channel selection registers. Channels to be converted in Group1 are configured in the Group1 Channel-Select Register (ADG1SEL), those to be converted in Group2 are configured in the Group2 Channel-Select Register (ADG2SEL), and those to be converted in the Event Group are configured in the Event Group Channel-Select Register (ADEVSEL).

15.3.4 How to start a conversion?

The conversion groups Group1 and Group2 are software-triggered by default. A conversion in these groups can be started just by writing the desired channels to the respective Channel-Select Registers. For example, in order to convert channels 0, 1, 2, and 3 in Group1 and channels 8, 9, 10, and 11 in Group2, the application just has to write 0x0000000F to ADG1SEL and 0x00000F00 to ADG2SEL. The ADC module will start by servicing the group that was triggered first, Group1 in this example.

The conversions for all groups are performed in ascending order of the channel number. For the Group1 the conversions will be performed in the order: channel 0 first, followed by channel 1, then channel 2, and then channel 3. The Group2 conversions will be performed in the order: channels 8, 9, 10, and 11.

The Event Group is hardware-triggered. There are up to eight hardware event trigger sources defined for the ADC module. Please check the specific device datasheet to identify the trigger sources available.

The trigger source to be used needs to be configured in the ADEVSRC register. Similar registers also exist for the Group1 and Group2 as these can also be configured to be event-triggered.

The polarity of the event trigger is also configurable, with a falling edge being the default.

An Event Group conversion starts when at least one channel is selected for conversion in this group, and when the defined event trigger occurs.

15.3.5 How are results stored in the results' memory?

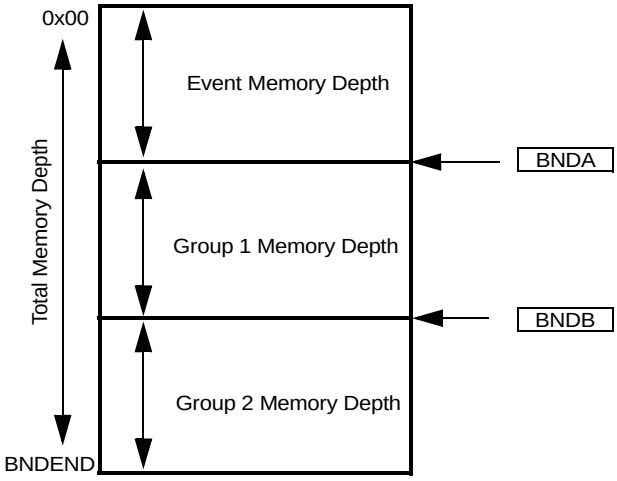
The ADC stores the conversion results in three separate memory regions in the ADC Results' RAM, one region for each group. Each memory region is a stack of buffers, with each buffer capable of holding one

conversion result. The number of buffers allocated for each group is programmed by configuring the ADC module registers ADBNDCR and ADBNDEND.

ADBNDCR contains two 9-bit pointers BNDA and BNDB. They are used to partition the total memory available into three memory regions as shown in Figure 15-3. Both BNDA and BNDB are pointers referenced from the start of the results' memory. BNDA specifies the number of buffers allocated for the Event Group conversion results in units of two buffers; BNDB specifies the number of buffers allocated for the Event Group plus Group1 in units of two buffers. Please refer to the Section 15.10.20 for more details on configuring the ADC results' memory.

ADBNDEND contains a 3-bit field called BNDEND that configures the total memory available. The ADC module can support up to 1024 buffers. Please refer to the specific device datasheet for the actual size of the ADC Results' RAM implemented on the device.

Figure 15-3. FIFO Implementation



15.3.6 How to read the results from the results' memory?

The CPU can read the conversion results in one of two ways:

a) By reading the group's conversion results from a FIFO queue, or

b) By reading the group's conversion results directly by accessing the correct ADC RAM location

15.3.6.1 Reading conversion results from the FIFO

The conversion results for each group can be accessed via a range of addresses provided to facilitate the use of the ARM Load-Multiple (LDM) instruction. A single read performed using the LDR instruction can be used to read out a single conversion result. The results are read out from the group's memory region as a FIFO queue by reading from any location inside this address range. The conversion result that got stored first gets read first. A result that is read from the memory in this method is removed from the memory. For example, a read from any address between offset 0x90 and 0xAF pulls out one conversion result from the Event Group memory.

Also, for debug purposes, each buffer has an emulation address that returns the next conversion result from the buffer without removing the result from the buffer itself. For example, reading from offset 0xF0 returns the next result in the Event Group buffer but does not actually remove that result from the buffer or change the amount of data held in the buffer.

15.3.6.2 Reading conversion results from the RAM

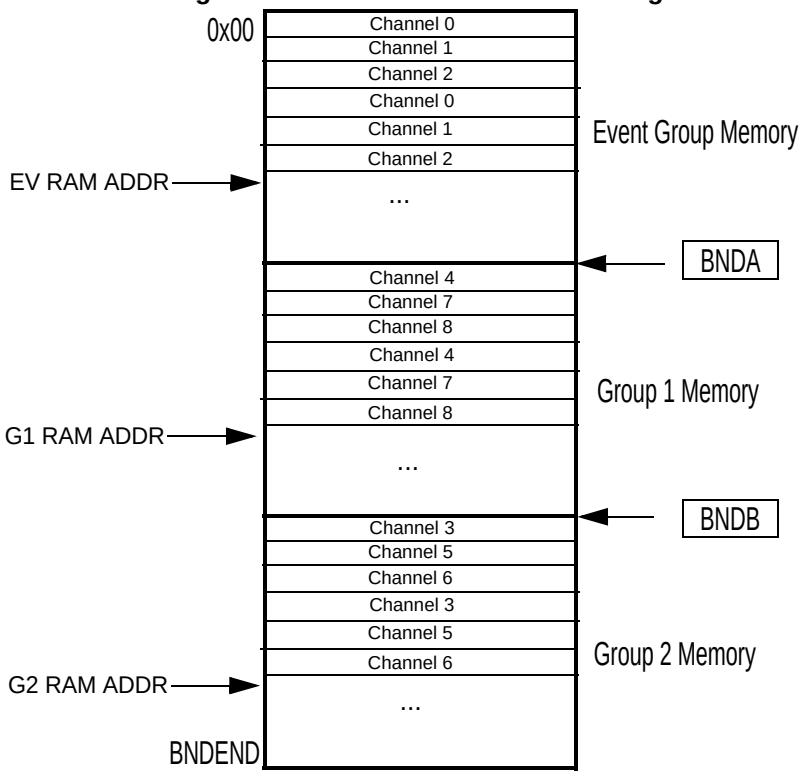
The ADC RAM is part of the device's memory map. Refer to the device datasheet to identify the ADC RAM's location in the specific device's memory map. The application needs to identify the address ranges for each

of the three memory regions for the three conversion groups after performing the segmentation as described in Section 15.3.5. It is up to the application to read the desired results from the three conversion groups. A separate register for each group holds the ADC RAM address for that group where the ADC will write the next conversion result. The application can therefore calculate how many valid conversion results are available to be read. This mode also allows the application to selectively read the conversion results for any particular input channel of interest without having to read other channels' conversion results first.

15.3.6.3 Example

Suppose that channels 0, 1, and 2 are selected for conversion in the Event Group, channels 4, 7, and 8 are selected for conversion in group 1, and channels 3, 5, and 6 are selected for conversion in group 2. The conversion results will get stored in the three memory regions as shown below:

Figure 15-4. Conversion results storage



Suppose that the CPU wants to read out the results for the Event Group from a FIFO queue. The CPU needs to read from any address in the range 0x90 to 0xAF multiple times, or do a "load multiple" from this range of addresses. This will cause the ADC to return the results for channel 0, then channel 1, then channel 2, then channel 0, and so on for each read access to this address range.

Now suppose that the application wants to read out the results for the group 1 from the RAM directly. The conversion results for the group 1 are accessible starting from address ADC RAM Base Address + BNDA. Also, it is known that the first result at this address is for the input channel 4, the next one is for input channel 7, and so on. So the application can selectively read the conversion results for only one channel if so desired.

15.3.7 How to stop a conversion?

A group's conversion can be stopped by clearing the group's channel select register. Writing a non-zero value to the group's channel select register causes the group's results' memory to be reset.

15.3.8 Example Sequence for Basic Configuration of ADC Module

The following sequence is necessary to configure the ADC to convert channels 0, 2, 4, and 8 in single-conversion mode using Group1:

1. Write 0 to the Reset Control Register (ADRSTCR) to release the module from the reset state
2. Write 1 to the Operating Mode Control Register (ADOPMODECR) to enable the ADC state machine
3. Optional step: Configure the ADCLK frequency by programming the desired divider into the Clock Control Register (ADCLOCKCR)
4. Optional Step: Configure the acquisition time for the group that is to be used. For example, configure the Group1 Sampling Time Control Register (ADG1SAMP) to set the acquisition time for Group1.
5. Select the channels that need to be converted in Group1 by writing to the Group1 Channel Select Register (ADG1SEL). In this example, a value of 0x115 needs to be written to ADG1SEL in order to select channels 0, 2, 4, and 8 for conversion in Group1.
6. Wait for the GP1 END bit to be set in the Group1 Conversion Status Register (ADG1SR). This bit gets set when all the channels selected for conversion in Group1 are converted and the results are stored in the Group1 memory.
7. Read the conversion results by reading from the Group1 FIFO access location or by reading directly from the Group1 results' memory.

15.4 Advanced Conversion Group Configuration Options

The below Table 15-1 shows the operating mode control registers and the status registers for each of the three conversion groups.

Table 15-1. ADC Groups' Operating Mode Control and Status Registers

Offset Address / Register	31 / 15	30 / 14	29 / 13	28 / 12	27 / 11	26 / 10	25 / 9	24 / 8	23 / 7	22 / 6	21 / 5	20 / 4	19 / 3	18 / 2	17 / 1	16 / 0
0x010 ADEVMODECR	Reserved															
	Reserved										EV CHID	OVR EV RAM IGN	Reserved	EV 8BIT	EV MODE	FRZ EV
0x014 ADG1MODECR	Reserved															
	Reserved										G1 CHID	OVR G1 RAM IGN	G1 HW TRIG	G1 8BIT	G1 MODE	FRZ G1
0x018 ADG2MODECR	Reserved															
	Reserved										G2 CHID	OVR G2 RAM IGN	G2 HW TRIG	G2 8BIT	G2 MODE	FRZ G2
0x06C ADEVSR	Reserved															
	Reserved												EV MEM EMPTY	EV BUSY	EV STOP	EV END
0x070 ADG1SR	Reserved															
	Reserved												G1 MEM EMPTY	G1 BUSY	G1 STOP	G1 END
0x074 ADG2SR	Reserved															
	Reserved												G2 MEM EMPTY	G2 BUSY	G2 STOP	G2 END

The following sections describe each of these group configuration options separately.

15.4.1 Single or Continuous Conversion Modes

The EV MODE, G1 MODE, and G2 MODE bits are used to select between either single or continuous conversion mode for each of the three groups.

15.4.1.1 Single Conversion Mode

A conversion group configured to be in single-conversion mode gets serviced only once by the ADC for each group trigger. The trigger can be a software trigger as in the case of Group1 and Group2 by default, or it could be a hardware event trigger as in the case of the Event Group.

The entire conversion sequence, from the acceptance of the group conversion request to the end of the last channel's conversion, is flagged for each group by the corresponding BUSY bit in that group's status register. After single-conversion mode is started, the BUSY bit is read as 1 until the conversion of the last channel is complete. The END bit for the group is set once all the channels in that group are converted.

For example, say channels 0, 2, 4, and 6 are selected for conversion in Group1 in single-conversion mode. When the Group1 gets serviced, the ADC will start conversion for channel 0, then channel 2, then channel 4, and then channel 6. It will then stop servicing the Group1, set the GP1 END status bit, and look to service the Event Group or the Group2, if required.

15.4.1.2 Continuous Conversion Mode

A conversion group configured to be in continuous-conversion mode gets serviced by the ADC continuously. The group still needs to be triggered appropriately for the first conversion to start. The conversions are performed continuously thereafter.

The entire conversion sequence, from the acceptance of the group conversion request to the end of the last channel's conversion, is flagged for each group by the corresponding BUSY bit in that group's status register. After continuous-conversion mode is started, the BUSY bit is read as 1 as long as the continuous-conversion mode for this group is selected.

As an example, say the channels 0, 2, 4, and 6 are selected for conversion in Group1, now in continuous-conversion mode. When the Group1 gets serviced, the ADC will complete conversions for channels 0, 2, 4 and 6, and then look to service the Event Group or the Group2. Once it is done servicing the Event Group or the Group2, it will return to service the Group1 again. The Group1 does not need to be triggered again for the repeated conversion.

Note: Configuring all conversion groups in continuous conversion mode
All the three groups cannot operate in continuous-conversion mode at the same time. If the application program configures all three groups to be in continuous-conversion mode, the Group2 is automatically reset to single-conversion mode, and the G2 MODE bit in the ADG2MODECR register is cleared to reflect the single-conversion mode of Group2.

15.4.2 Conversion Group Freeze Capability

Each conversion group can be configured to allow its conversions to be frozen whenever there is a request for conversion in another group.

For example, setting the FRZ EV bit in the ADEVMODECR register will allow the ADC to freeze ongoing Event Group conversions whenever there is a pending request, or a new request for a Group1 or Group2 conversion. The conversions for the Event Group will be frozen as long as the Group1 or Group2 conversions are active. Once the Group1 or Group2 conversions are completed, the Event Group conversions start from where they were frozen.

While a group's conversions are frozen, the group's STOP status bit is set. This bit is cleared once the group's conversions are restarted.

15.4.3 8-bit or 10-bit Result Mode

Some applications can use only the ADC conversion results as a byte. The ADC module can automatically shift a group's conversion results right by two bits as the result is being read out of the memory location. This eliminates the need for the application program to shift each conversion result itself, and results in more efficient code execution.

The 8-bit result mode is enabled by setting the 8BIT bit in the particular group's operating mode control register.

15.4.4 Group Memory Overrun Option

An overrun condition occurs when the ADC module tries to store more conversion results to a group's results' memory which is already full. In this case, the ADC allows two options.

If the OVR RAM IGN bit in the group's operating mode control register is set, then the ADC module ignores the contents of the group's results' memory and wraps around to overwrite the memory with the results of new conversions.

If the OVR RAM IGN bit is not set, then the application program has to read out the group's results' memory upon an overrun condition. Only then can the ADC continue to write new results to the memory.

15.4.5 Group Channel Id Storage Option

The ADC module allows the application program to also read out the analog input channel number along with its conversion result. This capability is enabled by setting the CHID bit in the group's operating mode control register.

If the CHID bit is not set, the bits [14-10] are forced to 00000 when the conversion results are read out from the group's results' memory.

If the CHID bit is set, the bits [14-10] in the group's results' memory contain the input channel number to which the conversion result belongs.

Note: Actual Storage of Channel Id

Regardless of whether the CHID bit is set or not, the channel number is **always stored** in the memory along with the conversion result. The CHID bit only affects whether the channel number is available with the conversion result **when the group's memory is read**. Therefore, the CHID bit for a group can be changed dynamically without affecting that group's ongoing conversions.

15.4.6 Group Trigger Options

The Group1 and Group2 operating mode control registers have an extra control bit: HW TRIG. This bit configures the group to be hardware event-triggered instead of software-triggered, which is the default.

When a group is configured to be event-triggered, the group conversion starts when at least one channel is selected for conversion in this group, and when the defined event trigger occurs. The event trigger source is defined for each group in the ADEVSRC, ADG1SRC and the ADG2SRC registers. The actual mappings of the available event trigger sources to a device's internal timer events are defined in the specific device specification.

15.5 ADC Module Basic Interrupts

This section describes the basic interrupts generated by the ADC module. The ADC module also has the capability to generate two more interrupts. These are discussed in the ADC Advanced Features section.

Table 15-2 shows the basic interrupt control and interrupt flag registers for the three groups. The base address for the control registers is 0xFFF7 C000.

Table 15-2. ADC Groups' Interrupt Control and Status Registers

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x028 ADEVINTENA	Reserved															
	Reserved												EV END INT EN	Reserved	EV OVR INT EN	EV THR INT EN
0x034 ADEVINTFL:G	Reserved															
	Reserved												EV END	EV MEM EMPTY	EV MEM OVER-FLOW	EV THR INT FLAG
0x040 ADEVINTCR	Reserved															
	Sign Extension							EVTHR[8-0]								
0x02C ADG1INTENA	Reserved															
	Reserved												G1 END INT EN	Reserved	G1 OVR INT EN	G1 THR INT EN
0x038 ADG1INTFLG	Reserved															
	Reserved												G1 END	G1 MEM EMPTY	G1 MEM OVER-FLOW	G1 THR INT FLAG
0x044 ADG1INTCR	Reserved															
	Sign Extension							G1THR[8-0]								
0x030 ADG2INTENA	Reserved															
	Reserved												G2 END INT EN	Reserved	G2 OVR INT EN	G2 THR INT EN

0x03C ADG2INTFLG	Reserved					
	Reserved		G2 END	G2 MEM EMPTY	G2 MEM OVER-FLOW	G2 THR INT FLAG
0x048 ADG2INTCR	Reserved					
	Sign Extension	G2THR[8-0]				

15.5.1 Group Conversion End Interrupt

The ADC module sets the group's conversion end flag (EV END, G1 END, or G2 END) in that group's interrupt flag register when all the channels selected for conversion in that group are converted. This causes a group conversion end interrupt to be generated if this interrupt is enabled by setting the group's END INT EN control bit (EV END INT EN, G1 END INT EN, or G2 END INT EN).

This interrupt can be easily used for conversion groups configured to be in the single-conversion mode. The application program can read out the conversion results, change the group's configuration if necessary, and restart the conversions by triggering the group from within the interrupt service routine.

15.5.2 Group Memory Threshold Interrupt

The ADC module has the ability to generate an interrupt for a fixed number of conversions for each group. A group memory threshold register determines how many conversion results must be in a group's memory region before the CPU is interrupted. This feature can be used to significantly reduce the CPU load when using interrupts for reading the conversion results.

The group's threshold register needs to be configured before the group conversions are triggered. This threshold register value behaves like a down-counter, which decrements each time the ADC writes a conversion result to this group's memory. This counter is incremented each time the application program reads a conversion result from the results' memory by accessing the FIFO queue. Simultaneous read (by application program) and write (by ADC module) operations from the group's results' memory leave the threshold counter unchanged.

The threshold counter can decrement past 0 and become negative. It always increments back to its original value when the memory region is emptied. To determine how many samples are in the memory region at a given moment, the threshold counter can be subtracted from the originally configured threshold count.

Whenever the threshold counter transitions from +1 to 0, it sets the group's threshold interrupt flag, and the CPU is interrupted if the group's threshold interrupt is enabled. The CPU is expected to clear the interrupt flag after reading the conversion results from the memory.

The interrupt flag is not set when the threshold counter stays at 0 or transitions from -1 to 0.

15.5.3 Group Memory Overrun Interrupt

An interrupt can be generated for each group if the number of ADC conversions for that group exceed the number of buffers allocated for that conversion group. The application program can choose to read out all the conversion results using the CPU or the DMA. Alternatively, the application program can set the group's OVR RAM IGN bit and allow the ADC module to overwrite the group's results' memory contents with new conversion results.

15.6 ADC Magnitude Threshold Interrupts

The ADC allows magnitude threshold interrupts to be generated. The magnitude comparison can be performed on up to six channels. The comparison parameters are programmed via the Magnitude Threshold Control Register. If the conversion result for the selected channel meets the condition programmed, an interrupt is generated.

The below table shows the control and status registers for the magnitude threshold interrupt feature.

Table 15-3. Control and Status Registers for Magnitude Threshold Interrupts

Offset Address / Register	31 / 15	30 / 14	29 / 13	28 / 12	27 / 11	26 / 10	25 / 9	24 / 8	23 / 7	22 / 6	21 / 5	20 / 4	19 / 3	18 / 2	17 / 1	16 / 0
0x128 AD MAG INT CR1	Reserved	MAG CHID1.4:0					MAG THRESHOLD1.9:0									
	Reserved			COMP CHID1.4:0					Reserved						CHN/ THR COMP	COMP GE/LT1
0x12C AD MAG INT1 MASK																
							MAG INT1 MASK.9:0									
0x130 AD MAG INT CR2	Reserved	MAG CHID2.4:0					MAG THRESHOLD2.9:0									
	Reserved			COMP CHID2.4:0					Reserved						CHN/ THR COMP	COMP GE/LT2
0x134 AD MAG INT2 MASK																
							MAG INT2 MASK.9:0									
0x138 AD MAG INT CR3	Reserved	MAG CHID3.4:0					MAG THRESHOLD3.9:0									
	Reserved			COMP CHID3.4:0					Reserved						CHN/ THR COMP	COMP GE/LT3
0x13C AD MAG INT3 MASK																
							MAG INT3 MASK.9:0									
0x140 AD MAG INT CR4	Reserved	MAG CHID4.4:0					MAG THRESHOLD4.9:0									
	Reserved			COMP CHID4.4:0					Reserved						CHN/ THR COMP	COMP GE/LT4

0x144 AD MAG INT4 MASK																	
												MAG INT4 MASK.9:0					
0x148 AD MAG INT CR5	Reserved		MAG CHID5.4:0						MAG THRESHOLD5.9:0								
	Reserved					COMP CHID5.4:0				Reserved						CHN/ THR COMP	COMP GE/LT5
0x14C AD MAG INT5 MASK																	
												MAG INT5 MASK.9:0					
0x150 AD MAG INT CR6	Reserved		MAG CHID6.4:0						MAG THRESHOLD6.9:0								
	Reserved					COMP CHID6.4:0				Reserved						CHN/ THR COMP	COMP GE/LT6
0x154 AD MAG INT6 MASK																	
												MAG INT6 MASK.9:0					
0x158 AD MAG THR INTENA SET													MAG INTENA SET6	MAG INTENA SET5	MAG INTENA SET4	MAG INTENA SET3	MAG INTENA SET2	MAG INTENA SET1
0x15C AD MAG THR INTENA CLR													MAG INTENA CLR6	MAG INTENA CLR5	MAG INTENA CLR4	MAG INTENA CLR3	MAG INTENA CLR2	MAG INTENA CLR1
0x160 AD MAG THR INTFLG													MAG THR INT FLG6	MAG THR INT FLG5	MAG THR INT FLG4	MAG THR INT FLG3	MAG THR INT FLG2	MAG THR INT FLG1
0x164 AD MAG THR INT OFFSET															INT OFF.3	INT OFF.2	INT OFF.1	INT OFF.0

15.6.1 Magnitude Threshold Interrupt Configuration

The following control fields are configurable for each of the six available magnitude threshold interrupts:

1. CHN/THR COMP: Specifies whether to compare two channels' conversion results, or to compare a channel's conversion result to a programmable threshold value.A value of 0 will select the programmable threshold to be compared, and a value of 1 will select the conversion result of the channel identified by the COMP CHID field to be compared.
2. MAG CHID: Specifies the channel number from 0 to 31 whose conversion result needs to be monitored.
3. COMP CHID: Specifies the channel number from 0 to 31 whose last conversion result is used for the comparison with the conversion result of the channel being monitored.
4. MAG THRESHOLD: Specifies the value for comparison with the conversion result of the channel identified by the MAG CHID field.
5. CMP GE/LT: Specifies whether the conversion result of the channel identified by MAG CHID is compared to be "greater than or equal to", or "less than" the reference value. The reference value can be the conversion result of another channel identified by the COMP CHID field, or it could be a threshold value specified in the MAG THRESHOLD field. A value of 0 in the CMP GE/LT field indicates a "less than" comparison and a value of 1 indicates a "greater than or equal to" comparison.

15.6.2 Magnitude Threshold Interrupt Comparison Mask Configuration

There is also a separate comparison mask register for each of the six magnitude threshold interrupts. This register is used to specify the bits that are masked off for the sake of the comparison. For example, the lower 4 bits of the conversion result can be masked off by writing 0xf to the interrupt comparison mask register, allowing a gross comparison to be made. By default, all the 10 bits of conversion are compared.

15.6.3 Magnitude Threshold Interrupt Enable / Disable Control

Each of the six magnitude interrupts also have separate interrupt enable set and clear registers. These are used to respectively enable and disable that particular magnitude threshold interrupt from being generated. To enable a magnitude threshold interrupt, write a 1 to the corresponding bit of the interrupt enable set register. Conversely, to disable a magnitude threshold interrupt, write a 1 to the corresponding bit of the interrupt enable clear register.

15.6.4 Magnitude Threshold Interrupt Flags

There is a separate register that holds the flags for these six interrupts. This flag gets set whenever the comparison condition for the corresponding interrupt is met. A magnitude threshold interrupt is generated if the corresponding flag is set inside the flag register, and the interrupt generation is enabled. This flag can be cleared by writing a 1 to the flag or by reading from the interrupt offset register in case of this interrupt being the current highest-priority pending interrupt.

15.6.5 Magnitude Threshold Interrupt Offset Register

It is possible to have multiple magnitude threshold interrupts pending at the same time. The magnitude threshold interrupt offset register holds the index of the currently pending highest priority magnitude threshold interrupt. The magnitude threshold interrupt 1 has the highest priority while the magnitude threshold interrupt 6 has the lowest priority. This is a read-only register and returns zeros if none of the magnitude threshold interrupts are pending. Writes to this register have no effect.

A read from this register updates the register to the next highest-priority pending magnitude threshold interrupt. This read also clears the corresponding flag from the magnitude threshold interrupt flag register. However, a read from the magnitude threshold interrupt offset register in emulation mode does not affect the interrupt flag register or the interrupt offset register.

15.7 ADC Results' RAM Special Features

The following sections describe some of the special features supported by the ADC module to enhance the results' RAM testability and integrity.

15.7.1 ADC Results' RAM Auto-Initialization

The ADC module allows the application to auto-initialize the ADC results' RAM to all zeros. The application must ensure that the ADC module is not in any of the conversion modes before triggering off the auto-initialization process.

The auto-initialization sequence is as follows:

1. Enable the global hardware memory initialization key by programming a value of 1010 to the bits [3-0] of the MINITGCR register of the System module.
2. Set the control bit for the ADC results' RAM in the MSINENA System module register. This bit is device-specific for each memory that support auto-initialization. Please refer to the device datasheet to identify the control bit for the ADC results' RAM. This starts the initialization process. The BUF INIT ACTIVE flag in the ADC module ADBNDEND register will get set to reflect that the initialization is ongoing.
3. When the memory initialization is completed, the corresponding status bit in the MINISTAT register will be set. Also, the BUF INIT ACTIVE flag will get cleared.

Please refer to the TMS470Px Platform Architecture Specification for more details on the memory auto-initialization process.

15.7.2 ADC Results' RAM Test Mode

In the defined conversion modes of the ADC, the application can only read from the ADC results' RAM. Only the ADC module is allowed to write to the results' RAM. A special test mode is defined to allow the application to also write into the ADC results' RAM - this mode is the ADC Results' RAM Test Mode. Only 32-bit reads and writes are allowed to the ADC results' RAM in this test mode.

Note: Contention on access to ADC Results' RAM

The ADC module cannot handle a contention between the application write to the results' RAM and the ADC writing a conversion result to the results' RAM. The application must ensure that the ADC is not likely to write a new conversion result to the results' RAM when the ADC Results' RAM Test Mode is enabled.

The ADC Results' RAM Test Mode is enabled by setting the RAM TEST EN bit in the ADC OP MODE CR.

15.8 ADC Special Modes

The ADC module supports some special modes for diagnostic and power saving purposes.

15.8.1 ADC Error Calibration Mode

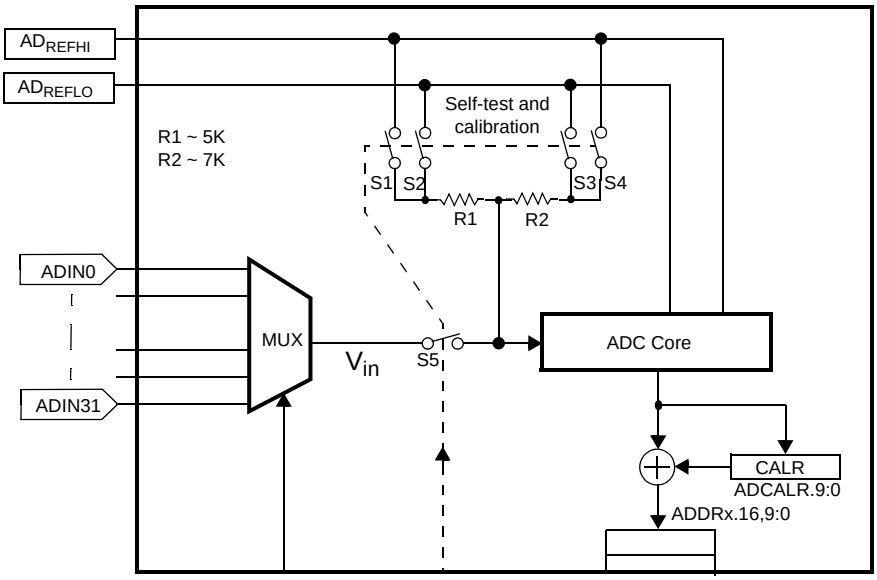
The application program can activate a calibration sequence any time self-test mode is disabled (SELF TST = 0). This calibration sequence includes the conversion of an embedded calibration reference voltage followed by the calculation of an offset error correction value.

Note: Disable Self-Test Mode Before Calibration

To avoid errors during the calibration operation, self-test mode must *not* be enabled during a calibration sequence. In addition, to ensure accurate results, calibrate the ADC in an environment with minimum noise.

Calibration mode is enabled by setting the CAL EN bit (ADCALCR.0). The application needs to ensure that no conversion group is being serviced when the calibration mode is enabled.

Figure 15-5. Self-Test and Calibration Logic



The input multiplexer gets disabled and only the reference voltage is connected to the ADC core input. Switch S5 of Figure 15-5 is opened. In addition, the digital result issued from a conversion is output from the ADC core to the calibration and offset error correction register, ADCALR. The ADC results' memory is not affected by the calibration conversion.

When calibration mode is disabled, the ADC can be configured for normal conversions.

15.8.1.1 Calibration Conversion

The calibration conversion also needs to meet the minimum sampling time specification for the ADC. This value is typically 1 us. The Event Group sample time register (ADEVSAMP) is used to specify the number of ADCLK cycles for the calibration conversion.

The BRIDGE EN and HILO bits (ADCALCR.9:8) control the voltage to the calibration reference device shown in Figure 15-7. The positions of the switches in calibration mode are listed in Table 15-4.

Table 15-4. Calibration Reference Voltages†

CAL EN	BRIDGE EN	HILO	S1	S2	S3	S4	S5	Reference Voltage
1	0	0	1	0	1	0	0	$(AD_{REFHI}*R1 + AD_{REFLO}*R2) / (R1 + R2)$
1	0	1	0	1	0	1	0	$(AD_{REFLO}*R1 + AD_{REFHI}*R2) / (R1 + R2)$
1	1	0	0	1	1	0	0	AD_{REFLO}
1	1	1	1	0	0	1	0	AD_{REFHI}
0	X	X	0	0	0	0	1	V_{in}

† The state of the switches in this table assumes that self-test mode is not enabled.

When CAL ST (ADCALCR.16) is set, a calibration conversion is started. The voltage source selected via the bits BRIDGE EN and HILO is converted once (single conversion mode) and the digital result is returned to the calibration and correction register, ADCALR, where it can be read by the CPU. The CAL ST bit acts as a flag and must be polled by the CPU. It is held set during the conversion process and automatically clears to indicate the end of the reference voltage conversion.

Note: No Interrupt for end of calibration
The ADC does not generate an interrupt to signal the end of the calibration conversion. The application must poll the CAL_STR bit to determine the end of the calibration conversion.

After the CAL ST bit is set by the application program, it can only be reset by the end of the ongoing conversion generated by the ADC core. If the calibration conversion is interrupted (CAL EN bit is cleared), the CAL ST bit is held at 1 until a new calibration conversion has been set and completed. Setting the CAL ST bit while calibration is disabled (CAL EN = 0) has no effect; however, in this situation, setting CAL EN immediately starts a calibration conversion. When the calibration conversion is interrupted by an ADC enable (ADC EN = 0, CAL EN = 1, and CAL ST = 1), a new conversion is automatically restarted as soon as the ADC enable bit is released (ADC EN = 1).

15.8.1.2 Calibration and Offset Error Correction Sequences

The number of measurements and the source to measure for an ADC calibration are application dependent. The CAL ST bit must be set for each calibration source to be measured. While calibration mode is enabled, any available calibration sources can be converted according to the BRIDGE EN and HILO bits (see Table 15-4). The digital results of the calibration measurements should be read from ADCALR by the application after each reference conversion so that a correction value can be computed and written back into ADCALR.

When the application has the necessary calibration data, it should compute the offset error correction value and load it into the calibration and correction register, ADCALR. After the CAL EN bit is cleared, normal conversion mode restarts, continuing from where it was frozen, but with the addition of self-correction data.

In normal mode, the self-correction system adds the correction value stored in ADCALR to each digital result before it is written to the respective group's FIFO.

The basic calibration routine is as follows:

1. Enable calibration via CAL EN (ADCALCR.0).
2. Select the voltage source via BRIDGE EN and HILO (ADCALCR.9:8).
3. Start the conversion with CAL ST (ADCALCR.16).
4. Wait for CAL ST to go to 0.
5. Get the results from ADCALR and save to memory.
6. Loop to step 2 until the calibration conversion data is collected for the desired reference voltages.

7. Compute the error correction value using calibration data saved in memory.
8. Load the ADCALR register with the 2s complement of the computed error correction value.
9. Disable calibration mode.

At this point, the ADC can be configured for normal operation and it corrects each digital result with the error correction value loaded in ADCALR.

Note: Prevent ADC Calibration Data From Being Overwritten
In calibration mode, the conversion result is written to ADCALR which overwrites any previous calibration data; therefore, the ADCALR register must be read before a new conversion is started.

For no correction, a value of 0x0000 must be written to ADCALR. In noncalibration mode, the ADCALR register can be read and written. Any value written to ADCALR in normal mode (CAL EN = 0) is added to each digital result from the ADC core.

15.8.1.3 Mid-Point Calibration

Because of its connections to the ADC's reference voltage (VrefHi, VrefLo), the precision of the calibration reference is voltage independent. On the other hand, the accuracy of the switched bridge resistor (R1 & R2) relies on the manufacturing process deviation. Consequently, the mid-point voltage's accuracy can be affected due to the imperfections in the two resistors (expected mismatch error is around 1.5%).

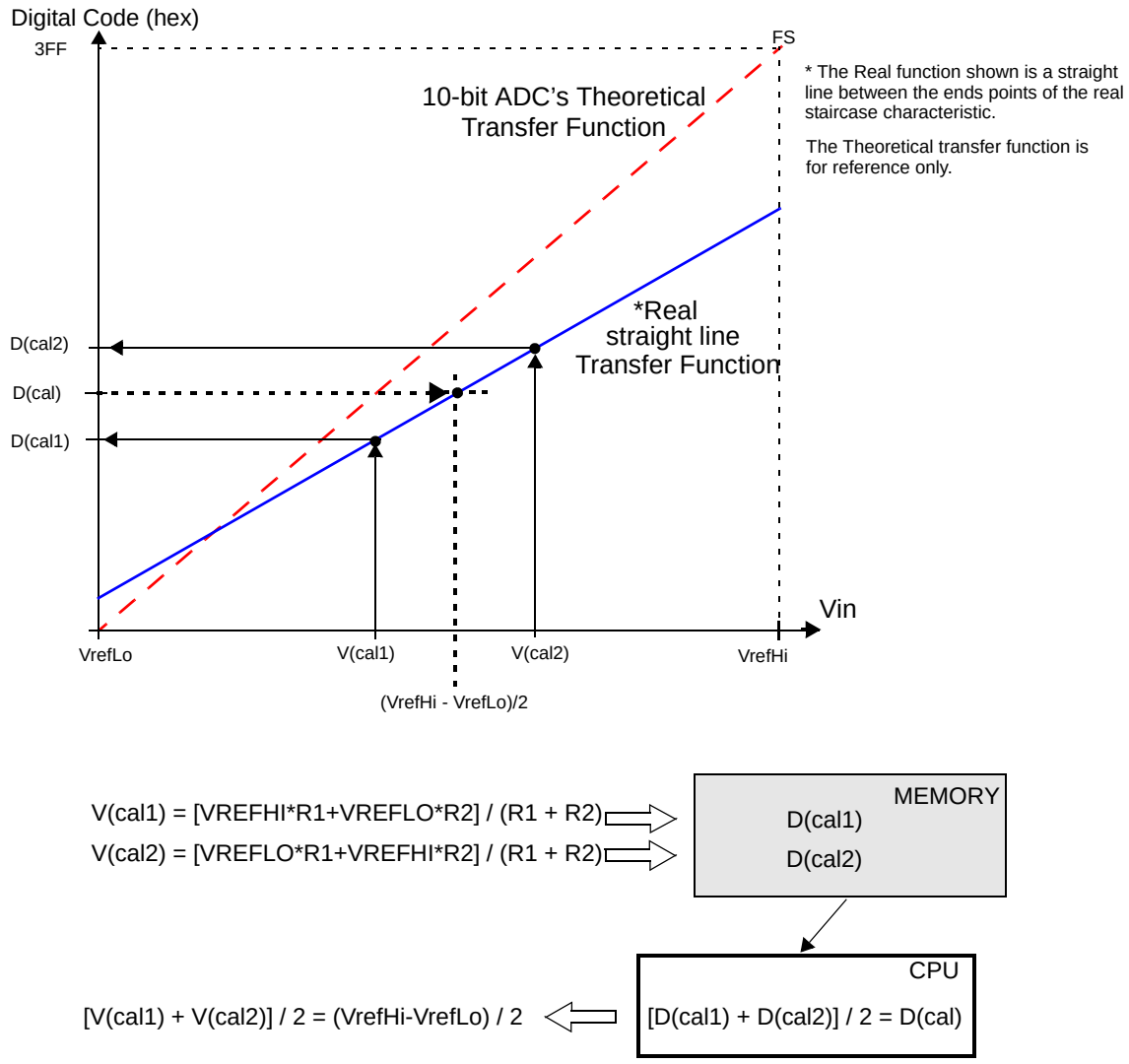
The switched reference voltage device has been specially designed to support a differential measurement of its mid-point voltage. This ensures the accuracy of the mid-point reference, and hence the efficiency of the calibration.

The differential mid-point calibration is software controlled; the algorithm (voltage source measurements and associated calculation) is inserted within the calibration software module included in the application program.

The basic differential mid-point calibration flow is illustrated here after

1. The application program connects the voltage VrefHi to R1 and VrefLo to R2, (BRIDGE_EN=0, HILO=0), launches a conversion of the input voltage V(cal1), and stores the digital result D(cal1) into the memory.
2. Then the application program switches the voltage VrefHi to R2 and VrefLo to R1 (BRIDGE_EN=0, HILO=1), converts this new input voltage V(cal2) and again stores the issued digital result D(cal2) into the memory.
3. The actual value of the real middle point is obtained by computing the average of these two results. [D(cal1)+D(cal2)] /2; The Figure below summarizes the mid-point calibration flow.

Figure 15-6. Mid-point value calculation

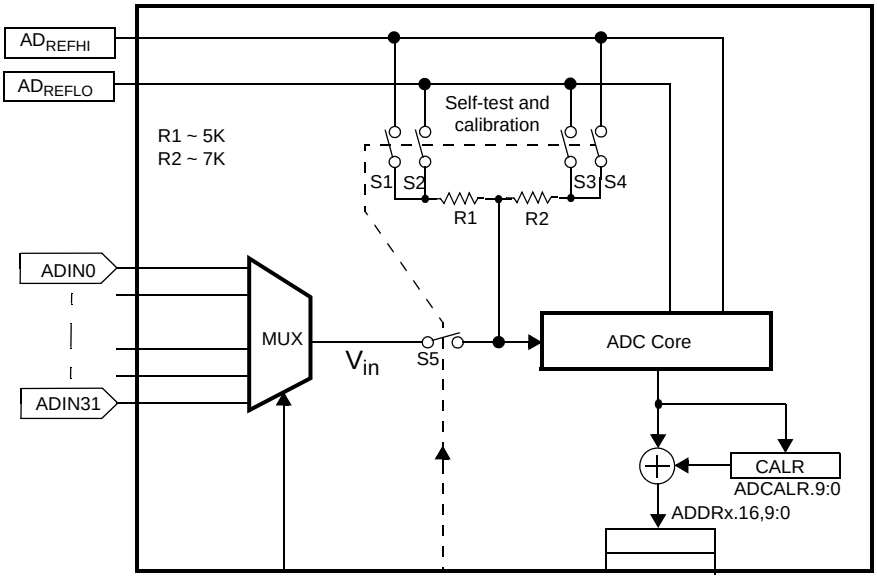


15.8.2 ADC Self-Test Mode

The ADC module supports a self-test mode which can be used to detect an open or a short on the ADC input channels. Self-test mode is enabled by setting the SELF TST bit (ADCALCR.24). Any conversion type (continuous or single conversion, freeze enabled or non-freeze enabled, interrupts enabled or disabled) can be performed in this mode.

In normal mode, setting the self-test mode while a conversion sequence is in process can corrupt the current channel conversion results. However, the next channel in the sequence is converted correctly during the additional self-test cycle. The logic associated with both self-test and calibration is shown in Figure 15-7.

Figure 15-7. Self-Test and Calibration Logic



In self-test mode, a test voltage defined by the HILO bit (ADCALCR.8) is provided to the ADC core input through a resistor (see Table 15-5). To change the test source, this bit can be toggled before any single conversion mode request. Changing this bit while a conversion is in progress *can* corrupt the results if the source switches during the acquisition period.

Please note that the switch S5 shown in the above figure is only for the purpose of explaining the self-test sequence. There is no physical switch.

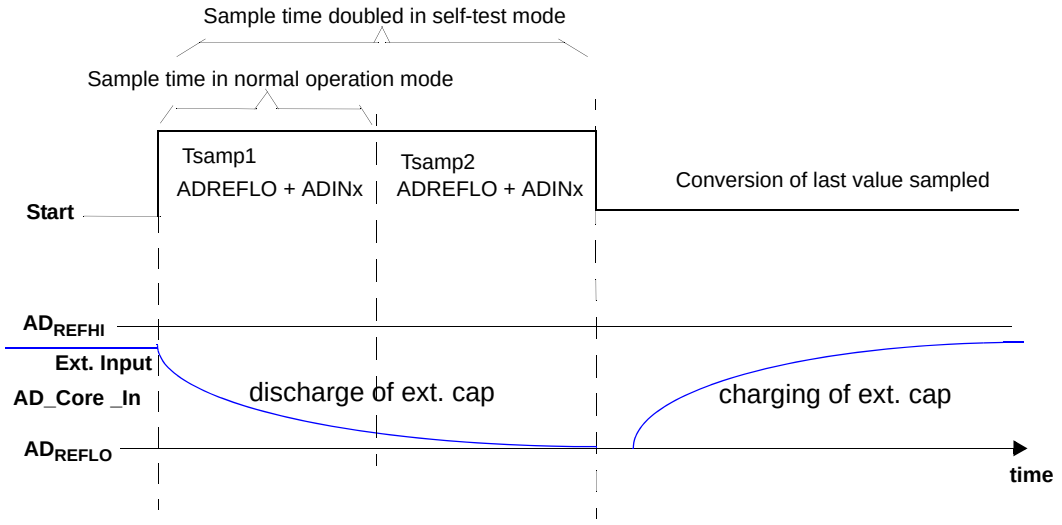
Table 15-5. Self-Test Reference Voltages[†]

SELF TST	HILO	S1	S2	S3	S4	S5	Reference Voltage
1	0	0	1	1	0	1	AD_{REFLO} via R1 || R2 connected to V_{in}
1	1	1	0	0	1	1	AD_{REFHI} via R1 || R2 connected to V_{in}
0	X	0	0	0	0	1	V_{in}

1 Switches refer to Figure 15-7.

Conversions in self-test mode are invoked by writing to ADG1SEL or ADG2SEL, or upon an external event. The conversion starts according to the configuration set in the three mode control registers and the sampling time control registers. The acquisition time for each conversion is extended to twice the normal configured acquisition time. The selected reference voltage and the input voltage from the ADINx input channel are both connected to the ADC internal sampling capacitor throughout this extended acquisition period. Figure 15-8 shows the self-test mode timing when the ADREFLO is chosen as the reference voltage for the self-test mode conversion. It also assumes an external capacitor connected to the ADC input channel.

Figure 15-8. Timing for Self-Test Mode



15.8.2.1 Use of Self-Test Mode to Determine Open/Short on ADC Input Channels

The following sequence needs to be used to deduce the ADC pin status:

- Convert the channel with self test enabled and with the reference voltage as Vreflo. Store the conversion result, say Vd.
- Convert the channel with self test enabled and with the reference voltage as Vrefhi. Store the conversion result, say Vu.
- Convert the channel with self test disabled. Store the conversion result, say Vn.

The results can be interpreted using the following table.

Table 15-6. Determination of ADC Input Channel Condition

Normal Conversion Result, Vn	Self-test Conversion Result, Vu	Self-test Conversion Result, Vd	Pin Condition
Vn	Vn < Vu < AD_{REFHI}	AD_{REFLO} < Vd < Vn	Good
AD_{REFHI}	AD_{REFHI}	approx. AD_{REFHI}	Shorted to AD_{REFHI}
AD_{REFLO}	approx. AD_{REFLO}	AD_{REFLO}	Shorted to AD_{REFLO}
Unknown	AD_{REFHI}	AD_{REFLO}	Open

15.8.3 ADC Powerdown Mode

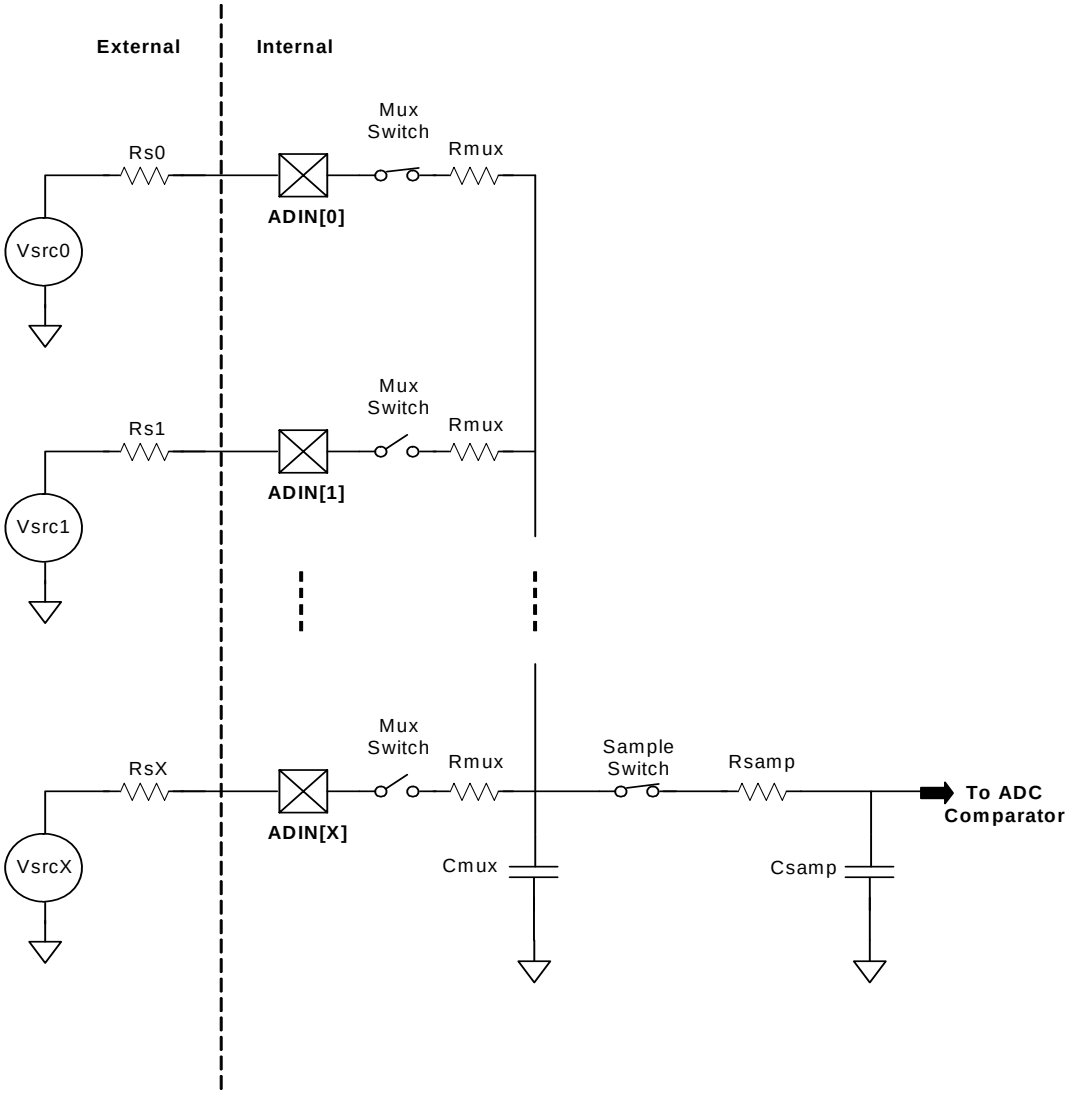
In the power-down mode, the clocks to the ADC module are stopped and the module is in a static state. The continuous bias current required for the analog stages is removed and the resistor ladder connected between AD_{REFHI} and AD_{REFLO} is opened. This results in the lowest possible ADC power consumption.

The ADC enters power-down mode when the appropriate control bit in the power down control register in the peripheral central resource registers frame is set. Please refer to the device datasheet to identify the appropriate control bit for powering down the ADC module.

15.8.4 ADC Input Impedance Measurement Mode

The Figure 15-9 shows the ADC input equivalent circuit.

Figure 15-9. ADC Input Equivalent Circuit



The ADC module closes the “Sample switch” shown in the above figure whenever a conversion is completed. This switch stays closed until the start of the sampling period of another conversion. The input impedance measurement mode is enabled by setting the CHN TEST EN bits of the ADC Operating Mode Control Register (ADOPMODECR). This mode allows the input mux switches of multiple ADC channels to be closed at the same time. Having multiple input mux switches closed along with the sample switch allows the input impedances of the ADC channels to be measured directly without performing any conversions.

The input channels that are enabled for this mode are selected by configuring the ADEVSEL register.

Note: Enabling Input Impedance Measurement Mode
This mode must not be enabled when the ADC is servicing any conversion group. Please make sure that the ADC is IDLE before enabling the input impedance measurement mode. The ADC EN must be cleared to ‘0’ before enabling this test mode.

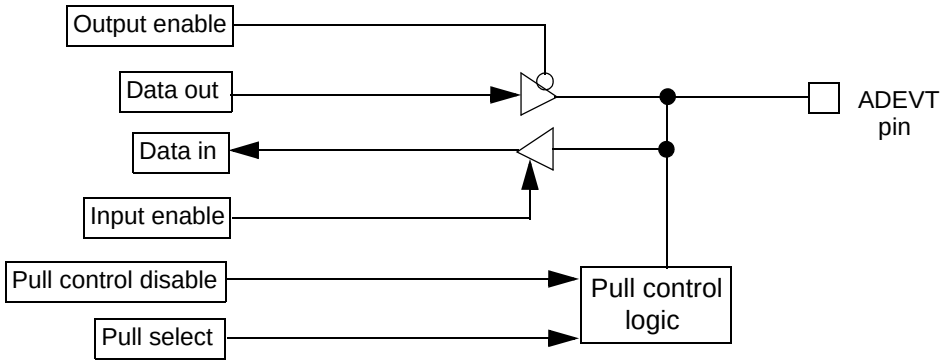
15.9 ADEVT Pin General Purpose I/O Functionality

The ADEVT pin, if available on the device, can be configured to be a general-purpose I/O pin. The following sections describe the different ways in which the application can configure the ADEVT pin.

15.9.1 GPIO Functionality

Figure 15-10 illustrates the GPIO functionality.

Figure 15-10. GPIO Functionality



15.9.2 Under Reset

The following apply if the device is under reset:

- Pull control. The reset pull control on the pins is enabled or disabled depending on a device-specific option. This feature is configurable for each module separately.
- Input buffer. If the reset pull control is enabled, then the input buffer is also enabled. If the reset pull control is disabled then the input buffer is also disabled.
- Output buffer. The output buffer is disabled.

15.9.3 Out of Reset

The following apply if the device is out of reset:

- Pull control. The pull control is enabled by clearing the PDIS (pull control disable) bit in the ADEVTPDIS register (Section 15.10.46). In this case, if the PSEL (pull select) bit in the ADEVTPSEL register (Section 15.10.47) is set, the pin will have a pull-up. If the PSEL bit is cleared, the pin will have a pull-down. If the PDIS bit is set in the control register, there is no pull-up or pull-down on the pin.
- Input buffer. The input buffer is disabled only if the pin direction is set as input in the ADEVTDIR register (Section 15.10.40) AND the pull control is disabled AND pull down is selected as the pull bias. In all other cases, the input buffer is enabled.

Note:
The pull-disable logic depends on the pin direction. It is independent of whether the device is in I/O or functional mode. If the pin is configured as output, then the pulls are disabled automatically. If the pin is configured as input, the pulls are enabled or disabled depending on bit PDIS in the pull disable register ADEVTPDIS.

- Output buffer. The ADEVT pin can be driven as an output pin if the ADEVTDIR bit is set in the pin direction control register AND the open-drain feature is not enabled.

15.9.4 Open-Drain Feature Enabled on a Pin

The following apply if the open-drain feature is enabled on the ADEVT pin.

- The output buffer is enabled if a low signal is being driven on to the pin.
- The output buffer is disabled (the direction control signal ADEVTDIR is internally forced low) if a high signal is being driven on to the pin.

15.9.5 Summary

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 15-7.

Table 15-7. Input Buffer, Output Buffer, and Pull Control Behavior as GPIO Pins

Device under Reset?	Pin Direction (DIR)	Pull Disable (PDIS)	Pull Select (PSEL)	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Device- and module-specific	Disabled	Depends on pull control
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Disabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

1 X = Don't care
2 DIR = 0 for input, 1 for output
3 PULDIS = 0 for enabling pull control
= 1 for disabling pull control
4 PULSEL= 0 for pull-down functionality
= 1 for pull-up functionality

15.10 ADC Control Registers

All registers in the ADC module are 32-bit, word-aligned; 8-bit, 16-bit and 32-bit accesses are allowed. The following table provides a quick reference to each of these registers. Specific bit descriptions are discussed in the following subsections.

All registers can be read by the CPU at any time without affecting an ongoing conversion or the ADC accuracy. Writing to the control registers or to the input-select registers (ADEVSEL, ADG1SEL, and ADG2SEL) while the corresponding group is being converted can disturb the ongoing channel conversion; however, the module recovers on the next conversion.

Table 15-8. ADC Registers Summary

Offset Address / Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 AD RST CR Page 787	Reserved															
	Reserved															Reset
0x04 AD OP MODE CR Page 788	Reserved							COS	Reserved			CHN TEST EN				RAM TEST EN
	Reserved							POW-ERDOWN	Reserved							ADC EN
0x08 AD CLOCK CR Page 790	Reserved															
	Reserved											PS.4:0				
0x0C AD CAL CR Page 791	Reserved							SELF TEST	Reserved							CAL ST
	Reserved						BRIDGE EN	HILO	Reserved							CAL EN
0x10 AD EV MODE CR Page 793	Reserved															
	Reserved										EV CHID	OVR EV RAM IGN		EV 8BIT	EV MODE	FRZ EV
0x14 AD G1 MODE CR Page 795																
											G1 CHID	OVR G1 RAM IGN	G1 HW TRIG	G1 8BIT	G1 MODE	FRZ G1
0x18 AD G2 MODE CR Page 798																
											G2 CHID	OVR G2 RAM IGN	G2 HW TRIG	G2 8BIT	G2 MODE	FRZ G2

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x1C AD EV SRC Page 801													EV EDG SEL	EVSRC.2:0		
0x20 AD G1 SRC Page 802													G1 EDG SEL	G1SRC.2:0		
0x24 AD G2 SRC Page 803													G2 EDG SEL	G2SRC.2:0		
0x28 AD EV INT ENA Page 804													EV END INT EN		EV OVR INT EN	EV THR INT EN
0x2C AD G1 INT ENA Page 806													G1 END INT EN		G1 OVR INT EN	G1 THR INT EN
0x30 AD G2 INT ENA Page 808													G2 END INT EN		G2 OVR INT EN	G2 THR INT EN
0x34 AD EV INT FLG Page 810													EV END	EV MEM EMPTY	EV MEM OVER-FLOW	EV THR INT FLAG
0x38 AD G1 INT FLG Page 812													G1 END	G1 MEM EMPTY	G1 MEM OVER-FLOW	G1 THR INT FLAG
0x3C AD G2 INT FLG Page 814													G2 END	G2 MEM EMPTY	G2 MEM OVER-FLOW	G2 THR INT FLAG

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x40 AD EV INT CR Page 816	Sign Extension								EVTHR.8:0							
0x44 AD G1 INT CR Page 817	Sign Extension								G1THR.8:0							
0x48 AD G2 INT CR Page 818	Sign Extension								G2THR.8:0							
0x58 AD BND CR Page 819								BNDA.8:0								
								BNDB.8:0								
0x5C AD BND END Page 820																BUF Init Active
														BNDEND.2:0		
0x60 AD EV SAMP Page 822						EVACQ.11:0										
0x64 AD G1 SAMP Page 823						G1ACQ.11:0										
0x68 AD G2 SAMP Page 824						G2ACQ.11:0										

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x6C AD EV SR Page 825																
													EV MEM EMPTY	EV BUSY	EV STOP	EV END
0x70 AD G1 SR Page 827																
													G1 MEM EMPTY	G1 BUSY	G1 STOP	G1 END
0x74 AD G2 SR Page 829																
													G2 MEM EMPTY	G2 BUSY	G2 STOP	G2 END
0x78 AD EV SEL Page 831	EV CHNSEL.31:16															
	EV CHNSEL.15:0															
0x7C AD G1 SEL Page 832	G1 CHNSEL.31:16															
	G1 CHNSEL.15:0															
0x80 AD G2 SEL Page 833	G2 CHNSEL.31:16															
	G2 CHNSEL.15:0															
0x84 AD CALR Page 834																
							ADCALR.9:0									
0x88 AD SM STATE Page 835																
													SMSTATE.3:0			
0x8C AD LAST CONV Page 837	LASTADIN.31:16															
	LASTADIN.15:0															

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x90 to 0xAF AD EV BUFFER Page 838	EV EMPTY	EVCHID.4:0					EVDR.9:0									
0xB0 to 0xCF AD G1 BUFFER Page 839	G1 EMPTY	G1CHID.4:0					G1DR.9:0									
0xD0 to 0xEF AD G2 BUFFER Page 840	G2 EMPTY	G2CHID.4:0					G2DR.9:0									
0xF0 AD EV EMU BUFFER Page 841	EV EMPTY	EVCHID.4:0					EVDR.9:0									
0xF4 AD G1 EMU BUFFER Page 842	G1 EMPTY	G1CHID.4:0					G1DR.9:0									
0xF8 AD G2 EMU BUFFER Page 843	G2 EMPTY	G2CHID.4:0					G2DR.9:0									
0xFC ADEVT DIR Page 844																ADEVT DIR
0x100 ADEVT OUT Page 845																ADEVT OUT

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x104 ADEVT IN Page 846																ADEVT IN
0x108 ADEVT SET Page 847																ADEVT SET
0x10C ADEVT CLR Page 848																ADEVT CLR
0x110 ADEVT PDR Page 849																ADEVT PDR
0x114 ADEVTPDIS Page 850																ADEVT PDIS
0x118 ADEVT PSEL Page 851																ADEVT PSEL
0x128 AD MAG INT CR1 Page 852	Reserved	MAG CHID1.4:0					MAG THRESHOLD1.9:0									
	Reserved			COMP CHID1.4:0				Reserved						CHN/ THR COMP		COMP GE/LT1
0x12C AD MAG INT1 MASK Page 854																
							MAG INT1 MASK.9:0									
0x130 AD MAG INT CR2 Page 852	Reserved	MAG CHID2.4:0					MAG THRESHOLD2.9:0									
	Reserved			COMP CHID2.4:0				Reserved						CHN/ THR COMP		COMP GE/LT2

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x134 AD MAG INT2 MASK Page 854							MAG INT2 MASK.9:0									
0x138 AD MAG INT CR3 Page 852	Reserved	MAG CHID3.4:0					MAG THRESHOLD3.9:0									
	Reserved				COMP CHID3.4:0				Reserved						CHN/ THR COMP	COMP GE/LT3
0x13C AD MAG INT3 MASK Page 854							MAG INT3 MASK.9:0									
0x140 AD MAG INT CR4 Page 852	Reserved	MAG CHID4.4:0					MAG THRESHOLD4.9:0									
	Reserved				COMP CHID4.4:0				Reserved						CHN/ THR COMP	COMP GE/LT4
0x144 AD MAG INT4 MASK Page 854							MAG INT4 MASK.9:0									
0x148 AD MAG INT CR5 Page 852	Reserved	MAG CHID5.4:0					MAG THRESHOLD5.9:0									
	Reserved				COMP CHID5.4:0				Reserved						CHN/ THR COMP	COMP GE/LT5
0x14C AD MAG INT5 MASK Page 854							MAG INT5 MASK.9:0									
0x150 AD MAG INT CR6 Page 852	Reserved	MAG CHID6.4:0					MAG THRESHOLD6.9:0									
	Reserved				COMP CHID6.4:0				Reserved						CHN/ THR COMP	COMP GE/LT6

Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x154 AD MAG INT6 MASK Page 854							MAG INT6 MASK.9:0									
0x158 AD MAG THR INTENA SET Page 855											MAG INT ENA SET6	MAG INT ENA SET5	MAG INT ENA SET4	MAG INT ENA SET3	MAG INT ENA SET2	MAG INT ENA SET1
0x15C AD MAG THR INTENA CLR Page 856											MAG INT ENA CLR6	MAG INT ENA CLR5	MAG INT ENA CLR4	MAG INT ENA CLR3	MAG INT ENA CLR2	MAG INT ENA CLR1
0x160 AD MAG THR INTFLG Page 857											MAG THR INT FLG6	MAG THR INT FLG5	MAG THR INT FLG4	MAG THR INT FLG3	MAG THR INT FLG2	MAG THR INT FLG1
0x164 AD MAG THR INT OFFSET Page 858													INT OFF.3	INT OFF.2	INT OFF.1	INT OFF.0
0x168 AD EV FIFO RESET CR Page 859	Reserved															
	Reserved															EV FIFO RESET
0x16C AD G1 FIFO RESET CR Page 860	Reserved															
	Reserved															G1 FIFO RESET
0x170 AD G2 FIFO RESET CR Page 861	Reserved															
	Reserved															G2 FIFO RESET
0x174 AD EV RAM ADDR Page 862	Reserved															
	Reserved							EV RAM ADDR.8:0								

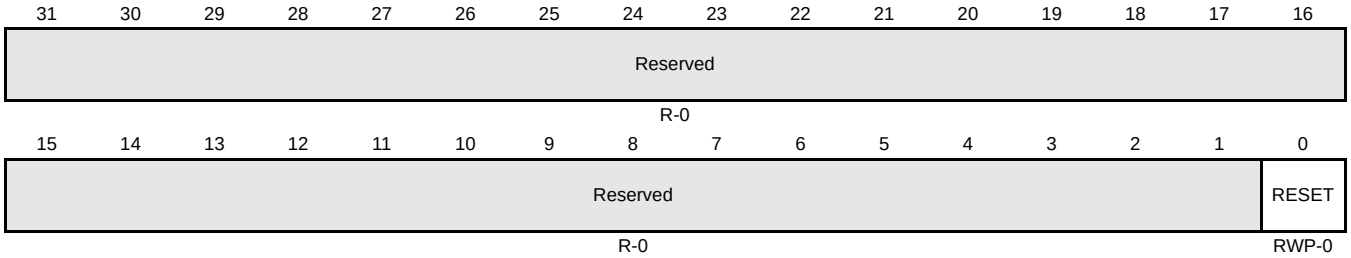
Table 15-8. ADC Registers Summary (Continued)

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x178 AD G1 RAM ADDR Page 863	Reserved															
	Reserved							G1 RAM ADDR.8:0								
0x17C AD G2 RAM ADDR Page 864	Reserved															
	Reserved							G2 RAM ADDR.8:0								

15.10.1 ADC Reset Control Register (ADRSTCR)

Figure 15-11 and Table 15-9 describe the ADRSTCR register.

Figure 15-11. ADC Reset Control Register (ADRSTCR) [offset = 0h]



RW = Read/Write, RC = Read/Clear, -*n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-9. ADC Reset Control Register (ADRSTCR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	RESET		This bit is used to reset the ADC internal state machines and control/status registers. This reset state is held until this bit is cleared. Privileged mode read/write:
		0	Module is released from the reset state.
		1	All the module's internal state machines and the control/status registers are reset.

15.10.2 ADC Operating Mode Control Register (ADOPMODECR)

Figure 15-12 and Table 15-10 describe the ADOPMODECR register.

Figure 15-12. ADC Operating Mode Control Register (ADOPMODECR) [offset = 4h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							COS				CHN TEST EN				RAM TEST EN
			R-0				RW-0		R-0			RW-1010			RW-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															ADC EN
							R-0								RW-0

RW = Read/Write, RC = Read/Clear, -*n* = value after reset

Table 15-10. ADC Operating Mode Control Register (ADOPMODECR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zeros, writes have no effect.
24	COS		This bit affects *emulation operation only*. It defines whether the ADC core clock (ADCLK) is immediately halted when the emulation system enters suspend mode or if it should continue operating normally. **Note**: If COS = 0 when the ADC module enters the emulation mode, then the accuracy of the conversion results can be affected depending on how long the module stays in the emulation mode. Any operation mode read/write: 0 ADC module halts all ongoing conversions immediately after emulation mode is entered. 1 ADC module continues all ongoing conversions as per the configurations of the three conversion groups.
23–21	Reserved		Reads return zeros, writes have no effect.
21-17	CHN TEST EN		Enable the input channels' impedance measurement mode. Please refer to the Section 15.8.4 for more details. Any operation mode read/write: 1010 Input impedance measurement mode is disabled. 0101 Input impedance measurement mode is enabled.

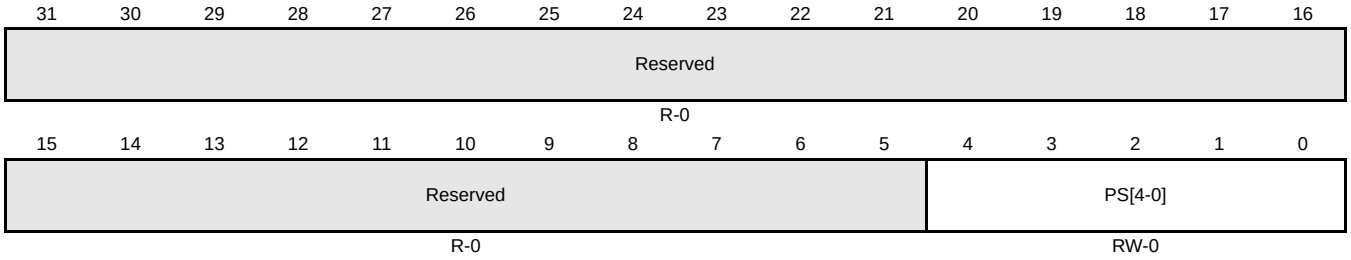
Table 15-10. ADC Operating Mode Control Register (ADOPMODECR) Field Descriptions (Continued)

Bit	Name	Value	Description
16	RAM TEST EN		Enable the ADC Results' RAM Test Mode. Please refer to Section 15.7.2 for more details. Any operation mode read/write:
		0	ADC RAM Test Mode is disabled. The application cannot write to the ADC RAM by the CPU or the DMA.
		1	ADC RAM Test Mode is enabled. The application can directly write to the ADC RAM by the CPU or the DMA.
15-1	Reserved		Reads return zeros, writes have no effect.
0	ADC EN		ADC Enable. This bit must be set to allow the ADC module to be configured to perform any conversions. Any operation mode read/write:
		0	No ADC conversions can occur. The input channel select registers: ADEVSEL, ADG1SEL, and ADG2SEL are held at their reset values.
		1	ADC conversions can now proceed as configured.

15.10.3 ADC Clock Control Register (ADCLOCKCR)

Figure 15-13 and Table 15-11 describe the ADCLOCKCR register.

Figure 15-13. ADC Clock Control Register (ADCLOCKCR) [offset = 8h]



RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

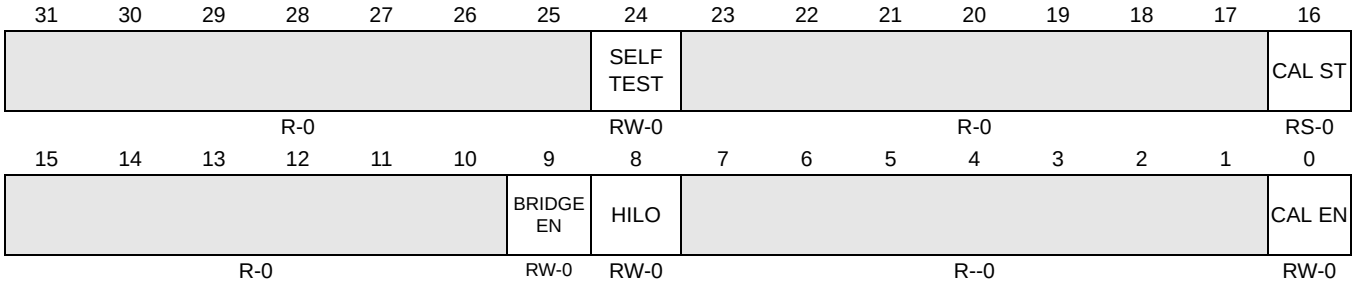
Table 15-11. ADC Clock Control Register (ADCLOCKCR) Field Descriptions

Bit	Name	Value	Description
31–5	Reserved	0	Reads return zeros, writes have no effect.
4-0	PS[4-0]	00000 to 11111	ADC Clock Prescaler. These bits define the prescaler value for the ADC core clock (ADCLK). The ADCLK is generated by dividing down the input bus clock (VCLK) to the ADC module. Any operation mode read/write: $t_{C(ADCLK)} = t_{C(VCLK)} * (PS[4-0] + 1)$, where $t_{C(ADCLK)}$ is the period of the ADCLK, and $t_{C(VCLK)}$ is the period of the VCLK.

15.10.4 ADC Calibration Mode Control Register (ADCALCR)

Figure 15-14 and Table 15-12 describe the ADCALCR register.

Figure 15-14. ADC Calibration Mode Control Register (ADCALCR) [offset = Oh]



RW = Read/Write, RC = Read/Clear, RS = Read/Set, -*n* = value after reset

Table 15-12. ADC Calibration Mode Control Register (ADCALCR) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return zeros, writes have no effect.
24	SELF TEST		ADC Self Test Enable. When this bit is Set, either AD_{REFHI} or AD_{REFLO} is connected through a resistor to the selected input channel. The desired conversion mode is configured in the group mode control registers. For more details on the ADC Self Test Mode, please refer to Section 15.8.2. Any operation mode read/write:
		0	ADC Self Test mode is disabled.
		1	ADC Self Test mode is enabled.
23–17	Reserved		Reads return zeros, writes have no effect.
16	CAL ST		ADC Calibration Conversion Start. Setting the CAL ST bit while the CAL EN bit is set starts conversion of the selected reference voltage. The ADC module uses the sample time configured in the Event Group sample time configuration register (ADEVSAMP) for the calibration conversion. Any operation mode:
		1	Read: Calibration conversion is in progress. Write: ADC module starts calibration conversion.
		0	Read: Calibration conversion has completed, or has not yet been started. Write: Writing 0 to this bit has no effect.
15–10	Reserved		Reads return zeros, writes have no effect.

Table 15-12. ADC Calibration Mode Control Register (ADCALCR) Field Descriptions (Continued)

Bit	Name	Value	Description
9	BRIDGE EN		Bridge Enable. When set with the HILO bit, BRIDGE EN allows a reference voltage to be converted in calibration mode. The Table 15-4, "Calibration Reference Voltages†," on page 770 defines the four different reference voltages that can be selected.
8	HILO		ADC Self Test mode and Calibration Mode Reference Source Selection. In the ADC Self Test mode, this bit defines the test voltage to be combined through a resistor with the selected input pin voltage. In the ADC Calibration Mode, this bit defines the reference source polarity. In the ADC module's normal operating mode, this bit has no effect.
7–1	Reserved		Reads return zeros, writes have no effect.
0	CAL EN		ADC Calibration Enable. When this bit is Set, the input channel multiplexor is disconnected and the calibration reference voltage is connected to the ADC core input. The calibration reference voltage is selected by the combination of the BRIDGE EN and HILO. The actual conversion of this reference voltage starts when the CAL ST bit is set. If the CAL ST bit is already set when the CAL EN bit is set, then the calibration conversion is immediately started. Please refer to Section 15.8.1 for more details on the ADC calibration mode. Any operation mode read/write: 0 Calibration mode is disabled. 1 Calibration mode is enabled.

15.10.5 ADC Event Group Operating Mode Control Register (*ADEVMODECR)*

Figure 15-15 and Table 15-13 describe the ADEVMODECR register.

Figure 15-15. ADC Event Group Operating Mode Control Register (ADEVMODECR) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										EV CHID	OVR EV RAM IGN		EV 8BIT	EV MODE	FRZ EV
0										RW-0	RW-0	0	RW-0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, RS = Read/Set, -*n* = value after reset

Table 15-13. ADC Event Group Operating Mode Control Register (ADEVMODECR) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved		Reads return zeros, writes have no effect.
5	EV CHID		Enable Channel Id for the Event Group conversion results to be read. This bit only affects the “read from FIFO” mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the “read from RAM” mode will return the 5-bit channel id along with the 10-bit conversion result. Any operation mode read/write:
		0	Bits 14-10, the channel id field, of the data read from the Event Group results’ FIFO is read as 00000b.
		1	Bits 14-10, the channel id field, of the data read from the Event Group results’ FIFO contains the number of the ADC analog input to which the conversion result belongs.
4	OVR EV RAM IGN		This bit allows the ADC module to overwrite the contents of the Event Group results’ memory under an overrun condition. Any operation mode read/write:
		0	The ADC cannot overwrite the contents of the Event Group results’ memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Event Group.
		1	When an overrun of the Event Group results’ memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Event Group, starting with the first location in this memory.
3	Reserved		Reads return zeros, writes have no effect.

Table 15-13. ADC Event Group Operating Mode Control Register (ADEVMODECR) Field Descriptions

Bit	Name	Value	Description
2	EV 8BIT		Event Group 8-bit result mode. This bit allows the Event Group conversion results to be read out in an 8-bit format. This bit only applies to the “read from FIFO” mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result. Any operation mode read/write:
		0	The Event Group conversion result is read out as a 10-bit value in the “read from Event Group FIFO” mode.
		1	The Event Group conversion result is read out as an 8-bit value in the “read from Event Group FIFO” mode.
1	EV MODE		Event Group Conversion Mode. This bit defines whether the input channels selected for conversion in the Event Group are converted only once per trigger, or are continuously converted. Any operation mode read/write:
		0	The channels selected for conversion in the Event Group are converted only once when the selected event trigger condition occurs.
		1	The channels selected for conversion in the Event Group are converted continuously when the selected event trigger condition occurs.
0	FRZ EV		Event Group Freeze Enable. This bit allows an Event Group conversion sequence to be frozen if a Group1 or a Group2 conversion is requested. The Event Group conversion is kept frozen until the Group1 or Group2 conversion is active, and continues from where it was frozen once the Group1 or Group2 conversions are completed. While the Event Group conversion is frozen, the EV STOP status flag in the ADEVST register indicates that the Event Group conversions have stopped. This bit gets cleared when the Event Group conversions resume. Any operation mode read/write:
		0	Event Group conversions cannot be frozen. All the channels selected for conversion in the Event Group are converted before the ADC can switch over to servicing any other conversion group.
		1	Event Group conversions are frozen whenever there is a request for conversion from Group1 or Group2.

15.10.6 ADC Group1 Operating Mode Control Register (*ADG1MODECR)*

Figure 15-16 and Table 15-14 describe the ADG1MODECR register.

Figure 15-16. ADC Group1 Operating Mode Control Register (ADG1MODECR) [offset = 14h]

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										G1 CHID	OVR G1 RAM IGN	G1 GW TRIG	G1 8BIT	G1 MODE	FRZ G1
										RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, RS = Read/Set, -*n* = value after reset

Table 15-14. ADC Event Group Operating Mode Control Register (ADG1MODECR) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved		Reads return zeros, writes have no effect.
5	G1 CHID		Enable Channel Id for the Group1 conversion results to be read. This bit only affects the “read from FIFO” mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the “read from RAM” mode will return the 5-bit channel id along with the 10-bit conversion result. Any operation mode read/write:
		0	Bits 14-10, the channel id field, of the data read from the Group1 results’ FIFO is read as 00000b.
		1	Bits 14-10, the channel id field, of the data read from the Group1 results’ FIFO contains the number of the ADC analog input to which the conversion result belongs.
4	OVR G1 RAM IGN		This bit allows the ADC module to overwrite the contents of the Group1 results’ memory under an overrun condition. Any operation mode read/write:
		0	The ADC cannot overwrite the contents of the Group1 results’ memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Group1.
		1	When an overrun of the Group1 results’ memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Group1, starting with the first location in this memory.

Table 15-14. ADC Event Group Operating Mode Control Register (ADG1MODECR) Field Descriptions

Bit	Name	Value	Description
3	G1 HW TRIG		Group1 Hardware Triggered. This bit allows the Group1 to be hardware triggered. The Group1 is software triggered by default. For more details on how to trigger a conversion group, please refer to Section 15.3.4.
			Any operation mode read/write:
		0	The Group1 is software-triggered. A Group1 conversion starts whenever the Group1 channel select register (ADG1SEL) is written with a non-zero value.
		1	The Group1 is hardware-triggered. A Group1 conversion starts whenever the Group1 channel select register has a non-zero value, and the specified hardware trigger occurs. The hardware trigger for the Group1 is specified in the Group1 Trigger Source register (ADG1SRC).
2	G1 8BIT		Group1 8-bit result mode. This bit allows the Group1 conversion results to be read out in an 8-bit format. This bit only applies to the “read from FIFO” mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result.
			Any operation mode read/write:
		0	The Group1 conversion result is read out as a 10-bit value in the “read from Group1 FIFO” mode.
		1	The Group1 conversion result is read out as an 8-bit value in the “read from Group1 FIFO” mode.
1	G1 MODE		Group1 Conversion Mode. This bit defines whether the input channels selected for conversion in the Group1 are converted only once, or are continuously converted.
			Any operation mode read/write:
		0	The channels selected for conversion in the Group1 are converted only once.
		1	The channels selected for conversion in the Group1 are converted continuously.

Table 15-14. ADC Event Group Operating Mode Control Register (ADG1MODECR) Field Descriptions

Bit	Name	Value	Description
0	FRZ G1		Group1 Freeze Enable. This bit allows a Group1 conversion sequence to be frozen if an Event Group or a Group2 conversion is requested. The Group1 conversion is kept frozen until the Event Group or Group2 conversion is active, and continues from where it was frozen once the Event Group or Group2 conversions are completed. While the Group1 conversion is frozen, the G1 STOP status flag in the ADG1ST register indicates that the Group1 conversions have stopped. This bit gets cleared when the Group1 conversions resume. Any operation mode read/write:
		0	Group1 conversions cannot be frozen. All the channels selected for conversion in the Group1 are converted before the ADC can switch over to servicing any other conversion group.
		1	Group1 conversions are frozen whenever there is a request for conversion from Event Group or Group2.

15.10.7 ADC Group2 Operating Mode Control Register (*ADG2MODECR)*

Figure 15-17 and Table 15-15 describe the ADG2MODECR register.

Figure 15-17. ADC Group2 Operating Mode Control Register (ADG2MODECR) [offset = 18h]

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										G2 CHID	OVR G2 RAM IGN	G2 GW TRIG	G2 8BIT	G2 MODE	FRZ G2
										RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, RS = Read/Set, -*n* = value after reset

Table 15-15. ADC Group2 Operating Mode Control Register (ADG2MODECR) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved		Reads return zeros, writes have no effect.
5	G2 CHID		Enable Channel Id for the Group2 conversion results to be read. This bit only affects the “read from FIFO” mode. The ADC always stores the channel id in the results RAM. Any 16-bit read performed in the “read from RAM” mode will return the 5-bit channel id along with the 10-bit conversion result. Any operation mode read/write:
		0	Bits 14-10, the channel id field, of the data read from the Group2 results’ FIFO is read as 00000b.
		1	Bits 14-10, the channel id field, of the data read from the Group2 results’ FIFO contains the number of the ADC analog input to which the conversion result belongs.
4	OVR G2 RAM IGN		This bit allows the ADC module to overwrite the contents of the Group2 results’ memory under an overrun condition. Any operation mode read/write:
		0	The ADC cannot overwrite the contents of the Group2 results’ memory. When an overrun of this memory occurs, the software needs to read out all the contents of this memory before the ADC is able to write a new conversion result for the Group2.
		1	When an overrun of the Group2 results’ memory occurs, the ADC proceeds to overwrite the contents with any new conversion results for the Group2, starting with the first location in this memory.

Table 15-15. ADC Group2 Operating Mode Control Register (ADG2MODECR) Field Descriptions

Bit	Name	Value	Description
3	G2 HW TRIG		Group2 Hardware Triggered. This bit allows the Group2 to be hardware triggered. The Group2 is software triggered by default. For more details on how to trigger a conversion group, please refer to Section 15.3.4. Any operation mode read/write:
		0	The Group2 is software-triggered. A Group2 conversion starts whenever the Group2 channel select register (ADG2SEL) is written with a non-zero value.
		1	The Group2 is hardware-triggered. A Group2 conversion starts whenever the Group2 channel select register has a non-zero value, and the specified hardware trigger occurs. The hardware trigger for the Group2 is specified in the Group2 Trigger Source register (ADG2SRC).
2	G2 8BIT		Group2 8-bit result mode. This bit allows the Group2 conversion results to be read out in an 8-bit format. This bit only applies to the “read from FIFO” mode. The lower 2 bits of the 10-bit conversion result are discarded and the upper 8 bits are shifted right two places to form the 8-bit conversion result. Any operation mode read/write:
		0	The Group2 conversion result is read out as a 10-bit value in the “read from Group2 FIFO” mode.
		1	The Group2 conversion result is read out as an 8-bit value in the “read from Group2 FIFO” mode.
1	G2 MODE		Group2 Conversion Mode. This bit defines whether the input channels selected for conversion in the Group2 are converted only once, or are continuously converted. Any operation mode read/write:
		0	The channels selected for conversion in the Group2 are converted only once.
		1	The channels selected for conversion in the Group2 are converted continuously.

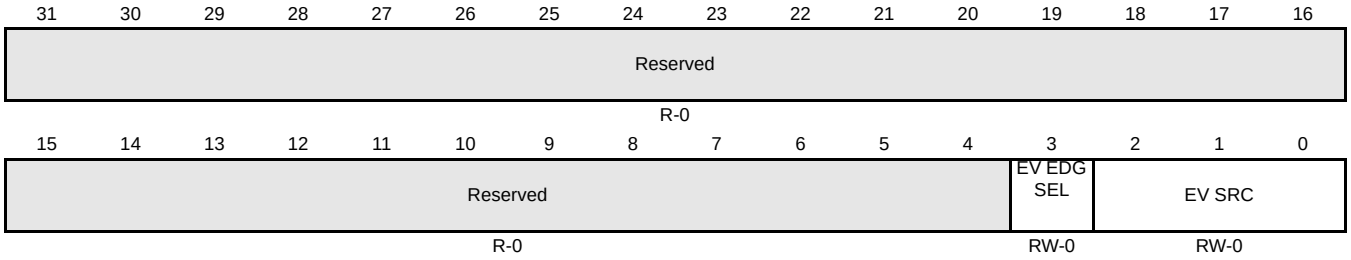
Table 15-15. ADC Group2 Operating Mode Control Register (ADG2MODECR) Field Descriptions

Bit	Name	Value	Description
0	FRZ G2		Group2 Freeze Enable. This bit allows a Group2 conversion sequence to be frozen if an Event Group or a Group1 conversion is requested. The Group2 conversion is kept frozen until the Event Group or Group1 conversion is active, and continues from where it was frozen once the Event Group or Group1 conversions are completed. While the Group2 conversion is frozen, the G2 STOP status flag in the ADG2ST register indicates that the Group2 conversions have stopped. This bit gets cleared when the Group2 conversions resume. Any operation mode read/write:
		0	Group2 conversions cannot be frozen. All the channels selected for conversion in the Group2 are converted before the ADC can switch over to servicing any other conversion group.
		1	Group2 conversions are frozen whenever there is a request for conversion from Event Group or Group1.

15.10.8 ADC Event Group Trigger Source Select Register (ADEVSRC)

Figure 15-18 and Table 15-16 describe the ADEVSRC register.

Figure 15-18. ADC Event Group Trigger Source Select Register (ADEVSRC) [offset = 1Ch]



RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-16. ADC Event Group Trigger Source Select Register (ADEVSRC) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	EV EDG SEL		Event Group Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Event Group conversion. Any operation mode read/write: 0 A high-to-low transition on the selected source will trigger the Event Group conversion. 1 A low-to-high transition on the selected source will trigger the Event Group conversion.
2–0	EV SRC		Event Group Trigger Source. Any operation mode read/write: 000 to 111 The ADC module allows a trigger source to be selected for the Event Group from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

15.10.9 ADC Group1 Trigger Source Select Register (ADG1SRC)

Figure 15-19 and Table 15-17 describe the ADG1SRC register.

Figure 15-19. ADC Group1 Trigger Source Select Register (ADG1SRC) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												G1 EDG SEL	G1 SRC		
R-0												RW-0	RW-0		

RW = Read/Write, RC = Read/Clear, -*n* = value after reset, RWP = Read in all modes, write in privileged mode only

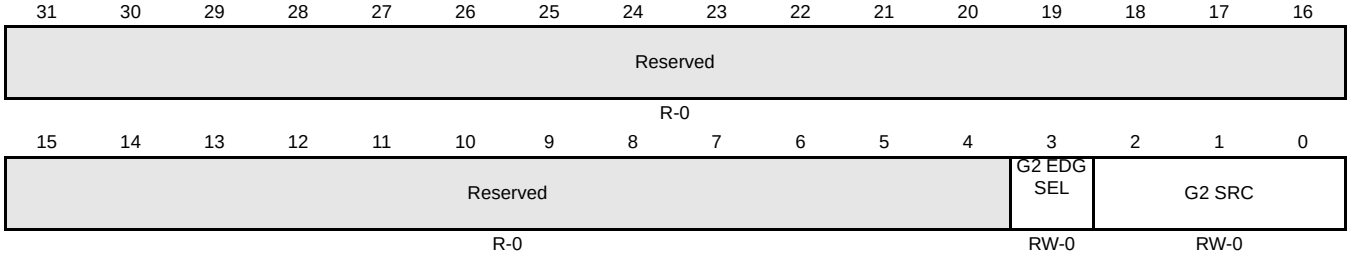
Table 15-17. ADC Group1 Trigger Source Select Register (ADG1SRC) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G1 EDG SEL		Group1 Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Group1 conversion. This bit is only applicable if the Group1 is configured to be hardware-triggered. Any operation mode read/write: 0 A high-to-low transition on the selected source will trigger the Group1 conversion. 1 A low-to-high transition on the selected source will trigger the Group1 conversion.
2–0	G1 SRC		Group1 Trigger Source. Any operation mode read/write: 000 to 111 The ADC module allows a trigger source to be selected for the Group1 from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

15.10.10ADC Group2 Trigger Source Select Register (ADG2SRC)

Figure 15-20 and Table 15-18 describe the ADG2SRC register.

Figure 15-20. ADC Group2 Trigger Source Select Register (ADG2SRC) [offset = 24h]



RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-18. ADC Group2 Trigger Source Select Register (ADG2SRC) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G2 EDG SEL		Group2 Trigger Edge Polarity Select. This bit determines the polarity of the transition on the selected source that triggers the Group2 conversion. This bit is only applicable if the Group2 is configured to be hardware-triggered. Any operation mode read/write: 0 A high-to-low transition on the selected source will trigger the Group2 conversion. 1 A low-to-high transition on the selected source will trigger the Group2 conversion.
2–0	G2 SRC		Group2 Trigger Source. Any operation mode read/write: 000 to 111 The ADC module allows a trigger source to be selected for the Group1 from up to eight options. These options are device-specific and the device specification must be referred to identify the actual trigger sources.

15.10.11ADC Event Group Interrupt Enable Control Register (ADEVINTENA)

Figure 15-21 and Table 15-19 describe the ADEVINTENA register.

Figure 15-21. ADC Event Group Interrupt Enable Control Register (ADEVINTENA) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												EV END INT EN		EV OVR INT EN	EV THR INT EN
R-0												RW-0	0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-19. ADC Event Group Interrupt Enable Control Register (ADEVINTENA) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	EV END INT EN		Event Group Conversion End Interrupt Enable. Please refer to Section 15.5.1 for more details on the conversion end interrupts. Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Event Group is done.
		1	An Event Group conversion end interrupt is generated when conversion of all the channels selected for conversion in the Event Group is done.
2	Reserved	0	Reads return zeros, writes have no effect.
1	EV OVR INT EN		Event Group Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Event Group results' memory which is already full. For more details on the overrun interrupts please refer to Section 15.5.3 Any operation mode read/write:
		0	No interrupt is generated if an Event Group memory overrun occurs.
		1	An Event Group memory overrun interrupt is generated if an Event Group memory overrun condition occurs.
0	EV THR INT EN		Event Group Threshold Interrupt Enable. An Event Group threshold interrupt occurs when the programmed Event Group threshold counter counts down to zero. Please refer to the Group Memory Threshold Interrupt description for more details. Any operation mode read/write:
		0	No interrupt is generated if the Event Group threshold counter reaches zero.

Table 15-19. ADC Event Group Interrupt Enable Control Register (ADEVINTENA) Field Descriptions

Bit	Name	Value	Description
		1	An Event Group threshold interrupt is generated if the Event Group threshold counter reaches zero.

15.10.12ADC Group1 Interrupt Enable Control Register (ADG1INTENA)

Figure 15-22 and Table 15-20 describe the ADG1INTENA register.

Figure 15-22. ADC Group1 Interrupt Enable Control Register (ADG1INTENA) [offset = 2Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												G1 END INT EN		G1 OVR INT EN	G1 THR INT EN
R-0												RW-0	0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-20. ADC Group1 Interrupt Enable Control Register (ADG1INTENA) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G1 END INT EN		Group1 Conversion End Interrupt Enable. Please refer to Section 15.5.1 for more details on the conversion end interrupts. Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Group1 is done.
		1	A Group1 conversion end interrupt is generated when conversion of all the channels selected for conversion in the Group1 is done.
2	Reserved	0	Reads return zeros, writes have no effect.
1	G1 OVR INT EN		Group1 Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Group1 results' memory which is already full. For more details on the overrun interrupts please refer to Section 15.5.3 Any operation mode read/write:
		0	No interrupt is generated if a Group1 memory overrun occurs.
		1	A Group1 memory overrun interrupt is generated if a Group1 memory overrun condition occurs.

Table 15-20. ADC Group1 Interrupt Enable Control Register (ADG1INTENA) Field Descriptions (Continued)

Bit	Name	Value	Description
0	G1 THR INT EN		Group1 Threshold Interrupt Enable. A Group1 threshold interrupt occurs when the programmed Group1 threshold counter counts down to zero. This counter decrements when the ADC module writes a new conversion result to the Group1 results' memory. The counter increments for each read of a conversion result from the Group1 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Group1 results' memory. Please refer to Section 15.5.2 for more details on the threshold interrupts. Any operation mode read/write:
		0	No interrupt is generated if the Group1 threshold counter reaches zero.
		1	A Group1 threshold interrupt is generated if the Group1 threshold counter reaches zero.

15.10.13ADC Group2 Interrupt Enable Control Register (ADG2INTENA)

Figure 15-23 and Table 15-21 describe the ADG1INTENA register.

Figure 15-23. ADC Group2 Interrupt Enable Control Register (ADG2INTENA) [offset = 30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												G2 END INT EN		G2 OVR INT EN	G2 THR INT EN
R-0												RW-0	0	RW-0	RW-0

RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-21. ADC Group2 Interrupt Enable Control Register (ADG2INTENA) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G2 END INT EN		Group2 Conversion End Interrupt Enable. Please refer to Section 15.5.1 for more details on the conversion end interrupts. Any operation mode read/write:
		0	No interrupt is generated when conversion of all the channels selected for conversion in the Group2 is done.
		1	A Group2 conversion end interrupt is generated when conversion of all the channels selected for conversion in the Group2 is done.
2	Reserved	0	Reads return zeros, writes have no effect.
1	G2 OVR INT EN		Group2 Memory Overrun Interrupt Enable. A memory overrun occurs when the ADC tries to write a new conversion result to the Group2 results' memory which is already full. For more details on the overrun interrupts please refer to Section 15.5.3 Any operation mode read/write:
		0	No interrupt is generated if a Group2 memory overrun occurs.
		1	A Group2 memory overrun interrupt is generated if a Group2 memory overrun condition occurs.

Table 15-21. ADC Group2 Interrupt Enable Control Register (ADG2INTENA) Field Descriptions (Continued)

Bit	Name	Value	Description
0	G2 THR INT EN		Group2 Threshold Interrupt Enable. A Group2 threshold interrupt occurs when the programmed Group2 threshold counter counts down to zero. This counter decrements when the ADC module writes a new conversion result to the Group2 results' memory. The counter increments for each read of a conversion result from the Group2 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Group2 results' memory. Please refer to Section 15.5.2 for more details on the threshold interrupts. Any operation mode read/write: 0 No interrupt is generated if the Group2 threshold counter reaches zero. 1 A Group2 threshold interrupt is generated if the Group2 threshold counter reaches zero.

15.10.14ADC Event Group Interrupt Flag Register (ADEVINTFLG)

Figure 15-21 and Table 15-19 describe the ADEVINTFLG register.

Figure 15-24. ADC Event Group Interrupt Flag Register (ADEVINTFLG) [offset = 34h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												EV END	EV MEM EMPTY	EV MEM OVER-RUN	EV THR INT FLG
R-0												RC-0	R-1	R-0	RC-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-22. ADC Event Group Interrupt Flag Register (ADEVINTFLG) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	EV END		Event Group Conversion End. This bit will be set only if the Event Group conversions are configured to be in the single-conversion mode. Any operation mode read: 0 All the channels selected for conversion in the Event Group have not yet been converted. 1 All the channels selected for conversion in the Event Group have been converted. An Event Group conversion end interrupt is generated, if enabled, when this bit gets set. This bit can be cleared by any one of the following ways: - By writing a ‘1’ to this bit. - By writing a ‘1’ to the Event Group status register bit 0 (EV END) - By reading one conversion result from the Event Group results’ memory in the “read from FIFO” mode - By writing a new set of channels to the Event Group channel select register.
2	EV MEM EMPTY		Event Group Results’ Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module. Any operation mode read: 0 The Event Group results’ memory is not empty. 1 The Event Group results’ memory is empty.
1	EV MEM OVER-RUN		Event Group Memory Overrun. This is a read-only bit; writes have no effect. Any operation mode read: 0 Event Group results’ memory has not overrun.

Table 15-22. ADC Event Group Interrupt Flag Register (ADEVINTFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Event Group results' memory has overrun.
0	EV THR INT FLG		Event Group Threshold Interrupt Flag. Any operation mode read: 0 The number of conversions completed for the Event Group is smaller than the threshold programmed in the Event Group interrupt threshold register. 1 The number of conversions completed for the Event Group is equal to or greater than the threshold programmed in the Event Group interrupt threshold register. This bit can be cleared by writing a '1' to it.

15.10.15ADC Group1 Interrupt Flag Register (ADG1INTFLG)

Figure 15-25 and Table 15-23 describe the ADG1INTFLG register.

Figure 15-25. ADC Group1 Interrupt Flag Register (ADG1INTFLG) [offset = 38h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												G1 END	G1 MEM EMPTY	G1 MEM OVER-RUN	G1 THR INT FLG
R-0												RC-0	R-1	R-0	RC-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-23. ADC Group1 Interrupt Flag Register (ADG1INTFLG) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G1 END		Group1 Conversion End. This bit will be set only if the Group1 conversions are configured to be in the single-conversion mode. Any operation mode read: 0 All the channels selected for conversion in the Group1 have not yet been converted. 1 All the channels selected for conversion in the Group1 have been converted. A Group1 conversion end interrupt is generated, if enabled, when this bit gets set. This bit can be cleared by any one of the following ways: - By writing a ‘1’ to this bit. - By writing a ‘1’ to the Group1 status register bit 0 (G1 END) - By reading one conversion result from the Group1 results’ memory in the “read from FIFO” mode - By writing a new set of channels to the Group1 channel select register.
2	G1 MEM EMPTY		Group1 Results’ Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module. Any operation mode read: 0 The Group1 results’ memory is not empty. 1 The Group1 results’ memory is empty.
1	G1 MEM OVER-RUN		Group1 Memory Overrun. This is a read-only bit; writes have no effect. Any operation mode read: 0 Group1 results’ memory has not overrun.

Table 15-23. ADC Group1 Interrupt Flag Register (ADG1INTFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Group1 results' memory has overrun.
0	G1 THR INT FLG		Group1 Threshold Interrupt Flag. Any operation mode read: 0 The number of conversions completed for the Group1 is smaller than the threshold programmed in the Group1 interrupt threshold register. 1 The number of conversions completed for the Group1 is equal to or greater than the threshold programmed in the Group1 interrupt threshold register. This bit can be cleared by writing a '1' to it.

15.10.16ADC Group2 Interrupt Flag Register (ADG2INTFLG)

Figure 15-26 and Table 15-24 describe the ADG2INTFLG register.

Figure 15-26. ADC Group2 Interrupt Flag Register (ADG2INTFLG) [offset = 3Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												G1 END	G1 MEM EMPTY	G1 MEM OVER-RUN	G1 THR INT FLG
R-0												RC-0	R-1	R-0	RC-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-24. ADC Group2 Interrupt Flag Register (ADG2INTFLG) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G2 END		Group2 Conversion End. This bit will be set only if the Group2 conversions are configured to be in the single-conversion mode. Any operation mode read: 0 All the channels selected for conversion in the Group2 have not yet been converted. 1 All the channels selected for conversion in the Group2 have been converted. A Group2 conversion end interrupt is generated, if enabled, when this bit gets set. This bit can be cleared by any one of the following ways: - By writing a ‘1’ to this bit. - By writing a ‘1’ to the Group2 status register bit 0 (G2 END) - By reading one conversion result from the Group2 results’ memory in the “read from FIFO” mode - By writing a new set of channels to the Group2 channel select register.
2	G2 MEM EMPTY		Group2 Results’ Memory Empty. This is a read-only bit; writes have no effect. It is not a source of an interrupt from the ADC module. Any operation mode read: 0 The Group2 results’ memory is not empty. 1 The Group2 results’ memory is empty.
1	G2 MEM OVER-RUN		Group2 Memory Overrun. This is a read-only bit; writes have no effect. Any operation mode read: 0 Group2 results’ memory has not overrun.

Table 15-24. ADC Group2 Interrupt Flag Register (ADG2INTFLG) Field Descriptions (Continued)

Bit	Name	Value	Description
		1	Group2 results' memory has overrun.
0	G2 THR INT FLG		Group2 Threshold Interrupt Flag. Any operation mode read:
		0	The number of conversions completed for the Group2 is smaller than the threshold programmed in the Group2 interrupt threshold register.
		1	The number of conversions completed for the Group2 is equal to or greater than the threshold programmed in the Group2 interrupt threshold register. This bit can be cleared by writing a '1' to it.

15.10.17ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR)

Figure 15-27 and Table 15-25 describe the ADEVTHRINTCR register.

Figure 15-27. ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR) [offset = 40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Sign Extension							EVTHR.8:0								
R-0							RW-0								

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

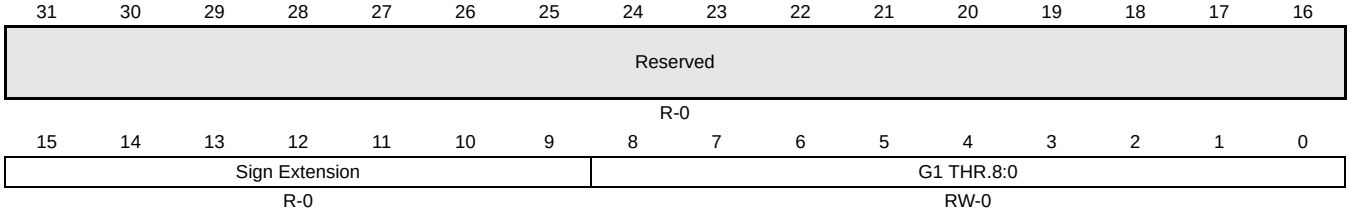
Table 15-25. ADC Event Group Threshold Interrupt Control Register (ADEVTHRINTCR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15–9	Sign Extension		These bits always read the same as the bit 8 of this register.
8–0	EV THR		Event Group Threshold Counter. Before ADC conversions begin on the Event Group, this field is initialized to the number of conversion results that the Event Group memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Event Group results' memory. The counter increments for each read of a conversion result from the Event Group results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Event Group results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Event Group FIFO will leave the threshold counter unchanged. In case of an Event Group Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Event Group threshold counter is not decremented. Please refer to Section 15.5.2 for more details on the threshold interrupts.

15.10.18ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR)

Figure 15-28 and Table 15-26 describe the ADG1THRINTCR register.

Figure 15-28. ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR) [offset = 44h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

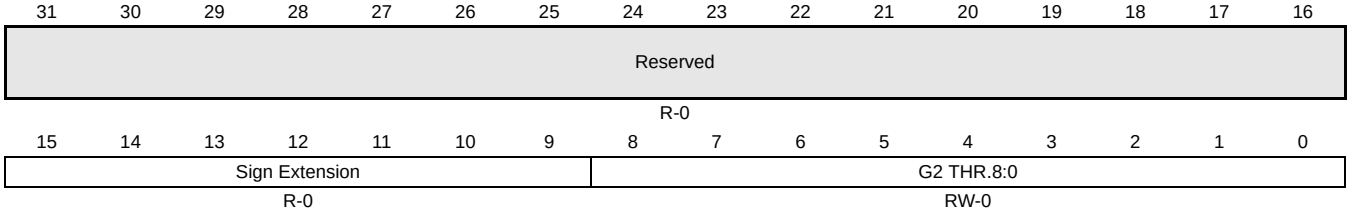
Table 15-26. ADC Group1 Threshold Interrupt Control Register (ADG1THRINTCR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15–9	Sign Extension		These bits always read the same as the bit 8 of this register.
8–0	G1 THR		Group1 Threshold Counter. Before ADC conversions begin on the Group1, this field is initialized to the number of conversion results that the Group1 memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Group1 results' memory. The counter increments for each read of a conversion result from the Group1 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Group1 results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Group1 FIFO will leave the threshold counter unchanged. In case of an Group1 Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Group1 threshold counter is not decremented. Please refer to Section 15.5.2 for more details on the threshold interrupts.

15.10.19ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR)

Figure 15-29 and Table 15-27 describe the ADG2THRINTCR register.

Figure 15-29. ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR) [offset = 48h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-27. ADC Group2 Threshold Interrupt Control Register (ADG2THRINTCR) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15–9	Sign Extension		These bits always read the same as the bit 8 of this register.
8–0	G2 THR		Group2 Threshold Counter. Before ADC conversions begin on the Group2, this field is initialized to the number of conversion results that the Group2 memory should contain before interrupting the CPU. This counter decrements when the ADC module writes a new conversion result to the Group2 results' memory. The counter increments for each read of a conversion result from the Group2 results' memory in the "read from FIFO" mode. The threshold counter is not affected for a direct read from the Group2 results' memory. Also, a simultaneous ADC write and a CPU/DMA read from the Group2 FIFO will leave the threshold counter unchanged. In case of an Group2 Results' memory overrun condition, if new conversion results are not allowed to overwrite the existing memory contents, then the Group2 threshold counter is not decremented. Please refer to Section 15.5.2 for more details on the threshold interrupts.

15.10.20ADC Results' Memory Configuration Register (ADBNDCR)

Figure 15-30 and Table 15-28 describe the ADBNDCR register.

Figure 15-30. ADC Results' Memory Configuration Register (ADBNDCR) [offset = 58h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							BNDA.8:0								
0							RW-0								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							BNDB.8:0								
0							RW-0								

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-28. ADC Results' Memory Configuration Register (ADBNDCR) Field Descriptions

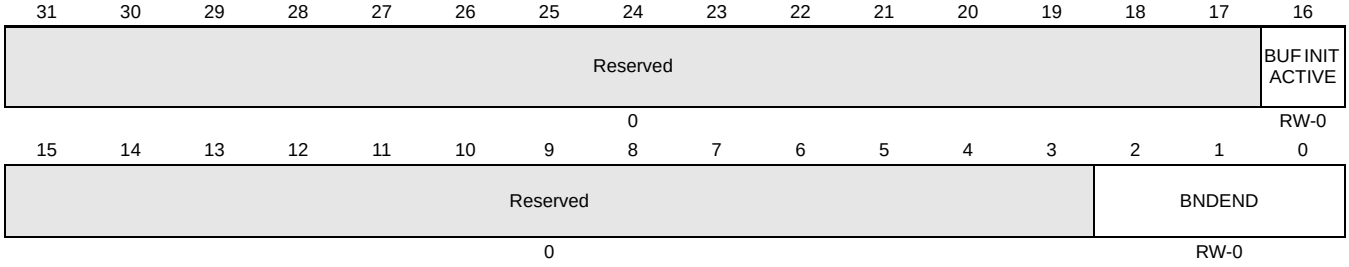
Bit	Name	Value	Description
31–25	Reserved	0	Reads return zeros, writes have no effect.
24–16	BNDA		Buffer Boundary A. These bits determine the memory available for the Event Group conversion results. The memory available is specified in terms of pairs of result buffers. Any operation mode read/write:
		0	Event Group conversions are not required. If Event Group conversions are performed with the BNDA value of zero, then the Event Group memory size will default to 1024 words. For proper usage of the ADC results' memory, configure the BNDA value to be non-zero and lower than the BNDB value.
		0x001 to 0x1FF	A total of (2 * BNDA) buffers are available in the ADC results' memory for storing Event Group conversion results.
15–9	Reserved	0	Reads return zeros, writes have no effect.
8–0	BNDB		Buffer Boundary B. These bits specify the number of buffers allocated for the Event Group plus the number of buffers allocated for the Group1. The number of buffer pairs allocated for storing Group1 conversion results can be determined by subtracting BNDA from BNDB. As a result, BNDB must always be specified as greater than or equal to BNDA. Any operation mode read:
		0	Event Group as well as Group1 conversions are not required.
		0x001 to 0x1FF	A total of 2 * (BNDB - BNDA) buffers are available in the ADC results' memory for storing Group1 conversion results.

Please refer to Section 15.3.5 for further details on how the conversion results are stored in the ADC results' RAM.

15.10.21ADC Results' Memory Size Configuration Register (ADBNDEND)

Figure 15-31 and Table 15-29 describe the ADBNDCR register.

Figure 15-31. ADC Results' Memory Size Configuration Register (ADBNDEND) [offset = 5Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-29. ADC Results' Memory Size Configuration Register (ADBNDEND) Field Descriptions

Bit	Name	Value	Description
31–17	Reserved	0	Reads return zeros, writes have no effect.
16	BUF INIT ACTIVE		ADC Results' Memory Auto-initialization Status. Any operation mode read/write:
		0	ADC Results' Memory is currently not being initialized, and the ADC is available. If this bit is read as '0' after triggering an auto-initialization of the ADC results' memory, then the ADC results' memory has been completely initialized to zeros. For devices requiring parity checking on the ADC results' memory, the parity bit in the results' memory will also be initialized according to the parity polarity. The parity polarity as well as the auto-initialization process is controlled by the System module. Please refer to the TMS470Px Platform Architecture specification for more details.
		1	ADC results' memory is being initialized, and the ADC is not available for conversion.
15–3	Reserved	0	Reads return zeros, writes have no effect.
2–0	BNDEND		Buffer Boundary End. These bits specify the total number of memory buffers available for storing the ADC conversion results. These bits should be programme to match the size of the ADC results' memory available on the specific device. Please refer to the specific device specification for information on the memory available for storing the ADC conversion results. Any operation mode read/write:
		000b	16 words available for storing ADC conversion results.
		001b	32 words available for storing ADC conversion results.
		010b	64 words available for storing ADC conversion results.
		011b	128 words available for storing ADC conversion results.

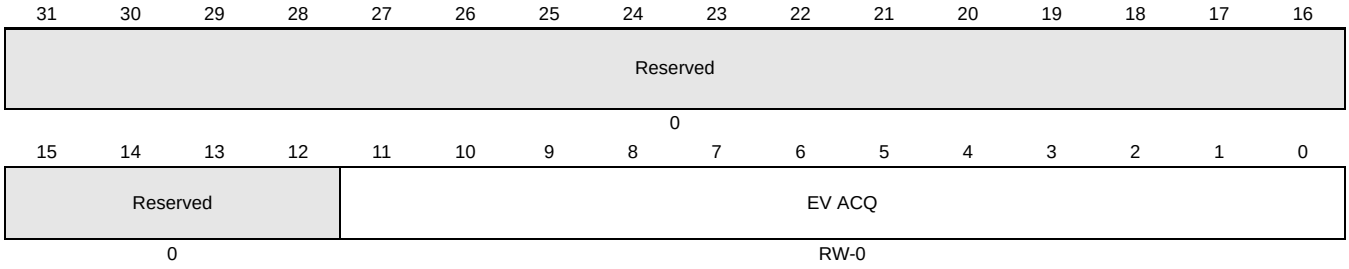
Table 15-29. ADC Results' Memory Size Configuration Register (ADBNDEND) Field Descriptions (Continued)

Bit	Name	Value	Description
		100b	192 words available for storing ADC conversion results.
		101b	256 words available for storing ADC conversion results.
		110b	512 words available for storing ADC conversion results.
		111b	1024 words available for storing ADC conversion results.

15.10.22ADC Event Group Sampling Time Configuration Register (ADEVSAMP)

Figure 15-32 and Table 15-30 describe the ADEVSAMP register.

Figure 15-32. ADC Event Group Sampling Time Configuration Register (ADEVSAMP) [offset = 60h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

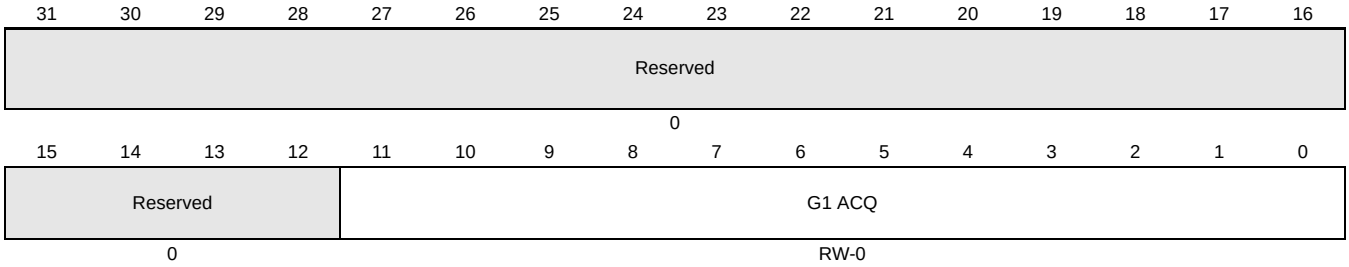
Table 15-30. ADC Event Group Sampling Time Configuration Register (ADEVSAMP) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved	0	Reads return zeros, writes have no effect.
11–0	EV ACQ		Event Group Acquisition Time. These bits define the sampling window (SW) for the Event Group conversions. SW = EV ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be guaranteed by configuring the EV ACQ value properly considering the frequency of the ADCLK signal. Please refer to the device specification to determine the minimum sampling time for this device.

15.10.23ADC Group1 Sampling Time Configuration Register (ADG1SAMP)

Figure 15-33 and Table 15-31 describe the ADG1SAMP register.

Figure 15-33. ADC Group1 Sampling Time Configuration Register (ADG1SAMP) [offset = 64h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

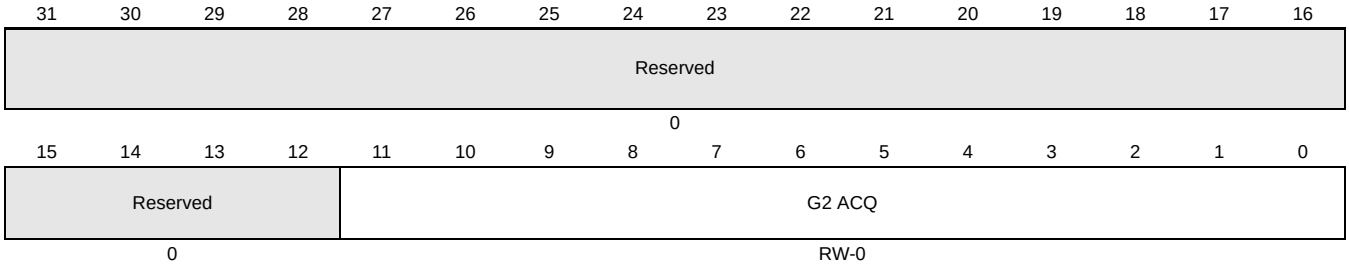
Table 15-31. ADC Group1 Sampling Time Configuration Register (ADG1SAMP) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved	0	Reads return zeros, writes have no effect.
11–0	G1 ACQ		Group1 Acquisition Time. These bits define the sampling window (SW) for the Group1 conversions. SW = G1 ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be guaranteed by configuring the G1 ACQ value properly considering the frequency of the ADCLK signal. Please refer to the device specification to determine the minimum sampling time for this device.

15.10.24ADC Group2 Sampling Time Configuration Register (ADG2SAMP)

Figure 15-34 and Table 15-32 describe the ADG2SAMP register.

Figure 15-34. ADC Group2 Sampling Time Configuration Register (ADG2SAMP) [offset = 68h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-32. ADC Group2 Sampling Time Configuration Register (ADG2SAMP) Field Descriptions

Bit	Name	Value	Description
31–12	Reserved	0	Reads return zeros, writes have no effect.
11–0	G2 ACQ		Group2 Acquisition Time. These bits define the sampling window (SW) for the Group2 conversions. SW = G2 ACQ + 2 in terms of ADCLK cycles. There are two factors that determine the minimum sampling window value required: First, the ADC module design requires that SW >= 3 ADCLK cycles. Second, the ADC input impedance necessitates a certain minimum sampling time. This needs to be guaranteed by configuring the G2 ACQ value properly considering the frequency of the ADCLK signal. Please refer to the device specification to determine the minimum sampling time for this device.

15.10.25ADC Event Group Status Register (ADEVSR)

Figure 15-35 and Table 15-33 describe the ADEVSR register.

Figure 15-35. ADC Event Group Status Register (ADEVSR) [offset = 6Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												EV MEM EMPTY	EVBUSY	EV STOP	EV END
R-0												R-1	R-0	R-0	R-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-33. ADC Event Group Status Register (ADEVSR) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	EV MEM EMPTY		Event Group Results' Memory Empty. This bit can be effectively used only when the conversion results are read out of the Event Group results' memory in the "read from FIFO" mode. Any operation mode read:
		0	The Event Group results' memory has valid conversion results.
		1	The Event Group results' memory is empty, or does not contain any unread conversion results.
2	EV BUSY		Event Group Conversion Busy. Any operation mode read:
		0	Event Group conversions are neither in progress nor frozen.
		1	Event Group conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Event Group is configured to be in the continuous conversion mode.
1	EV STOP		Event Group Conversion Stopped. Any operation mode read:
		0	Event Group conversions are not currently frozen.
		1	Event Group conversions are currently frozen.
0	EV END		Event Group Conversions Ended. Any operation mode read:

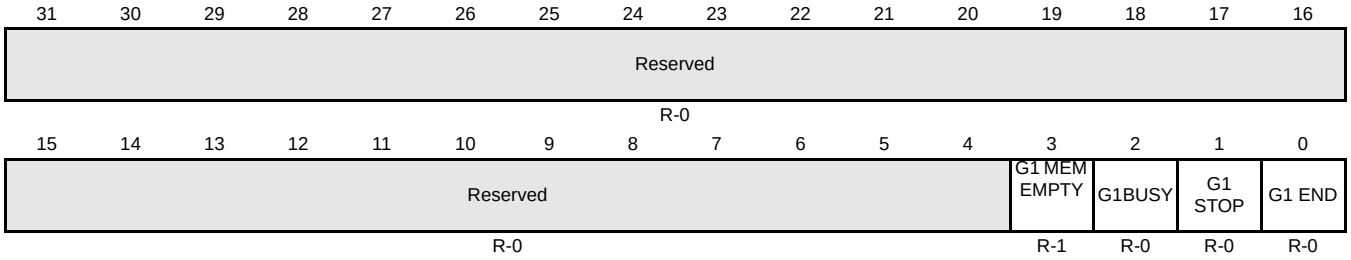
Table 15-33. ADC Event Group Status Register (ADEVSR) Field Descriptions (Continued)

Bit	Name	Value	Description
		0	Event Group conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Event Group has completed. This bit can be cleared under the following conditions: - By reading a conversion result from the Event Group results' memory in the "read from FIFO" mode - By writing a new value to the Event Group channel select register ADEVSEL - By writing a '1' to this bit - By disabling the ADC module by clearing the ADC EN bit

15.10.26ADC Group1 Status Register (ADG1SR)

Figure 15-36 and Table 15-34 describe the ADG1SR register.

Figure 15-36. ADC Group1 Status Register (ADG1SR) [offset = 70h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-34. ADC Group1 Status Register (ADG1SR) Field Descriptions

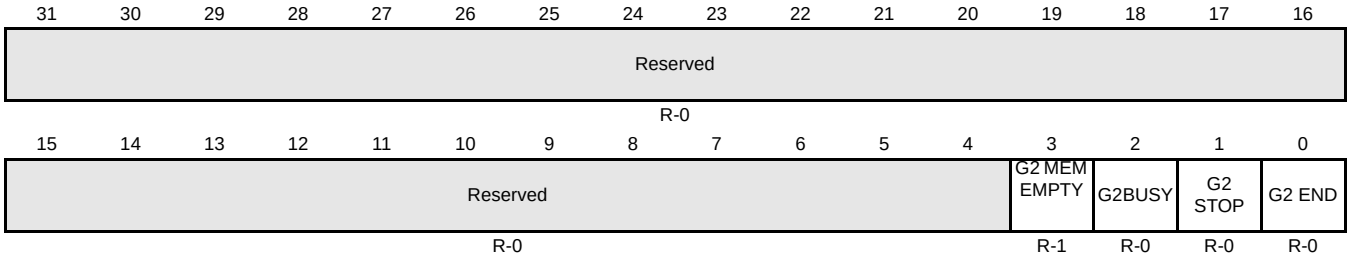
Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G1 MEM EMPTY		Group1 Results' Memory Empty. This bit can be effectively used only when the conversion results are read out of the Group1 results' memory in the "read from FIFO" mode. Any operation mode read:
		0	The Group1 results' memory has valid conversion results.
		1	The Group1 results' memory is empty, or does not contain any unread conversion results.
2	G1 BUSY		Group1 Conversion Busy. Any operation mode read:
		0	Group1 conversions are neither in progress nor frozen.
		1	Group1 conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Group1 is configured to be in the continuous conversion mode.
1	G1 STOP		Group1 Conversion Stopped. Any operation mode read:
		0	Group1 conversions are not currently frozen.
		1	Group1 conversions are currently frozen.
0	G1 END		Group1 Conversions Ended. Any operation mode read:

Bit	Name	Value	Description
		0	Group1 conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Group1 has completed. This bit can be cleared under the following conditions: - By reading a conversion result from the Group1 results' memory in the "read from FIFO" mode - By writing a new value to the Group1 channel select register ADG1SEL - By writing a '1' to this bit - By disabling the ADC module by clearing the ADC EN bit

15.10.27ADC Group2 Status Register (ADG2SR)

Figure 15-37 and Table 15-35 describe the ADG2SR register.

Figure 15-37. ADC Group2 Status Register (ADG2SR) [offset = 74h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-35. ADC Group2 Status Register (ADG2SR) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3	G2 MEM EMPTY		Group2 Results' Memory Empty. This bit can be effectively used only when the conversion results are read out of the Group2 results' memory in the "read from FIFO" mode. Any operation mode read:
		0	The Group2 results' memory has valid conversion results.
		1	The Group2 results' memory is empty, or does not contain any unread conversion results.
2	G2 BUSY		Group2 Conversion Busy. Any operation mode read:
		0	Group2 conversions are neither in progress nor frozen.
		1	Group2 conversions are either in progress, or are frozen for servicing some other group. This bit will always be set when the Group2 is configured to be in the continuous conversion mode.
1	G2 STOP		Group2 Conversion Stopped. Any operation mode read:
		0	Group2 conversions are not currently frozen.
		1	Group2 conversions are currently frozen.
0	G2 END		Group2 Conversions Ended. Any operation mode read:

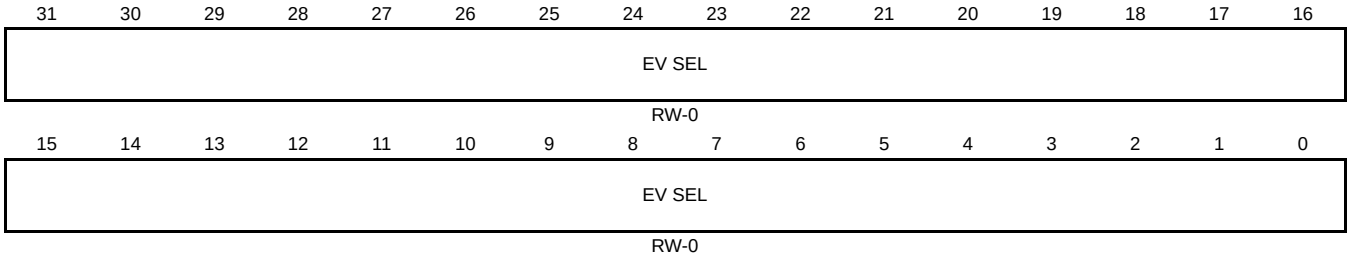
Table 15-35. ADC Group2 Status Register (ADG2SR) Field Descriptions (Continued)

Bit	Name	Value	Description
		0	Group2 conversions have either not been started or have not yet completed since the last time this status bit was cleared.
		1	The conversion for all the channels selected in the Group2 has completed. This bit can be cleared under the following conditions: - By reading a conversion result from the Group2 results' memory in the "read from FIFO" mode - By writing a new value to the Group2 channel select register ADG2SEL - By writing a '1' to this bit - By disabling the ADC module by clearing the ADC EN bit

15.10.28ADC Event Group Channel Select Register (ADEVSEL)

Figure 15-38 and Table 15-36 describe the ADEVSEL register.

Figure 15-38. ADC Event Group Channel Select Register (ADEVSEL) [offset = 78h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-36. ADC Event Group Channel Select Register (ADEVSEL) Field Descriptions

Bit	Name	Value	Description
31–0	EV SEL		Event Group channels selected. Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Event Group.
		Non-zero	The channels marked by the bit positions that are set to ‘1’ will be converted in ascending order when the Event Group is triggered.

Note: Clearing ADEVSEL During a Conversion
Writing 0x0000 to ADEVSEL stops Event Group conversions. This does not cause the ADC Event Group Results’ Memory pointer or the Event Group Threshold Register to be reset.

Note: Writing A Non-Zero Value To ADEVSEL During a Conversion
Writing a new value to ADEVSEL while a Channel in Event Group is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADEVSEL selection. This also causes the ADC Event Group Results’ Memory pointer to be reset so that the memory allocated for storing the Event Group conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

15.10.29ADC Group1 Channel Select Register (ADG1SEL)

Figure 15-39 and Table 15-37 describe the ADG1SEL register.

Figure 15-39. ADC Group1 Channel Select Register (ADG1SEL) [offset = 7Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
G1 SEL															
RW-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
G1 SEL															
RW-0															

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-37. ADC Group1 Channel Select Register (ADG1SEL) Field Descriptions

Bit	Name	Value	Description
31–0	G1 SEL		Group1 channels selected.
			Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Group1.
		Non-zero	The channels marked by the bit positions that are set to ‘1’ will be converted in ascending order when the Group1 is triggered.

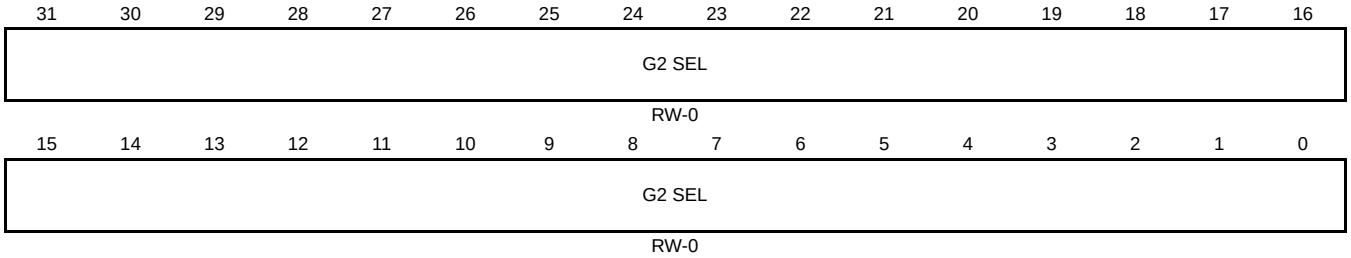
Note: Clearing ADG1SEL During a Conversion
Writing 0x0000 to ADG1SEL stops Group1 conversions. This does not cause the ADC Group1 Results’ Memory pointer or the Group1 Threshold Register to be reset.

Note: Writing A Non-Zero Value To ADG1SEL During a Conversion
Writing a new value to ADG1SEL while a Channel in Group1 is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADG1SEL selection. This also causes the ADC Group1 Results’ Memory pointer to be reset so that the memory allocated for storing the Group1 conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

15.10.30ADC Group2 Channel Select Register (ADG2SEL)

Figure 15-40 and Table 15-38 describe the ADG2SEL register.

Figure 15-40. ADC Group2 Channel Select Register (ADG2SEL) [offset = 80h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-38. ADC Group2 Channel Select Register (ADG2SEL) Field Descriptions

Bit	Name	Value	Description
31–0	G2 SEL		Group2 channels selected. Any operation mode read/write:
		0	No ADC input channel is selected for conversion in the Group2.
		Non-zero	The channels marked by the bit positions that are set to ‘1’ will be converted in ascending order when the Group2 is triggered.

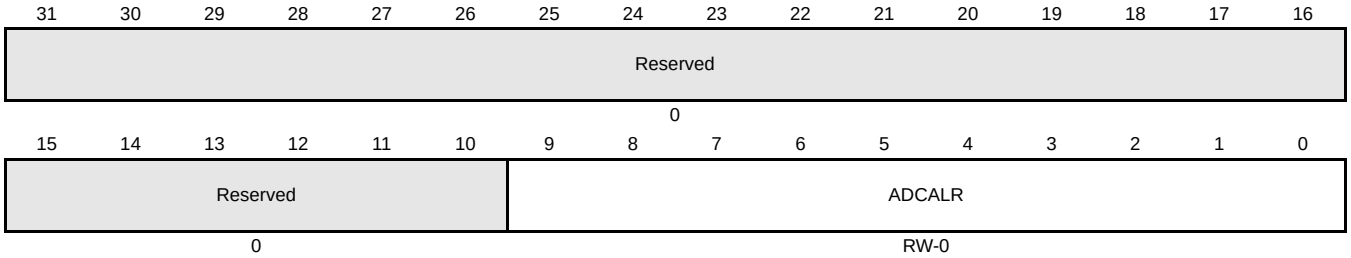
Note: Clearing ADG2SEL During a Conversion
Writing 0x0000 to ADG2SEL stops Group2 conversions. This does not cause the ADC Group2 Results’ Memory pointer or the Group2 Threshold Register to be reset.

Note: Writing A Non-Zero Value To ADG2SEL During a Conversion
Writing a new value to ADG2SEL while a Channel in Group2 is being converted results in a new conversion sequence starting immediately with the highest priority channel in the new ADG2SEL selection. This also causes the ADC Group2 Results’ Memory pointer to be reset so that the memory allocated for storing the Group2 conversion results gets overwritten. Care should be taken to re-program the corresponding Interrupt Threshold Counter or DMA Threshold Counter again so that correct number of conversions happen before a Threshold interrupt or Block DMA request is generated.

15.10.31ADC Calibration and Error Offset Correction Register (ADCALR)

Figure 15-41 and Table 15-39 describe the ADCALR register.

Figure 15-41. ADC Calibration and Error Offset Correction Register (ADCALR) [offset = 84h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-39. ADC Calibration and Error Offset Correction Register (ADCALR) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved	0	Reads return zeros, writes have no effect.
9–0	ADCALR		ADC Calibration Result and Offset Error Correction Value. The ADC module writes the results of the calibration conversions to this register. The application is required to use these conversion results and determine the ADC offset error. The application can then compute the correction for the offset error and this correction value needs to be written back to the ADCALR register in the 2's complement form. During normal conversion (when calibration is disabled), the ADCALR register contents are automatically added to each digital output from the ADC core before it is stored in the ADC results' memory. For more details on error calibration, please refer to the Section 15.8.1.

15.10.32ADC State Machine Status Register (ADSMSTATE)

Figure 15-42 and Table 15-40 describe the ADSMSTATE register.

Figure 15-42. ADC State Machine Status Register (ADSMSTATE) [offset = 88h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SMSTATE			
0												R-0			

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-40. ADC State Machine Status Register (ADSMSTATE) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.

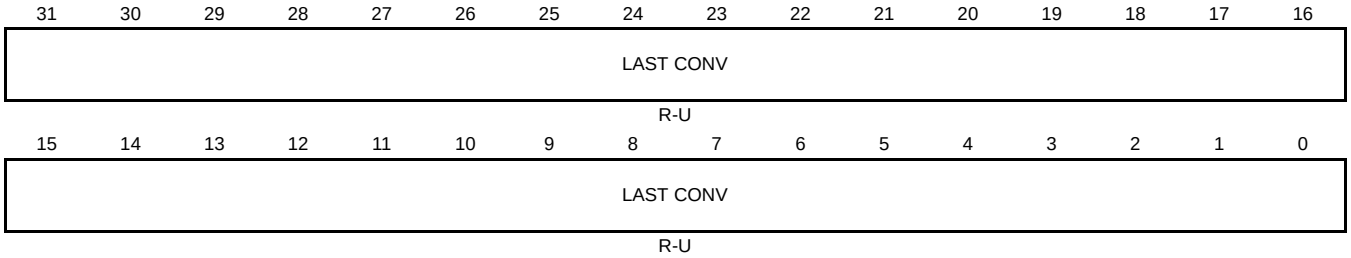
Table 15-40. ADC State Machine Status Register (ADSMSTATE) Field Descriptions (Continued)

Bit	Name	Value	Description
3–0	SM STATE		ADC State Machine Current State. These bits reflect the current state of the state machine. The ADC logic state machine goes through various states for converting each channel selected in each of the three conversion groups. Each of these states have been assigned a “state number”. This information is very useful for debug purposes. Any operating mode read:
		0000	IDLE
		0001	CONV_EV
		0010	CONV_SW1
		0011	CONV_SW2
		0100	CONV_CAL
		0101	START_EV
		0110	START_SW1
		0111	START_SW2
		1000	START_CAL
		1001	WAIT_EV
		1010	WAIT_SW1
		1011	WAIT_SW2
		1100	WAIT_CAL
		1101 to 1111	Invalid State. The ADC module state machine must never be in any of these states.

15.10.33ADC Channel Last Conversion Value Register (ADLASTCONV)

Figure 15-43 and Table 15-41 describe the ADLASTCONV register.

Figure 15-43. ADC Channel Last Conversion Value Register (ADLASTCONV) [offset = 8Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

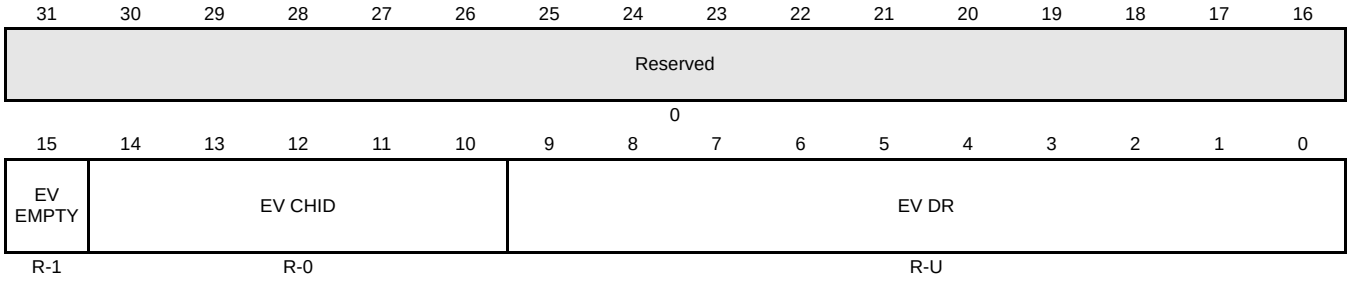
Table 15-41. ADC Channel Last Conversion Value Register (ADLASTCONV) Field Descriptions

Bit	Name	Value	Description
31–0	LAST CONV		ADC Input Channel's Last Converted Value. This register indicates whether the last converted value for a particular input channel was lower or higher than the mid-point of the reference voltage. In other words, this register acts as a digital input register and can be read by the application to determine the digital level at the input pins. This data is only valid for an input channel if it has been converted at least once. Any operation mode read for each bit of this register:
		0	A level lower than the midpoint reference voltage was measured at the last conversion for this channel
		1	A level higher than the midpoint reference voltage was measured at the last conversion for this channel.

15.10.34ADC Event Group Results' FIFO (ADEVBUFFER)

Figure 15-44 and Table 15-42 describe the ADEVBUFFER register.

Figure 15-44. ADC Event Group Results' FIFO (ADEVBUFFER) [offset = 90h - AFh]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

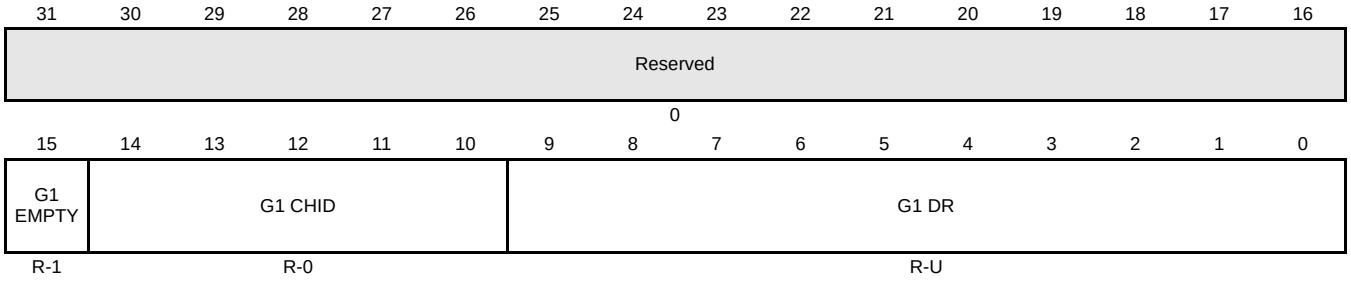
Table 15-42. ADC Event Group Results' FIFO (ADEVBUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	EV EMPTY		Event Group FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read: 0 The data in the EV DR field of this buffer is valid. 1 The data in the EV DR field of this buffer is not valid and there are no valid data in the Event Group results' memory.
14–10	EV CHID		Event Group Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read: 0 The conversion result in the EV DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Event Group mode control register (ADC Event Group Operating Mode Control Register (ADEVMODECR)). 00001b to 11111b The conversion result in the EV DR field of this buffer is from the ADC input channel number denoted by the EV CHID field.
9–0	EV DR		Event Group Digital Conversion Result. The Event Group results' FIFO location is aliased eight times, so that any word-aligned read from the address range 0x90 to 0xAF results in one conversion result to be read from the Event Group results' memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Event Group results' memory with just one instruction.

15.10.35ADC Group1 Results' FIFO (ADG1BUFFER)

Figure 15-45 and Table 15-43 describe the ADG1BUFFER register.

Figure 15-45. ADC Group1 Results' FIFO (ADG1BUFFER) [offset = B0h - CFh]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

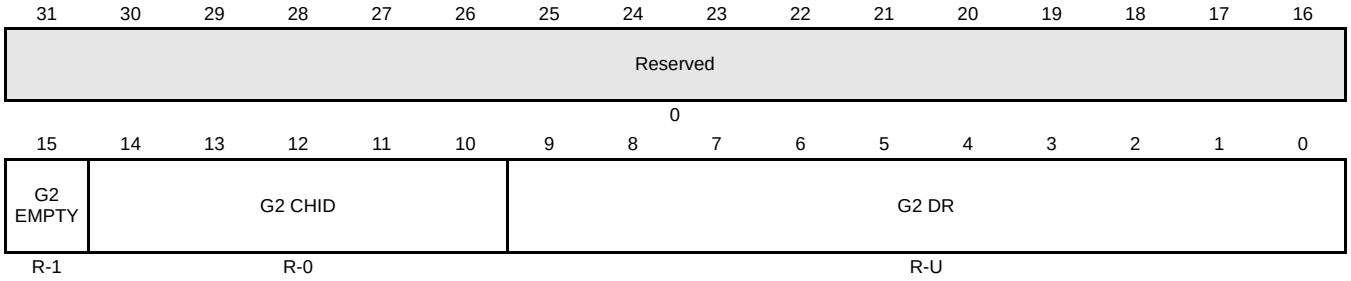
Table 15-43. ADC Group1 Results' FIFO (ADG1BUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	G1 EMPTY		Group1 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results. Any operation mode read: 0 The data in the G1 DR field of this buffer is valid. 1 The data in the G1 DR field of this buffer is not valid and there are no valid data in the Group1 results' memory.
14–10	G1 CHID		Group1 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results. Any operation mode read: 0 The conversion result in the G1 DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group1 mode control register (ADC Group1 Operating Mode Control Register (ADG1MODECR)). 00001b to 11111b The conversion result in the G1 DR field of this buffer is from the ADC input channel number denoted by the G1 CHID field.
9–0	G1 DR		Group1 Digital Conversion Result. The Group1 results' FIFO location is aliased eight times, so that any word-aligned read from the address range 0xB0 to 0xCF results in one conversion result to be read from the Group1 results' memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Group1 results' memory with just one instruction.

15.10.36ADC Group2 Results' FIFO (ADG2BUFFER)

Figure 15-45 and Table 15-43 describe the ADG2BUFFER register.

Figure 15-46. ADC Group2 Results' FIFO (ADG2BUFFER) [offset = D0h - EFh]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

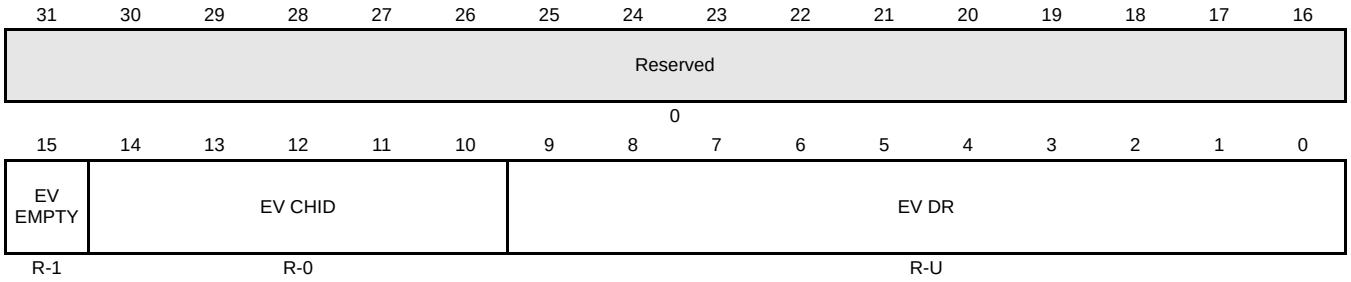
Table 15-44. ADC Group2 Results' FIFO (ADG2BUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	G2 EMPTY		Group2 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results. Any operation mode read: 0 The data in the G2 DR field of this buffer is valid. 1 The data in the G2 DR field of this buffer is not valid and there are no valid data in the Group1 results' memory.
14–10	G2 CHID		Group2 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results. Any operation mode read: 0 The conversion result in the G2 DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group2 mode control register (ADC Group2 Operating Mode Control Register (ADG2MODECR)). 00001b to 11111b The conversion result in the G2 DR field of this buffer is from the ADC input channel number denoted by the G2 CHID field.
9–0	G2 DR		Group2 Digital Conversion Result. The Group2 results' FIFO location is aliased eight times, so that any word-aligned read from the address range 0xD0 to 0xEF results in one conversion result to be read from the Group2 results' memory. This allows the ARM LDMIA instruction to read out up to 8 conversion results from the Group2 results' memory with just one instruction.

15.10.37ADC Event Group Results' Emulation FIFO (ADEVEMUBUFFER)

Figure 15-47 and Table 15-45 describe the ADEVBUFFER register. A read from this location also gives out one conversion result from the Event Group results' memory along with the "EV EMPTY" status bit and the optional channel id. However, this read will not affect any of the status flags in the Event Group interrupt flag register or the Event Group status register. This register is useful for debuggers.

Figure 15-47. ADC Event Group Results' Emulation FIFO (ADEVEMUBUFFER) [offset = F0h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

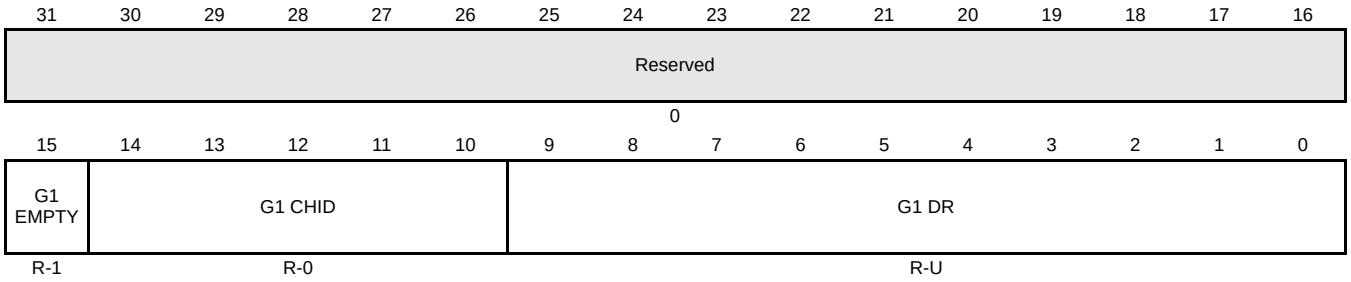
Table 15-45. ADC Event Group Results' Emulation FIFO (ADEVEMUBUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	EV EMPTY		Event Group FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read: 0 The data in the EV DR field of this buffer is valid. 1 The data in the EV DR field of this buffer is not valid and there are no valid data in the Event Group results' memory.
14–10	EV CHID		Event Group Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Event Group conversion results. Any operation mode read: 0 The conversion result in the EV DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Event Group mode control register (ADC Event Group Operating Mode Control Register (ADEVMODECR)). 00001b to 11111b The conversion result in the EV DR field of this buffer is from the ADC input channel number denoted by the EV CHID field.
9–0	EV DR		Event Group Digital Conversion Result.

15.10.38ADC Group1 Results' Emulation FIFO (ADG1EMUBUFFER)

Figure 15-48 and Table 15-46 describe the ADG1BUFFER register. A read from this location also gives out one conversion result from the Group1 results' memory along with the "G1 EMPTY" status bit and the optional channel id. However, this read will not affect any of the status flags in the Group1 interrupt flag register or the Group1 status register. This register is useful for debuggers.

Figure 15-48. ADC Group1 Results' Emulation FIFO (ADG1EMUBUFFER) [offset = F4h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-46. ADC Group1 Results' Emulation FIFO (ADG1EMUBUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	G1 EMPTY		Group1 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results. Any operation mode read: 0 The data in the G1 DR field of this buffer is valid. 1 The data in the G1 DR field of this buffer is not valid and there are no valid data in the Group1 results' memory.
14–10	G1 CHID		Group1 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group1 conversion results. Any operation mode read: 0 The conversion result in the G1 DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group1 mode control register (ADC Event Group Operating Mode Control Register (ADEVMODECR)). 00001b to 11111b The conversion result in the G1 DR field of this buffer is from the ADC input channel number denoted by the G1 CHID field.
9–0	G1 DR		Group1 Digital Conversion Result.

15.10.39ADC Group2 Results' Emulation FIFO (ADG2EMUBUFFER)

Figure 15-48 and Table 15-46 describe the ADG2BUFFER register. A read from this location also gives out one conversion result from the Group2 results' memory along with the "G2 EMPTY" status bit and the optional channel id. However, this read will not affect any of the status flags in the Group2 interrupt flag register or the Group2 status register. This register is useful for debuggers.

Figure 15-49. ADC Group2 Results' Emulation FIFO (ADG2EMUBUFFER) [offset = F8h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G2 EMPTY	G2 CHID					G2 DR									
R-1	R-0					R-U									

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-47. ADC Group2 Results' Emulation FIFO (ADG2EMUBUFFER) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved	0	Reads return zeros, writes have no effect.
15	G2 EMPTY		Group2 FIFO Empty. This bit is applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results. Any operation mode read: 0 The data in the G2 DR field of this buffer is valid. 1 The data in the G2 DR field of this buffer is not valid and there are no valid data in the Group2 results' memory.
14–10	G2 CHID		Group2 Channel Id. These bits are also applicable only when the "read from FIFO" mode is used for reading the Group2 conversion results. Any operation mode read: 0 The conversion result in the G2 DR field of this buffer is from the ADC input channel 0, or the channel id mode is disabled in the Group2 mode control register (ADC Event Group Operating Mode Control Register (ADEVMODECR)). 00001b to 11111b The conversion result in the G2 DR field of this buffer is from the ADC input channel number denoted by the G2 CHID field.
9–0	G2 DR		Group2 Digital Conversion Result.

15.10.40ADC ADEVT Pin Direction Control Register (ADEVTDIR)

Figure 15-50 and Table 15-48 describe the ADEVTDIR register.

Figure 15-50. ADC ADEVT Pin Direction Control Register (ADEVTDIR) [offset = FCh]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ADEVT DIR
0															RW-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

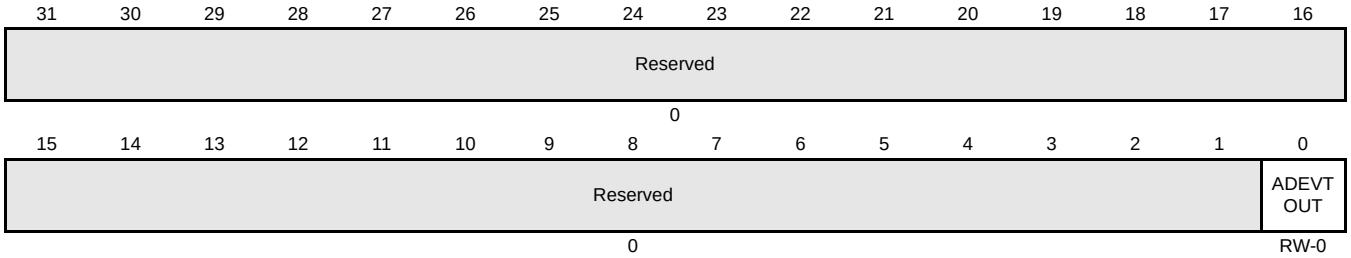
Table 15-48. ADC ADEVT Pin Direction Control Register (ADEVTDIR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT DIR		ADEVT Pin Direction. Any operating mode read/write:
		0	ADEVT is an input pin; the output buffer is disabled.
		1	ADEVT is an output pin; the output buffer is enabled.

15.10.41ADC ADEVT Pin Output Value Control Register (ADEVTOUT)

Figure 15-51 and Table 15-49 describe the ADEVTOUT register.

Figure 15-51. ADC ADEVT Pin Output Value Control Register (ADEVTOUT) [offset = 100h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

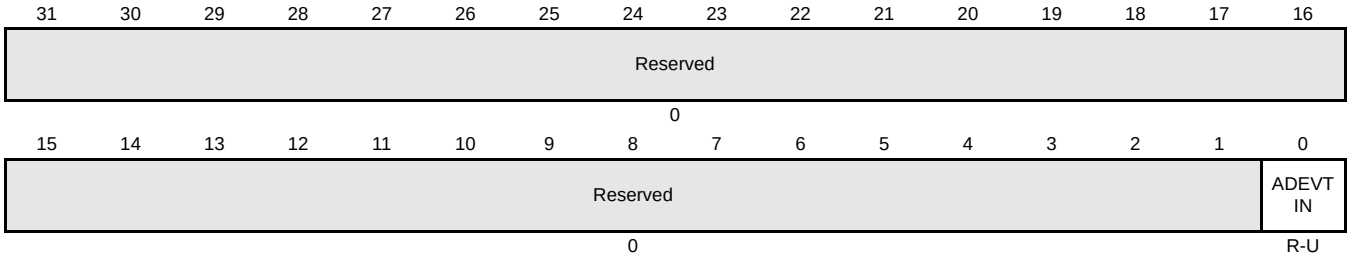
Table 15-49. ADC ADEVT Pin Output Value Control Register (ADEVTOUT) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT OUT		ADEVT Pin Output Value. This bit determines the logic level to be output to the ADEVT pin when the pin is configured to be an output pin. Any operating mode read/write:
		0	Output logic LOW on the ADEVT pin.
		1	Output logic HIGH on the ADEVT pin.

15.10.42ADC ADEVT Pin Input Value Register (ADEVTIN)

Figure 15-52 and Table 15-50 describe the ADEVTOUT register.

Figure 15-52. ADC ADEVT Pin Input Value Register (ADEVTIN) [offset = 104h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-50. ADC ADEVT Pin Input Value Register (ADEVTIN) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT IN		ADEVT Pin Input Value. This is a read-only bit which reflects the logic level on the ADEVT pin. Any operating mode read:
		0	Logic LOW present on the ADEVT pin.
		1	Logic HIGH present on the ADEVT pin.

15.10.43ADC ADEVT Pin Set Register (ADEVTSET)

Figure 15-53 and Table 15-51 describe the ADEVTSET register.

Figure 15-53. ADC ADEVT Pin Set Register (ADEVTSET) [offset = 108h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ADEVT SET
0															RW-0

R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

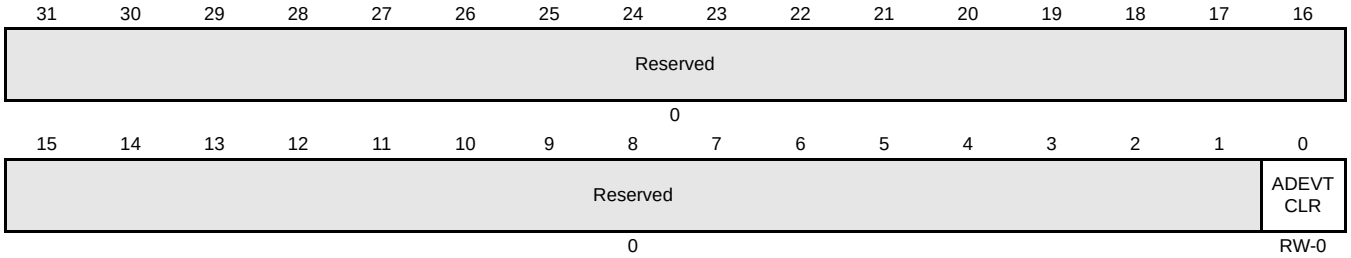
Table 15-51. ADC ADEVT Pin Set Register (ADEVTSET) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT SET		ADEVT Pin Set. A read from this bit always returns the current state of the ADEVT pin. Any operating mode read/write:
		0	Output value on the ADEVT pin is unchanged.
		1	Output logic HIGH on the ADEVT pin, if the pin is configured to be an output pin.

15.10.44ADC ADEVT Pin Clear Register (ADEVTCLR)

Figure 15-54 and Table 15-52 describe the ADEVTCLR register.

Figure 15-54. ADC ADEVT Pin Clear Register (ADEVTCLR) [offset = 10Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

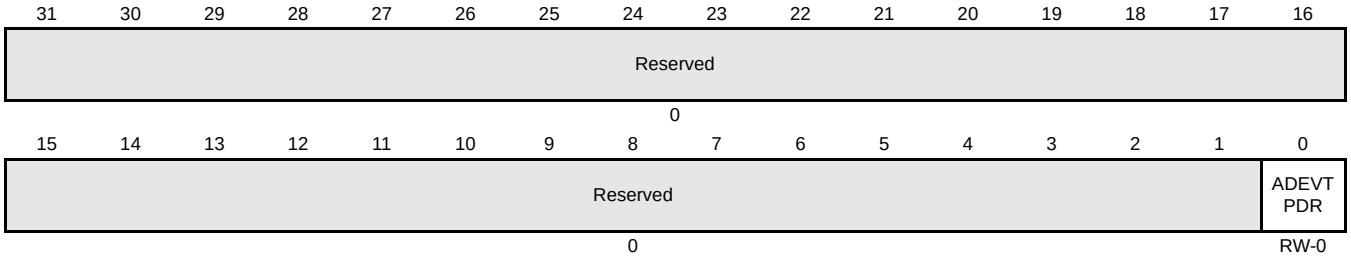
Table 15-52. ADC ADEVT Pin Clear Register (ADEVTCLR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT CLR		ADEVT Pin Clear. A read from this bit always returns the current state of the ADEVT pin. Any operating mode read/write: 0 Output value on the ADEVT pin is unchanged. 1 Output logic LOW on the ADEVT pin, if the pin is configured to be an output pin.

15.10.45ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR)

Figure 15-55 and Table 15-53 describe the ADEVTPDR register.

Figure 15-55. ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR) [offset = 110h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

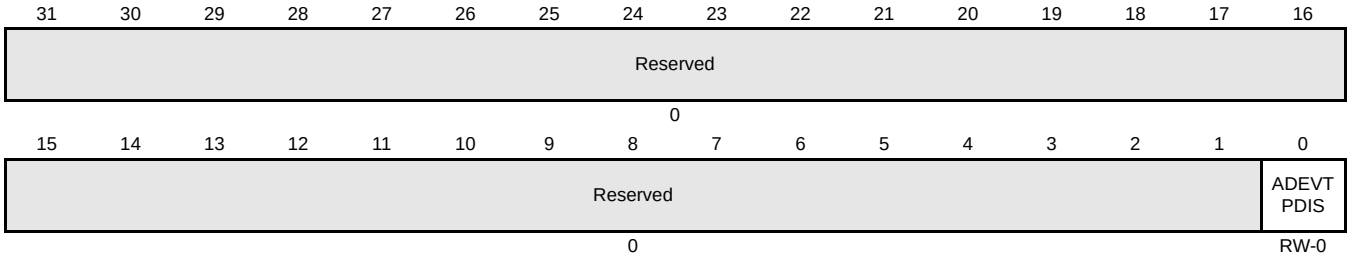
Table 15-53. ADC ADEVT Pin Open Drain Enable Register (ADEVTPDR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT PDR		ADEVT Pin Open Drain Enable. This bit enables the open-drain capability for the ADEVT pin if it is configured to be an output and a logic HIGH is being driven on to the pin. Any operating mode read/write: 0 Output value on the ADEVT pin is logic HIGH. 1 ADEVT pin is tri-stated.

15.10.46ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS)

Figure 15-56 and Table 15-54 describe the ADEVTPDIS register.

Figure 15-56. ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS) [offset = 114h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

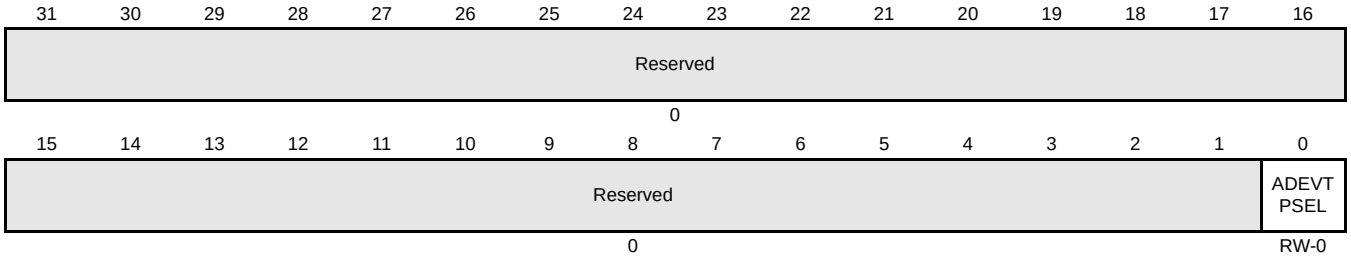
Table 15-54. ADC ADEVT Pin Pull Control Disable Register (ADEVTPDIS) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT PDIS		ADEVT Pin Pull Control Disable. This bit enables or disables the pull control on the ADEVT pin if it is configured to be an input pin. Any operating mode read/write:
		0	Pull control on the ADEVT pin is enabled.
		1	Pull control on the ADEVT pin is disabled.

15.10.47ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL)

Figure 15-57 and Table 15-55 describe the ADEVTPSEL register.

Figure 15-57. ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL) [offset = 118h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

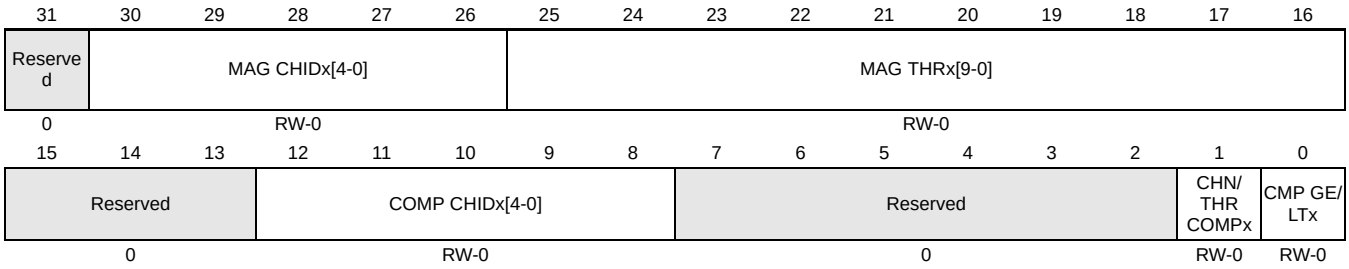
Table 15-55. ADC ADEVT Pin Pull Control Select Register (ADEVTPSEL) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	ADEVT PSEL		ADEVT Pin Pull Control Select. This bit selects the pull-up or pull-down functionality on the ADEVT pin if the pull control is enabled. Any operating mode read/write:
		0	Pull down is selected on the ADEVT pin.
		1	Pull up is selected on the ADEVT pin.
		0	Input buffer for ADEVT is disabled if PULLDIS = 1
		1	Input buffer for ADEVT is enabled if PULLDIS = 1

15.10.48ADC Magnitude Compare Interrupt Control Registers (ADMAGINTxCR)

Figure 15-58 and Table 15-56 describe the ADMAGINTxCR registers. The ADC module supports up to six magnitude compare interrupts. These registers are at offset addresses 128h, 130h, 138h, 140h, 148h, and 150h.

Figure 15-58. ADC Magnitude Compare Interruptx Control Registers (ADMAGINTxCR)



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-56. ADC Magnitude Compare Interruptx Control Register (ADMAGINTxCR) Field Descriptions

Bit	Name	Value	Description
31	Reserved	0	Reads return zeros, writes have no effect.
30–26	MAG CHIDx[4-0]		These bits specify the channel number from 0 to 31 for which the conversion result needs to be monitored by the ADC.
25–16	MAG THRx[9-0]		These bits specify the 10-bit compare value which the ADC will use for the comparison with the MAG CHIDx channel's conversion result.
15–13	Reserved	0	Reads return zeros, writes have no effect.
12–8	COMP CHIDx[4-0]		These bits specify the channel number from 0 to 31 whose last conversion result is compared with the MAG CHIDx channel's conversion result.
7–2	Reserved	0	Reads return zeros, writes have no effect.
1	CHN/THR COMPx		Channel OR Threshold comparison. Any operation mode read/write: 0 The ADC module will compare the MAG CHIDx channel's conversion result with the fixed threshold value specified by the MAG THRx field. 1 The ADC module will compare the MAG CHIDx channel's conversion result with the last conversion result for the COMP CHIDx channel. Both the MAG CHIDx and the COMP CHIDx channel must have been converted at least once for the ADC to perform the comparison.

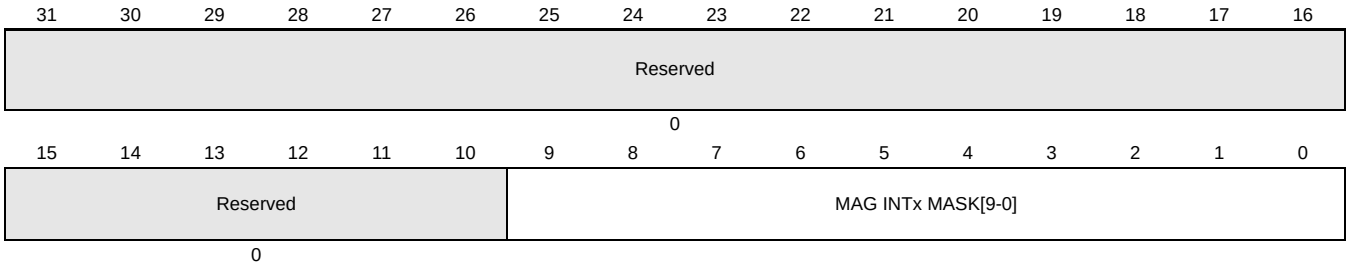
Table 15-56. ADC Magnitude Compare Interruptx Control Register (ADMAGINTxCR) Field Descriptions

Bit	Name	Value	Description
0	CMP GE/LTx		“Greater than or equal to” OR “Less than” comparison operator. Any operation mode read/write:
		0	The ADC module will check if the conversion result is lower than the reference value (fixed threshold or COMP CHIDx conversion result).
		1	The ADC module will check if the conversion result is greater than or equal to the reference value (fixed threshold or COMP CHIDx conversion result).

15.10.49ADC Magnitude Compare Interruptx Mask (ADMAGINTxMASK)

Figure 15-59 and Table 15-57 describe the ADMAGINTxMASK registers. There are six such registers for the six magnitude compare interrupts. These are located at address offsets 12Ch, 134h, 13Ch, 144h, 14Ch, and 154h.

Figure 15-59. ADC Magnitude Compare Interruptx Mask (ADMAGINTxMASK)



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

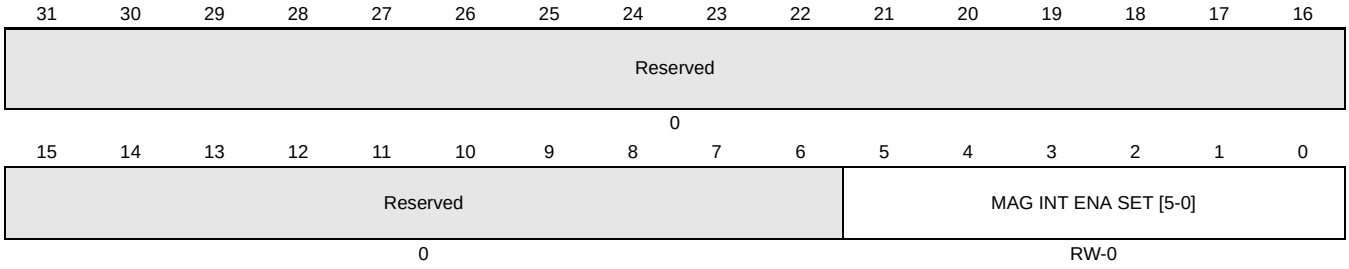
Table 15-57. ADC Magnitude Compare Interruptx Mask (ADMAGINTxMASK) Field Descriptions

Bit	Name	Value	Description
31–10	Reserved	0	Reads return zeros, writes have no effect.
9–0	MAG INTx MASK[9-0]		These bits specify the mask for the comparison in order to generate the magnitude compare interrupt # x. Any operation mode read/write:
		0	The ADC module will not mask the corresponding bit for the comparison.
		1	The ADC module will mask the corresponding bit for the comparison.

15.10.50ADC Magnitude Compare Interrupt Enable Set (ADMAGINTENASET)

Figure 15-60 and Table 15-58 describe the ADMAGINTENASET register.

Figure 15-60. ADC Magnitude Compare Interrupt Enable Set (ADMAGINTENASET) [offset = 158h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

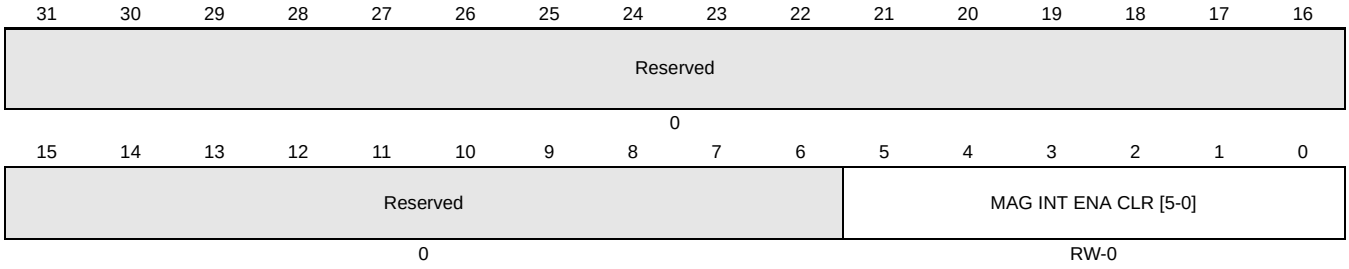
Table 15-58. ADC Magnitude Compare Interrupt Enable Set (ADMAGINTENASET) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved	0	Reads return zeros, writes have no effect.
5–0	MAG INT ENA SET[5-0]		Each of these six bits, when set, enable the corresponding magnitude compare interrupt. Any operation mode read/write for each bit:
		0	The enable status of the corresponding magnitude compare interrupt is left unchanged.
		1	The corresponding magnitude compare interrupt is enabled.

15.10.51ADC Magnitude Compare Interrupt Enable Clear (ADMAGINTENACLR)

Figure 15-61 and Table 15-59 describe the ADMAGINTENACLR register.

Figure 15-61. ADC Magnitude Compare Interrupt Enable Clear (ADMAGINTENACLR) [offset = 15Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

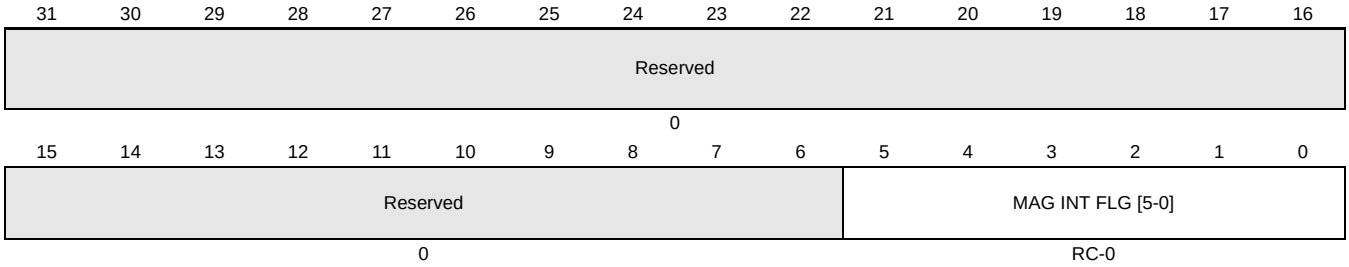
Table 15-59. ADC Magnitude Compare Interrupt Enable Clear (ADMAGINTENACLR) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved	0	Reads return zeros, writes have no effect.
5–0	MAG INT ENA CLR[5-0]		Each of these six bits, when set, disable the corresponding magnitude compare interrupt. Any operation mode read/write for each bit:
		0	The enable status of the corresponding magnitude compare interrupt is left unchanged.
		1	The corresponding magnitude compare interrupt is disabled.

15.10.52ADC Magnitude Compare Interrupt Flag (ADMAGINTFLG)

Figure 15-62 and Table 15-60 describe the ADMAGINTFLG register.

Figure 15-62. ADC Magnitude Compare Interrupt Flag (ADMAGINTFLG) [offset = 160h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

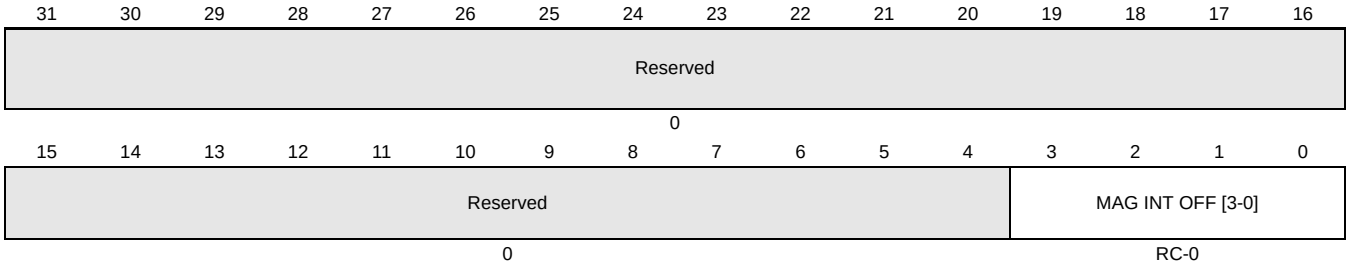
Table 15-60. ADC Magnitude Compare Interrupt Flag (ADMAGINTFLG) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved	0	Reads return zeros, writes have no effect.
5–0	MAG INT FLG [5-0]		Magnitude Compare Interrupt Flags. These bits can be polled by the application to determine if the magnitude compares have been evaluated as true. When a magnitude compare interrupt flag is set, the corresponding magnitude compare interrupt will be generated if enabled. Any operation mode, for each bit:
		0	Read: The corresponding magnitude comparison condition was false. Write: The corresponding flag is left unchanged.
		1	Read: The corresponding magnitude comparison condition was true. Write: The corresponding flag is cleared. The flag can also be cleared by reading from the magnitude compare interrupt offset register.

15.10.53ADC Magnitude Compare Interrupt Offset (ADMAGINTOFF)

Figure 15-63 and Table 15-61 describe the ADMAGINTOFF register.

Figure 15-63. ADC Magnitude Compare Interrupt Offset (ADMAGINTOFF) [offset = 164h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

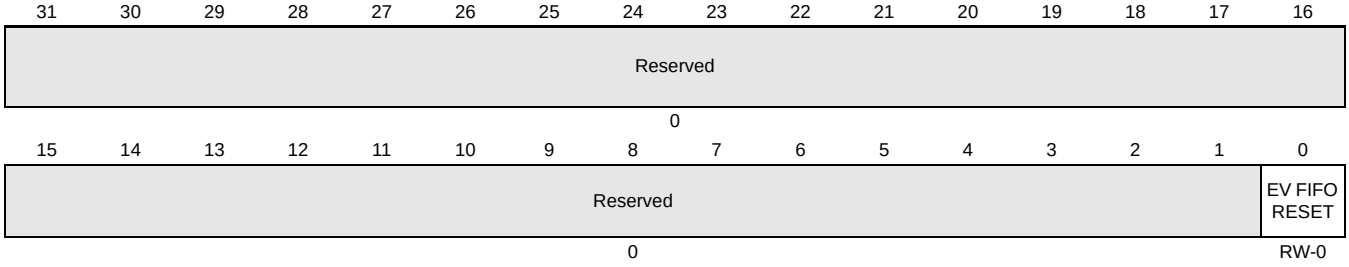
Table 15-61. ADC Magnitude Compare Interrupt Offset (ADMAGINTOFF) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved	0	Reads return zeros, writes have no effect.
3–0	MAG INT OFF [3-0]		Magnitude Compare Interrupt Offset. This field indexes the currently highest-priority magnitude compare interrupt. Interrupt 1 has the highest priority and interrupt 6 has the lowest priority among the magnitude compare interrupts. Writes to these bits have no effect. A read from this register clears this register as well as the corresponding magnitude compare interrupt flag in the ADMAGINTFLG register. However, a read from this register in emulation mode does not affect this register or the interrupt status flags. Any operation mode read:
		0000b	No magnitude compare interrupt is pending.
		0001b	Magnitude compare interrupt # 1 is pending.
		0010b	Magnitude compare interrupt # 2 is pending.
		0011b	Magnitude compare interrupt # 3 is pending.
		0100b	Magnitude compare interrupt # 4 is pending.
		0101b	Magnitude compare interrupt # 5 is pending.
		0110b	Magnitude compare interrupt # 6 is pending.

15.10.54ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR)

Figure 15-64 and Table 15-62 describe the ADEVFIFORESETCR register.

Figure 15-64. ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR) [offset = 168h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

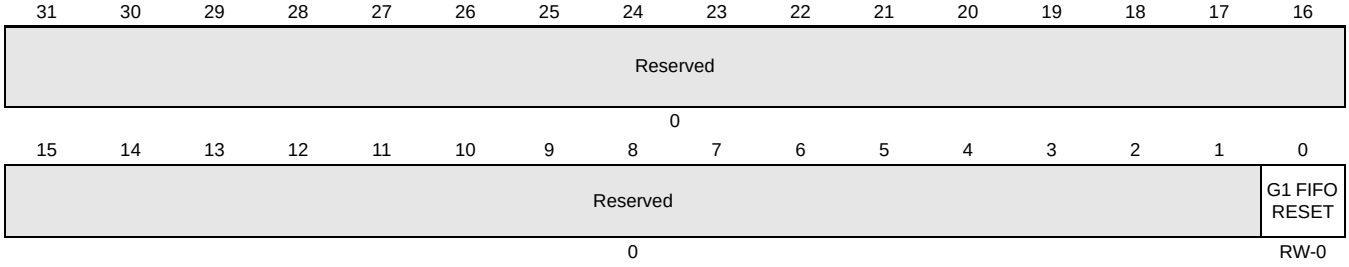
Table 15-62. ADC Event Group FIFO Reset Control Register (ADEVFIFORESETCR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	EV FIFO RESET		ADC Event Group FIFO Reset.The application can set this bit in case of an overrun condition. This allows the ADC module to overwrite the contents of the Event Group results' memory starting from the first location. When this bit is set to '1', the ADC module resets its internal Event Group results' memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Event Group results' memory to be overwritten only once each time this bit is set to '1'. As a result, the EV FIFO RESET bit will always be read as a '0'. The EV FIFO RESET bit will only have the desired effect when the Event Group results' memory is in an overrun condition. It must be used when the data already available in the results' memory can be discarded. If the application needs the Event Group memory to always be overwritten with the latest available conversion results, then the OVR EV RAM IGN bit in the Event Group operating mode control register needs to be set to '1'.

15.10.55ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR)

Figure 15-65 and Table 15-63 describe the ADG1FIFORESETCR register.

Figure 15-65. ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR) [offset = 16Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

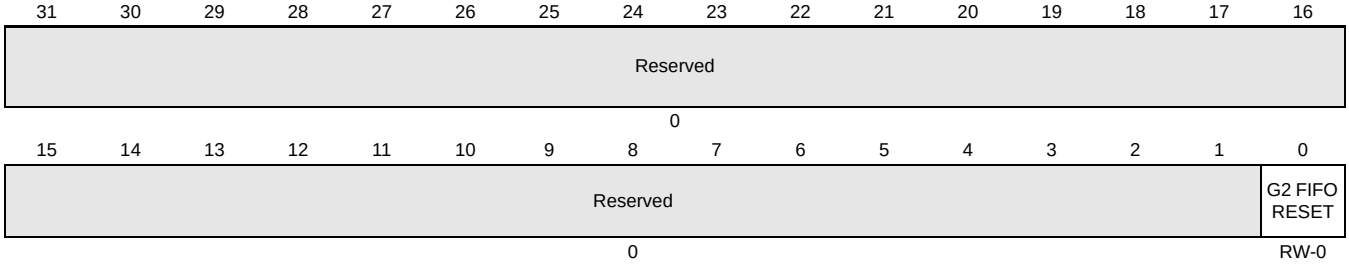
Table 15-63. ADC Group1 FIFO Reset Control Register (ADG1FIFORESETCR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	G1 FIFO RESET		ADC Group1 FIFO Reset.The application can set this bit in case of an over-run condition. This allows the ADC module to overwrite the contents of the Group1 results' memory starting from the first location. When this bit is set to '1', the ADC module resets its internal Group1 results' memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Group1 results' memory to be overwritten only once each time this bit is set to '1'. As a result, the G1 FIFO RESET bit will always be read as a '0'. The G1 FIFO RESET bit will only have the desired effect when the Group1 results' memory is in an overrun condition. It must be used when the data already available in the results' memory can be discarded. If the application needs the Group1 memory to always be overwritten with the latest available conversion results, then the OVR G1 RAM IGN bit in the Group1 operating mode control register needs to be set to '1'.

15.10.56ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR)

Figure 15-66 and Table 15-64 describe the ADG2FIFORESETCR register.

Figure 15-66. ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR) [offset = 170h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

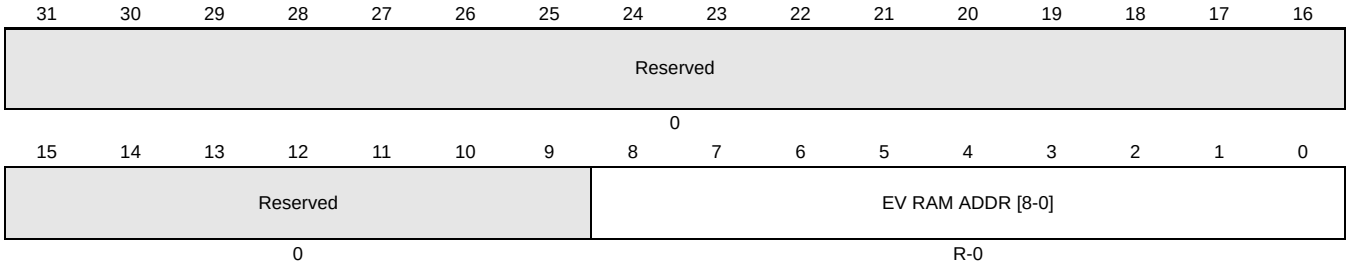
Table 15-64. ADC Group2 FIFO Reset Control Register (ADG2FIFORESETCR) Field Descriptions

Bit	Name	Value	Description
31–1	Reserved	0	Reads return zeros, writes have no effect.
0	G2 FIFO RESET		ADC Group2 FIFO Reset.The application can set this bit in case of an over-run condition. This allows the ADC module to overwrite the contents of the Group2 results' memory starting from the first location. When this bit is set to '1', the ADC module resets its internal Group2 results' memory pointers. Then this bit automatically gets cleared, so that the ADC module allows the Group2 results' memory to be overwritten only once each time this bit is set to '1'. As a result, the G2 FIFO RESET bit will always be read as a '0'. The G2 FIFO RESET bit will only have the desired effect when the Group2 results' memory is in an overrun condition. It must be used when the data already available in the results' memory can be discarded. If the application needs the Group2 memory to always be overwritten with the latest available conversion results, then the OVR G2 RAM IGN bit in the Group2 operating mode control register needs to be set to '1'.

15.10.57ADC Event Group RAM Write Address (ADEVRAMWRADDR)

Figure 15-67 and Table 15-65 describe the ADEVRAMWRADDR register.

Figure 15-67. ADC Event Group RAM Write Address (ADEVRAMWRADDR) [offset = 174h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

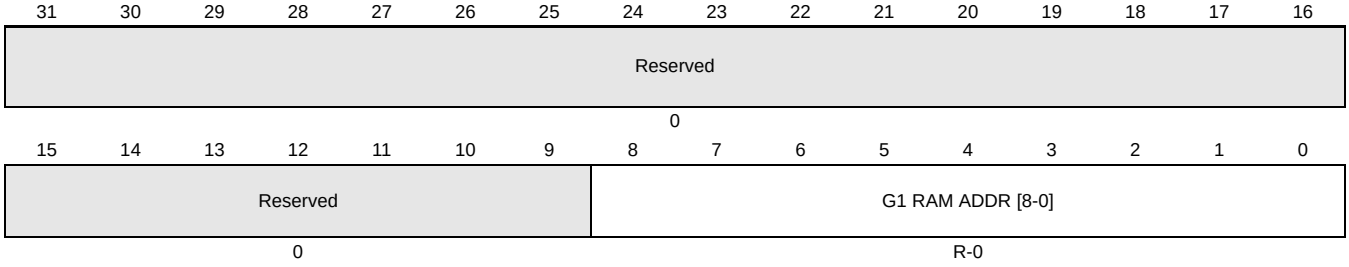
Table 15-65. ADC Event Group RAM Write Address (ADEVRAMWRADDR) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved	0	Reads return zeros, writes have no effect.
8–0	EV RAM ADDR [8-0]		Event Group results' memory write pointer. This field shows the address of the location where the next Event Group conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Event Group conversion results available till that time.

15.10.58ADC Group1 RAM Write Address (ADG1RAMWRADDR)

Figure 15-68 and Table 15-66 describe the ADG1RAMWRADDR register.

Figure 15-68. ADC Group1 RAM Write Address (ADG1RAMWRADDR) [offset = 178h]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

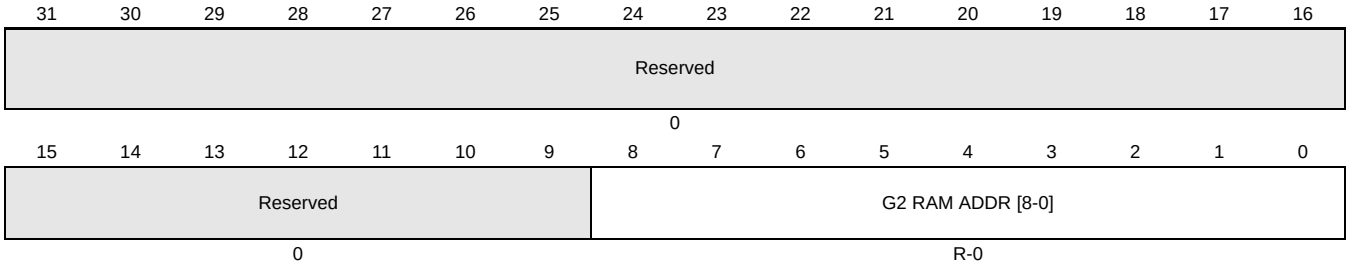
Table 15-66. ADC Group1 RAM Write Address (ADG1RAMWRADDR) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved	0	Reads return zeros, writes have no effect.
8–0	G1 RAM ADDR [8-0]		Group1 results' memory write pointer. This field shows the address of the location where the next Group1 conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Group1 conversion results available till that time.

15.10.59ADC Group2 RAM Write Address (ADG2RAMWRADDR)

Figure 15-69 and Table 15-67 describe the ADG2RAMWRADDR register.

Figure 15-69. ADC Group2 RAM Write Address (ADG2RAMWRADDR) [offset = 17Ch]



R = Read Only, RW = Read/Write, RC = Read/Clear, *-n* = value after reset, RWP = Read in all modes, write in privileged mode only

Table 15-67. ADC Group2 RAM Write Address (ADG2RAMWRADDR) Field Descriptions

Bit	Name	Value	Description
31–9	Reserved	0	Reads return zeros, writes have no effect.
8–0	G2 RAM ADDR [8-0]		Group2 results' memory write pointer. This field shows the address of the location where the next Group2 conversion result will be stored. This is specified in terms of the buffer number. The application can read this register to determine the number of valid Group2 conversion results available till that time.

Serial Communication Interface (SCI)/Buffered Local Interconnect Network (BLIN) Module

This document describes the serial communication interface (SCI)/buffered local interconnect network (BLIN) module. The SCI/BLIN is compliant to the LIN 2.0 protocol specified in the *LIN Specification Package*. This module uses the TMS470Px SCI as its core and augments the SCI's hardware features for LIN compatibility. This module can be configured to operate in either LIN mode or UART mode. This SCI/BLIN module is also compatible with the TMS470Px SCI.

Topic **Page**

16.1 Introduction and Features

This module uses the TMS470Px SCI as its core and augments the SCI's hardware features for LIN compatibility. The SCI/BLIN module is also compatible with the TMS470Px SCI.

16.1.1 Purpose

The SCI/BLIN provides a communications structure at the hardware and software level. It provides a low-cost solution where the bandwidth and fault tolerance of a communications area network (CAN) are not required.

16.1.2 Features

The following are the features of the TMS470Px SCI/BLIN module:

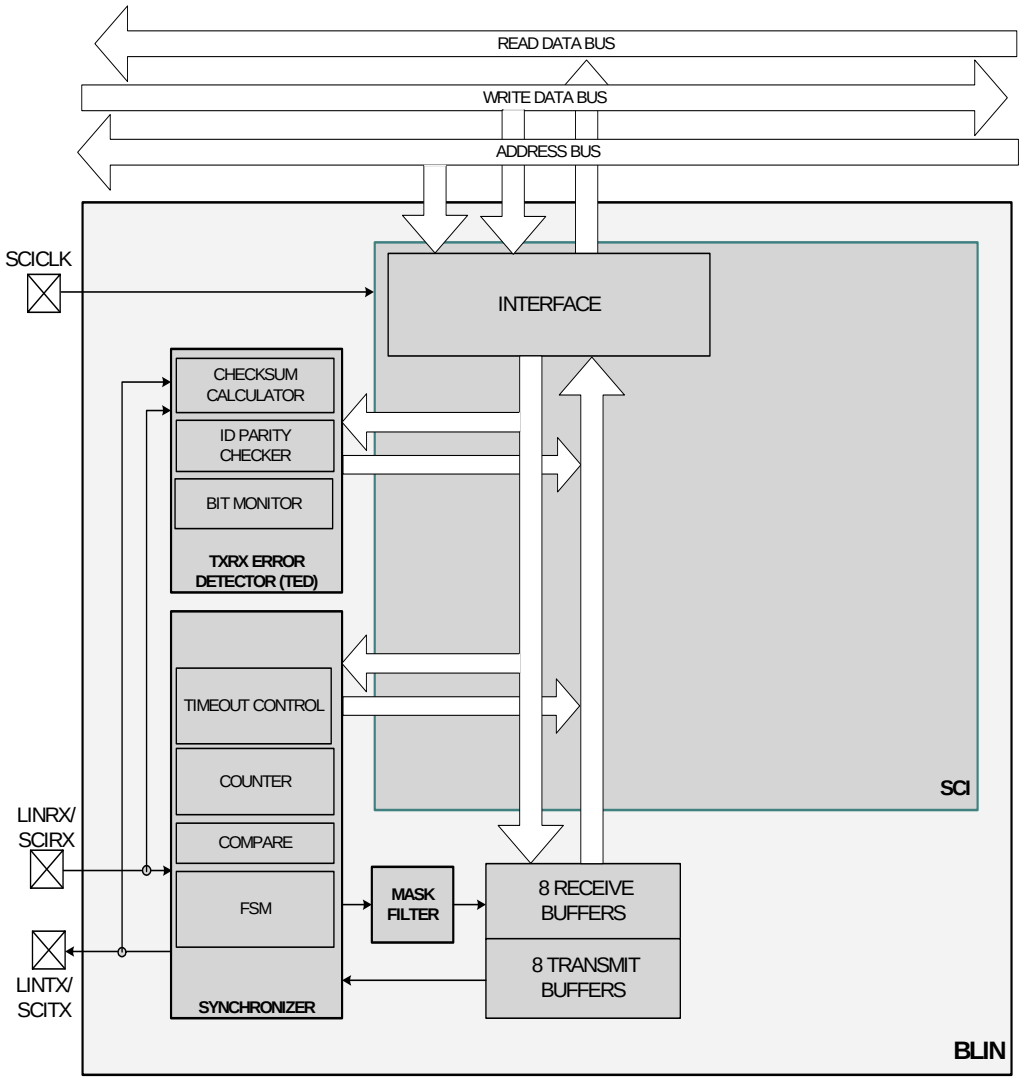
- Compatibility with LIN 1.3 or 2.0 protocols (*LIN Specification Package* 2002)
- Two external pins: LINRX and LINTX (SCICLK in compatibility mode)
- Fully compatible with the TMS470Px SCI
- Multi-buffered receive and transmit units
- Identification masks for message filtering
- Automatic master header generation
  - Programmable synchronization break field
  - Synchronization field
  - Identifier field
- Slave automatic synchronization
  - Synchronization break detection
  - Optional baud rate update
  - Synchronization validation
- 2^{31} programmable transmission rates with 7 fractional bits
- Wakeup on LINRX dominant level from transceiver
- Automatic wakeup support
  - Wakeup signal generation
  - Expiration times on wakeup signals
- Automatic bus idle detection
- Error detection
  - Bit error
  - Bus error
  - No-response error
  - Checksum error
  - Synchronization field error
  - Parity error
- 2 Interrupt lines with priority encoding for:
  - Receive
  - Transmit
  - ID, error, and status
- Support for LIN 2.0 checksum
- Enhanced synchronizer finite state machine (FSM) support for frame processing
- Enhanced handling of extended frames
- Enhanced baud rate generator

- Update wakeup/go to sleep

16.1.3 Block Diagram

The SCI/BLIN module is based on the TMS470Px SCI with the addition of an error detector (parity calculator, checksum calculator, and bit monitor), a mask filter, a synchronizer, and a multibuffered receiver and transmitter. The SCI interface, and the baud generator are modified as part of the hardware enhancements for LIN compatibility. Figure 16-1 shows the SCI/BLIN block diagram.

Figure 16-1. SCI/BLIN Block Diagram



16.1.4 Standards

The LIN standard is based on the SCI (UART) serial data link format. The communication concept is single-master/multiple-slave with a message identification for multi-cast transmission between any network nodes.

For compatibility with LIN2.0 standard the following additional features are implemented over LIN1.3:

i. Support for LIN 2.0 checksum
ii. Enhanced synchronizer FSM support for frame processing
iii. Enhanced handling of extended frames

iv. Enhanced baudrate generator

v. Update wakeup/go to sleep

The SCI/BLIN module covers the CPU performance-consuming features, defined in the *LIN Specification Package* Revision 1.3 and 2.0 by hardware.

16.2 Operation

The SCI/BLIN module can be used in LIN mode or UART mode. The enhancements for baud generation, and additional receive/transmit buffers necessary for LIN mode operation are part of the enhanced buffered SCI/BLIN module.

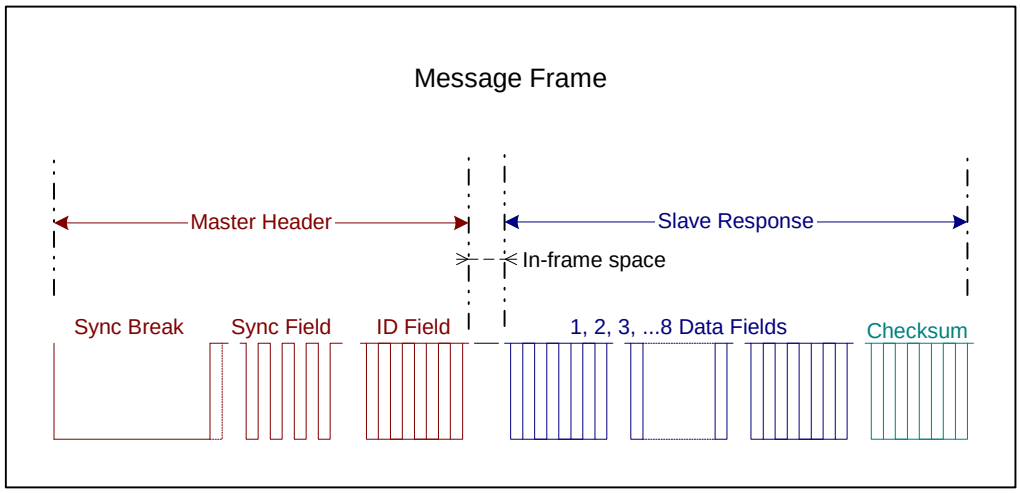
Note:
The SCI/BLIN is built around the SCI platform and uses a similar sampling scheme: 16 samples for each bit with majority vote on samples 8, 9, and 10.

The SCI/BLIN control registers are located at the TMS470Px SCI base address. For a detailed description of each register, see Section 16.6.

16.2.1 Message Frame

The LIN protocol defines a message frame format, illustrated in Figure 16-2. Each frame includes one master header, one response, one in-frame response space, and inter-byte spaces. In-frame-response and inter-byte spaces may be 0.

Figure 16-2. LIN Protocol Message Frame Format: Master Header and Slave Response



There is no arbitration in the definition of the LIN protocol; therefore, multiple slave nodes responding to a header might be detected as an error.

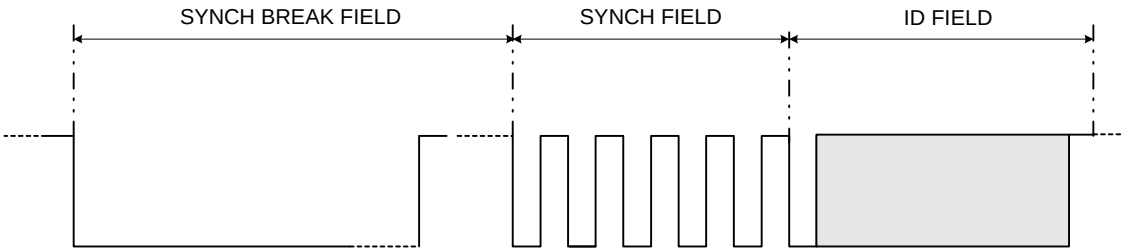
The LIN bus is a single channel wired-AND. The bus has a binary level: either dominant for a value of 0, or recessive for a value of 1.

16.2.1.1 Message Header

The header of a message is initiated by a master (see Figure 16-3) and consists of a three field-sequence:

- The synch break field signaling the beginning of a message
- The synch field conveying bit rate information of the LIN bus
- The ID field denoting the content of a message

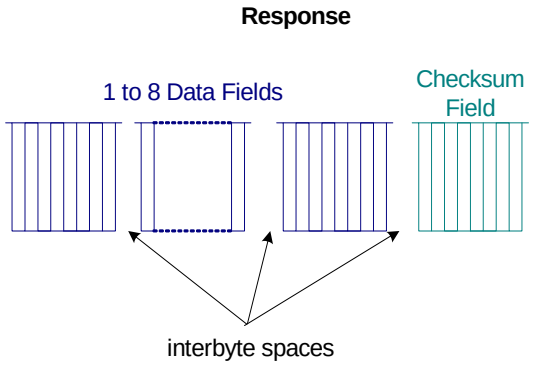
Figure 16-3. Header 3 fields: Synch Break, Synch, and ID



16.2.1.2 Response

The format of the response is as illustrated in Figure 16-4. There are two types of fields in a response: data and checksum. The data field consists of exactly one data byte, one start bit, and one stop bit, for a total of 10 bits. The LSB is transmitted first. The checksum field consists of one checksum byte, one start bit and one stop bit. The checksum byte is the inverted modulo-256 sum over all data bytes in the data fields of the response.

Figure 16-4. Response Format of LIN Message Frame



The format of the response is a stream of N data fields and one checksum field. Typically N is from 1 to 8, with the exception of the extended command frames. The length N of the response is indicated with the optional length control bits of the identifier, or by the SCIFORMAT(18–16) register (Section 16.6.11); see Table 16-1. The SCI/BLIN module supports response lengths from 1 to 8 bytes in compliance with LIN 2.0.

Table 16-1. Response Length with SCIFORMAT(18–16) programming

SCIFORMAT(18–16)	No. of Bytes
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

16.2.2 Synchronizer

The synchronizer has three major functions in the messaging between master and slave nodes. It generates the master header data stream, it synchronizes to the LIN bus for responding, and it locally detects timeouts.

A bit rate is programmed using the prescalers in the BRS register as indicated by the LIN_speed value in the LIN description file.

The SCI/BLIN synchronizer will perform the following functions: master header signal generation, slave detection and synchronization to message header with optional baud rate adjusting, response transmission timing and timeout control.

The SCI/BLIN synchronizer should be capable of detecting an incoming break and initialize communication at all times.

16.2.3 Baud Rate

The transmission baud rate of any node is configured by the CPU in the beginning; this defines the bit time T_{bit}. In SCI mode, the bit time is derived from the fields P and M in the baud rate selection register (BRS; Section 16.6.12). An additional 3-bit fractional divider value, field U in the BRS, further fine-tunes the data field baud rate.

The ranges for the prescaler values in the BRS are:

P = 0, 1, 2, 3, . . . , 2^{24} - 1

M = 0, 1, 2, . . . , 15

U = 0, 1, 2, 3, 4, 5, 6, 7

The BRS with P, M, and U values are user programmable. The U value is an additional 3-bit value determining that a T_{VCLK} (with a = 0,1) is added to each T_{bit} as explained in Section 16.2.3.2. All these divider values are automatically obtained in LIN mode during header reception when the synchronization field is measured if the ADAPT bit is set. Otherwise, the P and M dividers could be used for both SCI mode and LIN mode to select a baud rate.

The LIN protocol defines baud rate boundaries as follows:

$$1\text{kHz} \le F_{\text{LINCLK}} \le 20\text{kHz}$$

All transmitted bits are shifted in and out at T_{bit} periods.

16.2.3.1 Fractional Divider

The M field of the BRS register (Section 16.6.12) modifies the integer prescaler P for finer tuning of the baud rate. The M value adds in increments of 1/16 of the P value.

The bit time, T_{bit} is expressed in terms of the VCLK period T_{VCLK} as follows:

$$\text{For all P other than 0, and all M, } T_{bit} = 16\left(P + 1 + \frac{M}{16}\right)T_{VCLK}$$

$$\text{For P= 0 : } T_{bit} = 32T_{VCLK}$$

Therefore, the LINCLK frequency is given by

$$F_{LINCLK} = \frac{F_{VCLK}}{16\left(P + 1 + \frac{M}{16}\right)} \quad \text{For all P other than zero}$$

$$F_{LINCLK} = \frac{F_{VCLK}}{32} \quad \text{For P = 0}$$

16.2.3.2 Superfractional Divider

The superfractional divider scheme implemented by BRS(30–28), applies only in LIN mode. Building on the 4-bit fractional divider implemented by BRS(27–24), the superfractional divider uses an additional 3-bit modulating value shown in Table 16-2. The bits with a 1 in the table will have an additional VLCK period added to their T_{bit}.

Table 16-2. Superfractional Bit Modulation, LIN Master Mode and Slave Mode

BRS(30–28)	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Stop Bit
0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	1	0
2	1	0	0	0	1	0	0	0	1	0
3	1	0	1	0	1	0	0	0	1	0
4	1	0	1	0	1	0	1	0	1	0
5	1	1	1	0	1	0	1	0	1	1
6	1	1	1	0	1	1	1	0	1	1
7	1	1	1	1	1	1	1	0	1	1

The baud rate will vary over a LIN data field to average according to the BRS(30–28) value by a *d* fraction of the peripheral internal clock: 0<d<1.

The instantaneous bit time is expressed in terms of T_{VCLK} *as follows:*

$$\text{For all P other than 0, and all M and d, } T^{i}_{bit} = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK}$$

$$\text{For P= 0 } T_{bit} = 32T_{VCLK}$$

The averaged bit time is expressed in terms of T_{VCLK} as follows*:*

$$\text{For all P other than 0, and all M and d, } T^{a}_{bit} = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK}$$

$$\text{For P= 0 } T_{bit} = 32T_{VCLK}$$

With the superfractional divider, a LIN baud rate of 20 kbps is achievable with an internal clock VCLK of 726 kHz. Furthermore, a rate of 400 kbps is achievable with an VCLK of 14.6 MHz.

16.2.4 Header Generation

Automatic generation of the LIN protocol header data stream is supported without CPU interaction. The CPU will trigger a message header generation and the SCI/BLIN state machine will handle the generation itself. The header is always sent by the master to initiate a LIN communication and consists of three fields: synchronization break field, synchronization field, and identification field, as seen in Figure 16-5.

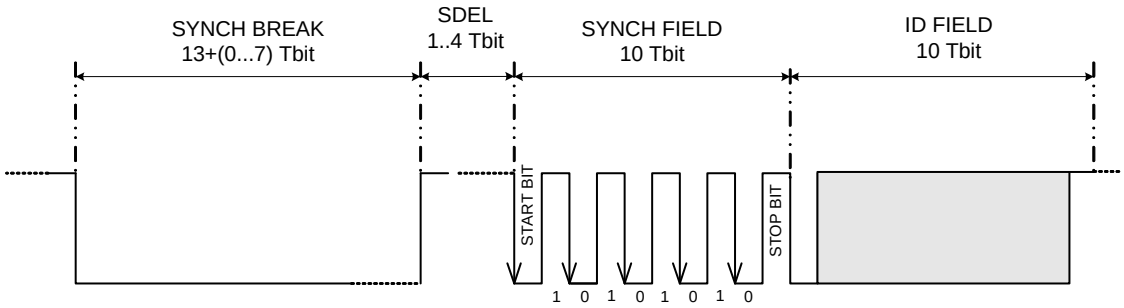
- The synchronization break field consists of two components.
  - The synchronization break (SYNCH BREAK) consists of a minimum of 13 (dominant) low bits to a maximum of 20 dominant bits. The synch break length may be extended from the minimum with the 3-bit SBREAK value in the LINCOMP register.
  - The synchronization break delimiter (SDEL) consists of a minimum of 1 (recessive) high bit to a maximum of 4 recessive bits. The delimiter marks the end of the synchronization break field. With a

minimum of 13 dominant T_{bit} followed by a minimum of 1 recessive T_{bit}. The synch break delimiter length depends on the 2-bit SDEL value in the LINCOMP register.

- The synchronization field (SYNCH FIELD) consists of one start bit, byte 0x55, and a stop bit. It is used to convey T_{bit} information and resynchronize LIN bus nodes.
- The identifier field's ID byte may use six bits as an identifier (optional length control within it), and two optional bits as parity of the identifier. The identifier parity is used and checked if the (SCIGCR1.2) PARITY enable bit is set. If length control bits are not used, then there can be a total of 64 identifiers plus parity. If neither length control or parity are used there can be up to 256 identifiers. See Figure 16-6 for an illustration of the ID field.

Note:
The LIN protocol uses the parity bits in the identifier. The control length bits are optional to the LIN protocol.

Figure 16-5. Message Header in Terms of T_{bit}



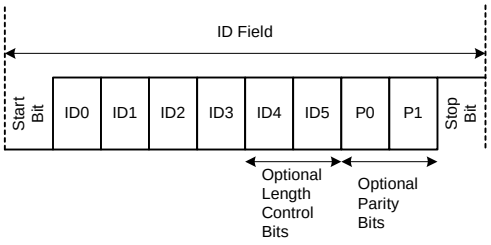
A master node initiates header generation on CPU writes to the IDBYTE in the LINID register (Section 16.6.26).

Note:
IDBYTE field (LINID[5:4]); Section 16.6.26] conveys response length information if compliant to standards earlier than LIN 1.3.

ID5	ID4	Number of Data bytes
0	0	2
0	1	2
1	0	4
1	1	8

The SCIFORMAT register (Section 16.6.11) stores the length of the response for later versions of the LIN protocol.

Figure 16-6. ID Field



16.2.4.1 Event Triggered Frame Handling Proposal

The LIN 2.0 protocol uses event-triggered frames that may occasionally cause collisions. Event-triggered frames have to be handled in software.

If no slave answers to an event triggered frame header, the master node will set the NRE flag, and a NRE interrupt will occur if enabled. If a collision occurs, a frame error and checksum error may arise before the NRE error. Those errors are flagged and the appropriate interrupts will occur, if enabled.

Frame errors and checksum errors depend on the behavior and synchronization of the responding slaves. If the slaves are totally synchronized and stop transmission once the collision occurred, it is possible that only the NRE error is flagged despite the occurrence of a collision. To detect if there has been a reception of one byte before the NRE error is flagged, the bus busy flag (SCIFLR.3) can be used as an indicator.

The bus busy flag is set on the reception of the first bit of the header and remains set until the header reception is complete, and again is set on the reception of the first bit of the response. In the case of a collision the flag is cleared in the same cycle as the NRE flag is set.

Software could implement the following sequence:

- Once the reception of the header is done (poll for RXID flag), wait for the bus busy flag to get set or NRE flag to get set.
- If bus busy flag is not set before NRE flag, then it is a true no response case (no data has been transmitted onto the bus).
- If bus busy flag gets set, then wait for NRE flag to get set or for successful reception. If NRE flag is set, then in this case a collision has occurred on the bus.

Even in the case of a collision, the received (corrupted) data is accessible in the RX buffers; registers LINRD0 and LINRD1.

16.2.4.2 Header Reception and Adaptive Baud Rate

A slave node baud rate might be adjusted to the detected bit rate as an option to the SCI/BLIN module. The adaptive baud rate option is enabled by setting the ADAPT bit in SCIGCR1 (Section 16.6.2). During header reception, a slave measures the baud rate during detection of the synch field. If SCIGCR1[9] (the ADAPT bit) is set, then the measured baud rate is compared to the slave node's programmed baud rate and adjusted to the LIN bus baud rate if necessary.

The SCI/BLIN synchronizer determines two measurements: BRK_count and BAUD_count (Figure 16-7). These values are always calculated during the Header reception for synch field validation (Figure 16-8).

Note:
When an inconsistent synch field (ISF) error occurs, suggested action for the application is
a) Reset the SW_nRESET bit.
b) Set the SW_nRESET bit to make sure that the internal FSMs are back to their normal states

Figure 16-7. Measurements for Synchronization

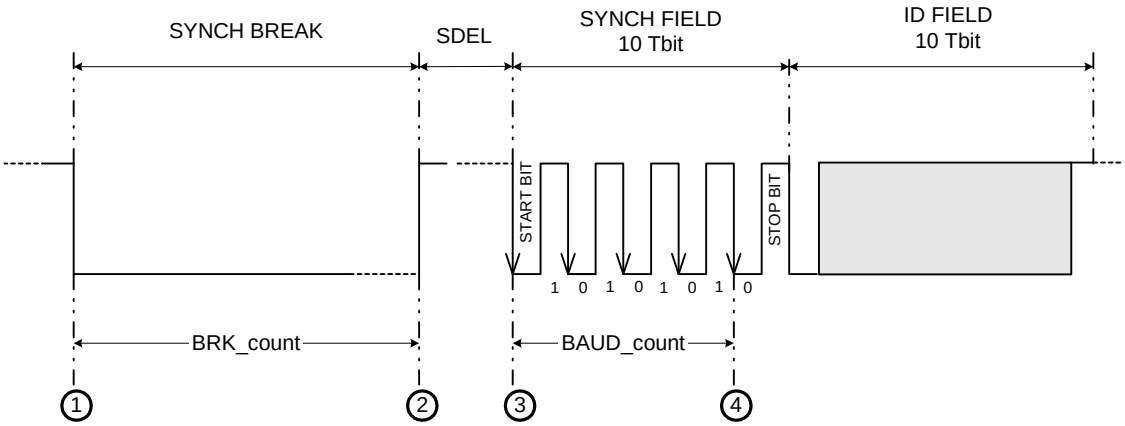
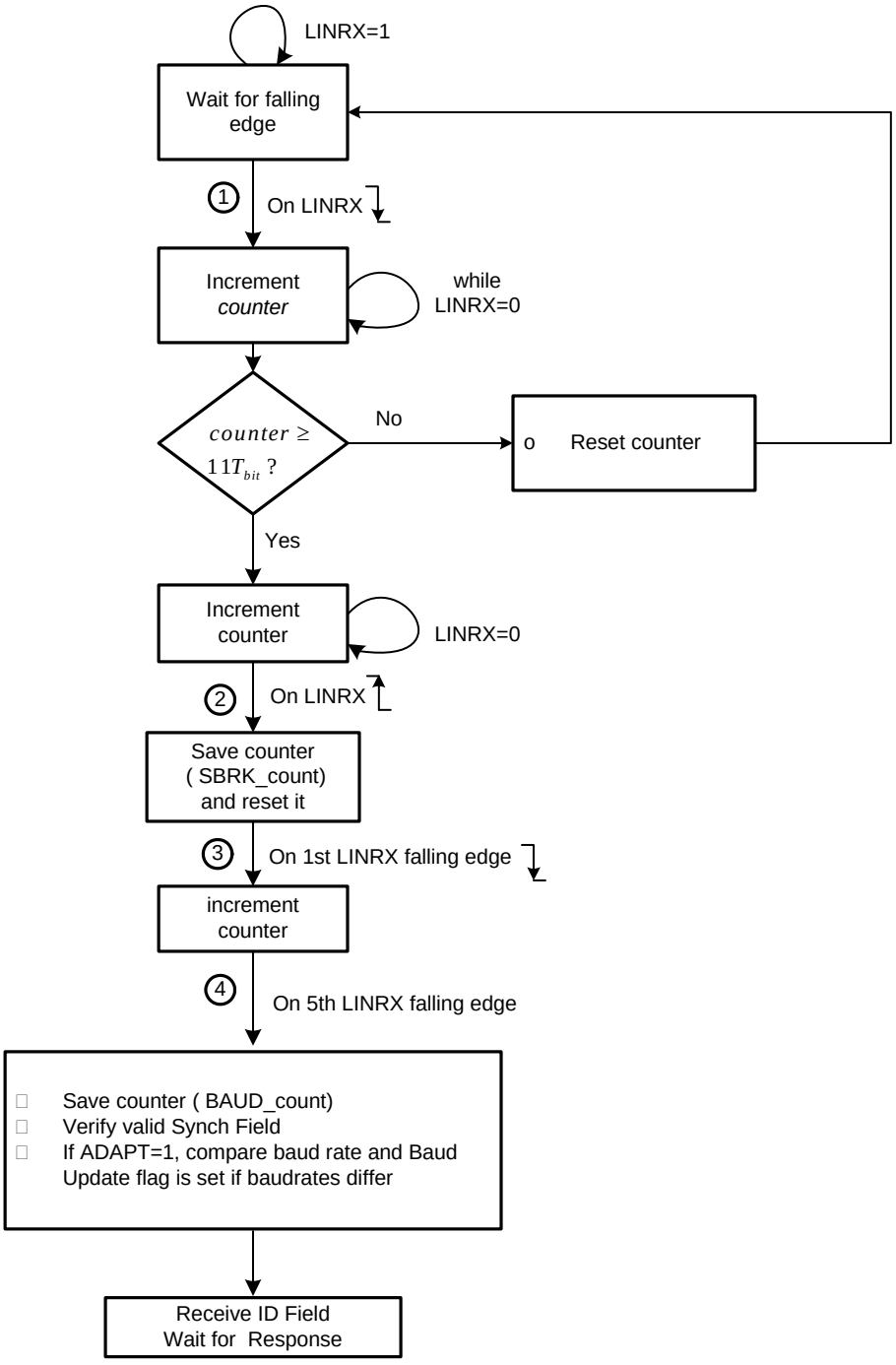


Figure 16-8. Synchronization Validation Process and Baud Rate Adjustment



By measuring the values BRK_count and BAUD_count, a valid synch break sequence can be detected as described in Figure 16-8. The four numbered events in Figure 16-7 signal the start/stop of the synchronizer counter. The synchronizer counter uses VCLK as the time base.

The synchronizer counter is used to measure the synch break relative to the detecting node T_{bit}. For a slave node receiving the synch break, a threshold of 11 T_{bit} is used as required by the LIN protocol. For detection

of the dominant data stream of the synch break, the synchronizer counter is started on a falling edge and stopped on a rising edge of the LINRX. On detection of the synch break delimiter, the synchronizer counter value is saved and then reset.

On detection of five consecutive falling edges, the BAUD_count is measured. Bit timing calculation and consistency to required accuracy is implemented following the recommendations of LIN revision 2.0. A slave node can calculate a single T_{bit} time by division of BAUD_count by 8. In addition, for consistency between the detected edges the following is evaluated:

$$BAUD\_count + BAUD\_count \gg 2 + BAUD\_count \gg 3 \leq BRK\_count$$

The BAUD_count value is shifted 3 times to the right and rounded using the first insignificant bit to obtain a T_{bit} unit. If the ADAPT bit is set, then the detected baud rate is compared to the programmed baud rate.

During the header reception processing as illustrated in Figure 16-8, if the measured BRK_count value is less than 11 T_{bit}, the synch break is not valid according to the protocol for a fixed rate. If the ADAPT bit is set, then the MBRS (Section 16.6.29) is used for measuring BRK_count and BAUD_count values and automatically adjusts to any allowed LIN bus rate (refer to *LIN Specification Package 2.0*).

Note:
In adaptive mode the MBRS divider should be set to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise a 0x00 data byte could mistakingly be detected as a synch break.

Note:
The synch-break-threshold relative to the slave node is 11 T_{bit}. The synch break is 13 T_{bit} as specified in LIN version 1.3.

If the synch field is not detected within the given tolerances, the inconsistent-synch-field-error (ISFE) flag will be set. An ISFE interrupt will be generated, if enabled by its respective bit in the SCISETINT register (Section 16.6.6). The ID byte should be received after the synch field validation was successful. Any time a valid synch break (larger than 11 T_{bit}) is detected, the receiver's state machine should reset to reception of this new frame. This reset condition is only valid during response state, not if an additional synch break occurs during header reception.

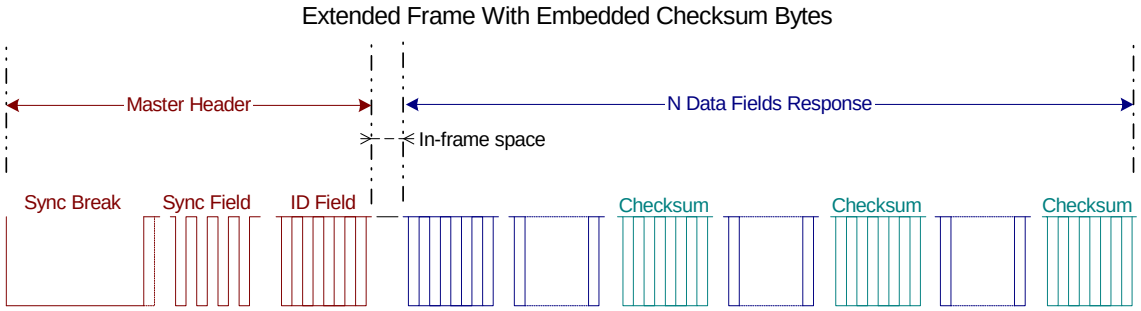
16.2.5 Extended Frames Handling

The LIN protocol includes two extended frames with identifiers 62 (user defined) and 63 (reserved extended). The length for these identifiers will be set at network configuration time to be shared with the LIN bus nodes.

Note:
Only identifier 62 (0x3E) applies to this special extended frame handling of unlimited response length.

Extended frame communication is triggered on reception of a header with identifier 0x3E; see Figure 16-9. Once the extended frame communication is triggered, unlike normal frames, this communication needs to be stopped before issuing another header. To stop the current extended frame communication, the LINMASK register (TX Mask and RX Mask; Section 16.6.25) must be programmed so that there will not be any TX or RX match. The current frame transmission/reception is completed before stopping the extended frames communication.

Figure 16-9. Optional Embedded Checksum in Response for Extended Frames



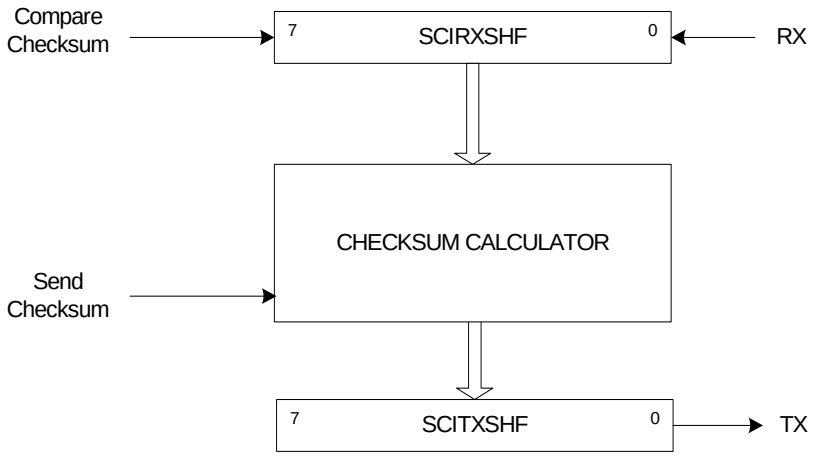
For the LIN 2.0 specification, the length of the user-defined frame is unlimited. Therefore, an ID interrupt will be generated (if enabled and there is a match) on reception of this identifier. This interrupt allows the CPU using a software counter to keep track of the bytes that are being sent out and decides when to calculate and insert a checksum byte (recommended at periodic rates). To handle this procedure, SCIGCR2[16] (Section 16.6.3) is used. A write to the SCIGCR2[16] bit will initiate an automatic send of the checkbyte. The last data field should be a checksum in compliance with the LIN protocol.

The periodicity of the checksum insertion, defined at network configuration time, is used by the receiving node to evaluate the checksum of the ongoing message, and has the benefit of enhanced reliability.

For the sending node, the checksum is automatically embedded when the SCIGCR2[16] bit is set. For the receiving node, the checksum is automatically compared when the SCIGCR2[17] bit is set; see Figure 16-10.

Note:
The LIN 2.0 enhanced checksum does not apply to the reserved identifiers. They will always use the classic checksum.

Figure 16-10. Checksum Compare and Send for Extended Frames



16.2.6 Timeout Control

Any SCI/BLIN node listening to the bus and expecting a response initiated from a master node could flag a no-response error timeout event. The LIN protocol defines four types of timeout events, which are all handled by the hardware of the SCI/BLIN module. The four LIN protocol events are:

- No-response timeout error
- Bus idle detection

- Timeout after wakeup signal
- Timeout after three wakeup signals

16.2.6.1 No-Response Error (NRE)

The no-response error will occur when any node expecting a response waits for T_{FRAME_MAX} time and the message frame is not fully completed within the maximum length allowed, T_{FRAME_MAX}. After this time a no-response error (NRE) is flagged in the NRE bit of the SCIFLR register. An interrupt is triggered if enabled.

As specified in the LIN 1.3 standard, the minimum time to transmit a frame is:

$T_{FRAME\_MIN} = T_{HEADER\_MIN} + T_{DATA\_FIELD} + T_{CHECKSUM\_FIELD}$

$= 44 + 10N$

where N = number of data fields.

And the maximum time frame is given by:

$T_{FRAME\_MAX} = T_{FRAME\_MIN} * 1.4$

$= (44 + 10N) * 1.4$

The timeout value T_{FRAME_MAX} is derived from the *N* number of data fields value. The *N* value is either embedded in the header's ID field for messages or is part of the description file. In the latter case, the 3-bit CHAR value in SCIFORMAT[18:16] in Section 16.6.11, will indicate the value for *N*.

Table 16-3. Timeout Values in T_{bit} Units

N	T_{DATA_FIELD}	T_{FRAME_MIN}	T_{FRAME_MAX}
1	10	54	76
2	20	64	90
3	30	74	104
4	40	84	118
5	50	94	132
6	60	104	146
7	70	114	160
8	80	124	174

Note:
The length coding of the ID field does not apply to two extended frame identifiers, ID fields of 0x3E (62) and 0x3F (63). In these cases, the ID field can be followed by an arbitrary number of data byte fields. **The LIN 2.0 protocol specification mentions that ID field 0x3F (63) cannot be used. For these two cases, the NRE will not be handled by the SCI/BLIN controller hardware.**

16.2.6.2 Bus Idle Detection

The second type of timeout can occur when a node detects an inactive LIN bus: no transitions between recessive and dominant values are detected on the bus. This happens after a minimum of 4 s (this is 80,000 F_{LINCLK} cycles with the fastest bus rate of 20 kbps). If a node detects no activity in the bus as the TIMEOUT bit is set (Section 16.6.8), then it can be assumed that the LIN bus is in sleep mode.

16.2.6.3 Timeout after Wakeup Signal and Timeout after Three Wakeup Signals

The third and fourth types of timeout are related to the wakeup signal. A node initiating a wakeup should expect a header from the master within a defined amount of time: timeout after wakeup signal. See Section 16.4.3 for more details.

16.2.7 TXRX Error Detector (TED)

The following sources of error are detected by the TXRX error detector logic (TED). The TED logic consists of a bit monitor, an ID parity checker, and a checksum error. The following errors are detected:

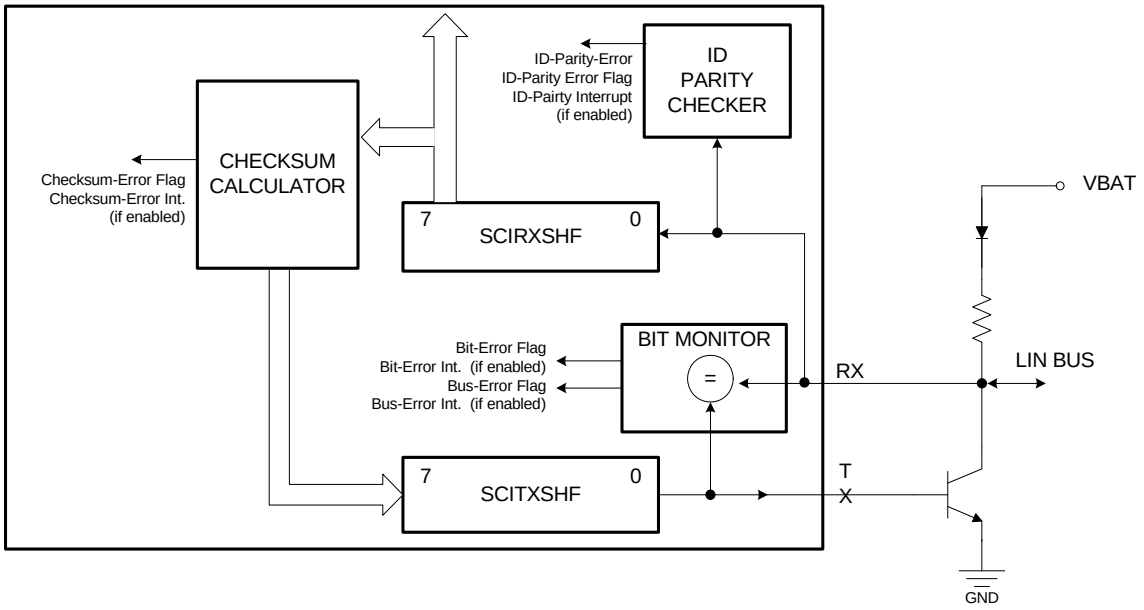
- Bit errors (BE)
- Physical bus errors (PBE)
- Identifier parity errors (PE)
- Checksum errors (CE)

All of these errors (BE, PBE, PE, CE) are flagged. An interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver if there is no error detected until the end of the frame.

16.2.7.1 Bit Errors

A bit error (BE) is detected at the bit time when the bit value that is monitored is different from the bit value that is sent. A bit error is indicated by the BE flag in SCIFLR (Section 16.6.8). After signaling a BE, the transmission has to be aborted no later than the next byte. The bit monitor ensures that the transmitted bit in LINTX is the correct value on the LIN bus by reading back on the LINRX pin as shown in Figure 16-11.

Figure 16-11. TXRX Error Detector



16.2.7.2 Physical Bus Errors

A Physical Bus Error (PBE) is detected by a master if no valid message can be generated on the bus (i.e., the bus is shorted to GND or V_{BAT}). The bit monitor can detect a physical bus error during header generation as it compares the actual value on the bus with the expected value at each bit time.

16.2.7.3 ID Parity Errors

If parity is enabled, an ID parity error (PE) is detected if any of the two parity bits of the sent ID byte are not equal to the calculated parity on the receiver node. The two parity bits are generated using the following mixed parity algorithm.

$$P0 = \overline{ID0 \oplus ID1 \oplus ID2 \oplus ID4} \text{ (even parity)}$$
$$P1 = \overline{ID1 \oplus ID3 \oplus ID4 \oplus ID5} \text{ (odd parity)}$$

If an ID-parity error is detected, the ID-parity error is flagged, and the received ID is not valid. See Section 16.2.8 for details.

16.2.7.4 Checksum Errors

A checksum error (CE) is detected and flagged at the receiving end if the calculated modulo-256 sum over all received data bytes (including the ID byte if it is the enhanced checksum type) plus the checkbyte does not result in 0xFF. The modulo-256 sum is calculated over each byte by adding with carry, where the carry bit of each addition is added to the LSB of its resulting sum.

For the transmitting node, the checkbyte sent at the end of a message is the inverted sum of all the data bytes (see Figure 16-12) for classic implementation. The checkbyte is the inverted sum of the identifier byte and all the data bytes (see Figure 16-13) for the LIN 2.0 compliant implementation. The classic implementation should always be used for reserved identifiers 60 to 63; therefore, the CTYPE bit (SCIGCR1[11]; Section 16.6.2) will be overridden in this case. For signal-carrying-frame identifiers (0 to 59) the type of checksum used depends on the CTYPE bit (SCIGCR1[11]).

Figure 16-12. Classic Checkbyte Generation at Transmitting Node

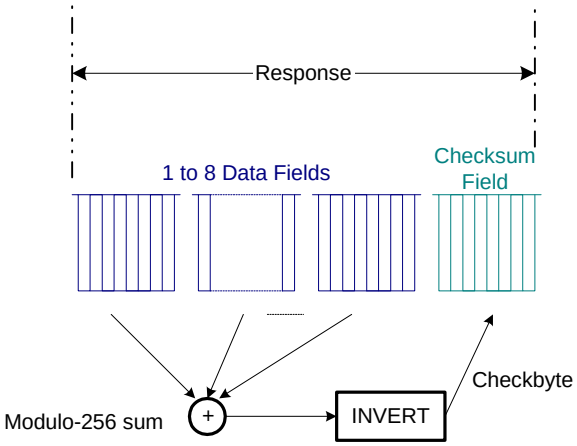
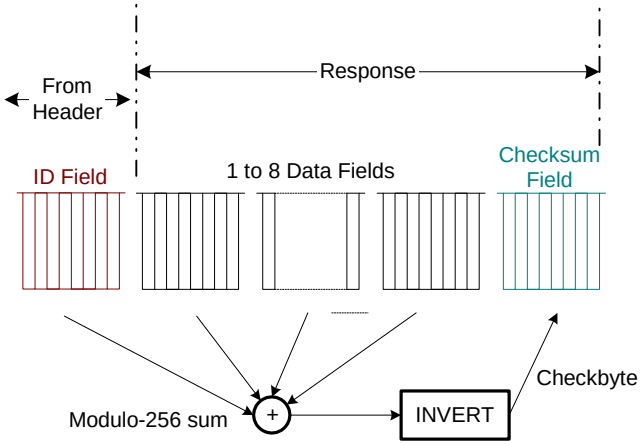


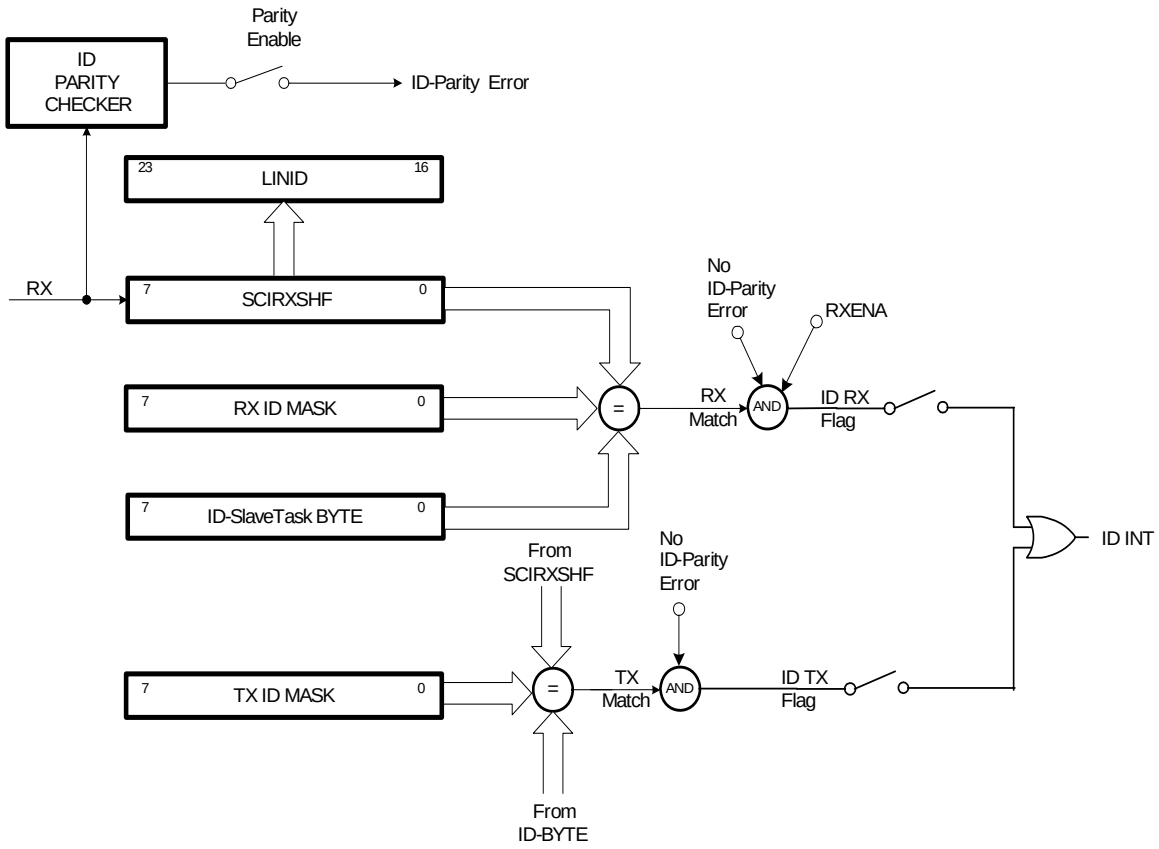
Figure 16-13. LIN 2.0-Compliant Checkbyte Generation at Transmitting Node



16.2.8 Message Filtering and Validation

Message filtering uses the entire identifier to determine which nodes will participate in a response, either receiving or transmitting a response. Therefore, two acceptance masks are used as shown in Figure 16-14.

Figure 16-14. ID Reception, Filtering and Validation



During header reception, all nodes filter the received ID-Field and decide whether they transmit or receive a response for the current message. There are two masks: one to accept a response reception; the other to initiate a response transmission. All nodes compare the received ID to the identifier stored in the ID-SlaveTask BYTE of the LINID register (Section 16.6.26) and use the RX ID MASK and the TX ID MASK fields in the LINMASK register (Section 16.6.25) to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA (SCIGCR1[24]; Section 16.6.2) bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. If there is a TX match with no parity error and the TXENA (SCIGCR1[25]; Section 16.6.2) bit is set, there will be an ID TX flag and an interrupt will be triggered if enabled in the SCISETINT register.

The masked bits become don't cares for the comparison. To build a mask for a set of identifiers, an xOR function could be used.

For example, to build a mask to accept IDs 0x26 and 0x25 using LINID[7:0] = 0x20; i.e., compare five most significant bits (MSBs) and filter three least significant bits (LSBs), the acceptance mask could be:

$$(0x26 + 0x25) \oplus 0x20 = 0x07$$

A mask of all zeros will compare all bits of the received identifier in the shift register with the ID-BYTE in LINID[7:0]. If HGEN CTRL in SCIGCR1 (Section 16.6.2) is set to 1, a mask of 0xFF will always cause a match. A mask of all 1s will filter all bits of the received identifier, and thus there will be an ID match regardless of the content of the ID-SlaveTask BYTE field in the LINID register (Section 16.6.26).

Note:
When the HGEN CTRL bit = 0, the LIN nodes compare the received ID to the ID-BYTE field in the LINID register, and use the RX ID MASK and the TX ID MASK in the

LINMASK register to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA (SCIGCR1[24]) bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. A mask of all 0s will compare all bits of the received identifier in the shift register with the ID-BYTE field in LINID[7:0]. A mask of all 1s will filter all bits of the received identifier and there will be no match.

Note:
For software compatibility with future LIN modules the HGEN CTRL bit (SCIGCR1[12]) must be set to 1, the RX ID Mask (LINMASK[23:16]) must be set to 0xFF and the TX ID MASK (LINMASK[7:0]) must be set to 0xFF.

During header reception, the received identifier is copied to the LINID[23:16] field. If there is no parity error and there is either a TX match or an RX match, then the corresponding TX or RX ID flag is set (SCIFLR[13] or SCIFLR[14]; Section 16.6.8). If the ID interrupt is enabled (SCISETINT[13] = 1; Section 16.6.4), then an ID interrupt is generated.

After the ID interrupt is generated, the CPU may read the LINID[23:16] buffer and determine what response to load into the transmit buffers.

Note:
When byte 0 is written to TD0 (LINTD0[31:24]), the response transmission is automatically generated.

In multibuffer mode, the TXRDY flag (Section 16.6.8) will be set when all the response data bytes and checkbyte are copied to the shift register SCITXSHF. In non multibuffer mode, the TXRDY flag is set each time a byte is copied to the SCITXSHF register, and also for the last byte of the frame after the checkbyte is copied to the SCITXSHF register.

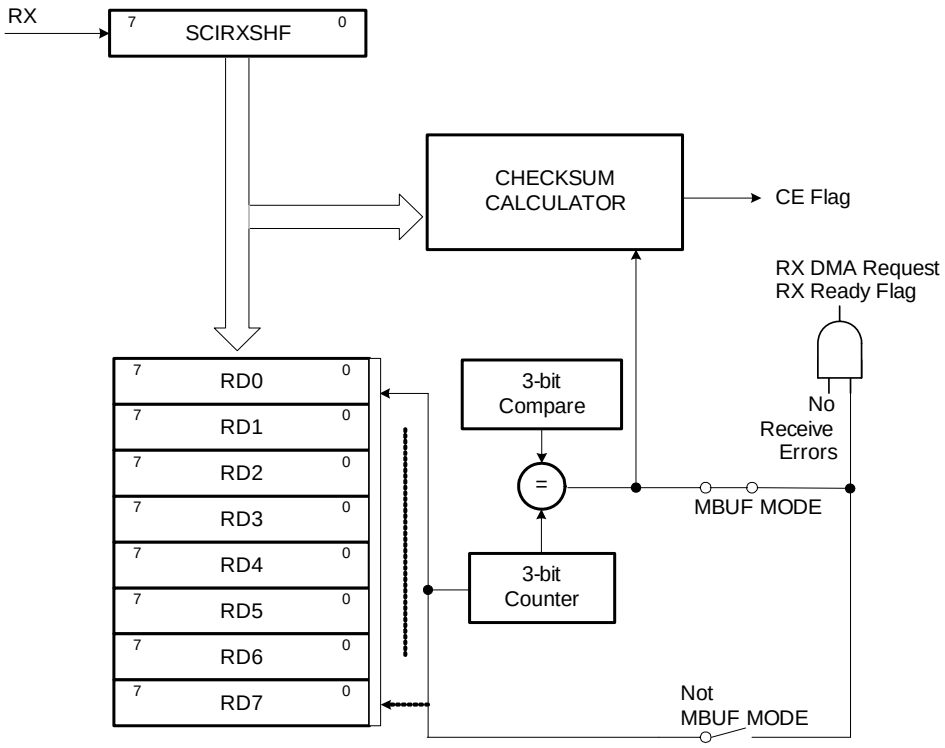
In multibuffer mode, the TXEMPTY flag is set when both the transmit buffer(s) TDy and the SCITXSHF shift register are emptied, and also after the checksum has been sent. In non multibuffer mode, TXEMPTY is set each time TD0 and SCITXSHF are emptied, except for the last byte of the frame where the checkbyte must also be transmitted. In multibuffer mode, TXEMPTY is set when the checkbyte has been transmitted.

If parity is enabled (SCIGCR1[3]), all slave receiving nodes will validate the identifier using all eight bits of the received ID byte. The TMS470Px SCI/BLIN will flag a corrupted identifier if an ID-parity error is detected.

16.2.9 Receive Buffers

To reduce CPU load when receiving a LIN N-byte (with N = 1–8) response in interrupt mode, the SCI/BLIN module has eight receive buffers; see Section 16-15. These buffers can store an entire LIN response in the RDy receive buffers. Also, these receive buffers may be used in buffered SCI mode (the LIN MODE bit, SCIGCR1[6], = 0 and the MBUF MODE bit, SCIGCR1[10], = 1).

Figure 16-15. Receive Buffers



The checkbyte following the data bytes is validated by the checksum calculator in the TED logic. The checksum error (CE) flag indicates a checksum error and a CE interrupt will be generated if enabled in the SCISETINT register.

The multibuffer 3-bit counter counts the data bytes transferred from the SCIRXSHF register to the RDy receive buffers (Section 16.6.23–Section 16.6.24) if multibuffer mode is enabled, or to RD0 if multibuffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be received. Therefore, it is loaded with the length control field (bits 4 and 5) of the ID BYTE field received during the header reception. However, in cases where the ID BYTE field does not convey message length, the LENGTH value, SCIFORMAT[18:16] (Section 16.6.11), indicates the expected length and is used to load the 3-bit compare register. Whether the length control field or the LENGTH value is used is selectable with the SCIGCR1[0] bit in communication mode.

Note: A receive interrupt (RX interrupt; see the SCIINTVECT0/1 registers in Section 16.6.9/ Section 16.6.10), and a receive ready RXRDY flag (SCIFLR[9]; Section 16.6.8) could occur after receiving a response if there are no response receive errors for the frame (checksum error, frame error, and overrun error). The checkbyte will be compared before acknowledging a reception.
The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.

16.2.10 Transmit Buffers

To reduce the CPU load when transmitting a LIN N-byte (with N = 1–8) response in interrupt mode , the SCI/ BLIN module has eight transmit buffers, TD0–TD7 in LINTDO and LINTD1 (Section 16.6.27 and Section 16.6.28). With these transmit buffers, an entire LIN response field can be preloaded in the TXy transmit

buffers. Also, these transmit buffers may be used in buffered SCI mode (LIN MODE bit, SCIGCR1[6], = 0 and MBUF MODE bit, SCIGCR1[10], = 1).

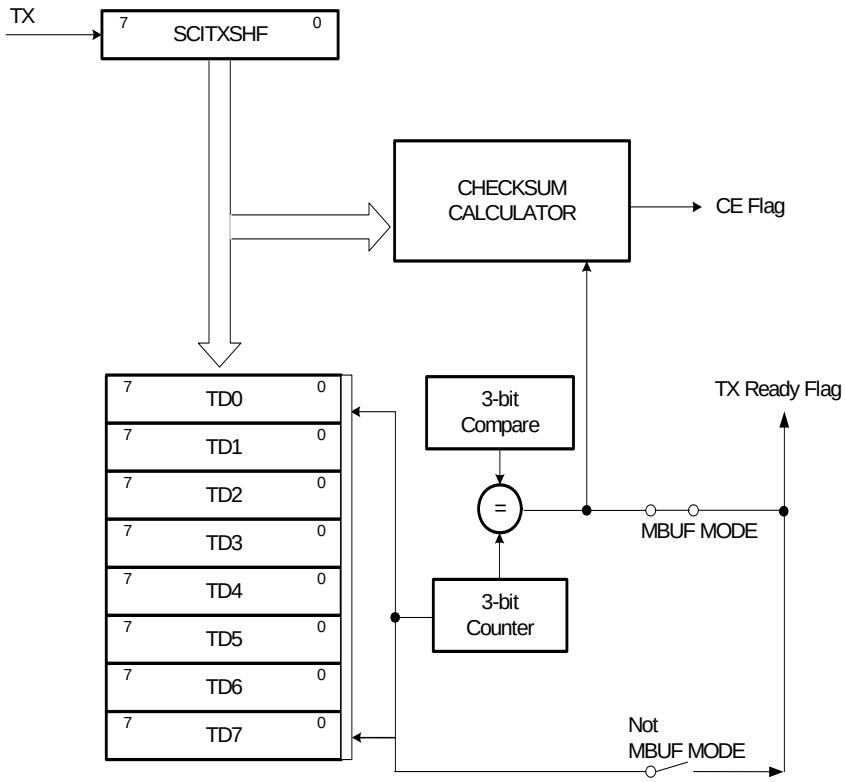
The multibuffer 3-bit counter counts the data bytes transferred from the TDy transmit buffers register if multibuffer mode is enabled, or from TD0 to SCITXSHF if multibuffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be transmitted. Therefore it is loaded with the length control field (bits 4 and 5) of the ID BYTE (Section 16.6.26) received during the header reception. However, if the ID field is not used to convey message length, the LENGTH value, SCIFORMAT[18:16], indicates the expected length and is used instead to load the 3-bit compare register. Which bit field is used is selectable with the COMM MODE bit in SCIGCR1[0].

A transmit interrupt (TX interrupt), and a transmit ready flag (TXRDY flag, SCIFLR[8])could occur after transmitting a response. Figure 16-16 illustrates the transmit buffers.

Note:
The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.

Note:
The transmit interrupt request can be eliminated until the next series of data is written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLEARINT register or by disabling the transmitter via the TXENA bit (SCIGCR1[25]=0).

Figure 16-16. Transmit Buffers



The checkbyte will be generated by the checksum calculator and sent after the data-fields transmission is finished. The multibuffer 3-bit counter counts the data bytes transferred from the TDy buffers into the SCITXSHF register.

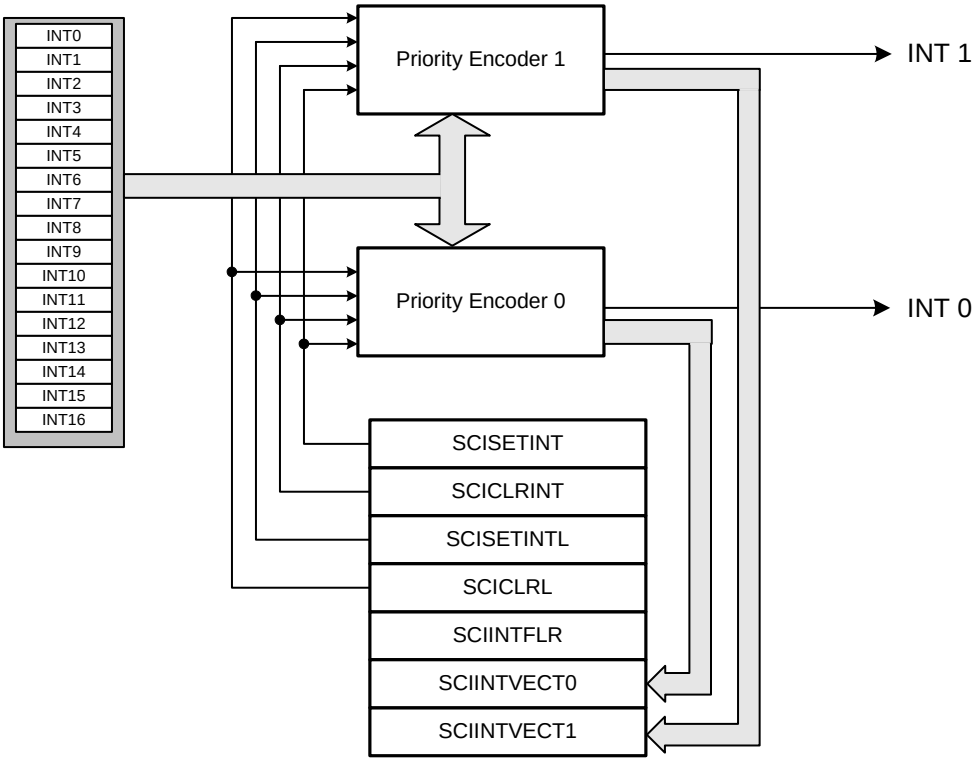
16.3 Interrupts

The SCI/BLIN module has two interrupt lines, level 0 and level 1, to the vectored interrupt manager (VIM) module (see Figure 16-17). Two offset registers (SCIINTVECT0 and SCIINTVECT1; Section 16.6.9 and Section 16.6.10) determine which flag triggered the interrupt according to the respective priority encoders. Each interrupt has a bit to enable/disable the interrupt in the SCISETINT and SCICLRINT registers (Section 16.6.6 and Section 16.6.7), respectively.

Each interrupt has a bit that can be set as interrupt level 0 or as interrupt level 1. By default, interrupts are in interrupt level 0. SCISETINTLVL sets a given interrupt to level1. SCICLEARINTLVL (Section 16.6.7) resets a given interrupt level to the default level 0.

The interrupt vector registers SCIINTVECT0 and SCIINTVECT1 (Section 16.6.9 and Section 16.6.10) return the vector of the pending interrupt line INT0 or INT1. If more than one interrupt is pending, the interrupt vector register holds the highest priority interrupt.

Figure 16-17. General Interrupt Scheme



There are 16 interrupt sources in the SCI/BLIN module, with 8 of them being LIN mode only, as seen in Table 16-4

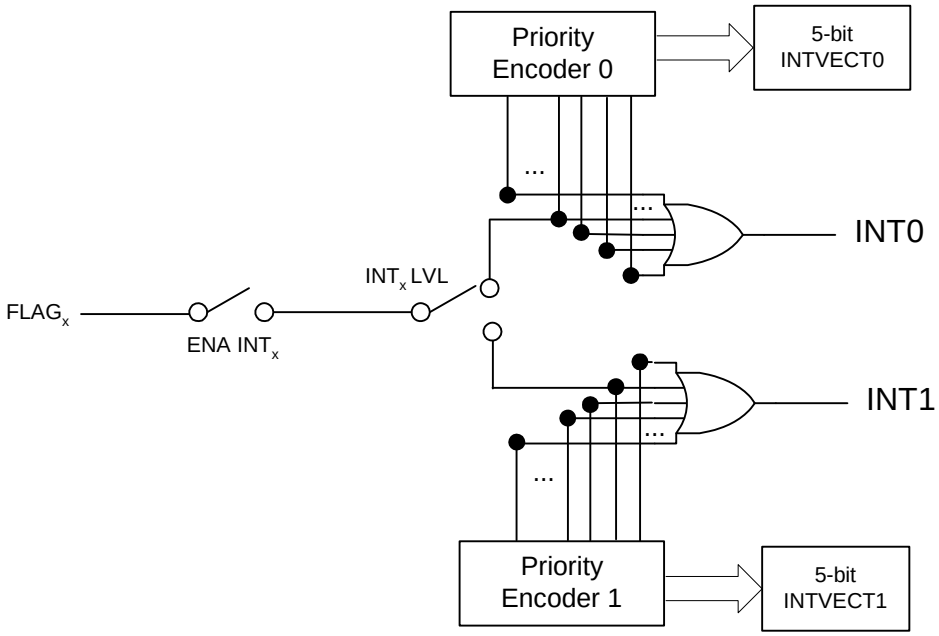
Table 16-4. SCI/BLIN Interrupts

Offset[1]	Interrupt	Comment
0	No interrupt	
1	Wakeup	SCI + LIN mode
2	Inconsistent-synch-field error	LIN mode only
3	Parity error	SCI + LIN mode
4	ID	LIN mode only
5	Physical bus error	LIN mode only
6	Frame error	SCI + LIN mode
7	Break detect	SCI mode only
8	Checksum error	LIN mode only
9	Overrun error	SCI + LIN mode
10	Bit error	SCI + LIN mode
11	Receive	SCI + LIN mode
12	Transmit	SCI + LIN mode
13	No-response error	LIN mode only
14	Timeout after wakeup signal (150 ms)	LIN mode only
15	Timeout after three wakeup signals (1.5 s)	LIN mode only
16	Timeout (Bus Idle, 4s)	LIN mode only

1 Offset 1 is the highest priority. Offset 16 is the lowest priority.

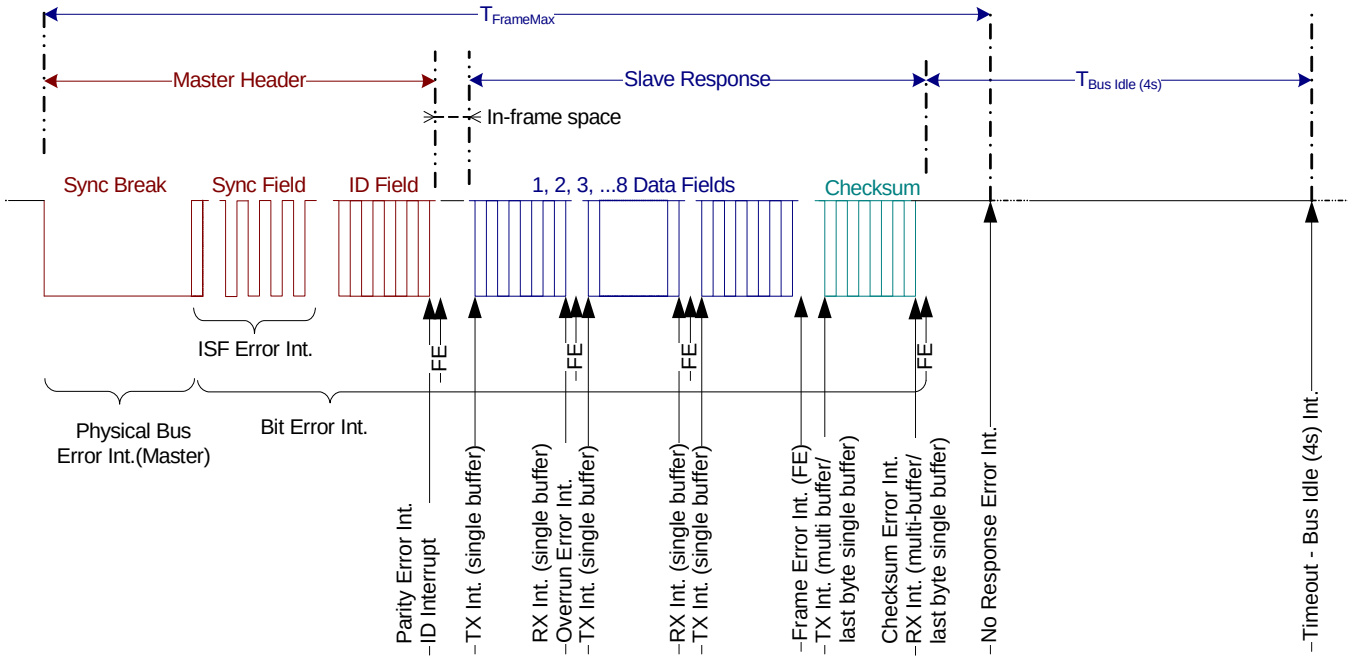
For each of the interrupt sources FLAGx, the SCISETINT and SCICLRINT register pair is used to enable/ disable the interrupt. For each source, the interrupt line can be chosen to be INT0 or INT1 with the SCISETINTLVL and SCICLRINTLVL register pair.

Figure 16-18. Interrupt Generation for Given Flags



A LIN message frame indicating the timing and sequence of the LIN interrupts that could occur is shown in Table 16-19.

Figure 16-19. LIN Message Frame Showing LIN Interrupt Timing and Sequence



16.4 Low-Power Mode

The SCI/BLIN module enters low-power mode when a sleep command frame is received. A wakeup signal will terminate the sleep mode of the LIN bus. On receipt of the sleep command, the POWERDOWN bit in SCIGCR2 (Section 16.6.3) must be set by the application software and the module enters local low-power mode.

Much like the TMS470Px SCI, the SCI/BLIN module can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI/BLIN module. During global low-power mode, all clocks to the SCI/BLIN are turned off so the module is completely inactive.

Note: Enabling Local Low-Power Mode During Receive and Transmit.
If the wakeup interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI/BLIN immediately generates a wake-up interrupt to clear the powerdown bit. Thus, the SCI/BLIN is prevented from entering low-power mode and completes the current reception. Otherwise, if the wakeup interrupt is disabled, the SCI/BLIN completes the current reception and then enters the low-power mode.

16.4.1 Entering Sleep Mode

In LIN mode, a sleep command is used to broadcast the sleep mode to all nodes. The sleep command consists of a diagnostic master request frame with identifier 0x3C (60), with the first data field as 0x00. There should be no activity in the bus once all nodes receive the sleep command: the bus is in sleep mode.

Local low-power mode is asserted by setting the POWERDOWN (SCIGCR2[0]; Section 16.6.3) bit; setting this bit stops the clocks to the SCI/BLIN internal logic and registers. Clearing the POWERDOWN bit causes SCI/BLIN to exit from local low-power mode. All the registers are accessible during local power-down mode. If a register is accessed in low-power mode, this access results in enabling the clock to the module for that particular access alone.

16.4.2 Wakeup

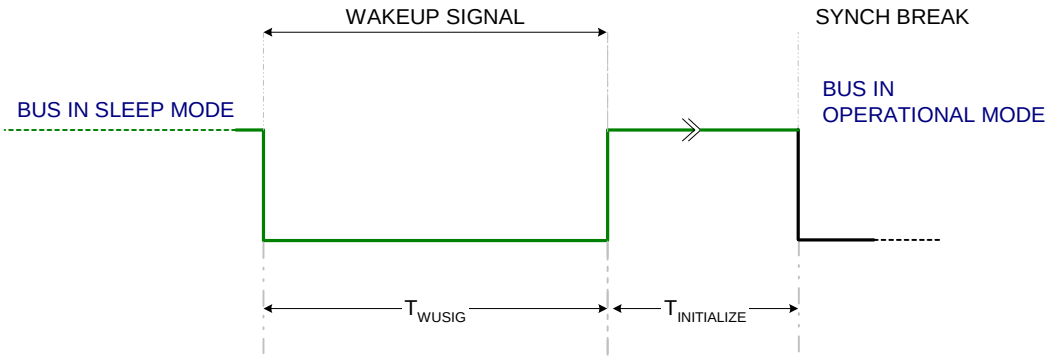
The wakeup interrupt is used to allow the SCI/BLIN module to automatically exit low-power mode. A SCI/BLIN wakeup is triggered when a low level is detected on the receive RX pin, and this clears the POWERDOWN bit.

Note:
If the wakeup interrupt is disabled (WAKEUP INT in the SCISETINT register is cleared), then the SCI/BLIN enters low-power mode whenever it is requested to do so, but a low level on the receive RX pin does NOT cause the SCI/BLIN to exit low-power mode.

In LIN mode, any node can terminate sleep mode by sending a wakeup signal; see Figure 16-20. A slave node that detects the bus in sleep mode, and with a wakeup request pending, will send a wakeup signal. The wakeup signal is generated with a dominant value on the LIN bus for T_{WUSIG}; this is at least 5 T_{bits} for the LIN bus baud rates. The wakeup signal is generated by sending an 0xF0 byte containing 5 dominant T_{bits} and 5 recessive T_{bits}.

$$0.25\text{ms} \leq T_{WUSIG} \leq 5\text{ms}$$

Figure 16-20. Wakeup Signal Generation



Assuming a perfect bus with no noise or loading effects, a write of 0xF0 to TD0 will load the transmitter to meet the wakeup signal timing requirement for T_{WUSIG}. Then, setting the GENWU bit (SCIGCR2[8]; Section 16.6.3) will transmit the preloaded value in TD0 for a wakeup signal transmission.

Note:
The GENWU bit can be set/reset only when SW_nRESET is set to '1' and the node is in power down mode. The bit will be cleared on a valid synch break detection. A master sending a wakeup request, will exit power down mode upon reception of the wakeup pulse. .

The TI TPIC1021 LIN transceiver, upon receiving a wakeup signal, will translate it to the TMS470Px microcontroller for wakeup with a dominant level on the RX pin, or a signal to the voltage regulator. While the POWERDOWN (SCIGCR2[0]; Section 16.6.3) bit is set, if the TMS470Px SCI/BLIN module detects a dominant level in the RX pin, it will generate a wakeup interrupt if enabled in the SCISETINT register (Section 16.6.4).

According to LIN protocol 2.0, the TI TPIC1021 LIN transceiver detecting a dominant level on the bus longer than 150 ms will detect it as a wakeup request. The SCI/BLIN controller's slave is ready to listen to the bus in less than 100 ms ($T_{INITIALIZE}$<100ms) after a dominant-to-recessive edge (end-of-wakeup signal).

16.4.3 Wakeup Timeouts

The LIN protocol defines the following timeouts for a wakeup sequence. After a wakeup signal has been sent to the bus, all nodes wait for the master to send a header. If no synch field is detected before 150 ms (3,000 cycles at 20 kHz) after dominant wakeup signal's ending edge, a new wakeup is sent by the same node that requested the first wakeup. This sequence is not repeated more than three times. After three attempts to wake up the LIN bus, wakeup signal generation is suspended for a 1.5 s (30,000 cycles at 20 kHz) period.

Note:
The SCI/BLIN controller handles the wakeup expiration times defined by the LIN protocol with a hardware implementation.

Note:
To achieve compatibility to LIN1.3 timeout conditions, the MBRS register (Section 16.6.29) must be set to assure that the LIN 2.0 (real-time-based) timings meet the LIN 1.3 bit time base. A node triggering the wakeup should set the MBRS register accordingly to meet the targeted time as 128 Tbits x programmed prescaler.

16.5 Emulation Mode

In emulation mode, the CONT bit of the SCIGCR1 (Section 16.6.2) determines how the SCI/BLIN operates when the program is suspended. The SCI/BLIN counters are affected by this bit during debug mode; when set, the counters are not stopped and when cleared, the counters are stopped.

Any reads in emulation mode to a SCI/BLIN register will not have any effect on the flags in the SCIFLR register (Section 16.6.8).

16.6 SCI/BLIN Control Registers

The SCI/BLIN module registers are based on the SCI registers, with added functionality registers enabled by the LIN MODE bit in the SCIGCR1 register.

These registers are accessible in 8-, 16-, and 32-bit reads or writes. The SCI/BLIN is controlled and accessed through the registers listed in Figure 16-21. Among the features that can be programmed are the LIN protocol mode, communication and timing modes, baud rate value, frame format, and interrupt configuration. The start address for the SCI/BLIN module is 0xFFF7 E400.

Figure 16-21. SCI/BLIN Control Registers Summary

Register Offset Address[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00 SCIGCR0 Page 898	Reserved															
	Reserved															RESET
0x04 SCIGCR1 Page 899	Reserved							TX ENA	RX ENA	Reserved					CONT	LOOP BACK
	Reserved			HGEN CTRL	CTYPE	MBUF MODE	ADAPT	SLEEP	SW nRST	LIN MODE	CLOCK	STOP	PARITY	PAR-ITY ENA	TIM-MING MODE	COMM MODE
0x08 SCIGCR2 Page 907	Reserved								Reserved						CC	SC
	Reserved							GEN WU	Reserved							POWER DOWN
0x0C SCISETINT Page 909	SET BE INT	SET PBE INT	SET CE INT	SET ISFE INT	SET NRE INT	SET FE INT	SET OE INT	SET PE INT	Reserved							
	Reserved		SET ID	Reserved			SET RX INT	SET TX INT	SET TOA3 WUS INT	SET TOA WUS INT	Reserved	SET TIME-OUT INT	Reserved		SET WAKE UP INT	SET BRKDT INT
0x10 SCICLEARINT Page 913	CLR BE INT	CLR PBE INT	CLR CE INT	CLR ISFE INT	CLR NRE INT	CLR FE INT	CLR OE INT	CLR PE INT	Reserved							
	Reserved		CLR ID INT	Reserved			CLR RX INT	CLR TX INT	CLR TOA3 WUS INT	CLR TOA WUS INT	Reserved	CLR TIME-OUT INT	Reserved		CLR WAKE UP INT	CLR BRKDT INT
0x14 SCI SETINTLVL Page 917	SET BE INT LVL	SET PBE INT LVL	SET CE INT LVL	SET ISFE INT LVL	SET NRE INT LVL	SET FE INT LVL	SET OE INT LVL	SET PE INT LVL	Reserved							
	Reserved		SET ID INT LVL	Reserved			SET RX INT LVL	SET TX INT LVL	SET TOA3 WUS INT LVL	SET TOA WUS INT LVL	Reserved	SET TIME-OUT INT LVL	Reserved		SET WAKE UP INT LVL	SET BRKDT INT LVL

1 The offset address is relative to the peripheral's beginning address.

Register Offset Address[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x18 SCICLEARINTLVL Page 920	CLR BE INT LVL	CLR PBE INT LVL	CLR CE INT LVL	CLR ISFE INT LVL	CLR NRE INT LVL	CLR FE INT LVL	CLR OE INT LVL	CLR PE INT LVL	Reserved							
	Reserved		CLR ID TX INT LVL	Reserved			CLR ID RX INT LVL	CLR TX INT LVL	CLR TOA3 WUS INT LVL	CLR TOA WUS INT LVL	Reserved	CLR TIME-OUT INT LVL	Reserved		CLR WAKE UP INT LVL	CLR BRKDT INT LVL
0x1C SCIFLR Page 924	BE	PBE	CE	ISFE	NRE	FE	OE	PE	Reserved							
	Reserved	ID RX	ID TX	RX WAKE	TX EMPTY	TX WAKE	RX RDY	TX RDY	TOA3 WUS	TOA WUS	Reserved	TIME-OUT	BUSY	IDLE	WAKE UP	BRKDT
0x20 SCIINTVECT0 Page 934	Reserved															
	Reserved														INTVECT0[4:0]	
0x24 SCIINTVECT1 Page 935	Reserved															
	Reserved														INTVECT1[4:0]	
0x28 SCIFORMAT Page 936	Reserved														LENGTH[2:0]	
	Reserved														CHAR[2:0]	
0x2C BRS Page 938	Reserved	U(2–0)			M(3–0)				PRESCALER P [23:16]							
	PRESCALER P [15:0]															
0x30 SCIED Page 941	Reserved															
	Reserved								ED[7:0]							
0x34 SCIRD Page 942	Reserved															
	Reserved								RD[7:0]							

1 The offset address is relative to the peripheral's beginning address.

Register Offset Address[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x38 SCITD Page 943	Reserved															
	Reserved								TD[7:0]							
0x3C SCIPIO0 Page 944	Reserved															
	Reserved													TX FUNC	RX FUNC	Res
0x40 SCIPIO1 Page 945	Reserved															
	Reserved													TX DIR	RX DIR	Res
0x44 SCIPIO2 Page 947	Reserved															
	Reserved													TX IN	RX IN	Res
0x48 SCIPIO3 Page 948	Reserved															
	Reserved													TX OUT	RX OUT	Res
0x4C SCIPIO4 Page 949	Reserved															
	Reserved													TX SET	RX SET	Res
0x50 SCIPIO5 Page 950	Reserved															
	Reserved													TX CLR	RX CLR	Res
0x58 SCIPIO7 Page 951	Reserved															
	Reserved													TX PD	RX PD	Res

1 The offset address is relative to the peripheral's beginning address.

Register Offset Address[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x5C SCIPIO8 Page 952	Reserved															
	Reserved													TX PSL	RX PSL	Res
0x60 LINCOMPARE Page 953	Reserved															
	Reserved							SDEL[1:0]			Reserved			SBREAK[2:0]		
0x64 LINRD0 Page 955	RD0[7:0]								RD1[7:0]							
	RD2[7:0]								RD3[7:0]							
0x68 LINRD1 Page 956	RD4[7:0]								RD5[7:0]							
	RD6[7:0]								RD7[7:0]							
0x6C LINMASK Page 957	Reserved								RX ID MASK[7:0]							
	Reserved								TX ID MASK[7:0]							
0x70 LINID Page 958	Reserved								Received ID[7:0]							
	ID-SlaveTask BYTE(7–0)								ID BYTE[7:0]							
0x74 LINTD0 Page 959	TD0[7:0]								TD1[7:0]							
	TD2[7:0]								TD3[7:0]							
0x78 LINTD1 Page 960	TD4[7:0]								TD5[7:0]							
	TD6[7:0]								TD7[7:0]							

1 The offset address is relative to the peripheral's beginning address.

Register Offset Address[1]	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x7C MBRS Page 961	Reserved															
	Reserved				MBR[12:0]											
0x90 IODFTCTRL Page 962	BEN	PBEN	CEN	ISFE	Reserved	FEN	PEN	BRKDT ENA	Reserved			PIN SAMPLE MASK	TX SHIFT[2:0]			
	Reserved				IODFTENA[3:0]				Reserved						LPB ENA	RXP ENA

1 The offset address is relative to the peripheral's beginning address.

16.6.1 SCI Global Control Register 0 (SCIGCR0)

The SCIGCR0 register defines the module reset. Figure 16-22 and Table 16-5 illustrate this register.

Figure 16-22. SCI Global Control Register 0 (SCIGCR0) [offset = 00h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RESET
R-0															RWP-0

R = Read, W = Write in all modes; RWP = Read/Write in privileged mode only; S = Set, U = Undefined, -*n* = Value after reset

.

Table 16-5. SCI Global Control Register 0 (SCIGCR0) Field Description

Bit	Name	Value	Description
31–1	Reserved		Reads return 0 and writes have no effect.
0	Reset		This bit resets the SCI/BLIN module. This bit is effective in SCI and LIN mode.
		0	SCI/BLIN module is in reset.
		1	SCI/BLIN module is out of reset.
			Note: Read/Write in privileged mode only.

16.6.2 SCI Global Control Register 1 (SCIGCR1)

The SCIGCR1 register defines the frame format, protocol, and communication mode used by the SCI. Figure 16-23 and Table 16-6 illustrate this register.

Figure 16-23. SCI Global Control Register 1 (SCIGCR1) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						TXENA	RXENA	Reserved						CONT	LOOP BACK
R-0						R/W-0	R/W-0	R-0						R/W-0	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			HGEN CTRL	CTYPE	MBUF MODE	ADAPT	SLEEP	SW nRST	LIN MODE	CLOCK	STOP	PARITY	PARITY ENA	TIMING MODE	COMM MODE
R-0	R-0	R-0	R/WL-0	R/WL-0	R/W-0	R/WL-0	R/W-0	R/W-0	RWP-0	R/W-0	R/WC-0	R/WC-0	R/W-0	R/WC-0	R/W-0

R = Read in all modes; W = Write in all modes; WL = Write in LIN mode only; WP = Write in privileged mode only; WC = Write in sci-compatible mode only; *-n* = Value after reset

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description

Bit	Name	Value	Description
31–26	Reserved		Reads return 0 and writes have no effect.
25	TXENA		Transmit enable. This bit is effective in LIN and SCI modes. Data is transferred from SCITD (Section 16.6.13.3), or the TDy (with y=0, 1, ...7) buffers (Section 16.6.27 and Section 16.6.28) in LIN mode to the SCITXSHF shift out register only when the TXENA bit is set.
		0	Transfers from SCITD or TDy to SCITXSHF are disabled.
		1	Transfers from SCITD or TDy to SCITXSHF are enabled.
			Note: Data written to SCITD or the transmit multi-buffer before TXENA is set is not transmitted. If TXENA is cleared while transmission is ongoing, the data previously written to SCITD is sent (including the checkbyte in LIN mode).

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
24	RXENA		Receive enable. This bit is effective in LIN and SCI modes. RXENA allows or prevents the transfer of data from SCIRXSHF to SCIRD or the receive multibuffers.
		0	The receiver will not transfer data from the shift buffer to the receive buffer or multi-buffers.
		1	The receiver will transfer data from the shift buffer to the receive buffer or multi-buffers.
			Note: Clearing RXENA stops received characters from being transferred into the receive buffer or multi-buffers, prevents the RX status flags (see Table 16-5) from being updated by receive data, and inhibits both receive and error interrupts. However, the shift register continues to assemble data regardless of the state of RXENA.
			Note: If RXENA is cleared before a frame is completely received, the data from the frame is not transferred into the receive buffer.
			Note: If RXENA is set before a frame is completely received, the data from the frame is transferred into the receive buffer. If RXENA is set while SCIRXSHF is in the process of assembling a frame, the status flags are not guaranteed to be accurate for that frame. To ensure that the status flags correctly reflect what was detected on the bus during a particular frame, RXENA should be set before the detection of that frame.
23–18	Reserved		Reads return 0 and writes have no effect.
17	CONT		Continue on suspend. This bit is effective in LIN and SCI modes. This bit has an effect only when a program is being debugged with an emulator, and it determines how the SCI/BLIN operates when the program is suspended. The SCI/BLIN counters are affected by this bit: when the bit is set the counters are not stopped, when the bit is cleared the counters are stopped during debug mode.
		0	When debug mode is entered, the SCI/BLIN state machine is frozen. Transmissions and LIN counters are halted and resume when debug mode is exited.
		1	When debug mode is entered, the SCI/BLIN continues to operate until the current transmit and receive functions are complete.

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
16	LOOP BACK		Loopback bit. This bit is effective in LIN and SCI modes. The self-checking option for the SCI/BLIN can be selected with this bit. If the LINITX and LINRX pins are configured with SCI/BLIN functionality, then the LINTX pin is internally connected to the LINRX pin. Externally, during loop back operation, the LINTX pin outputs a high value and the LINRX pin is in a high-impedance state. If this bit value is changed while the SCI/BLIN is transmitting or receiving data, errors may result.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled.
15–13	Reserved		Reads return 0 and writes have no effect.
12	HGEN CTRL		HGEN control. This bit is effective in LIN mode only. This bit controls the type of mask filtering comparison
		0	ID filtering using the ID-Byte field in LINID (Section 16.6.26) occurs. Mask of 0xFF in LINMASK register will result in no match.
		1	ID filtering uses ID-SlaveTask BYTE (recommended). Mask of 0xFF in LINMASK register will result in ALWAYS match. **Note: For software compatibility with future LIN modules the HGEN CTRL bit must be set to 1, the RX ID Mask (LINMASK[23:16]) must be set to 0xFF and the TX ID MASK (LINMASK[7:0]) must be set to 0xFF.**
11	CTYPE		Checksum type. This bit is effective in LIN mode only. This bit controls the type of checksum to be used: classic or enhanced.
		0	Classic checksum is used.
		1	Enhanced checksum is used.
10	MBUF MODE		Multibuffer mode. This bit is effective in LIN and SCI modes.This bit controls receive/transmit buffer usage, i.e., whether the RX/TX multibuffers are used or a single register, RD0/TD0, is used.
		0	The multi-buffer mode is disabled.
		1	The multi-buffer mode is enabled.

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
9	ADAPT		Adapt. This mode is effective in LIN mode only. This bit has an effect during the detection of the synch field. Two LIN protocol bit rate modes could be enabled with this bit according to the node capability file definition: automatic or select. The software and network configuration will decide which of these two modes are enabled. When this bit is cleared, the LIN 2.0 protocol fixed bit rate should be used. If the ADAPT bit is set, a SCI/BLIN slave node detecting the baud rate will compare it to the prescalers in BRS register and update it if they are different. The BRS register will be updated with the new value. If this bit is not set there will be no adjustment to the BRS register.
		0	Automatic baud rate adjustment is disabled.
		1	Automatic baud rate adjustment is enabled.

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
8	SLEEP		SCI sleep. This bit is effective in LIN and SCI modes. In a multiprocessor configuration, this bit controls the receive sleep function. Clearing this bit brings the SCI/BLIN out of sleep mode.
		0	Sleep mode is disabled.
		1	Sleep mode is enabled.
			Note: The receiver still operates when the SLEEP bit is set; however, RXRDY is updated and SCIRD is loaded with new data only when an address frame is detected. The remaining receiver status flags (see Table 16-5) are updated and an error interrupt is requested if the corresponding interrupt enable bit is set, regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the SCI is asleep, software can promptly deal with the error condition.
			Note: The SLEEP bit is *not* automatically cleared when an address byte is detected.
			See Section 16.4 for more information on using the SLEEP bit for multiprocessor communication.
7	SWnRESET		Software reset (active low). This bit is effective in LIN and SCI modes.
		0	The SCI/BLIN is in its reset state; no data will be transmitted or received. Writing a 0 to this bit initializes the SCI state machines and operating flags as defined in Table 16-7 and Table 16-8. All affected logic is held in the reset state until a 1 is written to this bit.
		1	The SCI/BLIN is in its ready state; transmission and reception can be done. After this bit is set to 1, the configuration of the module should not change.
			Note: The SCI/BLIN should only be configured while SW nRESET = 0.
6	LIN MODE		LIN mode. This bit is effective in LIN or SCI-compatible mode. This bit controls the module mode of operation.
		0	LIN mode is disabled; SCI compatibility mode is enabled.
		1	LIN mode is enabled; SCI compatibility mode is disabled.

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
5	CLOCK		SCI internal clock enable. The CLOCK bit determines the source of the module clock on the SCICLK pin. It also determines whether a LIN node is a slave or master.
			SCI-compatible mode
		0	The external SCICLK is the clock source.
			Note: If an external clock is selected, then the internal baud rate generator and baud rate registers are bypassed. The maximum frequency allowed for an externally sourced SCI clock is VCLK/16.
		1	The internal SCICLK is the clock source.
			LIN mode
		0	The node is in slave mode.
		1	The node is in master mode.
4	STOP		SCI number of stop bits. This bit is effective in SCI-compatible mode only.
		0	One stop bit is used.
		1	Two stop bits are used.
			Note: The receiver checks for only one stop bit. However in idle-line mode, the receiver waits until the end of the second stop bit (if STOP = 1) to begin checking for an idle period.
3	PARITY		SCI parity odd/even selection.This bit is effective in SCI-compatible mode only. If the PARITY ENA bit (SCICCR1[3]; Section 16.6.2) is set, PARITY (SCICCR[4]) designates odd or even parity.
		0	Odd parity is used.
		1	Even parity is used.
			Note: The parity bit is calculated based on the data bits in each frame and the address bit (in address-bit mode). The start and stop fields in the frame are not included in the parity calculation.
			Note: For odd parity, the SCI transmits and expects to receive a value in the parity bit that makes odd the total number of bits in the frame with the value of 1.
			Note: For even parity, the SCI transmits and expects to receive a value in the parity bit that makes even the total number of bits in the frame with the value of 1.

Table 16-6. SCI Global Control Register 1 (SCIGCR1) Field Description (Continued)

Bit	Name	Value	Description
2	PARITY ENA		Parity enable. This bit enables or disables the parity function.
			SCI compatibility or buffered SCI mode
		0	Parity is disabled; no parity bit is generated during transmission or is expected during reception.
		1	Parity is enabled. A parity bit is generated during transmission and is expected during reception.
			LIN mode
		0	ID-parity verification is disabled.
		1	ID-parity verification is enabled.
1	TIMING MODE		SCI timing mode bit. This bit is effective in SCI-compatible mode only. it selects the SCI timing mode.
		0	Synchronous timing is used.
		1	Asynchronous timing is used.
			See the SCI document for more information on the synchronous and asynchronous timing modes.
0	COMM MODE		SCI/BLIN communication mode bit. In compatibility mode it selects the SCI communication mode. In LIN mode it selects length control option for ID-field bits ID4 and ID5.
			SCI-compatible mode
		0	Idle-line mode is used.
		1	Address-bit mode is used.
			See the SCI document for more information on these communication modes.
			LIN mode
		0	ID4 and ID5 are not used for length control.
		1	ID4 and ID5 are used for length control.

Table 16-7. SCI Receiver Status Flags

SCI Flag	Register	Bit	Value After SW nRESET[1]
CE	SCIFLR	29	0
ISFE	SCIFLR	28	0
NRE	SCIFLR	27	0
FE	SCIFLR	26	0
OE	SCIFLR	25	0
PE	SCIFLR	24	0
RXWAKE	SCIFLR	12	0
RXRDY	SCIFLR	9	0
BUSY	SCIFLR	3	0
IDLE	SCIFLR	2	0
WAKE UP	SCIFLR	1	0
BRKDT	SCIFLR	0	0

1 The flags are frozen with their reset value while SW nRESET = 0.

Table 16-8. SCI Transmitter Status Flags

SCI Flag	Register	Bit	Value After SW nRESET[1]
BE	SCIFLR	31	0
PBE	SCIFLR	30	0
TX WAKE	SCIFLR	10	0
TX EMPTY	SCIFLR	11	1
TXRDY	SCIFLR	8	1

1 The flags are frozen with their reset value while SW nRESET = 0.

16.6.3 SCI Global Control Register 2 (SCIGCR2)

The SCIGCR2 register is used to send or compare a checkbyte during extended frames, to generate a wakeup and for low-power mode control of the LIN module. Figure 16-24 and Table 16-9 illustrate this register.

Figure 16-24. SCI Global Control Register 2 (SCIGCR2) [offset = 08h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CC	SC
R-0														R/WL-0	R/WL-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							GEN WU	Reserved							POWER DOWN
R-0							R/W-0	R-0							R/W-0

R = Read, W = Write in all modes, WL = Write in LIN mode only; -*n* = Value after reset

.

Table 16-9. SCI Global Control Register 2 (SCIGCR2) Field Description

Bit	Name	Value	Description
31–18	Reserved		Reads return 0 and writes have no effect.
17	CC		Compare checksum. This bit is effective in LIN mode only. This bit is used by the receiver for extended frames to trigger a checksum compare. In non-multibuffer mode, when the CC bit is set, the checksum will be compared on the byte that is expected to be the checkbyte. During multi-buffer mode, the following scenarios are associated with the CC bit: a) If the CC bit is set during the reception of the data, then the byte that is received after the reception of the programmed number of data bytes as indicated by SCIFORMAT[18:16] is treated as a checksum byte. b) If the CC bit is set during the idle period (i.e., during the inter-frame space), then the immediate next byte will be treated as a checksum byte. A CE will immediately be flagged if there is a checksum error. This bit is automatically cleared once the checksum is compared. See Section 16.2.5 for more details.
		0	No checksum compare will occur.
		1	Compare checksum on expected checkbyte.

Table 16-9. SCI Global Control Register 2 (SCIGCR2) Field Description (Continued)

Bit	Name	Value	Description
16	SC		Send checkbyte. This bit is effective in LIN mode only. This bit is used by the transmitter with extended frames to send a checkbyte. In non-multibuffer mode, the checkbyte will be sent after the current byte transmission. In multibuffer mode, the checkbyte will be sent after the last byte count, indicated by the SCIFORMAT[18:16]). See Section 16.2.5 for more details. This byte will be cleared after the checkbyte has been transmitted.
		0	No checkbyte will be sent.
		1	A checkbyte will be sent.
15–9	Reserved		Reads return 0 and writes have no effect.
8	GEN WU		Generate wakeup signal. This bit controls the generation of a wakeup signal, by transmitting the TDO buffer value. The LIN protocol specifies that this signal should be a dominant for T_{WUSIG}. This bit is cleared on reception of a valid synch break.
		0	No wakeup signal will be generated.
		1	The TDO buffer value will be transmitted for a wakeup signal.
			Note: To clear the GENWU bit while not in powerdown mode, the sequence below should be followed: **1) Set SWnRST to 0.** **2) Disable the LINMODE bit (Clears GENWU bit).** **3) Re-enable LINMODE bit.** **4) Reconfigure LIN registers if necessary.** **5) Set SWnRST to 1.**
7–1	Reserved		Reads return 0 and writes have no effect.
0	POWERDOWN		Power down. This bit is effective in LIN or SCI-compatible mode. When this bit is set, the SCI/BLIN module attempts to enter local low-power mode. If the POWERDOWN bit is set while the receiver is actively receiving data and the wakeup interrupt is disabled, then the SCI/BLIN will delay entering low-power mode until the reception is completed. In LIN mode, the user may set the POWERDOWN bit on receiving a sleep command or on idle bus detection (more than 4 seconds, i.e., 80,000 cycles at 20 kHz). See Section 16.4 for more information on low-power mode.
		0	The SCI/BLIN module is in normal operation.
		1	The SCI/BLIN module is entering local low-power mode.

16.6.4 SCI Set Interrupt Register (SCISETINT)

Figure 16-25 and Table 16-10 illustrate this register.

Figure 16-25. SCI Set Interrupt Register (SCISETINT) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SET BE INT	SET PBE INT	SET CE INT	SET ISFE INT	SET NRE INT	SET FE INT	SET OE INT	SET PE INT	Reserved							
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0	R-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SET ID INT	Reserved			SET RX INT	SET TX INT	SET TOA3 WUS INT	SET TOA WUS INT	Reserved	SET TIME-OUT INT	Reserved		SET WAKE UP INT	SET BRKDT INT
R-0		R/WL-0	R-0			R/W-0	R/W-0	R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

R = Read in all modes; W = Write in all modes; WL = Write in LIN mode only; WC = Write in sci-compatible mode only; -*n* = Value after reset

.

Table 16-10. SCI Set Interrupt Register (SCISETINT) Field Description

Bit	Name	Value	Description
31	SET BE INT		Set bit error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN module to generate an interrupt when there is a bit error.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
30	SET PBE INT		Set physical bus error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN module to generate an interrupt when a physical bus error occurs.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
29	SET CE INT		Set checksum-error Interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN module to generate an interrupt when there is a checksum error.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.

Table 16-10. SCI Set Interrupt Register (SCISETINT) Field Description (Continued)

Bit	Name	Value	Description
28	SET ISFE INT		Set inconsistent-synch-field-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN module to generate an interrupt when there is an inconsistent synch field error.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
27	SET NRE INT		Set no-response-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN module to generate an interrupt when a no-response error occurs.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
26	SET FE INT		Set framing-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN module to generate an interrupt when a framing error occurs.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
25	SET OE INT		Set overrun-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN module to generate an interrupt when an overrun error occurs.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
24	SET PE INT		Set parity interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN module to generate an interrupt when a parity error occurs.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
23–19	Reserved		Reads return 0 and writes have no effect.
18–16	Reserved		Reads return 0 and writes have no effect.
15–14	Reserved		Reads return 0 and writes have no effect.

Table 16-10. SCI Set Interrupt Register (SCISETINT) Field Description (Continued)

Bit	Name	Value	Description
13	SET ID INT		Set identification interrupt. This bit is effective in LIN mode only. This bit is set to enable an interrupt when a valid matching identifier is received. See Section 16.2.8 for more details.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
12–10	Reserved		Reads return 0 and writes have no effect.
9	SET RX INT		Receiver interrupt enable. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN to generate a receive interrupt after a frame has been completely received and the data is being transferred from SCIRXSHF to SCIRD.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
8	SET TX INT		Set transmitter interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN to generate a transmit interrupt as data is being transferred from SCITD to SCITXSHF and the TXRDY bit is being set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
7	SET TOA3WUS INT		Set timeout after three wakeup signals interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN to generate an interrupt when a timeout occurs after three wakeup signals have been sent.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
6	SET TOAWUS INT		Set timeout after wakeup signal interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN to generate an interrupt when a timeout occurs after one wakeup signal has been sent.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
5	Reserved		Reads return 0 and writes have no effect.

Table 16-10. SCI Set Interrupt Register (SCISETINT) Field Description (Continued)

Bit	Name	Value	Description
4	SET TIMEOUT INT		Set timeout interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/BLIN to generate an interrupt when no LIN bus activity occurs for at least four seconds.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
3–2	Reserved		Reads return 0 and writes have no effect.
1	SET WAKEUP INT		Set wakeup interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/BLIN to generate a wake-up interrupt and thereby exit low-power mode. If enabled, the wake-up interrupt is asserted when local low-power mode is requested while the receiver is busy or if a low level is detected on the SCIRX pin during low-power mode.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.
0	SET BRKDT INT		Set break-detect interrupt. This bit is effective in SCI-compatible mode only. Setting this bit enables the SCI/BLIN to generate an error interrupt if a break condition is detected on the SCIRX pin.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt is enabled.

16.6.5 SCI Clear Interrupt Register (SCICLEARINT)

Figure 16-26 and Table 16-11 illustrate this register.

Figure 16-26. SCI Clear Interrupt Register (SCICLEARINT) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR BE INT	SET PBE INT	CLR CE INT	CLR ISFE INT	CLR RE INT	CLR FE INT	CLR OE INT	CLR PE INT	Reserved							
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0	R-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		CLR ID INT	Reserved			CLR RX INT	CLR TX INT	CLR TOA3 WUS INT	CLR TOA WUS INT	Reserved	CLR TIME-OUT INT	Reserved		CLR WAKE UP INT	CLR BRKDT INT
R-0		R/WL-0	R-0			R/W-0	R/W-0	R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

R = Read in all modes; W = Write in all modes; WC = Write in sci-compatible mode only; WL = Write in LIN mode only; -*n* = Value after reset

.

Table 16-11. SCI Clear Interrupt Register (SCICLEARINT) Field Description

Bit	Name	Value	Description
31	CLR BE INT		Clear bit error interrupt. This bit is effective in LIN mode only. This bit disables the bit error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
30	CLR PBE INT		Clear physical bus error interrupt. This bit is effective in LIN mode only. This bit disables the physical-bus error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
29	CLR CE INT		Set checksum-error interrupt. This bit is effective in LIN mode only. This bit disables the checksum interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

Table 16-11. SCI Clear Interrupt Register (SCICLEARINT) Field Description (Continued)

Bit	Name	Value	Description
28	CLR ISFE INT		Clear inconsistent-synch-field-error (ISFE) interrupt. This bit is effective in LIN mode only. This bit disables the ISFE interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
27	CLR NRE INT		Clear no-response-error interrupt. This bit is effective in LIN mode only. This bit disables the NRE interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
26	CLR FE INT		Clear framing-error interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the framing-error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
25	CLR OE INT		Clear overrun-error interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the SCI/BLIN overrun error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
24	CLR PE INT		Clear parity interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the parity error interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
23–19	Reserved		Reads return 0 and writes have no effect.
18–16	Reserved		Reads return 0 and writes have no effect.
15–14	Reserved		Reads return 0 and writes have no effect.

Table 16-11. SCI Clear Interrupt Register (SCICLEARINT) Field Description (Continued)

Bit	Name	Value	Description
13	CLR ID INT		Clear ID interrupt. This bit is effective in LIN mode only. This bit disables the ID interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
12–10	Reserved		Reads return 0 and writes have no effect.
9	CLR RX INT		Clear receiver interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the receiver interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
8	CLR TX INT		Clear transmitter interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the transmitter interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
7	CLR TOA3WUS INT		Clear timeout-after-three-wakeup-signals interrupt. This bit is effective in LIN mode only. This bit disables the timeout after three wakeup signals interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
6	CLR TOAWUS INT		Clear timeout after wakeup signal interrupt. This bit is effective in LIN mode only. This bit disables the timeout after one wakeup signal interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
5	Reserved		Reads return 0 and writes have no effect.

Table 16-11. SCI Clear Interrupt Register (SCICLEARINT) Field Description (Continued)

Bit	Name	Value	Description
4	CLR TIMEOUT INT		Clear timeout interrupt. This bit is effective in LIN mode only. This bit disables the timeout (LIN bus idle) interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
3–2	Reserved		Reads return 0 and writes have no effect.
1	CLR WAKEUP INT		Clear wake-up interrupt. This bit is effective in LIN or SCI-compatible mode. This bit disables the wakeup interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.
0	CLR BRKDT INT		Clear break-detect interrupt. This bit is effective in SCI-compatible mode only. This bit disables the break-detect interrupt when set.
		0	*Read:* The interrupt is disabled. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt is enabled. *Write:* The interrupt is disabled.

16.6.6 SCI Set Interrupt Level Register (SCISETINTLVL)

Figure 16-27 and Table 16-12 illustrate this register.

Figure 16-27. SCI Set Interrupt Level Register (SCISETINTLVL) [offset = 14h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SET BE INT LVL	SET PBE INT LVL	SET CE INT LVL	SET ISFE INT LVL	SET NRE INT LVL	SET FE INT LVL	SET OE INT LVL	SET PE INT LVL	Reserved							
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0	R-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved Reserved		SET ID INT LVL	Reserved			SET RX INT LVL	SET TX INT LVL	SET TOA3 WUS INT LVL	SET TOA WUS INT LVL	Reserved	SET TIME-OUT INT LVL	Reserved		SET WAKE UP INT LVL	SET BRKDT INT LVL
R-0		R/WL-0	R-0			R/W-0	R/W-0	R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

R = Read in all modes; W = Write in all modes; WC = Write in sci-compatible mode only; WL = Write in LIN mode only; -*n* = Value after reset

.

Table 16-12. SCI Set Interrupt Level Register (SCISETINTLVL) Field Description

Bit	Name	Value	Description
31	SET BE INT LV		Set bit error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
30	SET PBE INT LVL		Set physical bus error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
29	SET CE INT LVL		Set checksum-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
28	SET ISFE INT LVL		Set inconsistent-synch-field-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

Table 16-12. SCI Set Interrupt Level Register (SCISETINTLVL) Field Description (Continued)

Bit	Name	Value	Description
27	SET NRE INT LVL		Set no-response-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
26	SET FE INT LVL		Set framing-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
25	SET OE INT LVL		Set overrun-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
24	SET PE INT LVL		Set parity error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
23–19	Reserved		Reads return 0 and writes have no effect.
18	Reserved		Reads return 0 and writes have no effect.
17–14	Reserved		Reads return 0 and writes have no effect.
13	SET ID INT LVL		Set ID interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
12–10	Reserved		Reads return 0 and writes have no effect.
9	SET RX INT LVL		Set receiver interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

Table 16-12. SCI Set Interrupt Level Register (SCISETINTLVL) Field Description (Continued)

Bit	Name	Value	Description
8	SET TX INT LVL		Set transmitter interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
7	SET TOA3WUS INT LVL		Set timeout after three wakeup signals interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
6	SET TOAWUS INT LVL		Set timeout after wakeup signal interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
5	Reserved		Reads return 0 and writes have no effect.
4	SET TIMEOUT INT LVL		Set timeout interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
3–2	Reserved		Reads return 0 and writes have no effect.
1	SET WAKEUP INT LVL		Set wake-up interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.
0	SET BRKDT INT		Set break-detect interrupt level. This bit is effective in SCI-compatible mode only
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read or write:* The interrupt level is mapped to the INT1 line.

16.6.7 SCI Clear Interrupt Level Register (SCICLEARINTLVL)

Figure 16-28 and Table 16-13 illustrate this register.

Figure 16-28. SCI Clear Interrupt Level Register (SCICLEARINTLVL) [offset =18h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR BE INT LVL	CLR PBE INT LVL	CLR CE INT LVL	CLR ISFE INT LVL	CLR NRE INT LVL	CLR FE INT LVL	CLR OE INT LVL	CLR PE INT LVL	Reserved							
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0	R-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CLR IDIX INT LVL	Reserved			CLR IDRX INT LVL	CLR TX INT LVL	CLR TOA3WUS INT LVL	CLR TOAWUS INT LVL	Reserved	CLR TIME-OUT INT LVL	Reserved		CLR WAKEUP INT LVL	CLR BRKDT INT LVL
R-0		R/WL-0	R-0			R/W-0	R/W-0	R/WL-0	R/WL-0	R-0	R/WL-0	R-0		R/W-0	R/WC-0

R = Read in all modes; W = Write in all modes; WC = Write in sci-compatible mode only; WL = Write in LIN mode only; -*n* = Value after reset

.

Table 16-13. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Description

Bit	Name	Value	Description
31	CLR BE INT LVL		Clear bit error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
30	CLR PBE INT LVL		Clear physical bus error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
29	CLR CE INT LVL		Clear checksum-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

Table 16-13. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Description (Continued)

Bit	Name	Value	Description
27	CLR NRE INT LVL		Clear no-response-error interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
26	CLR FE INT LVL		Clear framing-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
25	CLR OE INT LVL		Clear overrun-error interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
24	CLR PE INT LVL		Clear parity interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
23–19	Reserved		Reads return 0 and writes have no effect.
18	Reserved		Reads return 0 and writes have no effect.
17–14	Reserved		Reads return 0 and writes have no effect.
13	CLR ID INT LVL		Clear ID interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
12–10	Reserved		Reads return 0 and writes have no effect.

Table 16-13. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Description (Continued)

Bit	Name	Value	Description
9	CLR RX INT LVL		Clear receiver interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
8	CLR TX INT LVL		Clear transmitter interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
7	CLR TOA3WUS INT LVL		Clear timeout after three wakeup signals interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
6	CLR TOAWUS INT LVL		Clear timeout after wakeup signal interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
5	Reserved		Reads return 0 and writes have no effect.
4	CLR TIMEOUT INT LVL		Clear timeout interrupt level. This bit is effective in LIN mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.
3–2	Reserved		Reads return 0 and writes have no effect.
1	CLR WAKEUP INT LVL		Clear wake-up interrupt level. This bit is effective in LIN or SCI-compatible mode.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

Table 16-13. SCI Clear Interrupt Level Register (SCICLEARINTLVL) Field Description (Continued)

Bit	Name	Value	Description
0	CLR BRKDT INT		Clear break-detect interrupt level. This bit is effective in SCI-compatible mode only.
		0	*Read:* The interrupt level is mapped to the INT0 line. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The interrupt level is mapped to the INT1 line. *Write:* The interrupt level is mapped to the INT0 line.

16.6.8 SCI Flags Register (SCIFLR)

Figure 16-29 and Table 16-14 illustrate this register.

Figure 16-29. SCI Flags Register (SCIFLR) [offset = 1Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BE	PBE	CE	ISFE	NRE	FE	OE	PE	Reserved							
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/WL-0	R/W-0	R/W-0	R/W-0	R-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	ID RX	ID TX	RX WAKE	TX EMPTY	TX WAKE	RX RDY	TX RDY	TOA3 WUS	TOA WUS	Reserved	TIME-OUT	BUSY	IDLE	WAKE UP	BRKDT
R-0	R/WL-0	R/WL-0	R/WC-0	R/W-1	R/WC-0	R/W-0	R/W-1	R/WL-0	R/WL-0	R-0	R/WL-0	R/W-0	R-0	R/WL-0	R/WC-0

R = Read in all modes; W = Write in all modes; WC = Write in sci-compatible mode only; WL = Write in LIN mode only; -*n* = Value after reset

Table 16-14. SCI Flags Register (SCIFLR) Field Description

Bit	Name	Value	Description
31	BE		Bit error flag. This bit is effective in LIN mode only. This bit is set when a bit error has occurred. This is detected by the bit monitor in TED. See Section 16.2.7 for more information. The bit error flag is cleared by any of the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • On reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.
30	PBE		Physical bus error flag. This bit is effective in LIN mode only. This bit is set when a physical bus error has been detected by the bit monitor in TED. See Section 16.2.7 for more information. The physical bus error flag is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • On reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
29	CE		Checksum error flag. This bit is effective in LIN mode only. This bit is set when a checksum error has been detected by a receiving node. This error is detected by the TED logic. See Section 16.2.7 for more information. The type of checksum to be used depends on the CTYPE bit in SCIGCR1. The checksum error flag is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No error has been detected since this bit was last cleared. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An error has been detected since this bit was last cleared. *Write:* The bit is cleared to 0.
28	ISFE		Inconsistent synch field error flag. This bit is effective in LIN mode only. This bit is set when an inconsistent synch field error has been detected by the synchronizer during header reception. See Section 16.2.4.2 for more information.The inconsistent synch field error flag is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No inconsistent synch field error has been detected. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An inconsistent synch field error has been detected. *Write:* This bit is cleared to 0.
27	NRE		No-response error flag. This bit is effective in LIN mode only. This bit is set when there is no response to a master's header completed within TFRAME_MAX. This timeout period is applied for message frames of known length (identifiers 0 to 61). This error is detected by the synchronizer. See Section 16.2.6 for more information. The no-response error flag is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No no-response error has been detected since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A no-response error has been detected. *Write:* The bit is cleared to 0.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
26	FE		Framing error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when an expected stop bit is not found. In SCI compatibility mode, only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. Detection of a framing error causes the SCI/BLIN to generate an error interrupt if the SET FE INT bit is set in the register SCISETINT (Section 16.6.4). The framing error flag is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 • Reception of a new character/frame, depending on whether the module is in SCI compatible or LIN mode In multibuffer mode the frame is defined in the SCIFORMAT register (Section 16.6.11).
		0	*Read:* No framing error has been detected since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A framing error has been detected. *Write:* The bit is cleared to 0.
25	OE		Overrun error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when the transfer of data from SCIRXSHF to SCIRD overwrites unread data already in SCIRD or the RDy buffers in LINRD0 and LINRD1 (Section 16.6.23 and Section 16.6.24). Detection of an overrun error causes the LIN to generate an error interrupt if the SET OE INT bit = 1 (Section 16.6.4). The OE flag is reset by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No overrun error has been detected since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* An overrun error has been detected. *Write:* The bit is cleared to 0.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
24	PE		Parity error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when a parity error is detected in the received data. In address-bit mode, the parity is calculated on the data and address bit fields of the received frame. In idle-line mode, only the data is used to calculate parity. An error is generated when a character is received with a mismatch between the number of 1s and its parity bit. If the parity function is disabled (SCIGCR[2] = 0), the PE flag is disabled and read as 0. Detection of a parity error causes the LIN to generate an error interrupt if the SET PE INT bit = 1 (Section 16.6.4). The PE bit is reset by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reception of a new character or frame, depending on whether the module is in SCI compatible or LIN mode, respectively. • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No parity error has been detected since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A parity error has been detected. *Write:* The bit is cleared to 0.
23–15	Reserved		Reads return 0 and writes have no effect.
14	ID RX Flag		Identifier on receive flag. This bit is effective in LIN mode only. This flag is set once an identifier is received with an receive match and no ID-parity error. See Section 16.2.8 for more details. This flag indicates that a new valid identifier has been received on an RX match. This bit is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the LINID register • Reception of a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No valid ID has been received since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A valid ID RX has been received in LINID[23:16] on an RX match. *Write:* This bit is cleared to 0.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
13	ID TX Flag		Identifier on transmit flag. This bit is effective in LIN mode only. This flag is set when an identifier is received with a transmit match and no ID-parity error. See Section 16.2.8 for more details. This flag indicates that a new valid identifier has been received on a TX match. This bit is cleared by the following: • Setting the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the LINID register • Receiving a new synch break • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No valid ID has been received since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A valid ID TX has been received in LINID[23:16] on a TX match. *Write:* This bit is cleared to 0.
12	RXWAKE		Receiver wakeup detect flag. This bit is effective in SCI-compatible mode only. The SCI sets this bit to indicate that the data currently in SCIRD is an address. RXWAKE is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Upon receipt of a data frame.
		0	The data in SCIRD is not an address.
		1	The data in SCIRD is an address.
			See the SCI document for more information on using the RXWAKE bit with sleep mode.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
11	TX EMPTY		Transmitter empty flag. The value of this flag indicates the contents of the transmitter's buffer register(s) (SCITD/TDy) and shift register (SCITXSHF). In multibuffer mode, this flag indicates the value of the TDx registers and shift register (SCITXSHF). In non-multibuffer mode, this flag indicates the value of the LINTD0 byte and the shift register (SCITXSHF). **Note: The RESET bit, an active SW nRESET (SCIGCR1[7]) or a system reset sets this bit. This bit does not cause an interrupt request.** *SCI-compatible mode or LIN with no multibuffer*
		0	Transmitter buffer or shift register (or both) are loaded with data.
		1	Transmitter buffer and shift registers are both empty.
			In LIN mode using multibuffer mode:
		0	Multibuffer or shift register (or all) are loaded with data
		1	Multibuffer and shift registers are all empty.
10	TXWAKE		SCI transmitter wakeup method select. This bit is effective in SCI-compatible mode only. The TXWAKE bit controls whether the data in SCITD should be sent as an address or data frame using multi-processor communication format. This bit is set to 1 or 0 by software before a byte is written to SCITD and is cleared by the SCI when data is transferred from SCITD to SCITXSHF or by a system reset. **Note: TXWAKE is not cleared by the SW nRESET bit (SCIGCR1[7];** Section 16.6.2**).** *Address-bit mode*
		0	Frame to be transmitted will be data (address bit = 0).
		1	Frame to be transmitted will be an address (address bit = 1).
			Idle-line mode
		0	The frame to be transmitted will be data.
		1	The following frame to be transmitted will be an address (writing a 1 to this bit followed by writing dummy data to the SCITD will result in a idle period of 11 bit periods before the next frame is transmitted). See the SCI document for more information on using the TXWAKE bit in the available communication modes.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
9	RXRDY		Receiver ready flag. In SCI-compatible mode, the receiver sets this bit to indicate that the SCIRD contains new data and is ready to be read by the CPU. In *LIN mode,* RXRDY is set once a valid frame is received in multibuffer mode, a valid frame being a message frame received with no errors. In *non-multibuffer mode,* RXRDY is set for each received byte and will be set for the last byte of the frame if there are no errors. The SCI/BLIN generates a receive interrupt when RXRDY flag bit is set if the interrupt-enable bit is set (SCISETINT[9]; Section 16.6.4). RXRDY is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the SCIRD register in compatibility mode • Reading the last data byte RDy of the response in LIN mode- **Note: The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.**
		0	*Read:* No new data is in SCIRD. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* New data is ready to be read from SCIRD. *Write:* The bit is cleared to 0.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
8	TXRDY		Transmitter buffer register ready flag. When set, this bit indicates that the transmit buffer(s) register(s) (SCITD in compatibility mode and LINTD0/LINTD1 in multibuffer mode) are ready to get another character from a CPU write. *In SCI-compatible mode,* writing data to SCITD automatically clears this bit. In *LIN mode,* this bit is cleared once byte 0 (TD0) is written to LINTD0. This bit is set after the data of the TX buffer is shifted into the SCITXSHF register. The SCI/BLIN asserts a transmit interrupt after data is copied to the TX shift register SCITXSHF, if the interrupt TXINT (SCISETINT[8]; Section 16.6.4) is set. TXRDY is also set to 1 either by setting of the RESET bit, enabling SW nRST (SCIGCR1[7]; Section 16.6.2) or by a system reset. **Note: The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.** **Note: The transmit interrupt request can be eliminated until the next series of data written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLEARINT register or by disabling the transmitter via the TXENA bit (SCIGCR1[25]=0).** *SCI-compatible mode* 0 SCITD is full. 1 SCITD is ready to receive the next character. *LIN mode* 0 The multibuffers are full. 1 The multibuffers are ready to receive the next character(s). For more information on transmit interrupt handling, see the SCI document for compatibility mode and Section 16.2.8 for LIN mode.
7	TOA3WUS		Timeout after three wakeup signals flag. This bit is effective in LIN mode only. This flag is set if there is no synch break received after three wakeup signals and a period of 1.5 seconds has passed. Such expiration time is used before issuing another round of wakeup signals. This bit is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 16.4.3 for more information. 0 *Read:* No timeout occurs after three wakeup signals. *Write:* Writing a 0 to this bit has no effect.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
		1	*Read:* Timeout will occur after 3 wakeup signals and 1.5 seconds time. *Write:* The bit will be cleared to 0.
6	TOAWUS		Timeout after wakeup signal flag. This bit is effective in LIN mode only. This bit is set if there is no synch break received after a wakeup signal has been sent. A minimum of 150 ms expiration time is used before issuing another wakeup signal. This bit is cleared by the following: • Setting the SW nRST bit • Setting of the RESET bit • A system reset occurring • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 16.4.3 for more information.
		0	*Read:* No timeout occurs after one wakeup signal (150 ms). *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A timeout occur after one wakeup signal. *Write:* The bit will be cleared to 0.
5	Reserved		Reads return 0 and writes have no effect.
4	TIMEOUT		LIN bus idle timeout flag. This bit is effective in LIN mode only. This bit is set if there is no LIN bus activity for at least 4 s. LIN bus activity is a transition from recessive to dominant. This bit is cleared by the following: • Setting the SW nRST bit • Setting of the RESET bit • A system reset occurring • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 See Section 16.2.6 for more information.
		0	*Read:* No bus idle has been detected since this bit was cleared. *Write:* A write of 0 to this bit has no effect.
		1	*Read:* A LIN bus idle has been detected. *Write:* The bit is cleared to 0.
3	BUSY		Bus busy flag. This bit is effective in LIN mode and SCI-compatible mode.This bit indicates whether the receiver is in the process of receiving a frame. As soon as the receiver detects the beginning of a start bit, the BUSY bit is set to 1. When the reception of a frame is complete, the SCI/BLIN clears the BUSY bit. If SET WAKEUP INT is set and power down is requested while this bit is set, the SCI/BLIN automatically prevents low-power mode from being entered and generates wakeup interrupt. The BUSY bit is controlled directly by the SCI/BLIN receiver, but it is cleared by the RESET bit, by an active SW nRST or by a system reset.
		0	The receiver is not currently receiving a frame.

Table 16-14. SCI Flags Register (SCIFLR) Field Description (Continued)

Bit	Name	Value	Description
		1	The receiver is currently receiving a frame.
2	IDLE		SCI receiver in idle state. This bit is effective in SCI-compatible mode only. While this bit is set, the SCI looks for an idle period to resynchronize itself with the bit stream. The receiver does not receive any data while the bit is set. The bus must be idle for 11 bit periods to clear this bit. The SCI enters the idle state if one of the following events occurs: • A system reset • An SCI software reset • A power down • The RX pin is configured as a general I/O pin
		0	The idle period has been detected; the SCI is ready to receive.
		1	The idle period has not been detected; the SCI will not receive any data.
1	WAKEUP		Wake-up flag. This bit is effective in LIN mode only. This bit is set by the SCI/BLIN when receiver or transmitter activity has taken the module out of power-down mode. An interrupt is generated if the SET WAKEUP INT bit (SCISETINT[2]) is set. It is cleared by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1 For compatibility mode, see the SCI document for more information on low-power mode.
		0	*Read:* The module will not wake up from power-down mode. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* Wake up from power-down mode. *Write:* The bit is cleared to 0.
0	BRKDT		SCI break-detect flag. This bit is effective in SCI-compatible mode only. This bit is set when the SCI detects a break condition on the SCIRX pin. A break condition occurs when the SCIRX pin remains continuously low for at least 10 bits after a missing first stop bit, that is, after a framing error. Detection of a break condition causes the SCI to generate an error interrupt if the BRKDT INT ENA bit is set. The BRKDT bit is reset by the following: • Setting of the SW nRST bit • Setting of the RESET bit • A system reset • Writing a 1 to this bit • Reading the corresponding interrupt offset in SCIINTVECT0/1
		0	*Read:* No break condition has been detected since the last clear. *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* A break condition has been detected. *Write:* Writing a 1 to this bit clears it to 0.

16.6.9 SCI Interrupt Vector Offset 0 (SCIINTVECT0)

Figure 16-30 and Table 16-15 illustrate this register.

Figure 16-30. SCI Interrupt Vector Offset 0 (SCIINTVECT0) [offset = 20h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTVECT0[4:0]				
R-0											R-0				

R = Read in all modes; -*n* = Value after reset

.

Table 16-15. SCI Interrupt Vector Offset 0 (SCIINTVECT0) Field Description

Bit	Name	Value	Description
31–5	Reserved		Reads return 0 and writes have no effect.
4–0	INTVECT0[4:0]	0–1Fh	Interrupt vector offset for INT0. This register indicates the offset for interrupt line INT0. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 16-4 for a list of the interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

16.6.10 SCI Interrupt Vector Offset 1 (SCIINTVECT1)

Figure 16-31 and Table 16-16 illustrate this register.

Figure 16-31. SCI Interrupt Vector Offset 1 (SCIINTVECT1) [offset =24h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTVECT1[4:0]				
R-0											R-0				

R = Read in all modes; -*n* = Value after reset

.

Table 16-16. SCI Interrupt Vector Offset 1 (SCIINTVECT1) Field Description

Bit	Name	Value	Description
31–5	Reserved		Reads return 0 and writes have no effect.
5–0	INTVECT1[4:0]	0–1Fh	Interrupt vector offset for INT1. This register indicates the offset for interrupt line INT1. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag in SCIFLR corresponding to the offset that was read. See Table 16-4 for list of interrupts. **Note: The flags for the receive (SCIFLR[9]) and the transmit (SCIFLR[8]) interrupt cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register).**

16.6.11 SCI Format Control Register (SCIFORMAT)

Figure 16-32 and Table 16-17 illustrate this register.

Figure 16-32. SCI Format Control Register (SCIFORMAT) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													LENGTH[2:0]		
R-0													R/W-0		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CHAR[2:0]		
R-0													R/WC-0		

R = Read in all modes; W = Write in all modes; WC = Write in SCI-compatible mode only; -*n* = Value after reset

.

Table 16-17. SCI Format Control Register (SCIFORMAT) Field Description

Bit	Name	Value	Description
31–19	Reserved		Reads return 0 and writes have no effect.
18–16	LENGTH[2:0]	0–3h	Frame length control bits. In *LIN mode*, these bits indicate the number of bytes in the response field from 1 to 8 bytes. In *buffered SCI mode*, these bits indicate the number of characters. When these bits are used to indicate LIN response length (SCIGCR1[0] = 1), then when there is an ID RX match, this value should be updated with the expected length of the response. In buffered SCI mode, these bits indicate the transmitter/receiver format for the number of characters: 1 to 8. There can be up to eight characters with eight bits each.
		000	The response field has 1 byte/character.
		001	The response field has 2 bytes/characters.
		010	The response field has 3 bytes/characters.
		011	The response field has 4 bytes/characters.
		100	The response field has 5 bytes/characters.
		101	The response field has 6 bytes/characters.
		110	The response field has 7 bytes/characters.
		111	The response field has 8 bytes/characters.
15–3	Reserved		Reads return 0 and writes have no effect.

Table 16-17. SCI Format Control Register (SCIFORMAT) Field Description (Continued)

Bit	Name	Value	Description
2–0	CHAR[2:0]	0–7h	Character length control bits. These bits are effective in SCI compatible or buffered SCI modes only. These bits set the SCI character length from 1 to 8 bits. **Note: In compatibility mode or buffered SCI mode, when data of fewer than eight bits in length is received, it is left justified in SCIRD/RDy and padded with trailing zeros. Data read from the SCIRD should be shifted by software to make the received data right justified.** **Note: Data written to the SCITD should be right justified but does not need to be padded with leading zeros.**
		000	The character is 1 bit long.
		001	The character is 2 bits long.
		010	The character is 3 bits long.
		011	The character is 4 bits long.
		100	The character is 5 bits long.
		101	The character is 6 bits long.
		110	The character is 7 bits long.
		111	The character is 8 bits long.

16.6.12 Baud Rate Selection Register (BRS)

This section describes the baud rate selection register. Figure 16-33 and Table 16-18 illustrate this register.

Figure 16-33. Baud Rate Selection Register (BRS) [offset = 2Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	U[2:0]			M[3:0]				PRESCALER P [23:16]							
R-0	R/W-0			R/W-0				R/W-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRESCALER P [15:0]															
R/W-0															

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-18. Baud Rate Selection Register (BRS) Field Description

Bit	Name	Value	Description
31	Reserved		This bit is always read as 0. Writes have no effect.
30–28	U[2:0]	0–2h	SCI/BLIN super fractional divider selection. These bits are effective in LIN or SCI asynchronous mode. These bits are an additional fractional part for the baud rate specification. These bits allow a superfine tuning of the fractional baud rate with seven more intermediate values for each of the M fractional divider values. See Section 16.2.3.1 for more details.
27-24	M[3:0]	0–3h	SCI/BLIN 4-bit fractional divider selection. These bits are effective in LIN or SCI asynchronous mode. These bits are used to select a baud rate for the SCI/BLIN module, and they are a fractional part for the baud rate specification. The M divider allows fine-tuning of the baud rate over the P prescaler with 15 additional intermediate values for each of the P integer values. See Section 16.2.3.1 for more details.

Table 16-18. Baud Rate Selection Register (BRS) Field Description (Continued)

Bit	Name	Value	Description
23–0	PRESCALER P [23:0]	0–FF FFFFh	These bits are used to select a baud rate for the SCI/BLIN module. These bits are effective in LIN mode and SCI compatibility (asynchronous/iso-synchronous) mode • In SCI-compatible mode, if an external clock source is selected (via the CLOCK bit, SCIGCR1[5] in Section 16.6.2), then the SCI accepts an external clock on the SCICLK pin. • If an internal clock source is selected and both the CLK FUNC and CLK DIR bits (Section 16.6.14) are set, the SCICLK pin becomes the serial clock output pin. • If an internal clock is selected and CLK FUNC is set but CLK DIR is cleared, then the SCI uses an internally generated clock but does not output this clock signal on the SCICLK pin. The SCI/BLIN has an internally generated serial clock determined by the VCLK and the prescalers P and M in this register. The LIN uses the 24-bit integer prescaler P value of this register to select one of over 16,700,000 available baud rates. The additional 4-bit fractional divider M refines the baudrate selection PRESCALER[27:24]. **NOTE: In LIN mode, ONLY the asynchronous mode and baudrate values are used.** The baud rate can be calculated using the following formulas: *Asynchronous baud value* = $\left\lfloor \frac{\text{VCLK Frequency}}{16\left(P + 1 + \frac{M}{16}\right)} \right\rfloor$ *Isosynchronous baud value* = $\left(\frac{\text{VCLK Frequency}}{P + 1}\right)$ For P = 0, *Asynchronous baud value* = $\left(\frac{\text{VCLK Frequency}}{32}\right)$ *Isosynchronous baud value* = $\left(\frac{\text{VCLK Frequency}}{2}\right)$ Table 16-19 contains comparative baud values for different P values, with VCLK = 50 MHz, for asynchronous mode. Table 16-20 contains comparative baud values for different baud register values, with VCLK = 25 MHz, for the isosynchronous mode.

Table 16-19. Comparative Baud Values for Different P Values, Asynchronous Mode(1)(2)

24-Bit Register Value		Baud Selected		Percent Error
Decimal	Hex	Ideal	Actual	
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

1 VCLK = 50 MHz
2 Values are in decimal except for column 2.

Table 16-20. Comparative Baud Values for Different P Values, Isosynchronous Mode(1)(2)(3)

24-Bit PRESCALER Value		Baud Selected		Percent Error
Decimal	Hex	Ideal	Actual	
216	0000D8	115200	115207	0.01
433	0001B1	57600	57604	0.01
650	00028A	38400	38402	0.01
1301	000515	19200	19201	0.01
2403	000963	10400	10399	0.01
2603	000A2B	9600	9601	0.01
3199	000C7F	7812.5	7812.5	0.00
5207	001457	4,800	4,800	0.00
124999	01E847	200	200	0.00
4999999	4C4B3F	5	5	0.00

1 VCLK = 25 MHz
2 Values are in decimal except for column 2.
3 An external SCI clock should not exceed VCLK/8

16.6.13 SCI Data Buffers (SCIED, SCIRD, SCITD)

The SCI has three addressable registers in which transmit and receive data is stored.

16.6.13.1 Receiver Emulation Data Buffer (SCIED)

The SCIED register is addressed at a location different from SCIRD, but is physically the same register. Figure 16-34 and Table 16-21 illustrate this register.

Figure 16-34. Receiver Emulation Data Buffer (SCIED) [offset = 30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ED[7:0]							
R-0								R-0							

R = Read in all modes; W = Write in SCI-compatible mode only; -*n* = Value after reset

.

Table 16-21. Receiver Emulation Data Buffer (SCIED) Field Description

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	ED[7:0]	0–FFh	Emulator data. This bit is effective in SCI-compatible mode only. Reading SCIED[7:0] does not clear the RXRDY flag. This register should be used only by an emulator that must continually read the data buffer without affecting the RXRDY flag.

16.6.13.2 Receiver Data Buffer (SCIRD)

This register provides a location for the receiver data. Figure 16-35 and Table 16-22 illustrate this register.

Figure 16-35. Receiver Data Buffer (SCIRD) [offset = 34h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RD[7:0]							
R-0								R-0							

R = Read in all modes; W = Write in SCI-compatible mode only; *-n* = Value after reset

.

Table 16-22. Receiver Data Buffer (SCIRD) Field Description

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	RD[7:0]		Receiver data. This bit is effective in SCI-compatible mode only. When a frame has been completely received, the data in the frame is transferred from the receiver shift register SCIRXSHF to this register. As this transfer occurs, the RXRDY flag is set and a receive interrupt is generated if SET RX INT (SCISETINT[9]; Section 16.6.4) is set. **Note: When the data is read from SCIRD, the RXRDY flag is automatically cleared.**

Note:
When the SCI receives data that is fewer than eight bits in length, it loads the data into this register in a left-justified format padded with trailing zeros. Therefore, the user software should perform a logical shift on the data by the correct number of positions to make it right justified.

16.6.13.3 Transmit Data Buffer Register (SCITD)

Data to be transmitted is written to the SCITD register. Figure 16-36 and Table 16-23 illustrate this register.

Figure 16-36. Transmit Data Buffer Register (SCITD) [offset = 38h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TD[7:0]							
R-0								R/W-0							

R = Read in all modes; W = Write in SCI-compatible mode only; *-n* = Value after reset

.

Table 16-23. Transmit Data Buffer Register (SCITD) Field Description

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	TD[7:0]	0–FFh	Transmit data. This bit is effective in SCI-compatible mode only. The transfer of data from this register to the transmit shift register SCITXSHF sets the TXRDY flag (SCIFLR[8]; Section 16.6.8), which indicates that SCITD is ready to be loaded with another byte of data. **Note: If TX INT ENA (register SCISETINT[8]; Section 16.6.4) is set, this data transfer also causes an interrupt .**

Note:
Data written to the SCITD register that is fewer than eight bits long must be right justified, but it need not be padded with leading zeros.

16.6.14 SCI Pin I/O Control Register 0 (SCIPIO0)

Figure 16-37 and Table 16-24 illustrate this register.

Figure 16-37. SCI Pin I/O Control Register 0 (SCIPIO0) [offset = 3Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX FUNC	RX FUNC	Reserved
R-0													R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-24. SCI Pin I/O Control Register 0 (SCIPIO0) Field Description

Bit	Name	Value	Description
31–3	Reserved		This bit is always read as 0. Writes have no effect.
2	TX FUNC		Transfer function. This bit is effective in LIN or SCI-compatible mode. This bit defines the function of pin SCITX.
		0	SCITX is a general-purpose digital I/O pin.
		1	SCITX is the SCI/BLIN transmit pin.
1	RX FUNC		Receive function. This bit is effective in LIN or SCI-compatible mode. This bit defines the function of pin SCIRX.
		0	SCIRX is a general-purpose digital I/O pin.
		1	SCIRX is the SCI/BLIN receive pin.
0	Reserved		This bit is always read as 0. Writes have no effect.

16.6.15 SCI Pin I/O Control Register 1 (SCIPIO1)

Figure 16-38 and Table 16-25 illustrate this register.

Figure 16-38. SCI Pin I/O Control Register 1 (SCIPIO1) [offset = 40h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX DIR	RX DIR	Reserved
R-0													R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-25. SCI Pin I/O Control Register 1 (SCIPIO1)

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX DIR		Transmit data direction. This bit is effective in LIN or SCI-compatible mode. This bit determines the data direction on the SCITX pin if it is configured with general-purpose I/O functionality (TX FUNC = 0; Section 16.6.14). See Table 16-26 for the SCITX pin control with this bit and others.
		0	SCITX is a general-purpose input pin.
		1	SCITX is a general-purpose output pin.
1	RX DIR		Receive data direction. This bit is effective in LIN or SCI-compatible mode. This bit determines the data direction on the SCIRX pin if it is configured with general-purpose I/O functionality (RX FUNC = 0). See Table 16-27 for the SCIRX pin control with this bit and others.
		0	SCIRX is a general-purpose input pin.
		1	SCIRX is a general-purpose output pin.
0	Reserved		This bit is always read as 0. Writes have no effect.

Table 16-26. SCITX Pin Control

Function	TX IN[1]	TX OUT	TX FUNC	TX DIR
SCITX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

1 TX DATA IN is a read-only bit. Its value always reflects the level of the SCITX pin.

Table 16-27. SCIRX Pin Control

Function	RX IN[1]	RX OUT	RX FUNC	RX DIR
SCIRX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

1 RX DATA IN is a read-only bit. Its value always reflects the level of the SCIRX pin.

16.6.16 SCI Pin I/O Control Register 2 (SCIPIO2)

Figure 16-39 and Table 16-28 illustrate this register.

Figure 16-39. SCI Pin I/O Control Register 2 (SCIPIO2) [offset = 44h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX IN	RX IN	Reserved
R-0													R-X	R-X	R-X

R = Read in all modes; -*n* = Value after reset; -*x* = Indeterminate

.

Table 16-28. SCI Pin I/O Control Register 2 (SCIPIO2) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX IN		Transmit data in. This bit is effective in LIN or SCI-compatible mode. This bit contains the current value on the SCITX pin.
		0	The SCITX pin is at logic low (0).
		1	The SCITX pin is at logic high (1).
1	RX IN		Receive data in. This bit is effective in LIN or SCI-compatible mode. This bit contains the current value on the SCIRX pin.
		0	The SCIRX pin is at logic low (0).
		1	The SCIRX pin is at logic high (1).
0	Reserved		Reads return 0 and writes have no effect.

16.6.17 SCI Pin I/O Control Register 3 (SCIPIO3)

Figure 16-40 and Table 16-29 illustrate this register.

Figure 16-40. SCI Pin I/O Control Register 3 (SCIPIO3) [offset =48h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX OUT	RX OUT	Reserved
R-0													R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-29. SCI Pin I/O Control Register 3 (SCIPIO3) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX OUT		Transmit data out. This bit is effective in LIN or SCI-compatible mode. This pin specifies the logic to be output on pin SCITX if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.) See Table 16-26 for an explanation of this bit's effect in combination with other bits.
		0	The output on the SCITX is at logic low (0).
		1	The output on the SCITX pin is at logic high (1).
1	RX OUT		Receive data out. This bit is effective in LIN or SCI-compatible mode. This bit specifies the logic to be output on pin SCIRX if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.) See Table 16-27 for an explanation of this bit's effect in combination with the other bits.
		0	The output on the SCIRX pin is at logic low (0).
		1	The output on the SCIRX pin is at logic high (1).
0	Reserved		Reads return 0 and writes have no effect.

16.6.18 SCI Pin I/O Control Register 4 (SCIPIO4)

Figure 16-41 and Table 16-30 illustrate this register.

Figure 16-41. SCI Pin I/O Control Register 4 (SCIPIO4) [offset =4Ch] .

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX SET	RX SET	Reserved
R-0													R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

Table 16-30. SCI Pin I/O Control Register 4 (SCIPIO4) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX SET		Transmit data set. This bit is effective in LIN or SCI-compatible mode. This bit sets the logic to be output on pin SCITX if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.) See Table 16-26 for an explanation of this bit's effect in combination with other bits.
		0	*Read:* The output on SCITX is at logic low (0). *Write:* Writing a 0 to this bit has no effect.
		1	*Read and write:* The output on SCITX is at logic high (1).
1	RX SET		Receive data set. This bit is effective in LIN or SCI-compatible mode. This bit sets the data to be output on pin SCIRX if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.) See Table 16-27 for an explanation of this bit's effect in combination with the other bits.
		0	*Read:* The output on SCIRX is at logic low (0). *Write:* Writing a 0 to this bit has no effect.
		1	*Read and write:* The output on SCIRX is at logic high (1).
0	Reserved		Reads return 0 and writes have no effect.

16.6.19 SCI Pin I/O Control Register 5 (SCIPIO5)

Figure 16-42 and Table 16-31 illustrate this register.

Figure 16-42. SCI Pin I/O Control Register 5 (SCIPIO5) [offset = 50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX CLR	RX CLR	Reserved
R-0													R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-31. SCI Pin I/O Control Register 5 (SCIPIO5) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX CLR		Transmit data clear. This bit is effective in LIN or SCI-compatible mode. This bit clears the logic to be output on pin SCITX if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DIR = 1 (SCITX pin is a general-purpose output.)
		0	*Read:* The output on SCITX is at logic low (0). *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The output on SCITX is at logic high (1). *Write:* The output on SCITX is at logic low (0).
1	RX CLR		Receive data clear. This bit is effective in LIN or SCI-compatible mode. This bit clears the logic to be output on pin SCIRX if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DIR = 1 (SCIRX pin is a general-purpose output.)
		0	*Read:* The output on SCIRX is at logic low (0). *Write:* Writing a 0 to this bit has no effect.
		1	*Read:* The output on SCIRX is at logic high (1). *Write:* The output on SCIRX is at logic low (0).
0	Reserved		Reads return 0 and writes have no effect.

16.6.20 SCI Pin I/O Control Register 7 (SCIPIO7)

Figure 16-43 and Table 16-32 illustrate this register.

Figure 16-43. SCI Pin I/O Control Register 7 (SCIPIO7) [offset = 58h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX PD	RX PD	Reserved
R-0													R/W-n	R/W-n	R/W-n

R = Read in all modes; W = Write in all modes; -*n* = Value after reset, Refer to the Terminal Functions in the device datasheet for default pin settings.

.

Table 16-32. SCI Pin I/O Control Register 7 (SCIPIO7) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX PD		Transmit pin pull control disable. This bit is effective in LIN or SCI-compatible mode. This bit disables pull control capability on the input pin SCITX.
		0	The pull control on the SCITX pin is enabled.
		1	The pull control on the SCITX pin is disabled.
1	RX PD		Receive pin pull control disable. This bit is effective in LIN or SCI-compatible mode. This bit disables pull control capability on the input pin SCIRX.
		0	Pull control on the SCIRX pin is enabled.
		1	Pull control on the SCIRX pin is disabled.
0	Reserved		Reads return 0 and writes have no effect.

16.6.21 SCI Pin I/O Control Register 8 (SCIPIO8)

Figure 16-44 and Table 16-33 illustrate this register. This register controls the input buffers of the pins when the pull controls of the pins are **disabled** (xxPD of SCIPIO7 = 1). If the pull control of the pins are enabled, the input buffers are enabled.

Figure 16-44. SCI Pin I/O Control Register 8 (SCIPIO8) [offset = 5Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX PSL	RX PSL	Reserved
R-0													R/W-*n*	R/W-*n*	R/W-*n*

R = Read in all modes; W = Write in all modes; -*n* = Value after reset(1)

1 The reset values of the PSL bits is device dependent.

.

Table 16-33. SCI Pin I/O Control Register 8 (SCIPIO8) Field Description

Bit	Name	Value	Description
31–3	Reserved		Reads return 0 and writes have no effect.
2	TX PSL		TX pin pull select.
		0	Input buffer for TX is disabled if TXPD = 1
		1	Input buffer for TX is enabled if TXPD = 1
1	RX PSL		RX pin pull select.
		0	Input buffer for RX is disabled if RXPD = 1
		1	Input buffer for RX is enabled if RXPD = 1
0	Reserved		Reads return 0 and writes have no effect.

16.6.22 LIN Compare Register (LINCOMPARE)

Figure 16-45 and Table 16-34 illustrate this register.

Figure 16-45. LIN Compare Register (LINCOMPARE) [offset = 60h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SDEL[1:0]		Reserved					SBREAK[2:0]		
R-0						RWP-0		R-0					RWP-0		

R = Read in all modes; WL = Write in LIN mode only; RWP = Read/Write in privileged mode only; *-n* = Value after reset

.

Table 16-34. LIN Compare Register (LINCOMPARE) Field Description

Bit	Name	Value	Description
31–10	Reserved		Reads return 0 and writes have no effect.
9–8	SDEL[1:0]		2-bit synch delimiter compare. These bits are effective in LIN mode only. These bits are used to configure the number of T_{bit} for the synch delimiter in the synch field. The default value is 0x00. The time delay value for the synchronization delimiter is $T_{SDEL} = (SDEL + 1)T_{bit}$
		00	The synch delimiter has 1 T_{bit}.
		01	The synch delimiter has 2 T_{bit}.
		10	The synch delimiter has 3 T_{bit}.
		11	The synch delimiter has 4 T_{bit}.
7-3	Reserved		Reads return 0 and writes have no effect.

Table 16-34. LIN Compare Register (LINCOMPARE) Field Description (Continued)

Bit	Name	Value	Description
2–0	SBREAK[2:0]		Synch break extend. These bits are effective in LIN mode only. These bits are used to configure the number of T_{bit} for the synch break to extend the minimum 13 T_{bit} in the synch field to a maximum of 20 T_{bit}. **Note: The default value is 0x0, which adds nothing to the automatically generated SYNCH BREAK.** The time delay for the SYNCH BREAK is $T_{SYNBRK} = 13T_{bit} + SBREAK \times T_{bit}$
		000	The synch break has no additional T_{bit}.
		001	The synch break has 1 additional T_{bit}.
		010	The synch break has 2 additional T_{bit}.
		011	The synch break has 3 additional T_{bit}.
		100	The synch break has 4 additional T_{bit}.
		101	The synch break has 5 additional T_{bit}.
		110	The synch break has 6 additional T_{bit}.
		111	The synch break has 7 additional T_{bit}.

16.6.23 LIN Receive Buffer 0 Register (LINRD0)

Figure 16-46 and Table 16-35 illustrate this register.

Figure 16-46. LIN Receive Buffer 0 Register (LINRD0) [offset = 64h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD0[7:0]								RD1[7:0]							
R-0								R-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD2[7:0]								RD3[7:0]							
R-0								R-0							

R = Read in all modes; -*n* = Value after reset

.

Table 16-35. LIN Receive Buffer 0 Register (LINRD0) Field Description

Bit	Name	Value	Description
31–24	RD0[7:0]	0–FFh	Receive buffer 0. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy bit field (Section 16.6.23, Section 16.6.24) according to the number of bytes received. A read of this byte clears the RXDY byte. **Note: RD<x-1> is equivalent to data byte <x> of the LIN frame**
23–16	RD1[7:0]	0–FFh	Receive buffer 1. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.
15–8	RD2[7:0]	0–FFh	Receive buffer 2. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.
7–0	RD3[7:0]	0–FFh	Receive buffer 3. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received.

16.6.24 LIN Receive Buffer 1 Register (LINRD1)

Figure 16-47 and Table 16-36 illustrate this register.

Figure 16-47. LIN Receive Buffer 1 Register (RD1) [offset = 68h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD4[7:0]								RD5[7:0]							
R-0								R-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD6[7:0]								RD7[7:0]							
R-0								R-0							

R = Read in all modes; -*n* = Value after reset

.

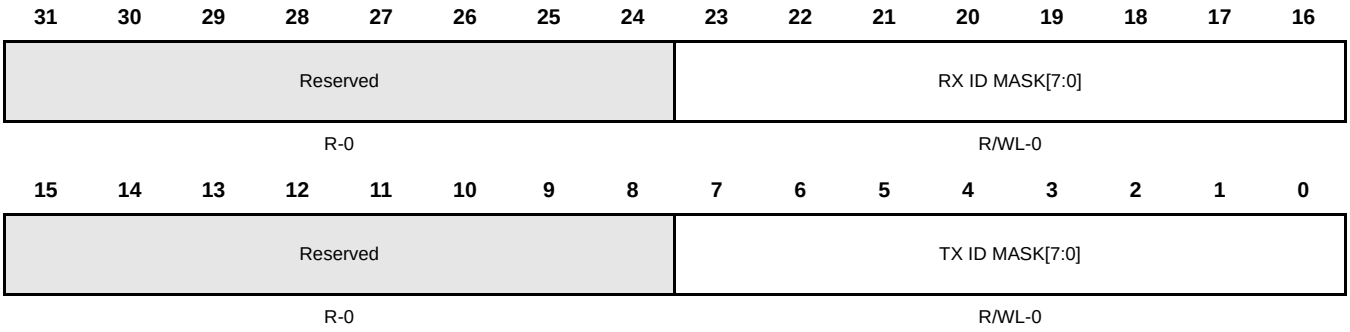
Table 16-36. LIN Receive Buffer 1 Register (RD1) Field Description

Bit	Name	Value	Description
31–24	RD4[7:0]	0–FFh	Receive buffer 4. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received. **Note: RD<x-1> is equivalent to data byte <x> of the LIN frame**
23–16	RD5[7:0]	0–FFh	Receive buffer 5. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.
15–8	RD6[7:0]	0–FFh	Receive buffer 6. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.
7–0	RD7[7:0]	0–FFh	Receive buffer 7. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding register according to the number of bytes received.

16.6.25 LIN Mask Register (LINMASK)

Figure 16-48 and Table 16-37 illustrate this register.

Figure 16-48. LIN Mask Register (LINMASK) [offset = 6Ch]



R = Read in all modes; WL = Write in LIN mode only; *-n* = Value after reset

.

Table 16-37. LIN Mask Register (LINMASK) Field Description

Bit	Name	Value	Description
31–24	Reserved		Reads of these bits return 0 and writes have no effect.
23–16	RX ID MASK[7:0]	0–FFh	Receive ID mask. These bits are effective in LIN mode only. This 8-bit mask is used for filtering an incoming ID message and comparing it to the ID-byte. A compare match of the received ID with the RX ID mask will set the ID RX flag and trigger an ID interrupt if enabled (SET RX INT; Section 16.6.6). A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that bit is filtered and therefore is not used in the compare.
15–8	Reserved		Reads of these bits return 0 and writes have no effect.
7–0	TX ID MASK[7:0]	0–FFh	Transmit ID mask. These bits are effective in LIN mode only. This 8-bit mask is used for filtering an incoming ID message and comparing it to the ID-byte. A compare match of the received ID with the TX ID mask will set the ID TX flag and trigger an ID interrupt if enabled (SET ID INT in SCISETINT; Section 16.6.6). A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that bit is filtered and therefore is not used for the compare.

Note:
For software compatibility with future LIN modules the HGEN CTRL bit (SCIGCR1[12]) must be set to 1, the RX ID MASK (LINMASK[23:16]) must be set to 0xFF and the TX ID MASK (LINMASK[7:0]) must be set to 0xFF.

16.6.26 LIN Identification Register (LINID)

Figure 16-49 and Table 16-38 illustrate this register.

Figure 16-49. LIN Identification Register (LINID) [offset = 70h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								RECEIVED ID[7:0]							
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID-SlaveTask BYTE[7:0]								ID BYTE[7:0]							
R/WL-0								R/WL-0							

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

Table 16-38. LIN Identification Register (LINID) Field Description

Bit	Name	Value	Description
31–24	Reserved		Reads return 0 and writes have no effect.
23–16	Received ID[7:0]	0–FFh	Received identification. These bits are effective in LIN mode only. This byte contains the current message identifier. During header reception the received ID is copied from the SCIRXSHF register to this byte if there is no ID-parity error and there has been an RX/TX match.
15–8	(7–0)ID-SLAVETASK BYTE[7:0]	0–FFh	ID-SlaveTask Byte. These bits are effective in LIN mode only. This field contains the identifier to which the received ID of an incoming header will be compared to decide whether a receive response, a transmit response, or no action needs to be performed by the LIN node when a header with a particular ID is received.
7–0	ID BYTE[7:0]	0–FFh	ID byte. This field is effective in LIN mode only. This byte is the LIN mode message ID. On a master node, a write to this register by the CPU initiates a header transmission. For a slave task, this byte is used for message filtering when HGENCTRL (SCIGCR1[12]; Section 16.6.2) = 0.

Note:
For software compatibility with future LIN modules the HGEN CTRL bit (SCIGCR1[12]) must be set to 1, the RX ID MASK (LINMASK[23:16]) must be set to 0xFF and the TX ID MASK (LINMASK[7:0]) must be set to 0xFF.

16.6.27 LIN Transmit Buffer 0 Register(LINTD0)

Figure 16-50 and Table 16-39 illustrate this register.

Figure 16-50. LIN Transmit Buffer 0 Register (LINTD0) [offset = 74h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TD0[7:0]								TD1[7:0]							
R/W-0								R/W-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TD2[7:0]								TD3[7:0]							
R/W-0								R/W-0							

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-39. LIN Transmit Buffer 0 Register (LINTD0) Field Description

Bit	Name	Value	Description
31-24	TD0[7:0]	0–FFh	8-Bit transmit buffer 0. Byte 0 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Once byte 0 is written in TD0 buffer, transmission will be initiated. **Note: TD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	TD1[7:0]	0–FFh	8-Bit transmit buffer 1. Byte 1 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
15-8	TD2[7:0]	0–FFh	8-Bit transmit buffer 2. Byte 2 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
7-0	TD3[7:0]	0–FFh	8-Bit transmit buffer 3. Byte 3 to be transmitted is written into this register and then copied to SCITXSHF for transmission.

16.6.28 LIN Transmit Buffer 1 Register (LINTD1)

Figure 16-51 and Table 16-40 illustrate this register.

Figure 16-51. LIN Transmit Buffer 1 Register (LINTD1) [offset = 78h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TD4[7:0]								TD5[7:0]							
R/W-0								R/W-0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TD6[7:0]								TD7[7:0]							
R/W-0								R/W-0							

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 16-40. LIN Transmit Buffer 1 Register (LINTD1) Field Description

Bit	Name	Value	Description
31-24	TD4[7:0]	0–FFh	8-Bit transmit buffer 4. Byte 4 to be transmitted is written into this register and then copied to SCITXSHF for transmission. **Note: TD<x-1> is equivalent to data byte <x> of the LIN frame.**
23-16	TD5[7:0]	0–FFh	8-Bit transmit buffer 5. Byte 5 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
15–8	TD6[7:0]	0–FFh	8-Bit transmit buffer 6. Byte 6 to be transmitted is written into this register and then copied to SCITXSHF for transmission.
7–0	TD7[7:0]	0–FFh	8-Bit transmit buffer 7. Byte 7 to be transmitted is written into this register and then copied to SCITXSHF for transmission.

16.6.29 Maximum Baud Rate Selection Register (MBRS)

Figure 16-52 and Table 16-41 illustrate this register.

Figure 16-52. Maximum Baud Rate Selection Register (MBRS) [offset = 7Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			MBR[12:0]												
R-0			R/WL-0	R/WL-1	R/WL-1	R/WL-0	R/WL-1	R/WL-1	R/WL-0	R/WL-1	R/WL-0	R/WL-1	R/WL-1	R/WL-0	R/WL-0

R = Read in all modes; WL = Write in LIN mode only; *-n* = Value after reset

.

Table 16-41. Maximum Baud Rate Selection Register (MBRS) Field Description

Bit	Name	Value	Description
31–13	Reserved		Reads return 0 and writes have no effect.
12–0	MBR[12:0]	0–1FFFh	Maximum baud rate prescaler. This bit is effective in LIN mode only. This 13-bit prescaler is used during the synchronization phase (see Section 16.2.4.2) of a slave module if the ADAPT bit is set (SCIGCR[9] = 1; Section 16.6.2). In this way, a SCI/BLIN slave using an automatic or select bit rate modes detects any LIN bus legal rate automatically. The MBR value should be programmed to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise, a 0x00 data byte could mistakingly be detected as a sync break. The default value for a 70Mhz VCLK is 0xDAC. This MBR prescaler is used by the wake-up and idle time counters for a constant expiration time relative to a 20 kHz rate. $MBR = \frac{0.9 \times VCLK}{maxbaudrate}$

16.6.30 Input/Output Error Enable (IODFTCTRL) Register

The LIN has the following error and status flags:

- Bit error
- Physical bus error
- Checksum error
- Inconsistent synch field error
- No response error
- Frame error
- Overrun error
- Parity error
- Timeout after wakeup signal
- Timeout after 3 wakeup signals
- Bus idle timeout
- Break detect

All the bits in the IODFTCTRL register are used in IODFT (I/O design for test) mode only. Figure 16-53 and Table 16-42 illustrate this register.

Figure 16-53. Input/Output Error Enable Register (IODFTCTRL) [offset = 90h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BEN	PBEN	CEN	ISFE	Reserved	FEN	PEN	BRKDT ENA	Reserved			PIN SAMPLE MASK[1:0]		TX SHIFT [2:0]		
R/WL-0	R/WL-0	R/WL-0	R/WL-0	R-0	R/W-0	R/WC-0	R/WC-0	R-0			R/W-0	R/W-0	R/W-0		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				IODFTENA[3:0]				Reserved						LPB ENA	RXP ENA
R-0				R/WP-0	R/WP-1	R/WP-0	R/WP-1	R-0						R/WP-0	R/WP-0

R = Read in all modes; W = Write in all modes; WL = Write in LIN mode only; WC = Write in sci-compatible mode only; WP = Write in privilege mode only.
-*n* = Value after reset

Table 16-42. Input/Output Error Enable Register (IODFTCTRL) Field Description

Bit	Name	Value	Description
31	BEN		Bit error enable. This bit is effective in LIN mode only. This bit is used to create a bit error.
		0	No bit error is created.
		1	The bit received is ORed with 1 and passed to the bit monitor circuitry.

Table 16-42. Input/Output Error Enable Register (IODFTCTRL) Field Description (Continued)

Bit	Name	Value	Description
30	PBEN		Physical bus error enable. This bit is effective in LIN mode only. This bit is used to create a physical bus error.
		0	No error is created.
		1	The bit received during synch break field transmission is ORed with 1 and passed to the bit monitor circuitry.
29	CEN		Checksum error enable. This bit is effective in LIN mode only. This bit is used to create a checksum error.
		0	No error is created.
		1	The polarity of the CTYPE (checksum type) in the receive checksum calculator is changed so that a checksum error is occurred.
28	ISFE		Inconsistent synch field (ISF) error enable. This bit is effective in LIN mode only. This bit is used to create an ISF error.
		0	No error is created
		1	The bit widths in the synch field are varied so that the ISF check fails and the error flag is set.
27	Reserved		Reads return 0 and writes have no effect.
26	FEN		Frame error enable. This bit is used to create a frame error.
		0	No error is created
		1	The stop bit received is ANDed with 0 and passed to the stop bit check circuitry.
25	PEN		Parity error enable. This bit is effective in compatibility mode only. This bit is used to create a parity error.
		0	No parity error occurs.
		1	The parity bit received is toggled so that a parity error occurs.
24	BRKD TENA		Break detect error enable. This bit is effective in SCI-compatibility mode only. This bit is used to create a BRKDT error.
		0	No error is created.
		1	The stop bit of the frame is ANDed with 0 and passed to the RSM so that a frame error occurs. Then the RX pin is forced to continuous low for 10 T_{BITS} so that a BRKDT error occurs.
23–21	Reserved		Reads return 0 and writes have no effect.

Table 16-42. Input/Output Error Enable Register (IODFTCTRL) Field Description (Continued)

Bit	Name	Value	Description
20–19	PIN SAMPLE MASK		Pin sample mask. These bits define the sample number at which the TX pin value that is being transmitted will be inverted to verify the receive pin samples majority detection circuitry.
		00	No mask is used.
		01	Invert the TX Pin value at TBIT_CENTER.
		10	Invert the TX Pin value at TBIT_CENTER + SCLK.
		11	Invert the TX Pin value at TBIT_CENTER + 2 SCLK.
18–16	TX SHIFT		Transmit shift. These bits define the amount by which the value on TX pin is delayed so that the value on the RX pin is asynchronous. This feature is not applicable to the start bit.
		000	No delay occurs.
		001	The value is delayed by 1 SCLK.
		010	The value is delayed by 2 SCLK.
		011	The value is delayed by 3 SCLK.
		100	The value is delayed by 4 SCLK.
		101	The value is delayed by 5 SCLK.
		110	The value is delayed by 6 SCLK.
		111	The value is delayed by 7 SCLK.
15–12	Reserved		Reads return 0 and writes have no effect.
11–8	IODFTENA		IO DFT enable key.
		1010	IODFT is enabled.
		All other values	IODFT is disabled.
7–2	Reserved		Reads return 0 and writes have no effect.
1	LPBENA		Module loopback enable. **Note: In analog loopback mode the complete communication path through the I/Os can be tested, whereas in digital loop-back mode the I/O buffers are excluded from this path.**
		0	Digital loopback is enabled.
		1	Analog loopback is enabled in module I/O DFT mode when IODFTENA = 1010.

Table 16-42. Input/Output Error Enable Register (IODFTCTRL) Field Description (Continued)

Bit	Name	Value	Description
0	RXPENA		Module analog loopback through receive pin enable. This bit defines whether the I/O buffers for the transmit or the receive pin are included in the communication path (in analog loop-back mode)
		0	Analog loopback through the transmit pin is enabled.
		1	Analog loopback through the receive pin is enabled.

Serial Communication Interface (SCI)

This reference guide contains a general description of the serial communication interface (SCI) module for the TMS470Px devices. The SCI is a universal asynchronous receiver-transmitter (UART) that implements the standard nonreturn-to-zero (NRZ) format.

Topic **Page**

17.1 Introduction and Feature Overview

The SCI module provides a serial (UART) communication interface for TMS470Px devices.

17.1.1 Features

Table 17-1 contains a brief description of the SCI, lists its significant pins, describes interrupts, and defines the low-power mode of operation.

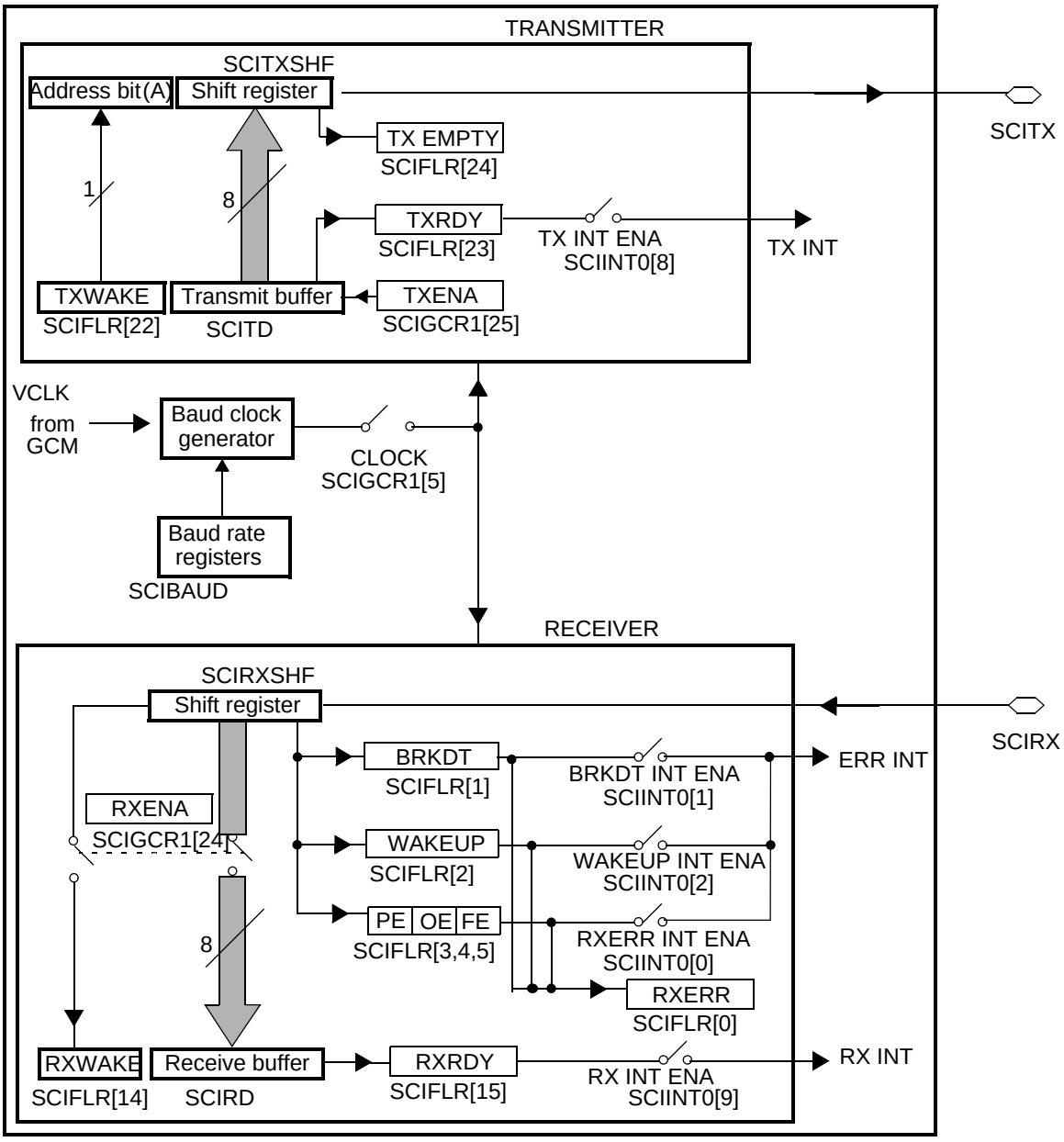
Table 17-1. SCI Description

Description	The SCI module is a universal asynchronous receiver-transmitter (UART) that implements the standard non-return to zero format. The SCI can be used to communicate, for example, through an RS-232 port or over a K-line.	
Pins	SCIRx	SCI receive pin
	SCITx	SCI transmit pin
Interrupts	The SCI has three interrupts: transmit, receive, and error. Each interrupt can be individually enabled.	
Sleep Mode	The SCI can be placed in low-power mode and then wake up to receive an incoming message.	
Features	Standard universal asynchronous receiver-transmitter (UART) communication Supports full- or half-duplex operation Standard NRZ format Double-buffered receive and transmit functions Configurable frame format of 3 to 13 bits per character based on the following: • Data word length programmable from one to eight bits • Additional address bit in address-bit mode • Parity programmable for zero or one parity bit, odd or even parity • Stop programmable for one or two stop bits Asynchronous communication mode The 24-bit programmable baud rate supports 2^{24} different baud rates and provide high-accuracy baud rate selection. Four error flags and six status flags provide detailed information regarding SCI events. Two multi-processor communication formats allow communication between more than two devices Sleep mode available to free CPU resources during multi-processor communication	

17.1.2 Functional SCI Block Diagram

Figure 17-1 provides a detailed view of the SCI module.

Figure 17-1. SCI Block Diagram



17.2 Functional Description

The programmable SCI module supports digital communications between the CPU and other asynchronous peripherals that use the standard non-return to zero (NRZ) format. The SCI receiver and transmitter are double-buffered, and each has its own separate enable and interrupt bits. The receiver and transmitter may each be operated independently or simultaneously in full duplex mode.

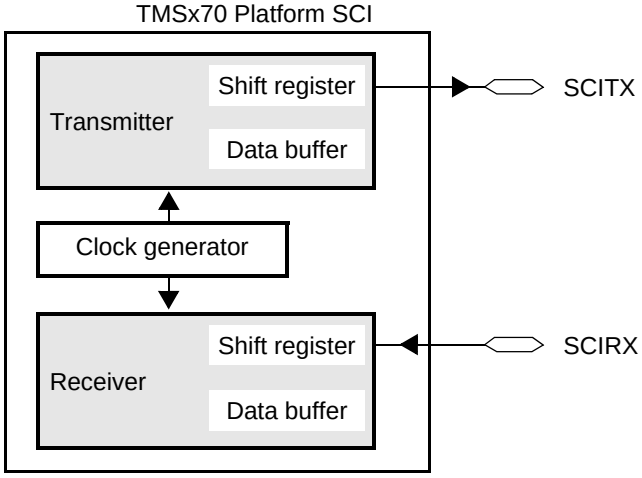
To ensure data integrity, the SCI checks the data it receives for breaks, parity errors, overrun errors, and framing errors. The bit rate (baud) is programmable to over 16 million different rates through a 24-bit baud-selection register.

17.2.1 Operation of the SCI Module

As shown in Figure 17-2, the major components of the SCI include:

- A transmitter (TX) with two major registers to perform double- buffering:
  - The transmitter data buffer register (SCITD) contains data loaded by the CPU to be transferred to the shift register for transmission.
  - The transmitter shift register (SCITXSHF) loads data from the data buffer (SCITD) and shifts data onto the SCITX pin, one bit at a time.
- A receiver (RX) with two major registers to perform double-buffering:
  - The receiver shift register (SCIRXSHF) shifts data in from the SCIRX pin one bit at a time and transfers completed data into the receive data buffer.
  - The receiver data buffer register (SCIRD) contains received data transferred from the receiver shift register.
- A programmable baud generator produces a UART clock scaled from VCLK (generated by the system module).

Figure 17-2. Simplified SCI Block Diagram



The SCI receiver and transmitter can operate independently in a half-duplex configuration (as only a receiver or only a transmitter) or simultaneously in a full-duplex configuration.

Note: Non-Addressable Internal Registers
In addition to communicating through its addressable registers, the SCI uses two non-addressable internal registers: the transmit shift register (SCITXSHF) and the receive shift register (SCIRXSHF). Throughout this document these registers are referenced only by their mnemonic. For more details, see Section 17.6.7.

17.3 SCI Communication Formats

The SCI module can be configured to meet the requirements of many applications. Because communication formats vary depending on each specific application, many attributes of the TMS470Px SCI are user configurable. These configuration options are as follows:

- Frame format

  The SCI uses a standard UART NRZ format with the option to configure parity, number of data bits, and number of stop bits.

- Timing modes

  Asynchronous mode does not use an external synchronizing clock

- Baud rate

  The SCI baud rate is configurable to one of 2^{24} possible data transmission rates.

- Multiprocessor modes

  The SCI permits networked communication between more than two UART devices by providing two multi-processor modes: idle-line and address-bit.

17.3.1 Frame Format

The SCI uses a programmable frame format. All frames consist of the following:

- One start bit
- One to eight data bits
- An optional address bit
- An optional parity bit
- One or two stop bits

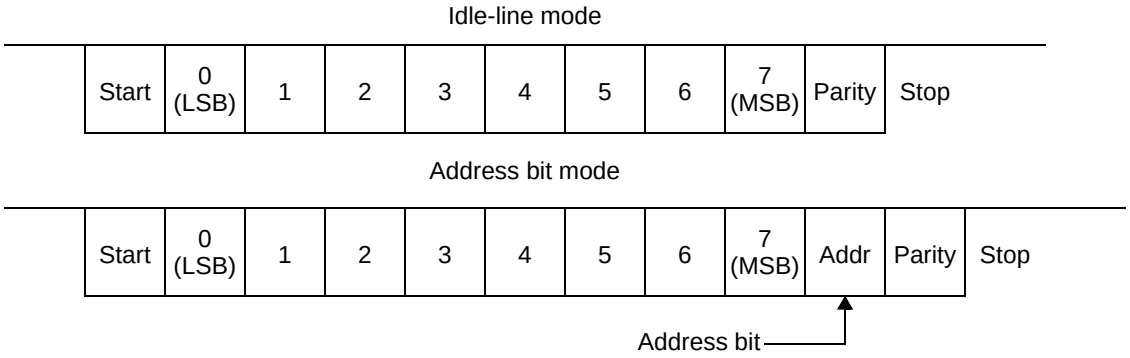
The frame format for both the transmitter and receiver is programmable via the SCIGCR1 register (see Section 17.6.2). Both receive and transmit data is encoded in NRZ format, which means that the transmit and receive lines are held logic-high (1) when idle. Each frame transmission begins with a start bit in which the transmitter pulls the SCI line logic-low. Following the start bit, frame data is sent and received, least-significant bit first (LSB).

An address bit is present in each frame if the SCI is configured to be in address-bit mode. In idle-line mode, this bit is not present in any frame. The format of frames with and without the address bit is illustrated in Figure 17-3.

A parity bit is present in every frame when the PARITY ENA bit (SCIGCR1[2]) is set. The value of the parity bit depends on the number of one bits in the frame and whether odd or even parity has been selected via the PARITY bit (SCIGCR1[3]). Both examples in Figure 17-3 have parity enabled.

All frames include one stop bit, which is always at logic high (1). This high level at the end of each frame is used to indicate the end of a frame to ensure synchronization between communicating devices. If the STOP bit (SCIGCR1[4]) is set, a second stop bits follows the first one. The examples shown in Figure 17-3 use one stop bit per frame.

Figure 17-3. Typical SCI Data Frame Formats



17.3.2 SCI Timing

The SCI uses asynchronous timing.

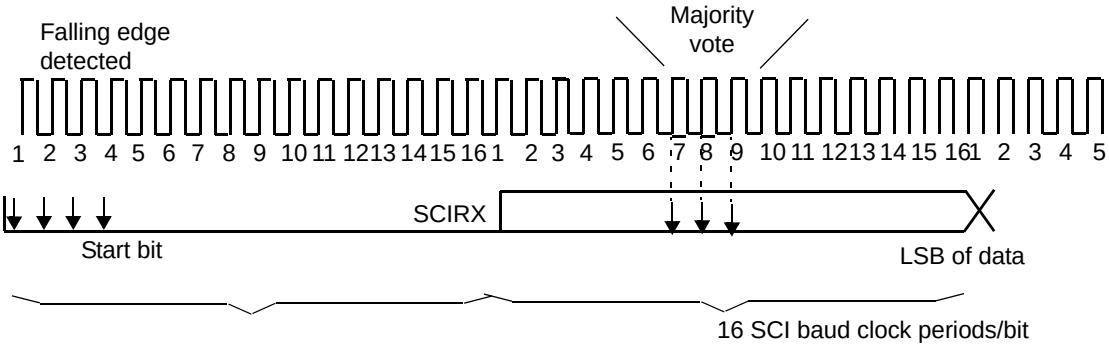
Asynchronous timing mode uses only the receive and transmit data lines to interface with devices using the standard universal asynchronous receiver- transmitter (UART) protocol.

17.3.2.1 Asynchronous Timing Mode

In the asynchronous timing mode, each bit in a frame has a duration of 16 SCI baud clock periods. Each bit, therefore, consists of 16 samples (one for each clock period). See Figure 17-4 for an illustration of asynchronous timing.

Note:
When the SCI is using asynchronous mode, the baud rates of all communicating devices must match as closely as possible. Receive errors result from devices communicating at different baud rates.

Figure 17-4. Asynchronous Communication Bit Timing



In this mode, the CLOCK bit (SCIGCR1[5]; Section 17.6.2) must be set to 1 to use the internal SCICLK.

With the receiver in the asynchronous timing mode, the SCI detects a valid start bit if the first four samples after a falling edge on the SCIRX pin are of logic level 0. As soon as a falling edge is detected on SCIRX, the SCI assumes that a frame is being received and synchronizes itself to the bus.

The SCI module provides some protection from noise causing unintended start bits or incorrect data. Without protection, a noise spike that brings an idle receive line low may be interpreted as a start bit. The SCI prevents this by requiring a start bit to bring the SCIRX line low for at least four contiguous SCI baud clock periods. If any of the receive samples during the first four SCI baud clock periods is not a logic low, then the SCI does not consider this a start bit and considers the receive line idle. When another falling edge is detected, the SCI checks for a valid, noise-free start bit. When a valid start bit is detected, the SCI determines the value of each

bit by sampling the SCI RX line value during the seventh, eighth, and ninth SCI baud clock periods. A majority vote of these samples is used to determine the value stored in the SCI receiver shift register. By sampling in the middle of the bit, the SCI reduces errors caused by propagation delays and rise and fall times. By taking a majority vote, the SCI reduces the likelihood of data corruption caused by data line noise. Figure 17-4 illustrates how the receiver samples a start bit and a data bit in asynchronous timing mode.

The transmitter transmits each bit for a duration of 16 SCI baud clock periods. During the first clock period for a bit, the transmitter shifts the value of that bit onto the SCITX pin. The transmitter then holds the current bit value on SCITX for the following 15 SCI baud clock periods.

17.3.3 SCI Baud Rate

The SCI uses an internal clock source to generate a baud clock for the SCI. If an internal clock is selected via the CLOCK bit (SCIGCR1[5]; Section 17.6.2), then the SCI uses the value in the 24-bit baud register (SCIBAUD; Section 17.6.6) to generate an internal baud clock for the SCI. This BAUD value is used to scale the VCLK signal fed to the SCI from the TMS470Px system module.

In asynchronous timing mode, the SCI generates a baud clock according to the following formula:

$$\textit{Asynchronous baud value} = \left(\frac{\text{VBUSPCLK Frequency}}{16(\text{BAUD} + 1)}\right) \quad \text{(EQ 1)}$$

For BAUD = 0,

$$\textit{Asynchronous baud value} = \left(\frac{\text{VBUSPCLK Frequency}}{32}\right) \quad \text{(EQ 2)}$$

Note: Transmit on Rising Edge; Receive on Falling Edge
The receiver samples data from SCIRX on the *falling* edge of the baud clock, and the transmitter shifts data onto SCITX on the *rising* edge of the baud clock.

17.3.4 SCI Multi-processor Communication Modes

In some applications, the TMS470Px SCI may be connected to more than one serial communication device. In such a multi-processor configuration, several frames of data may be sent to all connected devices or to an individual device. In the case of data sent to an individual device, the receiving devices must determine when they are being addressed. When a message is not intended for them, the devices can ignore the following data, which frees up the processor to carry out actions other than reading unimportant data from the SCI bus. When only two devices make up the SCI network, addressing is not needed, so multi-processor communication schemes are not required.

To interact with multi-processor bus configurations, the TMS470Px SCI includes the option for the use of either idle-line or address-bit (SCIGCR1[0] = 1) multi-processor communication mode.

In a multi-processor configuration, processors send an address character followed by one or more data characters. The SCI must be able to distinguish address characters from data characters. The idle-line and address-bit communication modes offer two methods of differentiating address and data information:

- Idle-line mode leaves a space of 10 or more idle bits before each frame containing an address and 9 or fewer idle bits before each frame containing data.
- Address-bit mode adds an extra bit (the address bit) into every frame. Frames containing an address have the address bit set to 1, and frames containing data have the address bit set to 0. In this mode, the idle space between frames is irrelevant.

When the SCI is not used in a multi-processor environment, software can consider all frames as data frames. In this case, the only distinction between the idle-line and address-bit modes is the presence of an extra bit (the address bit) in each frame sent with the address-bit protocol.

The SCI allows full-duplex communication, whereby data can be sent and received via the transmit and receive pins simultaneously. The protocol used by the SCI assumes that only one device transmits data on the bus at any one time. No arbitration is done by the SCI.

Note: Avoid Transmitting Simultaneously on the Same Serial Bus
The system designer must ensure that devices connected to the same serial bus line do not attempt to transmit simultaneously. If two devices are transmitting different data, the resulting bus conflict could damage the device.

17.3.4.1 Idle-Line Multi-processor Mode

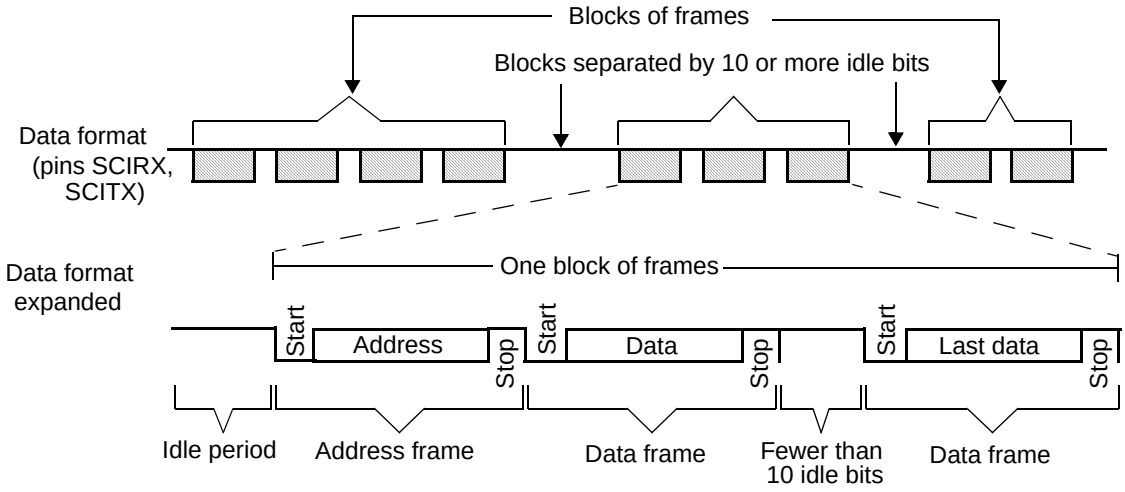
Table 17-2 defines some terms used in the discussion of idle-line mode and address-bit mode in a multiprocessor environment.

Table 17-2. Definition of Multi-processor Terms

Term	Definition
Bit	A single value to be transmitted or received. As discussed in Section 17.3.2.1, each bit sent or received with asynchronous timing consists of 16 SCI baud-clock periods
Idle bit	A bit of logic level 1 that occurs between frames
Idle period	A group of 10 or more idle bits
Frame	The group of bits to be transmitted or received in order to communicate a piece of data or an address. A frame always contains a start bit, a stop bit, and 1 to 8 data bits. It may also contain a parity bit, an address bit, and an extra stop bit. Each frame has a format as defined by the settings of the SCIGCR1 register (see Section 17.6.2.)
Address frame	A frame whose contents should be interpreted as an address
Data frame	A frame whose contents should be interpreted as data
Block	An address frame followed by zero or more data frames. A new block is begun at every address frame.

In idle-line multi-processor mode, a frame that is preceded by an idle period (10 or more idle bits) is an address frame. A frame that is preceded by fewer than 10 idle bits is a data frame. Figure 17-5 illustrates the format of several blocks and frames with idle-line mode.

Figure 17-5. Idle-Line Multiprocessor Communication Format



There are two ways to transmit an address frame using idle-line mode:

- **Method 1:** In software, deliberately leave an idle period between the transmission of the last data frame of the current block and the address frame of the next block.
- **Method 2:** Configure the SCI to automatically send an idle period between the last data frame of the current block and the address frame of the next block.

Method 1 is accomplished by a delay loop in software.

Method 2 can be implemented by using the transmit buffer and the TXWAKE bit (SCIFLR[22]; Section 17.6.4) in the following manner:

1. Write a 1 to the TXWAKE bit.
2. Write a dummy data value to the SCITD register (Section 17.6.7.3). This triggers the SCI to begin the idle period as soon as the transmitter shift register is empty.
3. Wait for the SCI to clear the TXWAKE flag.
4. Write the address value to SCITD.

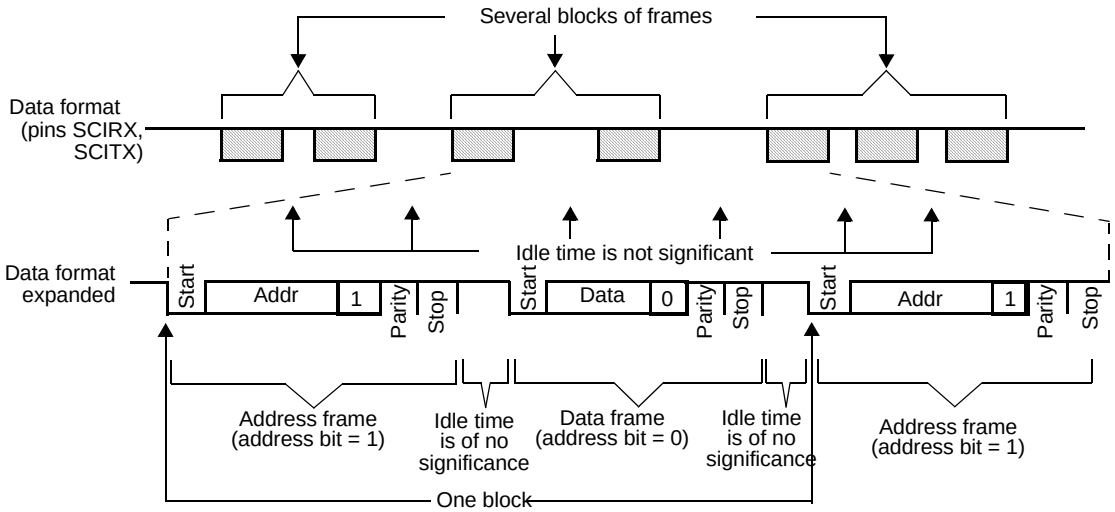
As indicated by Step 3, the software should wait for the SCI to clear the TXWAKE bit. However, the SCI clears the TXWAKE bit at the same time it sets TXRDY (that is, transfers data from SCITD into the transmit shift register). Therefore, if the TX INT ENA bit is set (Section 17.6.3), the transfer of data from SCITD to the transmit shift register causes an interrupt to be generated at the same time that the SCI clears the TXWAKE bit. If this interrupt method is used, software is not required to poll the TXWAKE bit.

> **Note:** When idle-line multi-processor communications are used, software must ensure that the idle time exceeds 10 bit periods before *addresses* (using one of the methods mentioned above), and software must also ensure that *data* frames are written to the transmitter quickly enough to be sent without a delay of 10 bit periods between frames. Failure to comply with these conditions will result in data interpretation errors by other devices receiving the transmission.

17.3.4.2 Address-Bit Multi-processor Mode

In the address-bit protocol, each frame has an extra bit immediately following the data field called an address bit. A frame with the address bit set to 1 is an address frame; a frame with the address bit set to 0 is a data frame. The idle period timing is irrelevant in this mode. Figure 17-6 illustrates the format of several blocks and frames in address-bit mode.

Figure 17-6. Address-Bit Multiprocessor Communication Format



When address-bit mode is used, the value of the TXWAKE bit is the value sent as the address bit. To send an address frame, software must set the TXWAKE bit (SCIFLR[22] = 1). This bit is cleared as the contents of the SCITD are shifted from the TXWAKE register so that all frames sent are data except when the TXWAKE bit is written as a 1.

No dummy write to SCITD is required before an address frame is sent in address-bit mode. The first byte written to SCITD after the TXWAKE bit is written to 1 is transmitted with the address bit set when address-bit mode is used.

17.3.4.3 Sleep Mode for Multi-processor Communication

When the SCI receives data and transfers that data from the receive shift register to SCIRD, the RXRDY bit is set. If RX INT ENA is set, the SCI also generates an interrupt, which tells the CPU to read the newly received frame. In multi-processor communication modes, this default behavior may be enhanced to provide selective indication of new data. When the TMSx70 Platform SCI receives an address frame that does not match its address, through software the device can ignore the data following this non-matching address until the next address frame by using sleep mode.

Sleep mode can be used with both idle-line and address-bit multi-processor modes. If sleep mode is enabled by the SLEEP bit (SCIGCR1[8] = 1; Section 17.6.2), then the SCI transfers data from receive shift register to SCIRD (Section 17.6.7.1) only for address frames. Therefore, in sleep mode, all data frames are assembled in the SCIRXSHF register without being shifted into the SCIRD and without initiating a receive interrupt . Upon reception of an address frame, the contents of the receive shift register are moved into SCIRD, and the software must read SCIRD and determine if the SCI is being addressed by comparing the received address against the address previously set in the software and stored somewhere in memory (the SCI does not have hardware available for address comparison). If the SCI is being addressed, the software must clear the SLEEP bit so that the SCI will load SCIRD with the data of the data frames that follow the address frame.

When the SCI has been addressed and sleep mode has been disabled (in software) to allow the receipt of data, the SCI should check the RXWAKE bit (SCIFLR[14]; Section 17.6.4) to determine when the next address has been received. This bit is set to 1 if the current value in SCIRD is an address and set to 0 if SCIRD contains data. If the RXWAKE bit is set, then software should check the address in SCIRD against its own address. If it is still being addressed, then sleep mode should remain disabled. Otherwise, the SLEEP bit should be set again.

The following is a sequence of events typical of sleep mode operation:

- The SCI is configured and both sleep mode and receive actions are enabled.
- An address frame is received and a receive interrupt is generated.

- Software compares the received address frame against that set by software and determines that the SCI is not being addressed, so the value of the SLEEP bit is not changed.
- Several data frames are shifted into the receive shift register, but no data is moved to SCIRD and no receive interrupts are generated.
- A new address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is being addressed and clears the SLEEP bit.
- Data shifted into the receive shift register is transferred to SCIRD, and a receive interrupt is generated after each data frame is received.
- In each interrupt routine, software checks RXWAKE to determine if the current frame is an address frame.
- Another address frame is received, RXWAKE is set, and software determines that the SCI is **not** being addressed and sets the SLEEP bit back to 1. No receive interrupts are generated for the data frames following this address frame.

By ignoring data frames that are not intended for the device, fewer interrupts are generated. These interrupts would otherwise require CPU intervention to read data that is of no significance to this specific device. Using sleep mode can help free some CPU resources.

Except for the RXRDY flag, the SCI continues to update the receiver status flags (see Table 17-3) while sleep mode is active. In this way, if an error occurs on the receive line, an application can immediately respond to the error and take the appropriate corrective action.

Because the RXRDY bit is not updated for data frames when sleep mode is enabled, the SCI can enable sleep mode and use a polling algorithm if desired. In this case, when RXRDY is set, software knows that a new address has been received. If the SCI is not being addressed, then the software should not change the value of the SLEEP bit and should continue to poll RXRDY.

17.4 Data Transfer

Data transmitted and received by the SCI must be moved to and from the SCI buffer registers from some location in device memory.

This section discusses the configuration of the SCI for interrupt-driven operation.

17.4.1 Interrupts

The SCI receiver and transmitter can both be controlled by interrupts. The receive and transmit interrupts allow efficient operation of the SCI by reading and writing character information to and from the SCI as new data arrives and when old data has just been sent. The RXRDY flag (SCIFLR[15]; Section 17.6.4) used by the receiver indicates that new data is available to be read.

The receiver also has various error interrupts that indicate when a particular error condition is active. An active error interrupt condition is indicated by the RXERR flag in SCIFLR. The exact source of an error interrupt can be determined by checking the parity error (PE), frame error (FE), overrrun error (OE), break-detect (BRKDT), and wake-up (WAKEUP) flags also located in SCIFLR. Additionally, the transmitter uses the TXRDY flag (SCIFLR[23]) to indicate that the transmitter is ready for new data to be written that will be sent to the bus.

Transmit, receive, and error interrupts are enabled or disabled through separate interrupt-enable bits. When not enabled, the interrupts are not asserted; however, polled operation of the SCI is still possible because the interrupt flags continue to indicate module events.

The SCI module generates three interrupt requests to the Vectored Interrupt Module (VIM): one each for transmitter, receiver, and error interrupts. Each of these interrupts must also be configured in the VIM before operation. Please see the VIM module user's guide for more information. The error interrupt should be set to have the highest priority, the receiver interrupt the next highest priority, and the transmitter the lowest priority. This prioritization scheme reduces the possibility of missed error conditions and receiver overruns.

17.4.1.1 Transmit Interrupt

The transmit ready (TXRDY) flag is set when the SCI transfers the contents of SCITD to the shift register, SCITXSHF. The TXRDY flag indicates that SCITD register (Section 17.6.7.3) is ready to be loaded with more data. In addition, the SCI sets the TX EMPTY bit if both the SCITD and transmit shift registers are empty. Transmit interrupts are enabled by the TX INT ENA bit (SCIINT0[8]). If the TX INT ENA bit (SCIINT0[8]) is set, then a transmit interrupt is generated when the TXRDY flag goes high. The transmit Interrupt is not generated immediately after setting the TX INT ENA bit (generated only after the first transfer from SCITD to the transmit shift register).

Writing data to the SCITD register clears the TXRDY bit. When this data has been moved to the transmit shift register, the TXRDY bit is set again. The interrupt request can be suspended by clearing the TX INT ENA bit; however, when the TX INT ENA bit is again set to 1, the TXRDY interrupt is asserted again. The transmit interrupt request can be stopped until the next series of values is written to SCITD by disabling the transmitter via the TXENA bit (SCIGCR1[25] = 0; Section 17.6.2), an SCI software reset, or by a device hardware reset.

Note: To use transmit interrupt functionality, the TX DMA ENA bit (SCIINT0[16]; Section 17.6.3) must be cleared.

17.4.1.2 Receive Interrupt

The receive-ready (RXRDY) flag is set when the SCI transfers newly received data from the receive shift register to SCIRD (Section 17.6.7.2). The RXRDY flag indicates that the SCI has new data to be read. Receive interrupts are enabled by the RX INT ENA bit (SCIINT0[9]; Section 17.6.3). If the RX INT ENA bit is set when the SCI sets the RXRDY flag, then a receive interrupt is generated.

17.4.1.3 Error Interrupts

The TMS470Px SCI provides hardware indication of error conditions to provide you with information about the status of module operation. According to the data being assembled by the receiver, the SCI monitors received data for errors and sets the parity error (PE), framing error (FE), and/or the break-detect (BRKDT) flag when these conditions are detected. In addition, the SCI sets the overrun error (OE) flag if a transfer of new data from the receive shift register to SCIRD (Section 17.6.7.2) overwrites unread data in SCIRD. If both overrun and parity errors occur, both flags are set. The SCI sets the wake-up flag (WAKEUP) if bus activity on the RX line either prevents power-down mode from being entered, or RX line activity causes an exit from power-down mode. Each of these flags is located in the receiver status register (SCIFLR; Section 17.6.4).

The RXERR flag, also in the SCIFLR register, is the logical OR of the parity error, framing error, overrun error, wake-up, and break-detect flags. This feature allows software that is polling for receiver errors to check only one bit, which will indicate if any or all of the five error conditions are active.

Error interrupts are controlled by three separate enable bits: RXERR INT ENA, BRKDT INT ENA, and WAKEUP INT ENA.

- If RXERR INT ENA (SCIINT0[0]) is set, an error interrupt is generated when the receiver detects either a parity, framing, or overrun error. The break-detect interrupt is enabled separately.
- If BRKDT INT ENA (SCIINT0[1]) is set, an error interrupt is generated if the receiver detects a break condition. A break condition occurs when the SCIRX line remains continuously low (active) for at least 10 bits immediately following a missed stop bit. This is further explained in the BRKDT bit description in Section 17.6.5.
- If WAKEUP INT ENA (SCIINT0[2]) is set, an error interrupt is generated when bus activity on the RX line either prevents power-down mode from being entered or RX line activity causes an exit from power-down mode.

17.5 Operating the SCI

Before the SCI sends or receives data, its registers should be properly configured. Upon power-up or a system-level reset, each bit in the SCI registers is set to a default state. The registers are writable only after

the RESET bit, SCIGCR0[0], is set to 1. Of particular importance is the SW nRESET bit (SCIGCR1[7]; Section 17.6.2). This active-low bit is initialized to 0 and keeps the SCI in a reset state until it is programmed to 1. Therefore, all SCI configuration should be completed before a 1 is written to the SW nRESET bit.

The following sections describe general configuration requirements for the SCI as well as the configuration process for the SCI receiver and SCI transmitter.

17.5.1 Configuration Requirements

The SCI contains some registers and bits that affect operation of both the transmitter and receiver. The control registers and bits listed below must be configured to suit the application whenever the SCI is used.

- SCIGCR0, Section 17.6.1.
- SCIGCR1, Section 17.6.2.
- SCIBAUD, Section 17.6.6.
- Continue on suspend bit (CONT); SCIGCR1[17], Section 17.6.2.
- Loop back bit (LOOP BACK); SCIGCR1[16], Section 17.6.2.
- SCI internal clock enable bit (CLOCK); SCIGCR1[5], Section 17.6.2.
- Power-down bit (POWERDOWN); SCIGCR1[9], Section 17.6.2.
- Software reset bit (SW nRESET); SCIGCR1[7], Section 17.6.2.

The following list details the configuration steps that software should perform before the transmission or reception of data. As long as SW nRESET is held low the entire time that the SCI is being configured, the order in which the registers are programmed is not important.

- Enable SCI by setting RESET bit of SCIGCR0.
- Clear SW nRESET to 0 before configuring the SCI.
- Select the desired frame format by programming SCIGCR1.
- Select the baud rate to be used for communication by programming SCIBAUD.
- Select whether an internal or external clock should be used by programming the CLOCK bit.
- Set LOOP BACK if you want to connect the transmitter to the receiver internally. (This feature can be used to perform a self-test.)
- Configure the receiver if data is to be received (see Section 17.5.2).
- Configure the transmitter if data is to be transmitted (see Section 17.5.3).
- Configure the SCIRX and SCITX pins for SCI functionality by setting the RX FUNC bit (SCIPIO0[1]) and the TX FUNC bit (SCIPIO0[2]) to 1.
- Configure the clock pin if the SCI is to output its clock or accept an external clock (see Section 17.6.8).
- Set SW nRESET to 1 after the SCI is configured.

Note: Set the COS bit if you do not want the SCI to halt for an emulation breakpoint until its current reception or transmission is complete. (This bit is used only in an emulation environment).

Table 17-3. SCI Receiver Status Flags

SCI Flag	Register	Bit	Value After SW nRESET(1)
RXWAKE	SCIFLR	14	0
RXRDY	SCIFLR	15	0
FE	SCIFLR	5	0
OE	SCIFLR	4	0
PE	SCIFLR	3	0
BRKDT	SCIFLR	1	0
RXERR	SCIFLR	0	0

1The flags are frozen with their reset value while SW nRESET = 0.

Table 17-4. SCI Transmitter Status Flags

SCI Flag	Register	Bit	Value After SW nRESET(1)
TX EMPTY	SCIFLR	24	1
TXRDY	SCIFLR	23	1

1The flags are frozen with their reset value while SW nRESET = 0.

17.5.2 Receiving Data

The following paragraphs describe the configuration and signal timing requirements to prepare the device to receive data.

17.5.2.1 Receiver Configuration

The receiver is configured by setting or clearing the following bits:

- RX enable (SCIGCR1[24])
- Sleep (SCIGCR1[8])
- RX error interrupt enable (SCIINT0[0])
- Break-detect interrupt enable (SCIINT0[1])
- Wake-up interrupt enable (SCIINT0[2])
- RX Interrupt enable (SCIINT0[9])
- RX pin control register (SCIPIOx[1])

The SCI receiver is enabled to receive messages if SW nRESET is inactive, and if the RX FUNC bit (SCIPIO0[1]; Section 17.6.8) and the RXENA bit (SCIGCR1[24]; Section 17.6.2) are set to 1. If the RX FUNC bit is not set, the SCIRX pin functions as a general purpose I/O pin rather than as an SCI function pin. No data is transferred into the receive buffer register while the RXENA bit is cleared to 0. The receiver assembles data into the receive shift register, but nothing is moved into the receiver buffer register (SCIRD). Both of these control bits allow the SCI receiver to be held inactive independently of the transmitter. Receive interrupts must also be configured if they are to be used. By default, the interrupts are disabled. For more information on receive interrupts, see Section 17.4.1.

When the receiver has been configured and enabled and SW nRESET has been set to 1, the receiver looks for an idle period (approximately 11 bit times). Until the receiver finds this idle period, the SCI regards any high bit sampled as being part of the idle period. If the detection of an idle period is interrupted by a low bit, then the SCI begins searching again for an idle period. When it is searching for an idle period, the SCI does not distinguish between high bits that are part of data being sent and high bits that are part of a true idle period.

An idle period must be detected before data can be received regardless of the mode of operation. This requirement is in place so that the SCI can safely connect to the receive bus after powerup and wakeup from low-power mode. The idle period is required so that the SCI does not begin monitoring the bus in the middle of communication with other peripherals and receive invalid frame data.

After the idle period is found, data is automatically received as it arrives on the SCIRX pin. Because the SCI sets the RXRDY bit (SCIFLR; Section 17.6.4) when it transfers newly received data from the receive shift register to SCIRD (Section 17.6.7.2), new data should be read each time RXRDY is set. The SCI clears the RXRDY bit after the SCIRD has been read.

Also, as data is transferred from the receive shift register to SCIRD, the SCI sets the FE, OE, or PE flags in SCIFLR if any of these error conditions were detected in the received data. The wake-up and break-detect status bits are also set if one of these errors occurs, but they do not necessarily occur at the same time that new data is being loaded into SCIRD.

It is not necessary to poll the RXRDY bit to wait for it to be set high. If RX INT ENA is set in the SCIINT0 register (Section 17.6.3), the SCI generates a receive interrupt while it is transferring data from the receive shift register to SCIRD. Also, if the RXERR INT ENA bit is set in the SCIINT0 register, then the SCI generates an error interrupt if any of the corresponding error conditions (PE, OE, or FE) are detected in the received data.

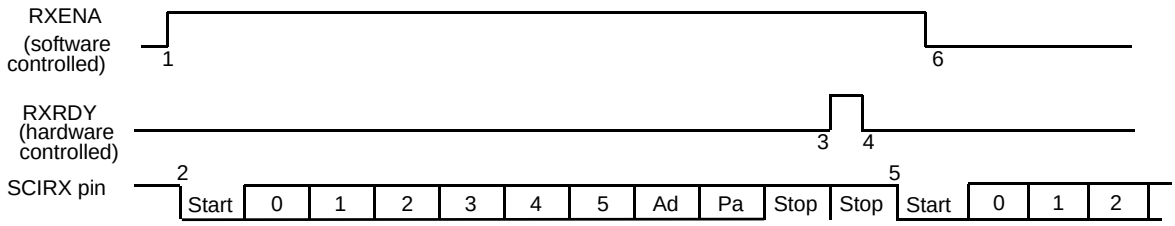
17.5.2.2 Receiver Signal Timing

Figure 17-7 is an example of the timing relationship between the RXENA and RXRDY bits and received data. In the figure, the SCI is assumed to be in the following configuration:

- Address-bit mode
- Six data bits per frame
- Parity enabled
- Two stop bits

The notes following Figure 17-7 correspond to the numbered steps in the diagram and explain the receive process. Note that when RXRDY is set, the receiver flags PE, OE, FE, RXERR, and RXWAKE become valid for the data being transferred into SCIRD. Also, the BUS BUSY flag is cleared when the RXRDY flag is set because this condition indicates that the receiver has completed the reception of a frame.

Figure 17-7. SCI RX Signals in Communication Modes



1 RXENA is set high to enable the receiver to load SCIRD with new data.
2 A start bit is detected (for this example, an idle period has already been detected).
3 Data is transferred from the receive shift register to SCIRD. An interrupt is driven if RX INT ENA is set.
4 SCIRD is read.
5 Another start bit is detected.
6 RXENA is set low to disable the receiver. Data is shifted into the receive shift register but is not transferred to SCIRD.

17.5.3 Transmitting Data

The following sections describe the configuration and signal timing requirements to prepare the device to transmit data.

17.5.3.1 Transmitter Configuration

The transmitter is configured by setting or clearing the following bits:

- TX enable (SCIGCR1[25]; Section 17.6.2)
- TX wake-up method select (SCIFLR[22]; Section 17.6.4)
- TX Interrupt enable (SCIINT0[8]; Section 17.6.3)
- TX pin control register (SCIPIOx[2])

The SCI transmitter is enabled to transmit messages if SW nRESET is inactive (SCIGCR1[7] = 1), and if the TX FUNC bit (SCIPIO0[2]) and the TXENA bit (SCIGCR1[25]) are set to 1. If the TX FUNC bit is not set, the SCITX pin functions as a general-purpose I/O pin rather than an SCI function pin. No data is transferred out of the shift register. The TXENA bit allows the transfer of data from the transmit buffer (SCITD) to the transmit shift register. While the TXENA bit is cleared to 0, software can write data to the transmit buffer, but it is not moved into the transmit shift register. Any value written to the SCITD before TXENA is set to 1 is not transmitted. If the TXENA bit is cleared while the transmitter is sending a byte, then both bytes in the SCITD and the transmit shift register are transmitted before the transmitter is disabled. Both of these control bits allow for the SCI transmitter to be held inactive independently of the receiver. The transmit interrupt must also be configured if it is to be used. By default, the interrupt are disabled. For more information on transmit interrupts, see Section 17.4.1.

When the transmitter has been configured and enabled and the SW nRESET has been set to 1, the SCI waits for data to be written to SCITD, transfers it to transmit shift register, and then transmits it on the TX pin. The flags TXRDY and TX EMPTY indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to SCITD, the TXRDY bit is set. Additionally, if both SCITD and the transmit shift register are empty, then the TX EMPTY bit is set.

Because the SCI sets the TXRDY bit each time it transfers data from SCITD to the transmit shift register, new data can be written to SCITD each time TXRDY is set. The SCI clears the TXRDY bit only when data is currently in SCITD. It is not necessary to poll the TXRDY bit to wait for it to be set high. If TX INT ENAis set, the SCI generates an interrupt while it is transferring data from SCITD to the transmit shift register and setting the TXRDY bit.

When the SCI has completed transmission of all pending frames, the transmit shift register and SCITD are empty, the TXRDY bit is set, and an interrupt is generated, if enabled. Because all data has been transmitted, the interrupt request should be halted by either disabling the transmit interrupt via the TX INT ENA bit (SCIINT0[8] = 0) or by disabling the transmitter (SCIGCR1[25] = 0).

Note:
Disabling the transmit interrupt halts the interrupt request only until the transmit interrupt is re-enabled.

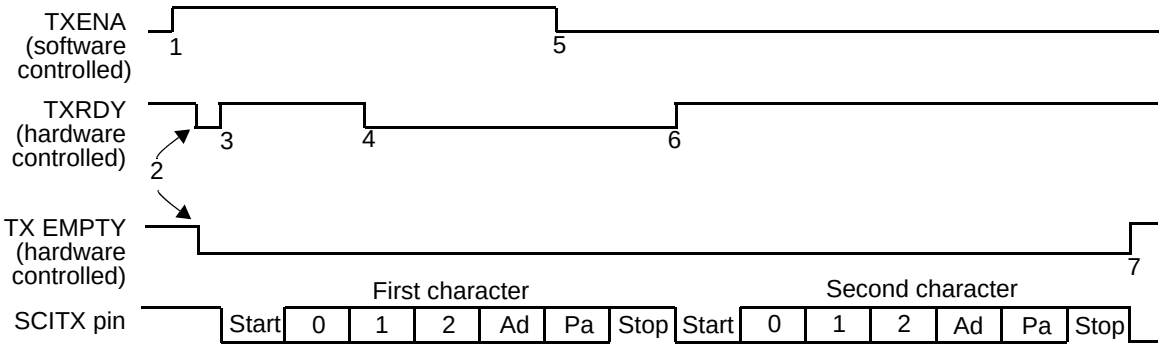
17.5.3.2 Transmitter Signal Timing

Figure 17-8 is an example of the timing relationship of TXENA, TX EMPTY, and TXRDY and transmitted data. In the figure, the SCI is assumed to be in the following configuration:

- Address-bit mode
- Three data bits per frame
- Parity enabled
- One stop bit

The notes following Figure 17-8 correspond to the numbered steps in the diagram and explain the transmit process.

Figure 17-8. SCI TX Signals in Communication Modes



1 TXENA is set high to enable the transmitter to send data.
2 A write occurs to SCITD.
3 The SCI transfers data from SCITD to the transmit shift register and begins transmitting. The transmitter requests an interruptif TX INT ENAis set.
4 A second write occurs to SCITD.
5 TXENA is set low to disable the transmitter.
6 The first character is completed, so the SCI transfers data from SCITD to the transmit shift register and continues transmitting. The transmitter does not request an interrupt because the transmitter is disabled.
7 Transmission is complete and both SCITD and the transmit shift register are empty.

17.5.4 Power-Down Mode

The SCI can be put in either local or global low-power mode. Global low-power mode is asserted when the device enters low power mode and is not controlled by the SCI. During global low-power mode, all clocks to the SCI are turned off so the module is completely inactive.

Power-down of the SCI can also be achieved by setting the POWERDOWN (SCIGCR1[9]) bit. However, it is strongly recommended to power-down the SCI by programming the SCI's peripheral frame bit in the System Peripheral Control Registers (PCR). Please see the System Architecture users guide. All the registers are accessible during local power-down mode as any register access enables the clock to SCI for that particular access alone.

The wake-up interrupt is used to allow the SCI to automatically exit low-power mode when a low level is detected on the SCIRX pin and also this clears the POWERDOWN bit. If this interrupt is disabled (WAKEUP INT ENA is cleared), then the SCI immediately enters low-power mode whenever it is requested and also any activity on the SCIRX pin does not cause the SCI to exit low-power mode.

Note: Enabling Local Low-Power Mode During Receive and Transmit
If the wake-up interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI immediately generates a wake-up interrupt to clear the POWERDOWN bit and prevents the SCI from entering low-power mode and thus completes the current reception. Otherwise, if the wake-up interrupt is disabled, then the SCI completes the current reception and then enters low-power mode.

17.6 SCI Registers

The SCI is controlled and accessed through the registers listed in Figure 17-9. Among the features that can be programmed are the communication and timing modes, baud value, frame format, and interrupt enables. These registers are accessible in 8-, 16-, and 32-bit reads or writes.

Figure 17-9. SCI Control Register Summary

Offset Addr[1]	Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x00	SCIGCR0 Page 987	Reserved															
		Reserved															RESET
0x04	SCIGCR1 Page 988	Reserved						TX ENA	RX ENA	Reserved						COS	LOOP BACK
		Reserved						POWER DOWN	SLEEP	SW nRST	Reserved		STOP	PARITY	PAR-ITY ENA	Reserved	COMM MODE
0x08	SCIINT0 Page 992	Reserved													RX DMA ALL	RX DMA ENA	TX DMA ENA
		Reserved						RX INT ENA	TX INT ENA	Reserved					WAKUP INT ENA	BRKDT INT ENA	RXERR INT ENA
0x0C	SCIFLR Page 994	Reserved							TX EMPTY	TX RDY	TX WAKE	Reserved					
		RX RDY	RX WAKE	Reserved						BUSY	IDLE	FE	OE	PE	WAKE UP	BRKDT	RX ERR
0x10	SCICHAR Page 998	Reserved															
		Reserved															CHAR
0x14	SCIBAUD Page 999	Reserved								BAUD							
		BAUD															
0x18	SCIED Page 1001	Reserved															
		Reserved								ED							

1 The absolute address of these registers is device specific. Consult the specific device data sheet to verify the SCI register addresses.

Offset Addr[1]	Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x1C	SCIRD	Reserved															
	Page 1002	Reserved								RD							
0x20	SCITD	Reserved															
	Page 1003	Reserved								TD							
0x24	SCIPIO0	Reserved															
	Page 1004	Reserved													TX FUNC	RX FUNC	Reserved
0x28	SCIPIO1	Reserved															
	Page 1005	Reserved													TX DIR	RX DIR	Reserved
0x2C	SCIPIO2	Reserved															
	Page 1006	Reserved													TX IN	RX IN	Reserved
0x30	SCIPIO3	Reserved															
	Page 1007	Reserved													TX OUT	RX OUT	Reserved
0x34	SCIPIO4	Reserved															
	Page 1009	Reserved													TX SET	RX SET	Reserved
0x38	SCIPIO5	Reserved															
	Page 1010	Reserved													TX CLR	RX CLR	Reserved
0x40	SCIPIO7	Reserved															
	Page 1011	Reserved													TX PD	RX PD	Reserved

1 The absolute address of these registers is device specific. Consult the specific device data sheet to verify the SCI register addresses.

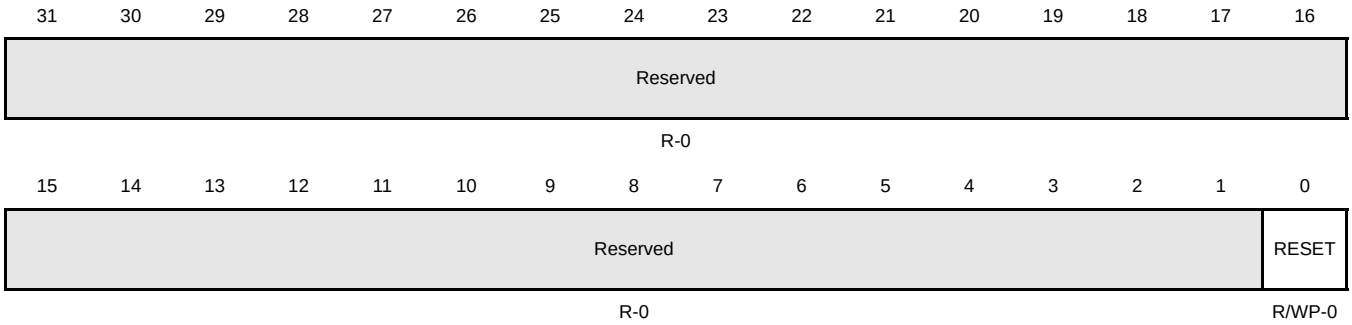
Offset Addr[1]	Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x44	SCIPIO8 Page 1011	Reserved															
		Reserved													TX PSL	RX PSL	Reserved
0x50	SCIIODCTRL Page 1013	Reserved					FEN	PEN	BRKDT ENA	Reserved			PIN SAMPLE MASK		TX SHIFT [2:0]		
		Reserved				IODFTENA[3:0]				Reserved						LPB ENA	RXP ENA

1 The absolute address of these registers is device specific. Consult the specific device data sheet to verify the SCI register addresses.

17.6.1 SCI Global Control Register 0 (SCIGCR0)

The SCIGCR0 register defines the module reset. Figure 17-10 and Table 17-5 illustrate this register.

Figure 17-10. SCI Global Control Register 0 (SCIGCR0) [offset = 00h]



R = Read; WP= Write in privileged mode only; C = Clear; -*n* = Value after reset

Table 17-5. SCI Global Control Register 0 (SCIGCR0) Field Descriptions

Bit	Name	Value	Description
31-1	Reserved		Reads return zeros and writes have no effect.
0	Reset		This bit acts as a module reset. Privileged mode write:
		0	SCI module is under reset
		1	SCI module is out of reset
			User and privileged mode read:
		0	SCI module is under reset
		1	SCI module is out of reset

17.6.2 SCI Global Control Register (SCIGCR1)

The SCIGCR1 register defines the frame format, protocol, and communication mode used by the SCI. Figure 17-10 and Table 17-6 illustrate this register.

Figure 17-11. SCI Global Control Register (SCIGCR1) [offset = 04h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						TXENA	RXENA	Reserved						CONT	LOOP BACK
R-0						R/W-0	R/W-0	R-0						R/W-0	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						POWER DOWN	SLEEP	SW nRST	Reserved		STOP	PARITY	PARITY ENA	Reserved	COMM MODE
R-0						R/W-0	R/W-0	R/W-0	R-0		R/W-0	R/W-0	R/W-0	R-0	R/W-0

R = Read in all modes; W = Write in all modes; WP = Write in privileged mode only; *-n* = Value after reset

Table 17-6. SCI Global Control Register (SCIGCR1) Field Descriptions

Bit	Name	Value	Description
31–26	Reserved		Reads return 0 and writes have no effect.
25	TXENA		Transmit enable. Data is transferred from the SCITD register (Section 17.6.7.3) to the transmit shift register only when the TXENA bit is set.
		0	Transfers from SCITD to the transmit shift register are disabled.
		1	Transfers from SCITD to the transmit shift register are enabled.
			Note: Data written to SCITD before TXENA is set is not transmitted. If TXENA is cleared while transmission is ongoing, the data previously written to SCITD is sent.

Table 17-6. SCI Global Control Register (SCIGCR1) Field Descriptions (Continued)

Bit	Name	Value	Description
24	RXENA		SCI receiver enable. RXENA allows or prevents the transfer of data from the receive shift register to SCIRD.
		0	The receiver will not transfer data from the receive shift register to SCIRD.
		1	The receiver will transfer data from the receive shift register to SCIRD.
			Note: Clearing RXENA stops received characters from being transferred into the receive buffer, prevents the RX status flags (see Table 17-3) from being updated by receive data, and inhibits both receive and error interrupts. However, the receive shift register continues to assemble data regardless of the state of RXENA.
			Note: If RXENA is cleared before a frame is completely received, the data from the frame is not transferred into SCIRD.
			Note: If RXENA is set before a frame is completely received, the data from the frame is transferred into SCIRD. If RXENA is set while the receive shift register is assembling a frame, the status flags are not guaranteed to be accurate for that frame. To ensure that the status flags correctly reflect what was detected on the bus during a particular frame, RXENA should be set before the detection of that frame.
23–18	Reserved		Reads return 0 and writes have no effect.
17	COS		**Continue on suspend.** This bit has an effect only when a program is being debugged with an emulator, and it determines how the SCI operates when program execution is suspended.
		0	When debug mode is entered, the SCI state machine is frozen. Transmissions are halted and resume when debug mode is exited.
		1	When debug mode is entered, the SCI continues to operate until the current transmit and receive functions are complete.
16	LOOP BACK		The self-checking option for the SCI can be selected with this bit. If the SCITX and SCIRX pins are configured with SCI functionality, then the SCITX pin is internally connected to the SCIRX pin. Externally, during loop back operation, the SCITX pin outputs a high value. The SCIRX pin must not be driven by an external device during self-test or errors may result.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled.
15–10	Reserved		Reads return 0 and writes have no effect.

Table 17-6. SCI Global Control Register (SCIGCR1) Field Descriptions (Continued)

Bit	Name	Value	Description
9	POWERDOWN		**Powerdown bit.** When the POWERDOWN bit is set, the SCI attempts to enter local low-power mode. If the POWERDOWN bit is set while the receiver is actively receiving data and the wake-up interrupt is enabled, then the SCI immediately asserts an error interrupt to prevent low-power mode from being entered. See Section 17.5.4 for more information on low-power mode.
		0	The module operates normally.
		1	Local low-power mode is requested.
8	SLEEP		**SCI sleep.** In a multi-processor configuration, this bit controls the receive-side sleep function. Clearing this bit brings the SCI out of sleep mode. See Section 17.3.4.3 for more information.
		0	Sleep mode is disabled.
		1	Sleep mode is enabled.
			Note: The receiver still operates when the SLEEP bit is set; however, RXRDY is updated and SCIRD is loaded with new data only when an address frame is detected. The remaining receiver status flags (see Table 17-3) are updated and an error interrupt is requested if the corresponding interrupt enable bit is set, regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the SCI is asleep, software can promptly deal with the error condition.
			Note: The SLEEP bit is *not* automatically cleared when an address byte is detected.
7	SWnRESET		**Software reset** (active low).
		0	The SCI is in its reset state; no data will be transmitted or received. Writing a 0 to this bit initializes the SCI state machines and operating flags as defined in Table 17-3 and Table 17-4. All affected logic is held in the reset state until a 1 is written to this bit.
		1	The SCI is in its ready state; transmission and reception can be performed. The configuration of the module should not change.
			Note: The SCI should only be configured while SWnRESET = 0.
6-5	Reserved		Reads return 0 and writes have no effect.
4	STOP		SCI number of stop bits.
		0	One stop bit is used.
		1	Two stop bits are used.
			Note: The receiver checks for only one stop bit. However in idle-line mode, the receiver waits until the end of the second stop bit (if STOP = 1) to begin checking for an idle period.

Table 17-6. SCI Global Control Register (SCIGCR1) Field Descriptions (Continued)

Bit	Name	Value	Description
3	PARITY		**SCI parity odd/even selection.** If the PARITY ENA bit in this register is set, PARITY designates odd or even parity.
		0	Odd parity is used.
		1	Even parity is used.
			Note: The parity bit is calculated based on the data bits in each frame and the address bit (in address-bit mode). The start and stop fields in the frame are not included in the parity calculation.
			Note: For odd parity, the SCI transmits and expects to receive a value in the parity bit that makes odd the total number of bits in the frame with the value of 1.
			Note: For even parity, the SCI transmits and expects to receive a value in the parity bit that makes even the total number of bits in the frame with the value of 1.
2	PARITY ENA		**Parity enable.** This bit enables or disables the parity function.
		0	Parity is disabled; no parity bit is generated during transmission or is expected during reception.
		1	Parity is enabled. A parity bit is generated during transmission and is expected during reception.
1	Reserved		Reads return 0 and writes have no effect.
0	COMM MODE		**SCI communication mode bit.**
		0	Idle-line mode is used.
		1	Address-bit mode is used.
			See Section 17.3.4 for more information on these communication modes.

17.6.3 SCI Set Interrupt Register (SCISETINT)

Figure 17-12 and Table 17-7 illustrate this register.

Figure 17-12. SCI Set Interrupt Register (SCISETINT) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RX INT ENA	TX INT ENA	Reserved					WAKUP INT ENA	BRKDT INT ENA	RXERR INT ENA
R-0						R/W-0	R/W-0	R-0					R/W-0	R/W-0	R/W-0

R = Read in all modes; W = Write in all modes; -*n* = Value after reset

.

Table 17-7. SCI Interrupt Control Register (SCIINT0) Field Descriptions

Bit	Name	Value	Description
31-10	Reserved		Reads return zero and writes have no effect.
9	RX INT ENA		**Receiver interrupt enable.** Setting this bit enables the SCI to generate a receive interrupt after a frame has been completely received and the data is being transferred from the receive shift register to SCIRD.
		0	The receive interrupt is disabled.
		1	The receive Interrupt is enabled.
8	TX INT ENA		**Transmitter interrupt enable.** Setting this bit enables the SCI to generate a transmit interrupt as data is being transferred from SCITD to transmit shift register and the TXRDY bit is being set.
		0	The transmit Interrupt is disabled.
		1	The transmit Interrupt is enabled.
7–3	Reserved		Reads return zero and writes have no effect.
2	WAKEUP INT ENA		**Wake-up interrupt enable.** Setting this bit enables the SCI to generate a wake-up interrupt and thereby exit low-power mode. If enabled, the wake-up interrupt is asserted when local low-power mode is requested while the receiver is busy or if a low level is detected on the SCIRX pin during low-power mode. This interrupt is occurs on the SCI error interrupt request line.
		0	The wake-up interrupt is disabled.
		1	The wake-up interrupt is enabled.
1	BRKDT INT ENA		**Break-detect interrupt enable.** Setting this bit enables the SCI to generate an error interrupt if a break condition is detected on the SCIRX pin.
		0	The break-detect interrupt is disabled.
		1	The break-detect interrupt is enabled.

Table 17-7. SCI Interrupt Control Register (SCIINT0) Field Descriptions (Continued)

Bit	Name	Value	Description
0	ERR INT ENA		**Receive error interupt enable.** Setting this bit enables the SCI to generate an error interrupt if any of the following conditions are detected on the SCIRX pin. • Framing error (FE) • Overrun error (OE) • Parity error (PE) **Note: Break-detect and wake-up interrupts are enabled separately.**
		0	The error interrupt is disabled.
		1	The error interrupt is enabled.

17.6.4 SCI Flags Register (SCIFLR)

This section describes the flag registers. Figure 17-13 and Table 17-8 illustrate this register.

Figure 17-13. SCI Flags Register (SCIFLR) [offset = 0Ch]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TX EMPTY	TX RDY	TX WAKE	Reserved					
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-1	R-1	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX RDY	RX WAKE	Reserved						BUSY	IDLE	FE	OE	PE	WAKE UP	BRKDT	RX ERR
RC-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-1	RC-0	RC-0	RC-0	RC-0	RC-0	R-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-8. SCI Flags Register (SCIFLR) Field Descriptions

Bit	Name	Value	Description
31-25	Reserved		Reads return zero and writes have no effect.
24	TX EMPTY		**Transmitter empty flag.** The value of this flag indicates the contents of the SCITD and transmit shift register. An active SW nRESET (SCIGCR1[7]) or a system reset sets this bit. This bit does not cause an interrupt request.
		0	The SCIRD or transmit shift register (or both) are loaded with data.
		1	The SCIRD and transmit shift register are both empty.
23	TXRDY		**Transmitter buffer register ready flag.** When set, this bit indicates that the SCITD is ready to receive another character. Writing data to SCITD automatically clears this bit. The SCI asserts a transmit interruptafter data is written to the SCITD, when it sets this flag if the interrupt, TX INT ENA (SCIINT0[8]), is set. TXRDY is also set to 1 either by enabling SWnRESET (SCIGCR1[7]) or by a system reset.
		0	SCITD is full.
		1	SCITD is ready to receive the next character.
			For more information on transmit interrupt handling, see Section 17.4.1.1.

Table 17-8. SCI Flags Register (SCIFLR) Field Descriptions (Continued)

Bit	Name	Value	Description
22	TXWAKE		**SCI transmitter wake-up method select.** The TXWAKE bit controls whether the data in SCITD should be sent as an address or data frame using multi-processor communication format. This bit is set to 1 or 0 by software before a byte is written to SCITD and is cleared by the SCI when data is transferred from SCITD to the transmit shift register or by a system reset. TXWAKE is not cleared by the SWnRESET bit (SCIGCR1[7]).
			Address-bit mode:
		0	Frame to be transmitted will be data (address bit = 0).
		1	Frame to be transmitted will be an address (address bit = 1).
			Idle-line mode:
		0	Frame to be transmitted will be data.
		1	Frame to be transmitted will be an address (writing a 1 to this bit followed by writing dummy data to the SCITD will result in a idle period of 11 bit periods before the next frame is transmitted).
			See Section 17.3.4 for more information on using the TXWAKE bit in the available communication modes.
21-16	Reserved		Reads return zero and writes have no effect.
15	RXRDY		**SCI receiver ready flag.** The receiver sets this bit to indicate that the SCIRD contains new data and is ready to be read by the CPU. The SCI generates a receive interrupt when it sets this bit if the interrupt-enable bit RX INT ENA is set. RXRDY is cleared by an active SWnRESET, a system reset, writing a 1 to this bit, or by reading from SCIRD.
		0	*Read:* No new data is in SCIRD. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* New data is ready to be read from SCIRD. *Write:* The bit is cleared to 0.
14	RXWAKE		**Receiver wake-up detect flag.** The SCI sets this bit to indicate that the data currently in SCIRD is an address. RXWAKE is cleared by an active SWnRESET, a system reset, or by the SCI upon receipt of a data frame.
		0	The data in SCIRD is not an address.
		1	The data in SCIRD is an address.
			See Section 17.3.4.3 for more information on using the RXWAKE bit with sleep mode.
13-8	Reserved		Reads return zero and writes have no effect.

Table 17-8. SCI Flags Register (SCIFLR) Field Descriptions (Continued)

Bit	Name	Value	Description
7	BUSY		**Bus busy flag.** This bit indicates whether the receiver is in the process of receiving a frame. As soon as the receiver detects the beginning of a start bit, the BUS BUSY bit is set to 1. When the reception of a frame is complete, the SCI clears the BUS BUSY bit. If WAKEUP INT ENA is set and power down is requested while this bit is set, the SCI automatically prevents low-power mode from being entered and generates an error interrupt. The BUS BUSY bit is controlled directly by the SCI receiver but is cleared by an active SWnRESET or by a system reset.
		0	The receiver is not currently receiving a frame.
		1	The receiver is currently receiving a frame.
6	IDLE		**SCI receiver in idle state.** While this bit is set, the SCI looks for an idle period to resynchronize itself with the bit stream. The receiver does not receive any data while the bit is set. The bus must be idle for 11 bit periods to clear this bit. The SCI enters this state if one of the following events occurs: a system reset, an SCI software reset, power down, and if the RX pin is configured as a general I/O pin.
		0	An idle period was detected and the SCI is ready to receive.
		1	An idle period was not detected and the SCI will not receive any data.
5	FE		**SCI framing error flag.** This bit is set when an expected stop bit is not found. Only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. Detection of a framing error causes the SCI to generate an error interrupt if the RXERR INT ENA bit is set. The framing error flag is reset by clearing the SWnRESET bit, by a system reset, or by writing a 1 to this bit.
		0	*Read:* No framing error has been detected since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	A framing error has been detected. *Write:* The bit is cleared to 0.
4	OE		**SCI overrun error flag.** This bit is set when the transfer of data from the receive shift register to the SCIRD overwrites unread data already in the SCIRD. Detection of an overrun error causes the SCI to generate an error interrupt if the RXERR INT ENA bit is set. The OE flag is reset by an active SWnRESET, a system reset, or by writing a 1 to this bit.
		0	*Read:* No overrun error has been detected since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	An overrun error has been detected. *Write:* The bit is cleared to 0.

Table 17-8. SCI Flags Register (SCIFLR) Field Descriptions (Continued)

Bit	Name	Value	Description
3	PE		**SCI parity error flag.** This bit is set when a parity error is detected in the received data. An error is generated when a character is received with a mismatch between the number of 1s and its parity bit. For more information on parity checking, see Section 17.6.2. If the parity function is disabled, the PE flag is disabled and read as 0. Detection of a parity error causes the SCI to generate an error interrupt if the RXERR INT ENA bit is set. The PE bit is reset by an active SWnRESET, a system reset, or by writing a 1 to this bit.
		0	*Read:* No parity error has been detected since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	A parity error has been detected. *Write:* The bit is cleared to 0.
2	WAKEUP		**Wake-up flag.** This bit is set by the SCI when receiver or transmitter activity has taken the module out of power-down mode. An interrupt is generated if the WAKEUP INT ENA bit (SCIINT0[2]; Section 17.6.3) is set. It is cleared by an active SWnRESET, a system reset, or by writing a 1 to this bit**.** See Section 17.5.4 for more information on low-power mode.
		0	*Read:* The module will not wake up from power-down mode. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The module will wake up from power-down mode. *Write:* The bit is cleared to 0.
1	BRKDT		**SCI break-detect flag.** This bit is set when the SCI detects a break condition on the SCIRX pin. A break condition occurs when the SCIRX pin remains continuously low for at least 10 bits after a missing first stop bit, that is, after a framing error. Detection of a break condition causes the SCI to generate an error interrupt if the BRKDT INT ENA bit is set. The BRKDT bit is reset by an active SWnRESET, a system reset, or by writing a 1 to this bit.
		0	*Read:* No break condition has been detected since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* A break condition has been detected. *Write:* The bit is cleared to 0.
0	RXERR		**SCI receiver error flag.** The RXERR bit is the logical OR of the framing error, overrun error, parity error, wake-up, and break-detect flags (bits FE, OE, PE, WAKEUP, and BRKDT bits 5, 4, 3, 2, and 1). This bit can be used for fast error-condition checking and is cleared by an active SWnRESET, by a system reset, or by clearing FE, OE, PE, WAKEUP, and BRKDT if any of these bits are set.
		0	*Read:* No error flags have been set since the last clear. *Write:* Writing a zero to this bit has no effect.
		1	*Read:* An error flag has been set. *Write:* The bit is cleared to 0.

17.6.5 SCI Character Control Register (SCICHAR)

This section describes the character control registers. Figure 17-14 and Table 17-9 illustrate this register.

Figure 17-14. SCI Character Control Register (SCICHAR) [offset = 10h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CHAR(2–0)		
R-0													R/W-0		

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

Table 17-9. SCI Character Control Register (SCICHAR) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2–0	CHAR(2–0)	0–3h	Character length control bits. This bit field sets the SCI data length from 1 to 8 bits.
		000	The data is 1 bit long.
		001	The data is 2 bit long.
		010	The data is 3 bit long.
		011	The data is 4 bit long.
		100	The data is 5 bit long.
		101	The data is 6 bit long.
		110	The data is 7 bit long.
		111	The data is 8 bit long.
			When data of fewer than eight bits in length is received, it is left justified in SCIRD and padded with trailing zeros. Data read from the SCIRD should be shifted by software to make the received data right justified.
			Data written to the SCITD should be right justified (it does not need to be padded with leading zeros).

17.6.6 SCI Baud Rate Selection Register (SCIBAUD)

This section describes the baud rate selection registers. Figure 17-15 and Table 17-10 illustrate this register.

Figure 17-15. SCI Baud Rate Selection Register (SCIBAUD) [offset = 14h]

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

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-10. SCI Baud Rate Selection Register (SCIBAUD) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return zero and writes have no effect.
23–0	BAUD(23–0)		SCI 24-bit baud selection. The internally-generated serial clock is determined by VCLK and the baud select register. The SCI uses the 24-bit value of this register to select 1 of over 16,700,000 available baud rates. The baud rate can be calculated using the following formulas: For BAUD > 0, $\textit{Asynchronous baud value} = \left(\frac{\text{VBUSPCLK Frequency}}{16(\text{BAUD} + 1)}\right)$ For BAUD = 0, $\textit{Asynchronous baud value} = \left(\frac{\text{VBUSPCLK Frequency}}{32}\right)$

The SCI receives data on the falling clock edges and transmits data on the rising clock edges. Table 17-11 contains comparative baud values for different register values (with VCLK = 25 MHz) for asynchronous mode.

Table 17-11. Comparative Baud Values for Asynchronous Mode(1).

24-Bit Register Value		Baud Selected		Percent Error
Decimal	Hex	Ideal	Actual	
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

1 For this example, VCLK = 25 MHz. Values are in decimal except for column 2.

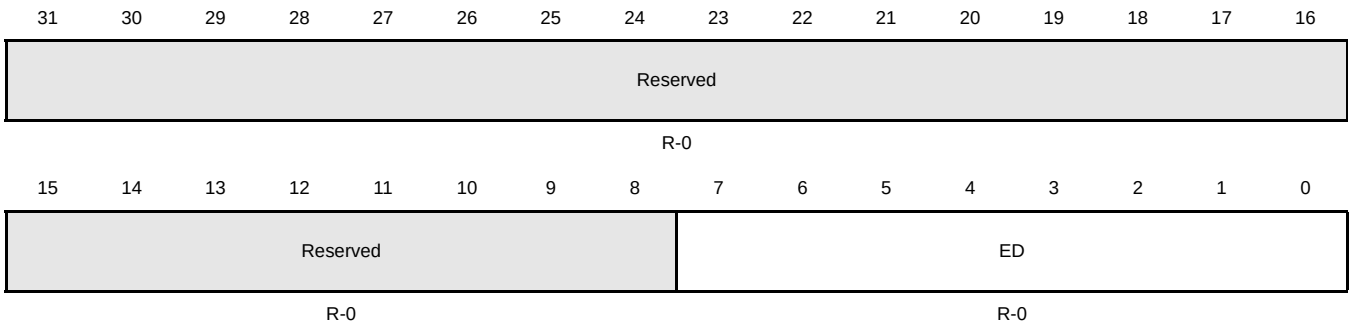
17.6.7 SCI Data Buffers (SCIED, SCIRD, SCITD)

The SCI has three addressable registers in which transmit and receive data is stored.

17.6.7.1 Receiver Emulation Data Buffer (SCIED)

The SCIED register is addressed at a location different from SCIRD but is physically the same register. Unlike SCIRD, however, reading SCIED does not clear the RXRDY flag. This register should be used only by an emulator which must continually read the data buffer without affecting the RXRDY flag. Figure 17-16 and Table 17-10 illustrate this register.

Figure 17-16. Receiver Emulation Data Buffer (SCIED) [offset = 18h]



LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

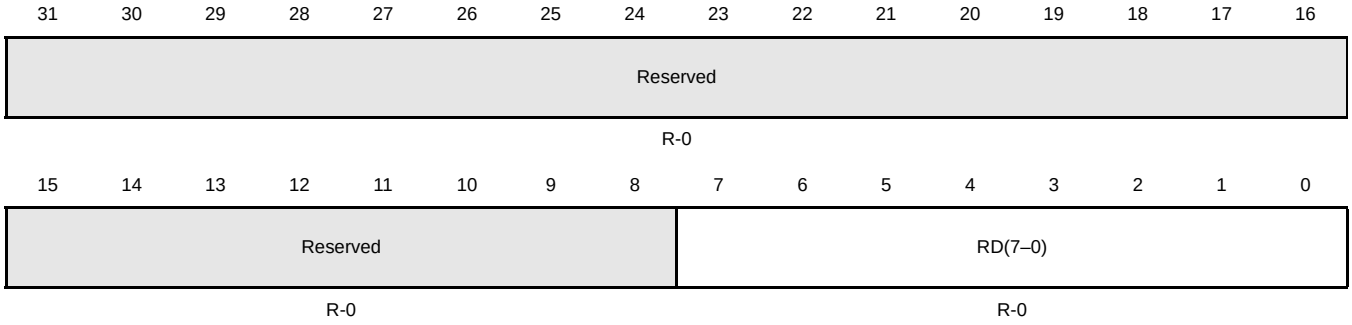
Table 17-12. Receiver Emulation Data Buffer (SCIED) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	ED(7–0)	0–FFh	Reading SCIED(7–0) does not clear the RXRDY flag. This register should be used only by an emulator, which must be able to read the data buffer without affecting the RXRDY flag.

17.6.7.2 Receiver Data Buffer (SCIRD)

When a frame has been completely received, the data in the frame is transferred from the receive shift register to this register. As this transfer occurs, the RXRDY flag is set and a receive interrupt is generated if RX INT ENA (SCIINT0.9) is set. When the data is read from SCIRD, the RXRDY flag is automatically cleared. Figure 17-17 and Table 17-13 illustrate this register.

Figure 17-17. Receiver Data Buffer (SCIRD) [offset = 1Ch]



LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

Note: Receive Buffer is Left Justified
For received data that is fewer than 8 bits in length, the SCI loads the data into this register in a left-justified format padded with trailing zeros. Therefore, software should perform a logical shift on the data by the correct number of positions to make it right justified.

.

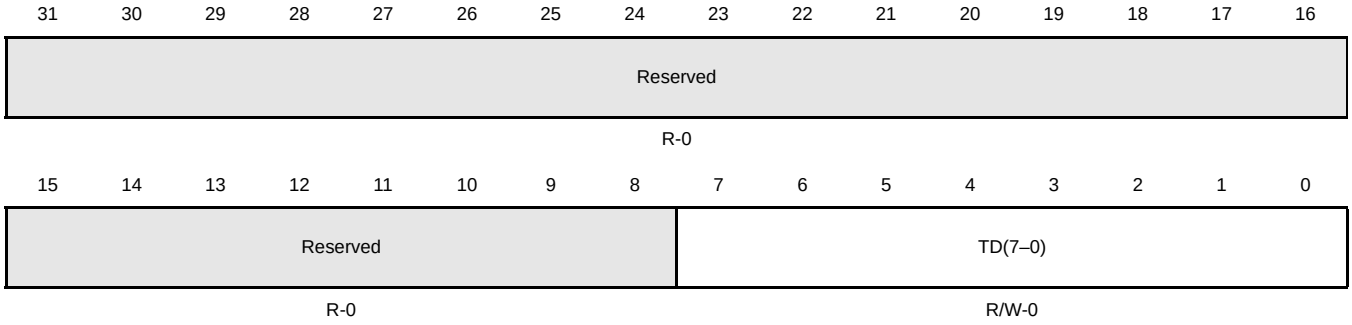
Table 17-13. Receiver Data Buffer (SCIRD) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	RD(7–0)		**Received data.** When a frame has been completely received, the data in the frame is transferred from the receive shift register to this register. As this transfer occurs, the RXRDY flag is set and a receive interrupt is generated if RX INT ENA is set. **Note: When the data is read from SCIRD, the RXRDY flag is automatically cleared.**

17.6.7.3 Transmit Data Buffer Register (SCITD)

Data to be transmitted is written to the SCITD register. The transfer of data from this register to the transmit shift register sets the TXRDY flag (register SCIFLR.23), which indicates that SCITD is ready to be loaded with another byte of data. If the TX INT ENA bit is set, this data transfer also causes an interrupt. Figure 17-18 and Table 17-14 illustrate this register.

Figure 17-18. Transmit Data Buffer Register (SCITD) [offset = 20h]



LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

Note: Receive Buffer Must Be Right Justified
Data written to SCITD register that is fewer than 8 bits long must be right justified but is not required to be padded with leading zeros.

.

Table 17-14. Transmit Data Buffer Register (SCITD) Field Descriptions

Bit	Name	Value	Description
31–8	Reserved		This bit is always read as 0. Writes have no effect.
7–0	TD(7–0)	0–FFh	**Transmit data.** The transfer of data from this register to the transmit shift register sets the TXRDY flag (SCIFLR[23]; Section 17.6.4), which indicates that SCITD is ready to be loaded with another byte of data. **Note: If TX INT ENA is set, this data transfer also causes an interrupt.**

17.6.8 SCI Pin I/O Control Register 0 (SCIPIO0)

This section describes the pin I/O control 0 register. Figure 17-19 and Table 17-15 illustrate this register.

Figure 17-19. SCI Pin I/O Control Register 0 (SCIPIO0) [offset = 24h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX FUNC	RX FUNC	Reserved
R-0													R/W-0	R/W-0	R-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-15. SCI Pin I/O Control Register 0 (SCIPIO0) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX FUNC		Transmit function. This bit defines the function of pin SCITX.
		0	SCITX is a general-purpose digital I/O pin.
		1	SCITX is the SCI transmit pin
1	RX FUNC		Receive function. This bit defines the function of pin SCIRX.
		0	SCIRX is a general-purpose digital I/O pin.
		1	SCIRX is the SCI receive pin.
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.9 SCI Pin I/O Control Register 1 (SCIPIO1)

This section describes the pin I/O control 1 register. Figure 17-20 and Table 17-16 illustrate this register.

Figure 17-20. SCI Pin I/O Control Register 1 (SCIPIO1) [offset = 28h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX DIR	RX DIR	Reserved
R-0													R/W-0	R/W-0	R-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

Table 17-16. SCI Pin I/O Control Register 1 (SCIPIO1) Field Descriptions

Bit	Name	Value	Description
31-2	Reserved		Reads return zero and writes have no effect.
2	TX DATA DIR		Transmit data direction. This bit determines the data direction on the SCITX pin if it is configured with general-purpose I/O functionality (TX FUNC = 0). See Table 17-19 for bit values.
		0	SCITX is a general-purpose input pin.
		1	SCITX is a general-purpose output pin.
1	RX DATA DIR.		Determines the data direction on the SCIRX pin if it is configured with general-purpose I/O functionality (RX FUNC = 0). See Table 17-20 for bit values.
		0	SCIRX is a general-purpose input pin.
		1	SCIRX is a general-purpose output pin.
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.10 SCI Pin I/O Control Register 2 (SCIPIO2)

This section describes the pin I/O control 2 register. Figure 17-21 and Table 17-17 illustrate this register.

Figure 17-21. SCI Pin I/O Control Register 2 (SCIPIO2) [offset = 2Ch]

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

LEGEND: R = Read; W = Write; C = Clear; -*n* = Value after reset

.

Table 17-17. SCI Pin I/O Control Register 2 (SCIPIO2) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX DATA IN		Transmit data in. This bit contains the current value on pin SCITX.
		0	SCITX value is at logic low (0).
		1	SCITX value is at logic high (1).
1	RX DATA IN		Receive data in. This bit contains the current value on pin SCIRX.
		0	The SCIRX value is at logic low (0).
		1	The SCIRX value is at logic high (1).
0	Reserved		Reads are undefined and writes have no effect.

17.6.11 SCI Pin I/O Control Register 3 (SCIPIO3)

This section describes the pin I/O control 3 register. Figure 17-22 and Table 17-18 illustrate this register.

Figure 17-22. SCI Pin I/O Control Register 3 (SCIPIO3) [offset =30h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX OUT	RX OUT	Reserved
R-0													R/W-0	R/W-0	R-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-18. SCI Pin I/O Control Register 3 (SCIPIO3) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX DATA OUT		Transmit data out. This bit contains the data to be output on pin SCITX if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DATA DIR = 1 (SCITX pin is a general-purpose output.)
		0	The output value on pin SCITX is at logic low (0).
		1	The output value on pin SCITX is at logic high (1).
1	RX DATA OUT		Receive data out. This bit contains the data to be output on pin SCIRX if the following conditions are met: RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) RX DATA DIR = 1 (SCIRX pin is a general-purpose output.)
		0	The output value on pin SCIRX is at logic low (0).
		1	The output value on pin SCIRX is at logic high (1).
0	Reserved		Reads return zero. Always write zero to this bit.

Table 17-19. SCITX Pin Control

Function	TX DATA IN(1)	TX DATA OUT	TX FUNC	TX DATA DIR
SCITX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

(1) TX DATA IN is a read-only bit. Its value always reflects the level of the SCIRX pin.

Table 17-20. SCIRX Pin Control

Function	RX DATA IN[1]	RX DATA OUT	RX FUNC	RX DATA DIR
SCIRX	X	X	1	X
General purpose input	X	X	0	0
General purpose output, high	X	1	0	1
General purpose output, low	X	0	0	1

(1) RX DATA IN is a read-only bit. Its value always reflects the level of the SCIRX pin.

17.6.12 SCI Pin I/O Control Register 4 (SCIPIO4)

This section describes the pin I/O control 4 register. Figure 17-23 and Table 17-21 illustrate this register.

Figure 17-23. SCI Pin I/O Control Register 4 (SCIPIO4) [offset = 34h]

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

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-21. SCI Pin I/O Control Register 4 (SCIPIO4) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX DATA SET		Transmit data set. This bit sets the data to be output on pin SCITX if the following conditions are met: • TX FUNC = 0 (SCITX pin is a general-purpose I/O.) • TX DATA DIR = 1 (SCITX pin is a general-purpose output.)
		0	*Read:* The TXDOUT bit is at logic low (0). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The TXDOUT bit is at logic high (1). *Write:* SCITX is at logic high (1).
1	RX DATA SET		Sets the data to be output on pin SCIRX if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DATA DIR = 1 (SCIRX pin is a general-purpose output.)
		0	*Read:* The RXDOUT bit is at logic low (0). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The RXDOUT bit is at logic high (1). *Write:* SCIRX is at logic high (1).
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.13 SCI Pin I/O Control Register 5 (SCIPIO5)

This section describes the pin I/O control 5 register. Figure 17-24 and Table 17-22 illustrate this register.

Figure 17-24. SCI Pin I/O Control Register 5 (SCIPIO5) [offset = 38h]

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

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset

.

Table 17-22. SCI Pin I/O Control Register 5 (SCIPIO5) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX DATA CLR		Clears the data to be output on pin SCITX if the following conditions are met: TX FUNC = 0 (SCITX pin is a general-purpose I/O.) TX DATA DIR = 1 (SCITX pin is a general-purpose output.)
		0	*Read:* The TXDOUT bit is at logic high (0). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The TXDOUT bit is at logic low (1). *Write:* SCITX is at logic low (0).
1	RX DATA CLR		Clears the data to be output on pin SCIRX if the following conditions are met: • RX FUNC = 0 (SCIRX pin is a general-purpose I/O.) • RX DATA DIR = 1 (SCIRX pin is a general-purpose output.)
		0	*Read:* The RXDOUT bit is at logic high (0). *Write:* Writing a zero to this bit has no effect.
		1	*Read:* The RXDOUT bit is at logic low (1). *Write:* SCIRX is at logic low (0).
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.14 SCI Pin I/O Control Register 7 (SCIPIO7)

This section describes the pin I/O control 7 register. Figure 17-25 and Table 17-23 illustrate this register.

Figure 17-25. SCI Pin I/O Control Register 7 (SCIPIO7) [offset = 40h]

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

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset. See device datasheet Terminal Functions for default pin settings

.

Table 17-23. SCI Pin I/O Control Register 7 (SCIPIO7) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX PD		TX pin pull control disable. This bit disables pull control capability in the output pin SCITX.
		0	Pull control on the SCITX pin is enabled.
		1	Pull control on the SCITX pin is disabled.
1	RX PD		RX pin pull control disable. This bit disables pull control capability in the output pin SCIRX.
		0	Pull control on the SCIRX pin is enabled.
		1	Pull control on the SCIRX pin is disabled.
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.15 SCI Pin I/O Control Register 8 (SCIPIO8)

This section describes the pin I/O control 8 registers. Figure 17-26 and Table 17-24 illustrate this register. This register controls the input buffers of the pins when the pull controls of the pins are disabled (xxPD of SCIPIO7 = 1). If the pull control of the pins are enabled, the input buffers are enabled.

Figure 17-26. SCI Pin I/O Control Register 8 (SCIPIO8) [offset = 44h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TX PSL	RX PSL	Reserved
R-0													R/W-n	R/W-n	R-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset; See device datasheet Terminal Functions for default pin settings

Table 17-24. SCI Pin I/O Control Register 8 (SCIPIO8) Field Descriptions

Bit	Name	Value	Description
31-3	Reserved		Reads return zero and writes have no effect.
2	TX PSL		TX pin pull select.
		0	Input buffer for TX is disabled if PULLDIS = 1
		1	Input buffer for TX is enabled if PULLDIS = 1
1	RX PSL		RX pin pull select.
		0	Input buffer fo RX is disabled if PULLDIS = 1
		1	Input buffer for RX is enabled if PULLDIS = 1
0	Reserved		Reads return zero. Always write zero to this bit.

17.6.16 IODFT for SCI Module Register (SCIIODCTRL)

The SCI has the following error and status flags:

- Frame error (FE)
- Overrun error (OE)
- Parity error (PE)
- Break detect (BD)

For IODFT (I/O design for test) mode only, an additional register (SCIIODCTRL) is added as shown in Figure 17-27 and Table 17-25. This register is used for TI internal test.

Figure 17-27. IODFT for SCI Module Register (SCIIODCTRL) [offset = 50h]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					FEN	PEN	BENA	Reserved			PIN SAMPLE MASK		TX SHIFT		
R-0					R/W-0	R/W-0	R/W-0	R-0			R/W-0		R/W-0		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				IODFTENA				Reserved						LPBENA	RXPENA
R-0				R/W-0				R-0						R/W-0	R/W-0

LEGEND: R = Read; W = Write; C = Clear; *-n* = Value after reset; n = Device Dependant

All the bits are used in IODFT mode only.

.

Table 17-25. IODFT for SCI Module Register (SCIIODCTRL) Field Descriptions

Bit	Name	Value	Description
31-27	Reserved		Reads return zero and writes have no effect.
26	FENA		**Frame error enable.**
		0	No frame error is generated.
		1	A frame error is generated. The stop bit received is ANDed with ’0’ and passed to the stop bit check circuitry.
25	PENA		**Parity error enable.**
		0	No parity error is generated.
		1	A parity error is generated. The parity bit received is toggled so that a parity error occurs.
24	BENA		**Break detect error enable.**
		0	No break detect error is generated.
		1	The stop bit of the frame is ANDed with ‘0’ and passed to the RSM so that a frame error occurs. Then the RX pin is forced to continuous low for 10 T_{BITS} so that a break detect error occurs.
23-21	Reserved		Writes have no effect and reads give out 0’s.

Table 17-25. IODFT for SCI Module Register (SCIIODCTRL) Field Descriptions (Continued)

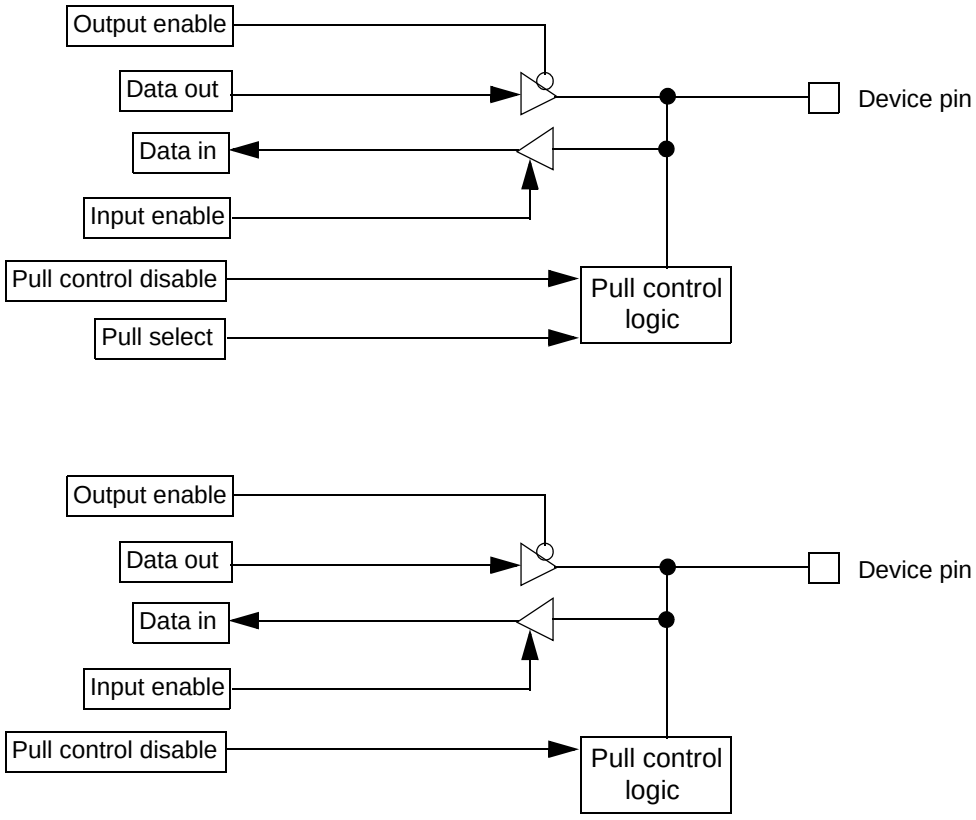
Bit	Name	Value	Description
20-19	PIN SAMPLE MASK		**Pin sample mask.** These bits define the sample number at which the TX Pin value that is being transmitted will be inverted to verify the receive pin sample majority detection circuitry.
		00	No mask is used.
		01	Invert the TX pin value at the 7th SCLK.
		10	Invert the TX pin value at the 8th SCLK.
		11	Invert the TX pin value at the 9th SCLK.
18-16	TX SHIFT		**Transmit shift.** These bits define the delay by which the value on the TX pin is delayed so that the value on the RX pin is asynchronous. This does not apply to the start bit.
		000	No delay occurs.
		001	The delay is 1 SCLK.
		010	The delay is 2 SCLK.
		011	The delay is 3 SCLK.
		100	The delay is 4 SCLK.
		101	The delay is 5 SCLK.
		110	The delay is 6 SCLK.
		111	No delay occurs.
15-12	Reserved		Reads return zero and writes have no effect.
11-8	IODFTENA		**IO DFT enable key.**
		1010	IODFT is enabled.
		All other values	IODFT is disabled.
7-2	Reserved		Reads return zero and writes have no effect.
1	LPBENA		**Module loopback enable.**
		0	Digital loopback is enabled.
		1	Analog loopback is enabled in module I/O DFT mode (when IODFTENA = 1010).
0	RXPENA		**Module analog loopback through receive pin enable.**
		0	Analog loopback is through the transmit pin.
		1	Analog loopback is through the receive pin.

The following sections apply to all device pins that can be configured as functional or general-purpose I/O pins.

17.7 Pin I/O Functionality

Figure 17-28 illustrates how the buffers are connected internally at the pins.

Figure 17-28. Pin I/O Functionality



17.7.1 Under Reset

The following apply if a device is under reset:

- **Pull control.** The reset pull control on the pins is enabled or disabled depending on a device-specific option.
- **Input buffer.** If the reset pull control is enabled, then the input buffer is also enabled. If the reset pull control is disabled then the input buffer is also disabled.
- **Output buffer.** The output buffer is disabled.

17.7.2 Out of Reset

The following apply if the device is out of reset:

- **Pull control.** The pull control is enabled by clearing the xPD (pull control disable) bit in the SCIPIO7 register (Section 17.6.14). In this case, if the xPSL (pull select) bit in the SCIPIO8 register (Section 17.6.15) is set, the pin will have a pull-up. If the xPSL bit is cleared, the pin will have a pull-down. If the xPD bit is set in the control register, there is no pull-up or pull-down on the pin.
- **Input buffer.** The input buffer is disabled only if the pin direction is set as input in the SCIPIO1 register (Section 17.6.9) AND the pull control is disabled AND pull down is selected as the pull bias (the PSL bit in the SCIPIO8 register; Section 17.6.15). AND the PSL bit in the SCIPIO8 register is a 0; Section 17.6.15). In all other cases, the input buffer is enabled.

Note:
The pull-disable logic depends on the pin direction. It is independent of whether the device is in I/O or functional mode. If the pin is configured as output or transmit, then the pulls are disabled automatically. If the pin is configured as input or receive, the pulls are enabled or disabled depending on bit PD in the pull disable register SCIPIO7 (Section 17.6.14).

- **Output buffer.** A pin can be driven as an output pin if the TX DIR bit is set in the pin direction control register (SCIPIO1; Section 17.6.9) AND the open-drain feature is not enabled in the SCIPIO6 register.

17.7.3 Open-Drain Feature Enabled on a Pin

The following apply if the open-drain feature is enabled on a pin.

- The output buffer is enabled if a low signal is being driven on to the pin.
- The output buffer is disabled (the direction control signal DIR is internally forced low) if a high signal is being driven on to the pin.

Note:
The open-drain feature is available only in I/O mode (SCIPIO0; Section 17.6.8).

17.7.4 Pin I/O Summary

The behavior of the input buffer, output buffer, and the pull control is summarized in Table 17-26.

Table 17-26. Input Buffer, Output Buffer, and Pull Control Behavior as GPIO Pins

Device under Reset?	Pin Direction (DIR)	Pull Disable (PULDIS)	Pull Select (PULSEL)	Pull Control	Output Buffer	Input Buffer
Yes	X	X	X	Device- and module-specific	Disabled	Depends on pull control
No	0	0	0	Pull down	Disabled	Enabled
No	0	0	1	Pull up	Disabled	Enabled
No	0	1	0	Disabled	Disabled	Disabled
No	0	1	1	Disabled	Disabled	Enabled
No	1	X	X	Disabled	Enabled	Enabled

1 X = Don't care
2 DIR = 0 for input, 1 for output
3 PULDIS = 0 for enabling pull control
= 1 for disabling pull control
4 PULSEL= 0 for pull-down functionality
= 1 for pull-up functionality

Platform SCC/HECC Controller Area Network (CAN)

This document describes the platform controller area network (CAN). The CAN uses established protocol to communicate serially with other controllers in harsh environments. This chapter describes the TMS470Px CAN controllers: the Standard CAN Controller (SCC) and the High-end CAN Controller (HECC). Unless otherwise stated in the text, all information is related to both the SCC and HECC.

Topic **Page**

18.1 CAN Overview

The TMS470Px CAN controller is available in two different implementations that are both fully compliant with the CAN protocol, version 2.0B. The two different CAN controller versions use the same CAN protocol kernel module to perform the basic CAN protocol tasks. Only the message controller differs between the two CAN controller versions. See Table 18-1.

Key features of the CAN module include:

- Common CAN protocol kernel (CPK) to perform protocol tasks
- Standard CAN controller (SCC) for standard CAN applications
  - Sixteen message objects
  - Three receive identifier masks
- High-end CAN controller (HECC) for complex applications
  - Thirty-two message objects
  - Thirty-two receive identifier masks

Table 18-1. SCC Features vs. HECC Features

Feature	SCC	HECC
Number of message objects	16 Rx/Tx	32 Rx/Tx
Number of receive identifier masks	3	32
CAN, version 2.0B compliant	X	X
Low-power mode	X	X
Programmable wake-up on bus activity	X	X
Programmable interrupt scheme	X	X
Automatic reply to a remote request	X	X
Automatic retransmission in case of error	X	X
Protect against reception of new message	X	X
Dual clock support	X	X
32-bit time stamp		X
Local network time counter		X
Programmable priority register for each message		X
Programmable transmission and reception time-out		X

18.1.1 CAN Protocol Processor Features

The CAN protocol kernel (CPK) performs the basic CAN protocol tasks according to CAN protocol specification 2.0B. The CPK is the basic module of all CAN controller implementations.

The CPK provides the following features:

- Full implementation of CAN protocol, version 2.0B
  - Standard and extended identifiers
  - Data and remote frames

- Bus speed of up to 1 Mbps
- Programmable prescaler from 1 to 256
- Programmable bit time according to the CAN specification
- Programmable sampling mode
  - One or three samples used to determine the received bit value
- Selectable edge of receive bit flow for synchronization
  - Both edges or falling edge only
- Automatic retransmission of a frame in case of loss of arbitration
- Low-power mode
- Bus failure diagnostic
  - Bus on/off
  - Error passive/active
  - Bus error warning
  - Bus stuck dominant
  - Frame error report: cyclic redundancy check (CRC), stuff, form, bit, and acknowledgment errors
  - Readable error counters
- Self-test mode
  - Operate in a loop-back mode (receiving its own message)
  - Dummy acknowledge

18.1.2 SCC Features

The TMSx70 device family has the following SCC features:

- All the CPK features: Full implementation of CAN protocol, version 2.0B
- Sixteen message objects, each with the following properties:
  - Configurable as receive or transmit
  - Configurable with standard or extended identifier
  - Protects against reception of new message
  - Supports data and remote frame
  - Composed of 0 to 8 bytes of data
  - Employs a programmable interrupt scheme with two interrupt levels
  - Has a message priority based on object number
- Two programmable local identifier masks for objects 0–2 and 3–5
  - Configurable as standard or extended message identifier
  - Has an acceptance mask register for identifier extension bit
- One global programmable identifier mask for objects 6–15
  - Configurable as standard or extended message identifier
  - Has an acceptance mask register for identifier extension bit
- Low-power mode
- Programmable wake-up on bus activity
- Automatic reply to a remote request message
- Automatic retransmission of a frame in case of loss of arbitration or error
- Dual clock support to avoid CAN bit-timing jitter

18.1.3 HECC Features

The TMS470PVF24xB/34xB has the following HECC features:

- All the CPK features: Full implementation of CAN protocol, version 2.0B
- Thirty-two message objects, each with the following properties:
  - Configurable as receive or transmit
  - Configurable with standard or extended identifier
  - Has a programmable receive mask
  - Supports data and remote frame
  - Composed of 0 to 8 bytes of data
  - Uses a 32-bit time stamp on receive and transmit message
  - Protects against reception of new message
  - Holds the dynamically programmable priority of transmit message
  - Employs a programmable interrupt scheme with two interrupt levels
  - Employs a programmable alarm on transmission or reception time-out
- Low-power mode
- Programmable wake-up on bus activity
- Automatic reply to a remote request message
- Automatic retransmission of a frame in case of loss of arbitration or error
- 32-bit local network time counter synchronized by a specific message (communication in conjunction with mailbox 16)
- SCC-compatible mode
  - Upwardly compatible software
  - Software written for the SCC can run without any changes on the HECC
  - SCC-compatible mode is automatically activated after RESET
- Dual clock support to avoid CAN bit-timing jitter

18.2 Overview of the CAN Network and Module

The controller area network (CAN) uses a serial multimaster communication protocol that efficiently supports distributed real-time control, with a very high level of security, and a communication rate of up to 1 Mbps. The CAN bus is ideal for applications operating in noisy and harsh environments, such as in the automotive and other industrial fields that require reliable communication or multiplexed wiring.

Prioritized messages of up to 8 bytes in data length can be sent on a multimaster serial bus using an arbitration protocol and an error-detection mechanism for a high level of data integrity.

18.2.1 CAN Protocol Overview

The CAN protocol supports four different frame types for communication:

- Data frames that carry data from a transmitter node to the receiver nodes
- Remote frames that are transmitted by a node to request the transmission of a data frame with the same identifier
- Error frames that are transmitted by any node on a bus-error detection
- Overload frames that provide an extra delay between the preceding and the succeeding data frames or remote frames.

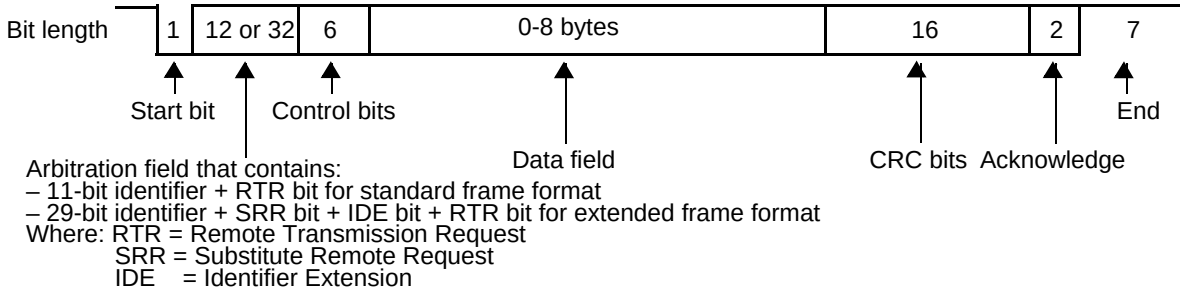
In addition, CAN specification version 2.0B defines two different formats that differ in the length of the identifier field: standard frames with an 11-bit identifier and extended frames with a 29-bit identifier.

CAN standard data frames contain from 44–108 bits and CAN extended data frames contain 64–28 bits. Furthermore, up to 23 stuff bits can be inserted in a standard data frame, and up to 28 stuff bits in an extended data frame, depending on the data-stream coding. The overall maximum data frame length is then 131 bits for a standard frame and 156 bits for an extended frame.

Bit fields within the data frame, shown in Figure 18-1, identify:

- Start of the frame
- Arbitration field containing the identifier and the type of message being sent
- Control field containing the number of data
- Up to 8 bytes of data
- Cyclic redundancy check (CRC)
- Acknowledgment
- End-of-frame bits

Figure 18-1. CAN Data Frame



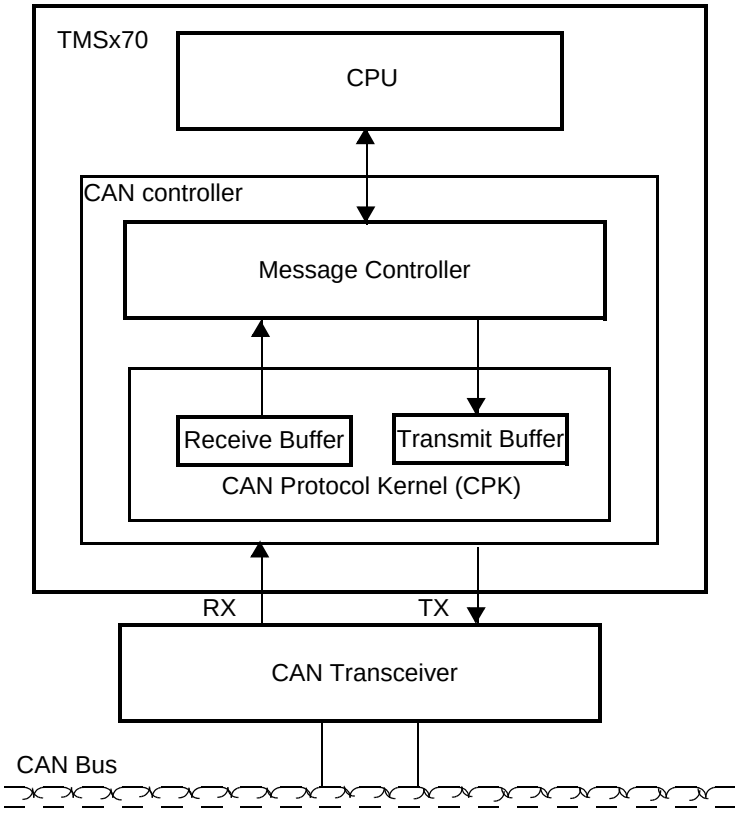
Note: Unless otherwise noted, numbers are the number of bits in the field.

18.2.2 CAN Controller Overview

The TMSx70 CAN controllers provide the CPU with full functionality of the CAN protocol, version 2.0B. The CAN controller minimizes the CPU's load in communication overhead and enhances the CAN standard by providing additional features.

The architecture of both the SCC and the HECC controllers, shown in Figure 18-2, is composed of a CAN protocol kernel (CPK) and a message controller.

Figure 18-2. Architecture of the SCC and HECC CAN Controllers



Two functions of the CPK are to decode all messages received on the CAN bus according to the CAN protocol and to transfer these messages into a receive buffer. Another CPK function is to transmit messages on the CAN bus according to the CAN protocol.

The message controller of a CAN controller is responsible for determining if any message received by the CPK must be preserved for the CPU use or be discarded. At the initialization phase, the CPU specifies to the message controller all message identifiers used by the application. The message controller is also responsible for sending the next message to transmit to the CPK according to the message's priority.

The SCC and the HECC differ only by their message controller— the CPK always offers the same services. The message management of the message controller is not part of the CAN protocol specification.

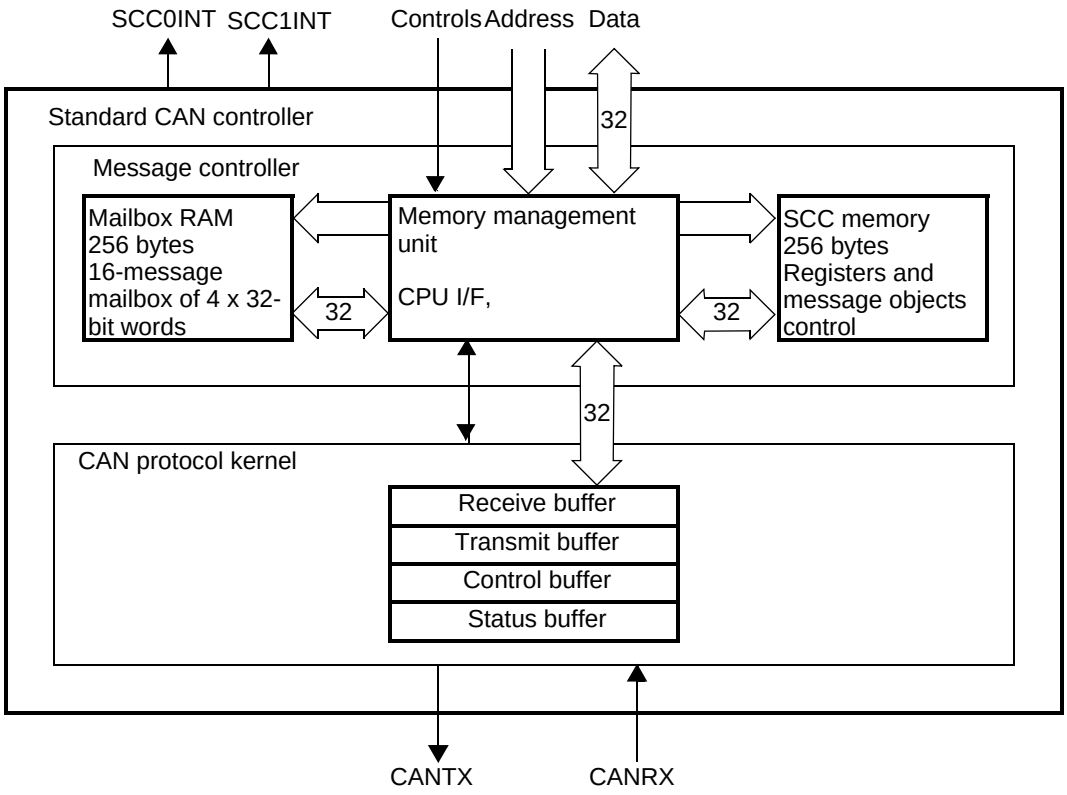
18.3 Standard CAN Controller (SCC) Overview

The SCC is a standard CAN controller with an internal 32-bit architecture, upwardly compatible with HECC.

The SCC, shown in Figure 18-3, consists of the following:

- The CAN protocol kernel (CPK)
- The message controller comprising the following:
  - The memory management unit (MMU), including the CPU interface and the receive control unit (acceptance filtering)
  - 256 bytes of mailbox RAM enabling the storage of 16 messages
  - 256 bytes of space register containing the global and the local identifier masks.

Figure 18-3. SCC Functional Block Diagram



After the reception of a valid message by the CPK, the receive control unit of the message controller determines if the message must be stored into one of the 16 message objects of the mailbox RAM. The receive control unit checks the state, the identifier, and the mask of all message objects to determine the appropriate mailbox location. The received message is stored into the first mailbox passing the acceptance filtering. If the receive control unit could not find any mailbox to store the received message, the message is discarded.

A message comprises an identifier of 11 or 29 bits (contained in the arbitration field), a control field, and up to 8 bytes of data.

When a message must be transmitted, the message controller transfers the message into the transmit buffer of the CPK to start the message transmission at the next bus-idle state. When more than one message must be transmitted, the message with the highest priority is transferred into the CPK by the message controller.

The memory registers contain the global identifier mask and the two dedicated identifier masks for the message objects 0–2 and 3–5. After the reception of a valid identifier, the receive control unit checks the state

and the identifier of all objects to determine if the received message must be stored into one of the message object's buffers.

To initiate a data transfer, the transmission-request bit has to be set in the corresponding control register. The entire transmission procedure and possible error handling is then done without any CPU involvement. If a mailbox has been configured to receive messages, the CPU easily reads its data registers using CPU read instructions. The mailbox may be configured to interrupt the CPU after every successful message transmission or reception.

To avoid jitter on the CAN bit timing caused by using a frequency-modulated PLL to generate the peripheral clock for the CAN module, the CPK clock can be separated from the rest of the peripheral clock domain and can be supplied with a jitter-free clock source. Please refer to section 18.6.1 for details about the dual clock feature.

The separation of the CPK clock from the peripheral clock is controlled by a dedicated bit of the microcontroller. For details about the separation of the CPK clock, please refer to the specific device documentation.

18.3.1 SCC Memory Map

The SCC module has two different offset addresses mapped in the TMS470Px memory.

The first offset address, *SCC_Offset*, is used to access the control and status registers, and the acceptance masks of the message objects. This memory range can be accessed 8-bit, 16-bit, and 32-bit wide.

The second offset address, *Mailbox_Offset,* is used to access the message mailboxes. This memory range can be accessed 8-bit, 16-bit and 32-bit wide. Each of these two memory blocks uses 256 bytes of address space.

The message storage is implemented by a RAM that can be addressed by the CAN controller or the CPU. The CPU controls the CAN controller by modifying the various mailboxes in the RAM or the additional registers. The contents of the various storage elements are used to perform acceptance filtering, message transmission, and interrupt handling.

The mailbox module in the SCC provides sixteen message mailboxes of 8-byte data length, a 29-bit identifier, and several control bits. Each mailbox can be configured to either transmit or receive data. Figure 18-4 shows the memory map.

Note: Unused Message Mailboxes
All RAM areas are byte-writable and may be used as normal memory. In this case, it must be ensured that no CAN function uses the RAM area. This assurance is reached by disabling the corresponding mailbox or by disabling the corresponding functions.

Figure 18-4. SCC Memory Map

1 Mailbox_offset = Mailbox_offset + (Mailbox# x 0x10)

18.3.2 SCC Registers

The SCC and HECC registers, listed in Table 18-2, are used by the CPU to configure and control the CAN controller and the message objects. The start address for the SCC module is 0xFFF7 DF00.

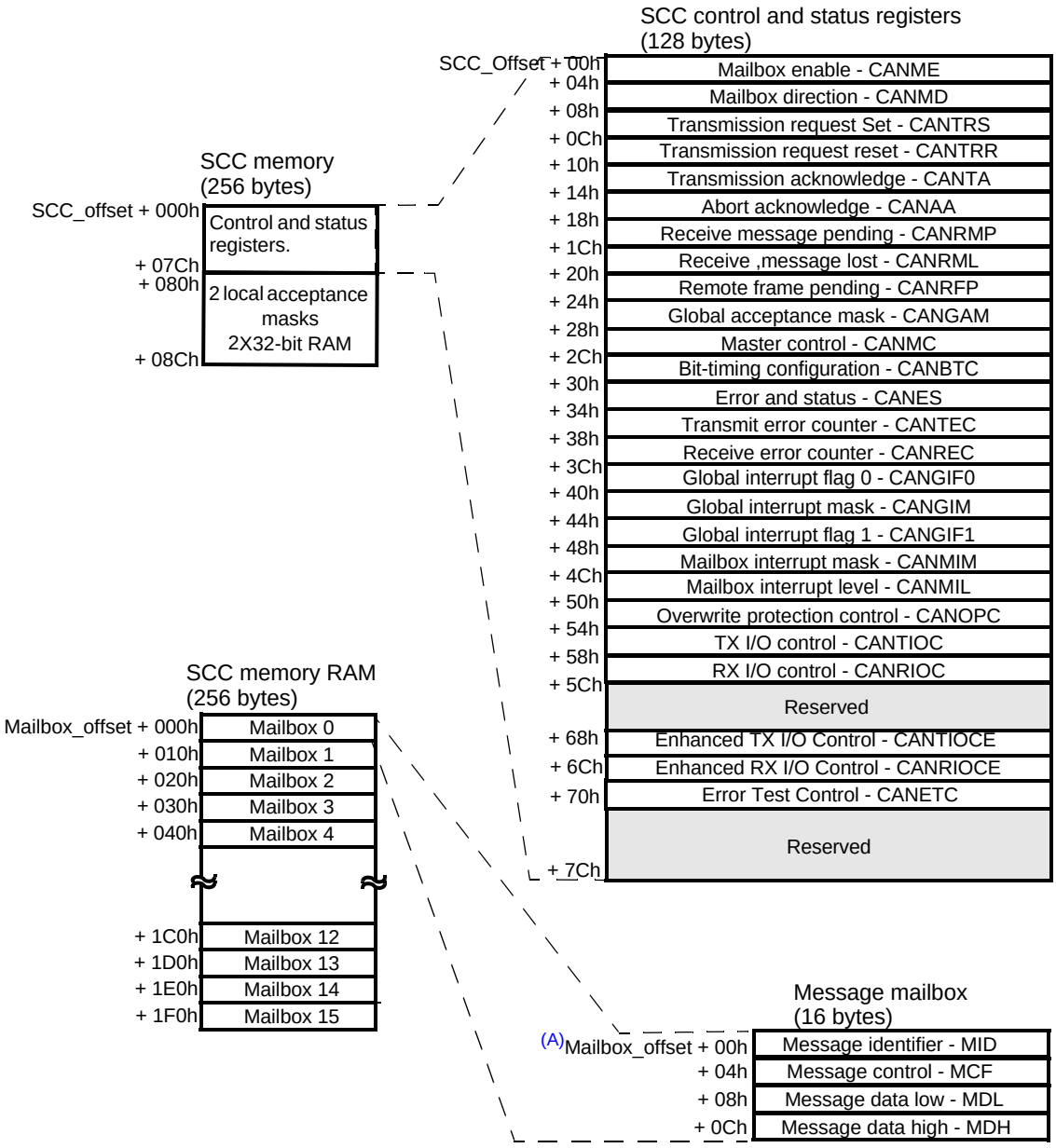


Table 18-2. Combined SCC/HECC Register Map

Offset Address[(1)]	Mnemonic	SCC Name	HECC Name	Section
0x00	CANME	Mailbox enable	Mailbox enable	section 18.11.1
0x04	CANMD	Mailbox direction	Mailbox direction	section 18.11.2
0x08	CANTRS	Transmit request set	Transmit request set	section 18.11.3

1 The actual addresses of the registers are device-specific. See the specific device data sheet to verify the module register addresses. The offset address of the device must be added to the actual register addresses.

Offset Address[1]	Mnemonic	SCC Name	HECC Name	Section
0x0C	CANTRR	Transmit request reset	Transmit request reset	section 18.11.4
0x10	CANTA	Transmission acknowledge	Transmission acknowledge	section 18.11.5
0x14	CANAA	Abort acknowledge	Abort acknowledge	section 18.11.6
0x18	CANRMP	Receive message pending	Receive message pending	section 18.11.7
0x1C	CANRML	Receive message lost	Receive message lost	section 18.11.8
0x20	CANRFP	Remote frame pending	Remote frame pending	section 18.11.9
0x24	CANGAM	Global acceptance map	Reserved	section 18.11.10
0x28	CANMC	Master control	Master control	section 18.11.11
0x2C	CANBTC	Bit-Timing configuration	Bit-timing configuration	section 18.11.12
0x30	CANES	Error and status	Error and status	section 18.11.13
0x34	CANTEC	Transmit error counter	Transmit error counter	section 18.11.14
0x38	CANREC	Receive error counter	Receive error counter	section 18.11.14
0x3C	CANGIF0	Global interrupt flag 0	Global interrupt flag 0	section 18.11.15
0x40	CANGIM	Global interrupt mask	Global interrupt mask	section 18.11.16
0x44	CANGIF1	Global interrupt flag 1	Global interrupt flag 1	section 18.11.15
0x48	CANMIM	Mailbox interrupt mask	Mailbox interrupt mask	section 18.11.17
0x4C	CANMIL	Mailbox interrupt level	Mailbox interrupt level	section 18.11.18
0x50	CANOPC	Overwrite protection control	Overwrite protection control	section 18.11.19
0x54	CANTIOC	TX I/O control	TX I/O control	section 18.11.20
0x58	CANRIOC	RX I/O control	RX I/O control	section 18.11.20
0x5C	CANLNT	Reserved	Local network time	section 18.9.2
0x60	CANTOC	Reserved	Time-out control	section 18.10.2
0x64	CANTOS	Reserved	Time-out status	section 18.10.3
0x68	CANTIOCE	Enhanced TX I/O control	Enhanced TX I/O controlt	Page 1097
0x6C	CANRIOCE	Enhanced RX I/O control	Enhanced RX I/O control	Page 1097
0x70	CANETC	Error Test Control	Error Test Control	section 18.11.22

1 The actual addresses of the registers are device-specific. See the specific device data sheet to verify the module register addresses. The offset address of the device must be added to the actual register addresses.

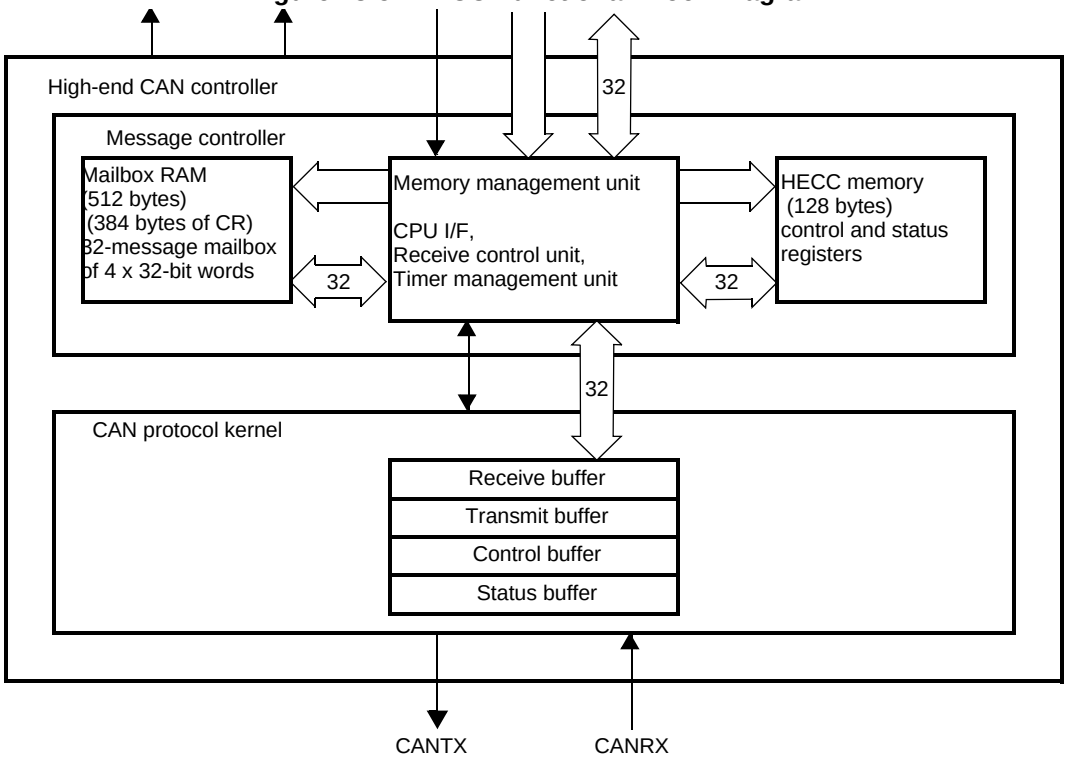
18.4 High-End CAN Controller (HECC) Overview

The HECC is a new-generation, TI, advanced CAN controller with an internal 32-bit architecture.

The HECC, shown in Figure 18-5, consists of the following:

- The CAN protocol kernel (CPK)
- The message controller comprising the following:
  - The memory management unit (MMU), including the CPU interface and the receive control unit (acceptance filtering), and the timer management unit
  - 512 bytes of mailbox RAM enabling the storage of 32 messages
  - 128 bytes of memory comprising the registers and the message objects control
  - 384 bytes of control RAM comprising the message objects control registers

Figure 18-5. HECC Functional Block Diagram



After the CPK receives a valid message, the receive control unit of the message controller determines if the received message must be stored into one of the 32 message objects of the mailbox RAM. The receive control unit checks the state, the identifier, and the mask of all message objects to determine the appropriate mailbox location. The received message is stored into the first mailbox passing the acceptance filtering. If the receive control unit cannot find any mailbox to store the received message, the message is discarded.

A message comprises an 11- or 29-bit identifier, a control field, and up to 8 bytes of data.

When a message must be transmitted, the message controller transfers the message into the transmit buffer of the CPK to start the message transmission at the next bus-idle state. When more than one message must be transmitted, the message with the highest priority (defined by the message-object-priority register) that is ready to be transmitted is transferred into the CPK by the message controller.

The timer management unit comprises a local network time counter and affixes a time stamp to all messages received or transmitted. The timer management unit controls all message reception and transmission, and generates an alarm when a message has not been received or transmitted during an allowed period of time (time-out).

To initiate a data transfer, the transmission request bit has to be set in the corresponding control register. The entire transmission procedure and possible error handling are then performed without any CPU involvement. If a mailbox has been configured to receive messages, the CPU easily reads its data registers using CPU read instructions. The mailbox may be configured to interrupt the CPU after every successful message transmission or reception.

To avoid jitter on the CAN bit timing caused by using a frequency-modulated PLL to generate the peripheral clock for the CAN module, the CPK clock can be separated from the rest of the peripheral clock domain and can be supplied with a jitter-free clock source. Please refer to section 18.6.1 for details about the dual clock feature.

The separation of the CPK clock from the peripheral clock domain is implemented by a dedicated bit of the microcontroller. For details about the separation of the CPK clock, please refer to the specific device documentation.

18.4.1 SCC-Compatible Mode

The HECC can be used in SCC mode. In this mode, all functions specific to the HECC are not available and the HECC behaves exactly as the SCC does. This mode is selected by default to allow any application software written for the SCC to run on the HECC without any modification.

The SCC-compatible mode is selected with bit SCM (MC.13).

Note: HECC Used in SCC Mode
When the HECC is configured in SCC mode, the HECC behaves exactly as the SCC does, and you should refer to the SCC specification only.

18.4.2 HECC Memory Map

The HECC module has two different offset addresses mapped in the TMS470Px memory.

The first offset address, *HECC_Offset*, is used to access the control register and the status register. The access to the control and status registers (memory range: *HECC_Offset ... HECC_Offset + 0x07C*) is 8-bit, 16-bit and 32-bit wide. The HECC control and status registers, shown in Figure 18-6, uses 128 bytes of address space.

The second offset address, *Mailbox_Offset,* is used to access the mailboxes. This memory range can be accessed 8-bit, 16-bit, and 32-bit wide. The acceptance mask, time stamp, and time-out of the message objects are implemented in a control RAM block, which has a base address of Mailbox_Offset + 0x1000. This memory range can also be accessed 8-bit, 16-bit, and 32-bit wide.

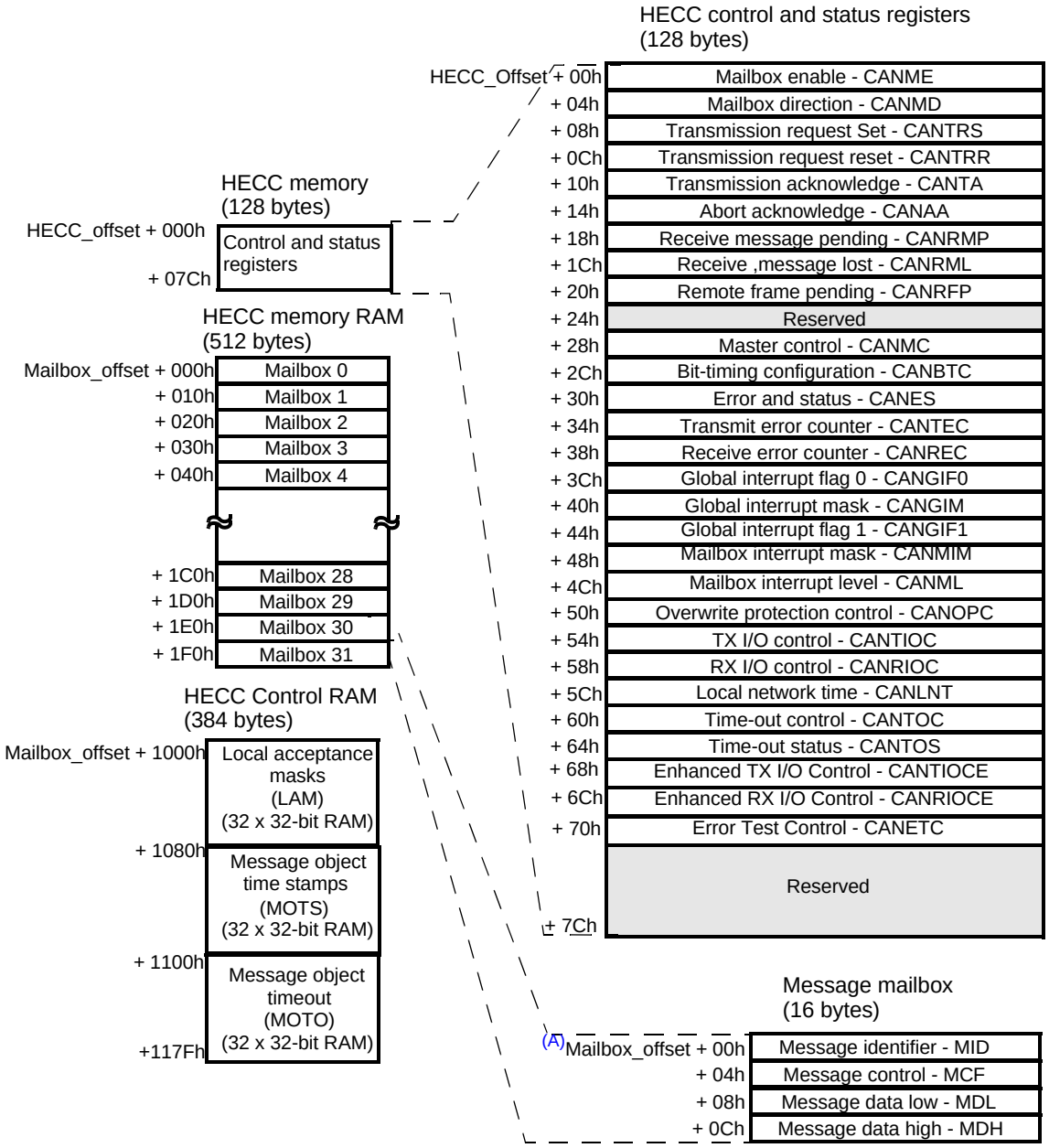
The message storage is implemented by a RAM that can be addressed by the CAN controller or the CPU. The CPU controls the CAN controller by modifying the various mailboxes in the RAM or the additional registers. The contents of the various storage elements are used to perform the functions of the acceptance filtering, message transmission, and interrupt handling.

The mailbox module in the HECC provides 32 message mailboxes of 8-byte data length, a 29-bit identifier, and several control bits. Each mailbox can be configured as either transmit or receive. In the HECC, each mailbox has an individual acceptance mask.

Note: Unused Message Mailboxes
All RAM areas are byte-writable and can be used as normal memory. In this case, it must be ensured that no CAN function uses the RAM area. This assurance is obtained by disabling the corresponding mailbox or by disabling the corresponding functions.

Note: HECC Used in SCC Mode
When the HECC is used in SCC mode, you should refer to the SCC memory map in Figure 18-4.

Figure 18-6. HECC Memory Map



1 Mailbox_offset = Mailbox_offset + (Mailbox# x 0x10)

18.4.3 HECC Registers

The SCC and HECC registers listed in Table 18-3 are used by the CPU to configure and control the CAN controller and the message objects. The start address of the HECC module is 0xFFF7 DC00.

Table 18-3. Combined SCC/HECC Register Map

Offset Address[1]	Mnemonic	SCC Name	HECC Name	Section
0x00	CANME	Mailbox enable	Mailbox enable	section 18.11.1
0x04	CANMD	Mailbox direction	Mailbox direction	section 18.11.2
0x08	CANTRS	Transmit request set	Transmit request set	section 18.11.3
0x0C	CANTRR	Transmit request reset	Transmit request reset	section 18.11.4
0x10	CANTA	Transmission acknowledge	Transmission acknowledge	section 18.11.5
0x14	CANAA	Abort acknowledge	Abort acknowledge	section 18.11.6
0x18	CANRMP	Receive message pending	Receive message pending	section 18.11.7
0x1C	CANRML	Receive message lost	Receive message lost	section 18.11.8
0x20	CANRFP	Remote frame pending	Remote frame pending	section 18.11.9
0x24	CANGAM	Global acceptance map	Reserved	section 18.11.10
0x28	CANMC	Master control	Master control	section 18.11.11
0x2C	CANBTC	Bit-Timing configuration	Bit-timing configuration	section 18.11.12
0x30	CANES	Error and status	Error and status	section 18.11.13
0x34	CANTEC	Transmit error counter	Transmit error counter	section 18.11.14
0x38	CANREC	Receive error counter	Receive error counter	section 18.11.14
0x3C	CANGIF0	Global interrupt flag 0	Global interrupt flag 0	section 18.11.15
0x40	CANGIM	Global interrupt mask	Global interrupt mask	section 18.11.16
0x44	CANGIF1	Global interrupt flag 1	Global interrupt flag 1	section 18.11.15
0x48	CANMIM	Mailbox interrupt mask	Mailbox interrupt mask	section 18.11.17
0x4C	CANMIL	Mailbox interrupt level	Mailbox interrupt level	section 18.11.18
0x50	CANOPC	Overwrite protection control	Overwrite protection control	section 18.11.19
0x54	CANTIOC	TX I/O control	TX I/O control	section 18.11.20
0x58	CANRIOC	RX I/O control	RX I/O control	section 18.11.20
0x5C	CANLNT	Reserved	Local network time	section 18.9.2
0x60	CANTOC	Reserved	Time-out control	section 18.10.2
0x64	CANTOS	Reserved	Time-out status	section 18.10.3
0x68	CANTIOCE	Enhanced TX I/O control	Enhanced TX I/O control	section 18.11.21
0x6C	CANRIOCE	Enhanced RX I/O control	Enhanced RX I/O control	section 18.11.21
0x70	CANETC	Error Test Control	Error Test Control	section 18.11.22

1 The actual addresses of the registers are device-specific. See the specific device data sheet to verify the module register addresses. The offset address of the device must be added to the actual register addresses.

18.5 Message Objects

The following sections describe the SCC message objects, the HECC message objects, the CAN message mailbox, and the CAN acceptance filter.

18.5.1 SCC Message Objects

The message controller of the SCC can handle 16 different message objects.

Each message object can be configured to either transmit or receive. In the SCC, message objects 0–2, 3–5, and 6–15 share the same acceptance masks.

A SCC message object consists of 20 bytes of RAM distributed, as shown in Table 18-4, as:

- A message mailbox comprising the following:
  - The 29-bit message identifier
  - The message control field register
  - 8 bytes of message data
- A 29-bit acceptance mask

Furthermore, corresponding control and status bits located in the SCC registers allow control of the message objects.

Table 18-4. RAM Distribution in SCC Message Object

Offset / Address(1)	Mnemonic	Name	Section
Mailbox_Offset + (Object# x 0x10) + 0x00	MID	Message identifier	Section 18.5.3.5
Mailbox_Offset + (Object# x 0x10) + 0x04	MCF	Message control field	Section 18.5.3.6
Mailbox_Offset + (Object# x 0x10) + 0x08	MDL	Message data low word	Section 18.5.3.7
Mailbox_Offset + (Object# x 0x10) + 0x0C	MDH	Message data high word	Section 18.5.3.7
SCC_Offset + 0x80 (for objects 0 to 2) or	LAM(0)	Local acceptance mask	Section 18.5.4.1
SCC_Offset + 0x8C (for objects 3 to 5) or	LAM(3)	Local acceptance mask	Section 18.5.4.1
SCC_Offset + 0x24 (for objects 6 to 15)	GAM	Global acceptance mask	Section 18.11.10

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the SCC module memory offset and RAM offset. Object# specifies the number of the message object.

Note: Unused Message Mailboxes
Message mailboxes not used by the application for CAN message (disabled in the CANME register) may be used as general memory by the CPU.

18.5.2 HECC Message Objects

The message controller of the HECC can handle 32 different message objects.

Each message object can be configured to either transmit or receive. In the HECC, each message object has an individual acceptance mask.

A HECC message object consists of 28 bytes of RAM distributed, as shown in Table 18-5, as:

- A message mailbox comprising the following:
  - The 29-bit message identifier
  - The message control field registers

  - 8 bytes of message data
- A 29-bit acceptance mask
  - A 32-bit time stamp
  - A 32-bit time-out

Furthermore, corresponding control and status bits located in the HECC registers allow control of the message objects.

Table 18-5. RAM Distribution in HECC Message Object

Offset / Address[1]	Mnemonic	Name	Section
Mailbox_RAM_Offset + (Object# x 0x10) + 0x00	MID	Message identifier	Section 18.5.3.5
Mailbox_RAM_Offset + (Object# x 0x10) + 0x04	MCF	Message control field	Section 18.5.3.6
Mailbox_RAM_Offset + (Object# x 0x10) + 0x08	MDL	Message data low word	Section 18.5.3.7
Mailbox_RAM_Offset + (Object# x 0x10) + 0x0C	MDH	Message data high word	Section 18.5.3.7
HECC_Offset + (Object# x 4) + 0x1000	LAM	Local acceptance mask	Section 18.5.4.3
HECC_Offset + (Object# x 4) + 0x1080	MOTS	Message object time stamp	Section 18.9.3
HECC_Offset + (Object# x 4) + 0x1100	MOTO	Message object time-out	Section 18.10.1

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the HECC module memory offset and RAM offset.

Note: HECC Used in SCC Mode
When the HECC is used in SCC mode, you must refer to the SCC message objects description (see Table 18-4).

Note: Unused Message Mailboxes
Message mailboxes not used by the application for CAN messages (disabled in the CANME register) may be used as general memory by the CPU.

18.5.3 CAN Message Mailbox

The message mailboxes are the RAM area where the CAN messages are actually stored after they were received or before they are transmitted.

The CPU may use the RAM area of the message mailboxes that are not used for storing messages as normal memory. This RAM area, unlike the register area, can be accessed by byte.

Each mailbox contains:

- The message identifier
  - 29 bits for extended identifier
  - 11 bits for standard identifier
- The identifier extension bit, IDE (MID.31)
- The acceptance mask enable bit, AME (MID.30)
- The auto answer mode bit, AAM (MID.29)
- The remote transmission request bit, RTR (MCF.4)
- The data length code, DLC (MCF.3–0)

- Up to eight bytes for the data field
- A transmit priority level, TPL, register on the HECC (MCF.13:8)

Each of the mailboxes can be configured as one of four message object types (see Table 18-6). Transmit and receive message objects are used for data exchange between one sender and multiple receivers (1–*n* communication link), whereas request and reply message objects are used to set up a one-to-one communication link.

Table 18-6. Message Object Behavior Configuration

Message Object Behavior	Mailbox Direction Register (CANMD)	Auto-Answer Mode Bit (AAM)	Remote Transmission Request Bit (RTR)
Transmit message object	0	0	0
Receive message object	1	0	0
Request message object	1	0	1
Reply message object	0	1	0

18.5.3.1 Transmit Mailbox

The CPU stores the data to be transmitted in a mailbox configured as transmit mailbox. After writing the data and the identifier into the RAM, the message is sent if the corresponding TRS[*n*] bit has been set.

If more than one mailbox is configured as a transmit mailbox and more than one corresponding TRS[*n*] is set, the messages are sent one after another in falling order, beginning with the mailbox with the highest priority.

In the SCC, the priority of the mailbox transmission depends on the mailbox number. The highest mailbox number (15) has the highest transmit priority.

In the HECC, the priority of the mailbox transmission depends on the setting of the TPL field in the message control field (CANMCF) register. The mailbox with the highest value in the TPL is transmitted first. Only when two mailboxes have the same value in the TPL register is the higher numbered mailbox transmitted first. See section 18.5.3.6 for more information about the CANMCF.

If a transmission fails because of a loss of arbitration or an error, the message transmission will be reattempted. Before reattempting the transmission, the CAN module checks if other transmissions are requested and then transmits the mailbox with the highest priority.

18.5.3.2 Receive Mailbox

The identifier of each incoming message is compared to the identifiers held in the receive mailboxes using the appropriate mask. When equality is detected, the received identifier, the control bits, and the data bytes are written into the matching RAM location. At the same time, the corresponding receive-message-pending bit (RMP[*n*] where RMP.31–0), is set and a receive interrupt is generated if enabled. If no match is detected, the message is not stored.

When a message is received, the message controller starts looking for a matching mailbox at the mailbox with the highest mailbox number. Mailbox 15 of the SCC and of the HECC in SCC compatible mode has the highest receive priority; mailbox 31 has the highest receive priority of the HECC in HECC mode.

RMP[*n*] (RMP.31–0) has to be reset by the CPU after reading the data. If a second message has been received for this mailbox and the RMP bit is already set, the corresponding message-lost bit (RML[*n*], where RML.31–0) is set. In this case, the stored message is overwritten with the new data if the overwrite-protection bit (OPC[*n*], where OPC.31-0) is cleared; otherwise, the next mailboxes are checked.

18.5.3.3 Handling of Remote Frames

If a remote frame is received (the incoming message has RTR (MCF.4) = 1), the CAN module compares the identifier to all identifiers of the mailboxes using the appropriate masks starting at the highest mailbox number in descending order.

In the case of a matching identifier (with the message object configured as send mailbox and AAM (MID.29) in this message object is set) this message object is marked as to be sent (TRS[n] is set).

In the case of a matching identifier (with the message object configured as send mailbox and bit AAM in this message object is not set) this message is not received.

After finding a matching identifier in a send mailbox, no further comparison is done.

In the case of a matching identifier and the message object configured as receive mailbox, this message is handled like a data frame and the corresponding bit in the receive message pending (CANRMP) register is set. The CPU then has to decide how to handle this situation. See section 18.11.7 for information about the CANRMP register.

If the CPU wants to change the data in a message object that is configured as remote frame mailbox (AAM (MID.29) = 1), it has to set the mailbox number MBNR (MC.4-0) and the change data request bit CDR (MC.8) first. The CPU may then perform the access and clear CDR to tell the CAN module that the access is finished. Until CDR is cleared, the transmission of this mailbox is not performed by the CAN module. Since TRS (TRS.31-0) is not affected by CDR, a pending transmission is started after CDR is cleared. Thus, the newest data will be sent.

To change the identifier in that mailbox, the message object first must be disabled (ME[*n*] = 0).

If the CPU wants to request data from another node, it may configure the message object as receive mailbox and set TRS. In this case, the module sends a remote frame request and receives the data frame in the same mailbox that sent the request. Therefore, only one mailbox is necessary to do a remote request. Note that the CPU must set RTR (MCF.4) to enable a remote frame transmission.

The behavior of the message object *n* is configured with MD[*n*] (MD.31-0), the AAM (MID.29), and RTR (MCF.4). It shows how to configure a message object according to the desired behavior.

To summarize, a message object can be configured with four different behaviors:

- A transmit message object is only able to transmit messages.
- A receive message object is only able to receive messages.
- A request message object is able to transmit a remote request frame and to wait for the corresponding data frame.
- A reply message object is able to transmit a data frame whenever a remote request frame is received for the corresponding identifier.

Note: Remote Transmission Request Bit

When a remote transmission request is successfully transmitted with a message object configured in request mode, the CANTA register is not set and no interrupt is generated. When the remote reply message is received, the behavior of the message object is the same as a message object configured in receive mode.

18.5.3.4 CPU Message Mailbox Access

Write accesses to the identifier can only be accomplished when the mailbox is disabled (ME[*n*] (ME.31-0) = 0). During access to the data field, it is critical that the data does not change while the CAN module is reading it. Hence, a write access to the data field is disabled for a receive mailbox.

For send mailboxes, an access is usually denied if the TRS (TRS.31–0) or the TRR (TRR.31-0) flag is set. In these cases, an interrupt may be asserted. A way to access those mailboxes is to set CDR (MC.8) before accessing the mailbox data.

After the CPU access is finished, the CPU must clear the CDR flag by writing a 0 to it. The CAN module checks for that flag before and after reading the mailbox. If the CDR flag is set during those checks, the CAN module does not transmit the message but continues to look for other transmit requests. The setting of the CDR flag also stops the write-denied interrupt (WDI) from being asserted.

18.5.3.5 Message Identifier Register (MID)

This register can only be written if mailbox *n* is disabled (ME[*n*] (ME.31–0) = 0). Figure 18-7 and Table 18-6 describe this register.

Figure 18-7. Message Identifier Register (MID) [offset = 00h](1)

31	30	29	28 ... 18	17 ... 16
IDE	AME	AAM	ID.28:18	ID.17:16
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x

15 ... 0
ID.15:0
R/W-x

R = Read any time; W = write when mailbox is disabled; x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)

Table 18-7. Message Identifier Register (MID) Field Descriptions

Bit	Name	Value	Description
31	**IDE**		Identifier extension.
		0	Standard identifier (11 bits). The received message or the message to be sent has a standard identifier.
		1	Extended identifier (29 bits). The received message or the message to be sent has an extended identifier.
30	AME		Acceptance mask enable. AME is only used for receiver mailboxes. It must not be set for automatic reply (AAM[*n*]=1, MD[*n*]=0) mailboxes, otherwise the mailbox behavior is undefined. This bit is not modified by a message reception.
		0	No acceptance mask will be used, all identifier bits must match to receive the message.
		1	The corresponding acceptance mask is used.
29	AAM		Auto answer mode. This bit is only valid for message mailboxes configured as transmit. For receive mailboxes, this bit has no effect: the mailbox is always configured for normal receive operation. This bit is not modified by a message reception.
		0	Normal transmit mode. The mailbox does not reply to remote requests. The reception of a remote request frame has no effect on the message mailbox.
		1	Auto answer mode. If a matching remote request is received, the CAN module answers to the remote request by sending the contents of the mailbox.

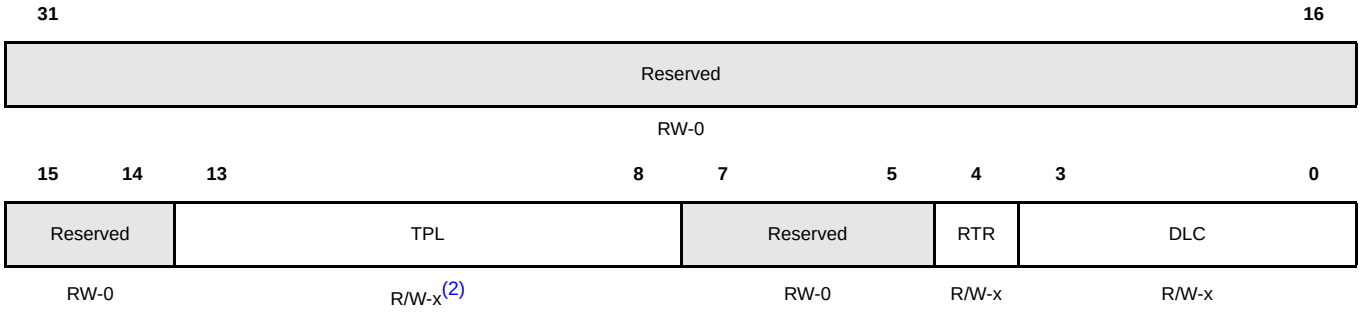
Table 18-7. Message Identifier Register (MID) Field Descriptions (Continued)

Bit	Name	Value	Description
28–0	ID(28–0)		Message identifier. In standard identifier mode, if the IDE bit (MID.31 = 0), the message identifier is stored in bits ID(28–18). In this case, bits ID(17–0) have no meaning. In extended identifier mode, if the IDE bit [MID(31) = 1], the message identifier is stored in bits ID(28–0).

18.5.3.6 Message Control Field Register (MCF)

The register MCF(*n*) can only be written if mailbox *n* is configured for transmission (MD[*n*] = 0) or if the mailbox is disabled (ME[*n*] = 0). Figure 18-8 and Table 18-8 describe this register.

Figure 18-8. Message Control Field Register (MCF) [offset = 04h][1]



R = Read any time; W = Write when mailbox is disabled or configured for transmission; -*n* = Value after reset; x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)
2 HECC only, reserved in the SCC

Table 18-8. Message Control Field Register (MCF) Field Descriptions

Bit	Name	Value	Description
31–14	Reserved		Reserved for future enhancement.
13–8	TPL(5–0)		Transmit priority level. This 6-bit field in the HECC (reserved in the SCC) defines the priority of this mailbox as compared to the other 31 mailboxes. The highest number has the highest priority. In the case that two mailboxes have the same priority, the one with the higher mailbox number is transmitted. TPL applies only for transmit mailboxes. TPL is not used when in SCC-compatibility mode.
7–5	Reserved		Reserved for future enhancement.
4	RTR		Remote transmission request.
		0	No remote frame is requested.
		1	*For receive mailbox:* If the TRS flag is set, a remote frame is transmitted and the corresponding data frame will be received in the same mailbox. *For transmit mailbox:* If the TRS flag is set, a remote frame is transmitted, but the corresponding data frame has to be received in another mailbox.
3–0	DLC(3–0)	0–Fh	Data length code. The number in these bits determines how many data bytes are sent or received. Valid value range is from 0–8. Values from 9–15 are not allowed.

18.5.3.7 Message Data Registers (MDL, MDH)

Eight bytes of the mailbox are used to store the data field of a CAN message. The setting of DBO (MC.10) determines the ordering of stored data.

The data is transmitted or received from the CAN bus, starting with byte 0.

- When DBO (MC[10]) = 1, the data is stored or read starting with the least significant byte of the CANMDL register and ending with the most significant byte of the CANMDH register.
- When DBO (MC[10]) = 0, the data is stored or read starting with the most significant byte of the CANMDL register and ending with the least significant byte of the CANMDH register.

The registers MDL[*n*] and MDH[*n*] can only be written if mailbox *n* is configured for transmission (MD[*n*] = 0) or the mailbox is disabled (ME[*n*] = 0).

If TRS[*n*] = 1, the registers MDL[*n*] and MDH[*n*] cannot be written, unless CDR (MC[8] = 1, with MBNR (MC[4-0] set to *n.* These settings also apply for a message object configured in reply mode (AAM (MID[29] = 1). Figure 18-9 through Figure 18-12 illustrate these registers.

Figure 18-9. Message Data Low Register with DBO = 0 (MDL) [offset = 08h](1)

31 24	23 16	15 8	7 0
Byte 0	Byte 1	Byte 2	Byte 3
R/W-x		R/W-x	

RW = Read any time, Write when mailbox is disabled or configured for transmission. -*n* = Value after reset, x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)

Figure 18-10. Message Data High Register with DBO = 0 (MDH) [offset = 0Ch](1)

31 24	23 16	15 8	7 0
Byte 4	Byte 5	Byte 6	Byte 7
R/W-x		R/W-x	

R = Read any time; W = Write when mailbox is disabled or configured for transmission; x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)

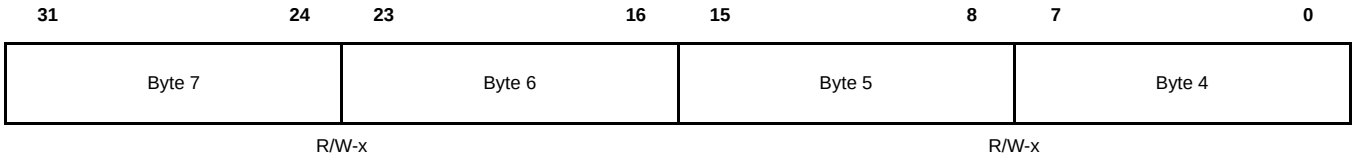
Figure 18-11. Message Data Low Register with DBO = 1 (MDL) [offset = 08h](1)

31 24	23 16	15 8	7 0
Byte 3	Byte 2	Byte 1	Byte 0
R/W-x		R/W-x	

R = Read any time; W = Write when mailbox is disabled or configured for transmission; x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)

Figure 18-12. Message Data High Register with DBO = 1 (MDH) [offset = 0Ch](1)



R = Read any time; W = Write when mailbox is disabled or configured for transmission; x = indeterminate

1 Relative address = Mailbox_Offset + (Object# x 0x10)

Note: Data Field
The data field beyond the valid received data is modified by any message reception and is indeterminate.

18.5.4 CAN Acceptance Filter

The SCC and the HECC implement the handling of the masks for the mailboxes differently. If the SCC compatibility bit is in the CANMC register is set, the HECC behaves like the SCC.

The identifier of the incoming message is first compared to the message identifier of the mailbox (which is stored in the mailbox). Then the appropriate acceptance mask is used to mask out the bits of the identifier that should not be compared.

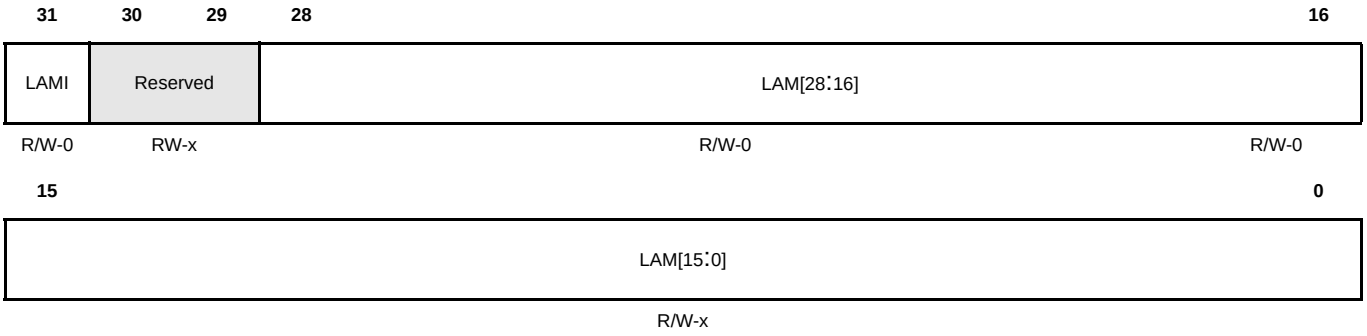
18.5.4.1 SCC Acceptance Filtering

In the SCC (or in the HECC in SCC-compatible mode), the global acceptance mask (CANGAM) is used for the mailboxes 15–6. An incoming message is stored in the highest numbered mailbox with a matching identifier. If there is no matching identifier in message objects 15–6, the incoming message is compared to the identifiers for message objects 5–3 and then 2–0.

The message objects 5–3 use the local acceptance mask LAM(3) of the SCC registers and the message objects 2–0 use the local acceptance mask LAM(0) of the SCC registers. After a hardware or software reset, LAM(0) and LAM(3) are reset to zero. See section 18.5.4.3 for specific uses.

Figure 18-13 and Table 18-9 describe the SCC local-acceptance-mask registers.

Figure 18-13. *SCC Local Acceptance Mask Register (LAM) [offset = 80h or 8Ch]*[1]



R = Read; W = Write; -*n* = Value after reset; x = indeterminate

1 Relative address = + SCC_Offset + 0x80 + (Object# x 4)

Table 18-9. SCC Local Acceptance Mask Register (LAM) Field Descriptions

Bit	Name	Value	Description
31	LAMI		Local acceptance mask identifier extension.
		0	The identifier extension bit stored in the mailbox determines which messages shall be received.
		1	Standard and extended frames can be received. In case of an extended frame, all 29 bits of the identifier are stored in the mailbox and all 29 bits of the local acceptance mask register are used for the filter. In case of a standard frame, only the first eleven bits (bits 28–18) of the identifier and the local acceptance mask are used.
30–29	Reserved		Reads return 0 and writes have no effect.

Table 18-9. SCC Local Acceptance Mask Register (LAM) Field Descriptions

28–0	LAM[28:0]		Local acceptance mask. These bits enable the masking of any identifier bit of an incoming message.
		0	Received identifier bit value must match the corresponding identifier bit of the MID register.
		1	Accept a 0 or a 1 (don't care) for the corresponding bit of the received identifier.

18.5.4.2 HECC Acceptance Filtering

Each of the 32 mailboxes of the HECC has its own local acceptance mask LAM(0) to LAM(31). There is no global acceptance mask in the HECC.

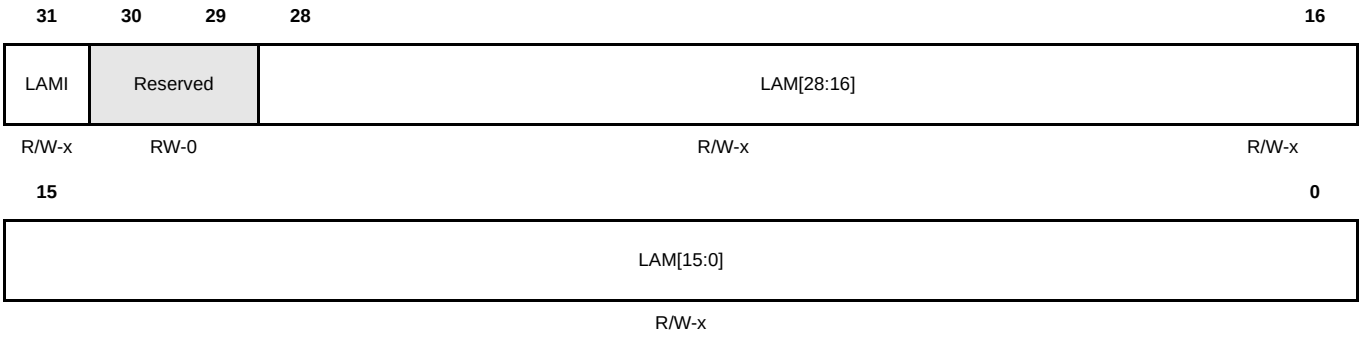
18.5.4.3 HECC Local Acceptance Mask Register (LAM)

The HECC local acceptance filtering allows the user to locally mask (don't care) any identifier bits of the incoming message.

After a hardware or a software reset of the HECC, the LAM registers are not modified.

In the HECC, each mailbox (0–31) has its own mask register, LAM[0] to LAM[31]. An incoming message is stored in the highest numbered mailbox with a matching identifier. Figure 18-14 and Table 18-10 describe this register.

Figure 18-14. HECC Local Acceptance Mask Register (LAM) [offset = 1000h][1]



R = Read; W = Write; -*n* = Value after reset; x = indeterminate

1 Relative address = + HECC_Mailbox_Offset + 0x1000 + (Object# x 4)

Table 18-10. HECC Local Acceptance Mask Register (LAM) Field Descriptions

Bit	Name	Value	Description
31	LAMI		Local acceptance mask identifier extension.
		0	The identifier extension bit stored in the mailbox determines which messages shall be received.
		1	Standard and extended frames can be received. In case of an extended frame, all 29 bits of the identifier are stored in the mailbox and all 29 bits of the local acceptance mask register are used for the filter. In case of a standard frame, only the first eleven bits (bits 28–18) of the identifier and the local acceptance mask are used.
30–29	Reserved		Reserved for future enhancement.
28–0	LAM[28:0]		Local acceptance mask. These bits enable the masking of any identifier bit of an incoming message.
		0	Received identifier bit value must match the corresponding identifier bit of the MID register.
		1	Accept a 0 or a 1 (don't care) for the corresponding bit of the received identifier.

18.6 CAN Module Initialization

The CAN module must be initialized before the utilization. Initialization is only possible if the module is in initialization mode. See Figure 18-15.

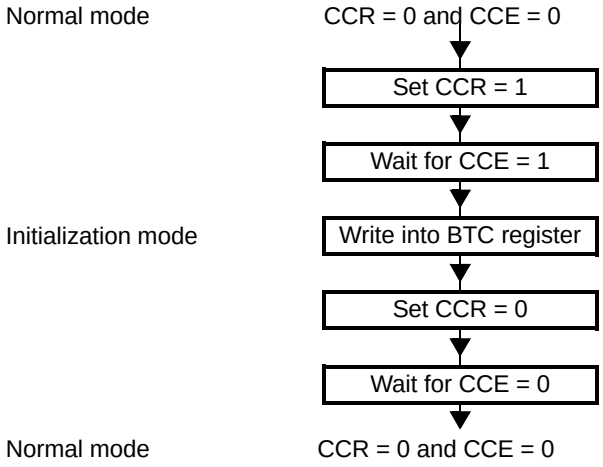
Programming CCR (MC[12]) = 1 sets the initialization mode. The initialization can be performed only when CCE (ES[4]) = 1. Afterwards, the configuration registers may be written.

The module is activated again by programming CCR (MC[12]) = 0.

After hardware reset, the initialization mode is active.

Note: Bit-Timing Configuration (CANBTC) Register With 0 Value
If the CANBTC register is programmed with a 0 value, or left with the initial value, the CAN module will never leave the initialization mode, i.e., the CCE (ES[4]) bit will remain at 1 when clearing the CCR bit.

Figure 18-15. Configuration Sequence



Note: Enter/Exit Initialization Mode
The transition between initialization mode and normal mode and vice-versa is performed in synchronization with the CAN network. That is, the CAN controller waits until it detects a bus idle sequence (11 recessive bits) before it changes the mode. If a stuck-to-dominant bus error occurs, the CAN controller can't detect a bus-idle condition and therefore is unable to perform a mode transition.

18.6.1 Dual Clock Support

Two seperate clock domains are implemented in the the SCC/HECC modules: the peripheral clock domain (P_CLK) and the CAN protocol kernel clock domain (CPK_CLK). Typically, the source for both clock domains is the peripheral clock derived from the PLL clock output. If frequency modulation of the PLL is enabled, then, due to the high precision clocking requirements of the CAN kernel, a separate clock without any modulation has to be applied to the CPK_CLK domain. Therefore the CPK_CLK has to be configured to be derived from a non frequency modulated clock source, like i.e. from the OSCIN clock. Please see the system module reference guide and the device datasheet for more information how to configure the relevant clock sources registers in the system module.

Note: Note: Enable Dual Clock Support
If the dual clock functionality is used, then P_CLK must always be higher or equal to CPK_CLK, in order to guarantee proper synchronization between the two clock domains. Here also the frequency shift of the modulated P_CLK must be considered.

18.6.2 CAN Bit-Timing Configuration

As shown in Figure 18-16, the CAN protocol specification partitions the nominal bit time into four different time segments:

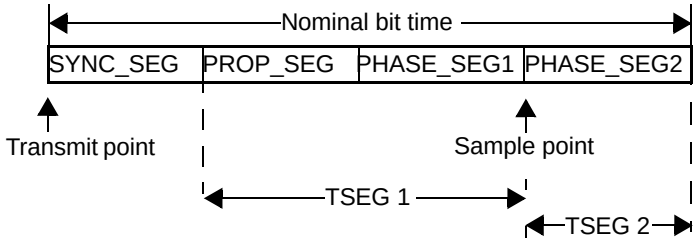
SYNC_SEG: this part of bit time is used to synchronize the various nodes on the bus. An edge is expected to lie within this segment. This segment is always 1 time quantum (TQ).

PROP_SEG: this part of the bit time is used to compensate for the physical delay times within the network. It is twice the sum of the signal's propagation time on the bus line, the input comparator delay, and the output driver delay. This segment is programmable from 1–8 time quanta (TQ).

PHASE_SEG1: this phase is used to compensate for positive edge phase error. This segment is programmable from 1–8 time quanta (TQ) and can be lengthened by resynchronization.

PHASE_SEG2: this phase is used to compensate for negative edge phase error. This segment is programmable from 2–8 time quanta (TQ) and can be shortened by resynchronization.

Figure 18-16. Partition of the Bit Time



All controllers on a CAN bus must have the same bit rate and bit length. At different clock frequencies of the individual controllers, the bit rate has to be adjusted by the time segments.

In the SCC/HECC modules, the length of a bit on the CAN bus is determined by the parameters TSEG1 [BTC(6–3)], TSEG2 [BTC(2–0)], and BRP [BTC(23–16)]. *BRP* is the binary value of BRP(7–0) + 1.

TSEG1 combines the two time segments PROP_SEG and PHASE_SEG1 as defined by the CAN protocol. TSEG2 defines the length of the time segment PHASE_SEG2.

The following bit timing rules have to be fulfilled when determining the bit segment values:

- $TSEG1_{CALC(min)} \geq TSEG2_{CALC}$
- Information processing time (IPT) $\leq TSEG1_{CALC} \leq 16$ TQ
- IPT $\leq TSEG2_{CALC} \leq 8$ TQ
- IPT = 3 / BRP_{CALC} (the resulting IPT has to be rounded up to the next integer value)

Note: Special Case of Baud Rate Prescaler Value BRP_{CALC} = 1

For the special case of baud rate prescaler value BRPCALC = 1, the IPT is equal to 3 time quanta. This parameter is not compliant to the ISO 11898 standard, where the IPT is defined to be less than or equal to 2 time quanta.
Thus, using this mode (BRPCALC = 1) is not allowed.

- 1 TQ $\leq SJW_{CALC} \leq$ min[4 TQ, $TSEG2_{CALC}$] (SJW = Synchronization Jump Width)
- To utilize three-time sampling mode, $BRP_{CALC} \geq 5$ has to be selected

18.7 CAN Interrupts

There are two different types of interrupts. One type of interrupt is a message-object related interrupt, for example, the receive-message-pending interrupt or the abort-acknowledge interrupt. The other type of interrupt is a system interrupt that handles errors or system-related interrupt sources, for example, the error-passive interrupt or the wake-up interrupt. See Figure 18-17 and Figure 18-18.

The following events may initiate one of the two interrupts:

- Message object interrupts
  - Message reception interrupt: a message was received
  - Message transmission interrupt: a message was transmitted successfully
  - Abort-acknowledge interrupt: a sent transmission was aborted
  - Receive-message-lost interrupt: an old message was overwritten by a new one
  - Message alarm interrupt (HECC only): one of the messages was not transmitted or received within a predefined time frame
  -
- System interrupts
  - Write-denied interrupt: the CPU tried to write to a mailbox but was not allowed to
  - Wake-up interrupt: this interrupt is generated after a wake up
  - Bus-off interrupt: the CAN module enters the bus-off state
  - Error-passive interrupt: the CAN module enters the error-passive mode
  - Warning level interrupt: one or both error counters are greater than or equal to 96
  - Time counter overflow interrupt (HECC only): the local network time stamp counter had an overflow

18.7.1 Interrupts Scheme

The interrupt flags are set if the corresponding interrupt condition occurred. The system interrupt flags are set depending on the setting of SIL (bit 2 of the CANGIM register). If set to 1, the global interrupts will set the bits in the CANGIF1 register, otherwise they will set in the CANGIF0 register.

The GMIF0/GMIF1 (bit 15 of the CANGIFn registers) bit is set depending on the setting of the MIL[*n*] bit of the CANMIL register that corresponds to the message object originating that interrupt. If the MIL[*n*] bit is set, the corresponding message object interrupt flag GMIF[*n*] will set the GMIF1 flag in the CANGIF1 register, otherwise, it will set the GMIF0 flag.

If all interrupt flags are cleared and a new interrupt flag is set, the CAN module interrupt output line (SCC0INT/HECC0INT or SCC1INT/HECC1INT) is activated if the corresponding interrupt mask bit is set. The interrupt line stays active until the interrupt flag is cleared by the CPU by writing a 1 to the appropriate bit.

The GMIF0 (bit 15 of CANGIF0) or GMIF1 (bit 15 of CANGIF1) bit must be cleared by writing a 1 to the appropriate bit in the CANTA register or the CANRMP register (depending on mailbox configuration). These bits cannot be cleared in the CANGIF0/CANGIF1 register.

After clearing one or more interrupt flags, and one or more interrupt flags are still pending, a new interrupt is generated. The interrupt flags are cleared by writing a 1 to the corresponding bit location. If the GMIF0 or GMIF1 bit is set, the mailbox interrupt vector MIV0 (bits 4–0 of CANGIF0) or MIV1 (bits 4–0 of CANGIF1) indicates the mailbox number of the mailbox that caused the setting of the GMIF0/1. It will always display the highest mailbox interrupt vector assigned to that interrupt line.

Figure 18-17. SCC Interrupts Scheme

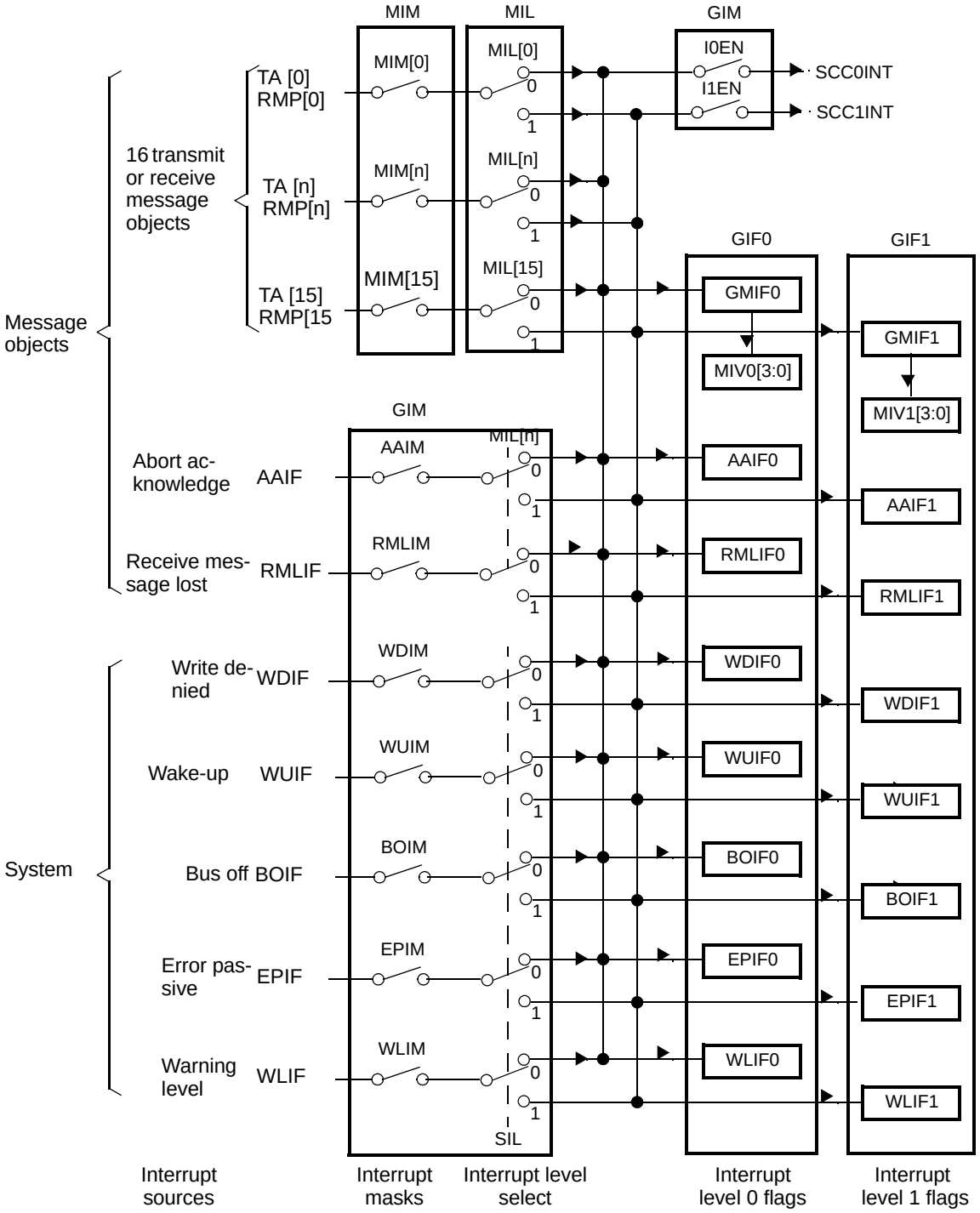
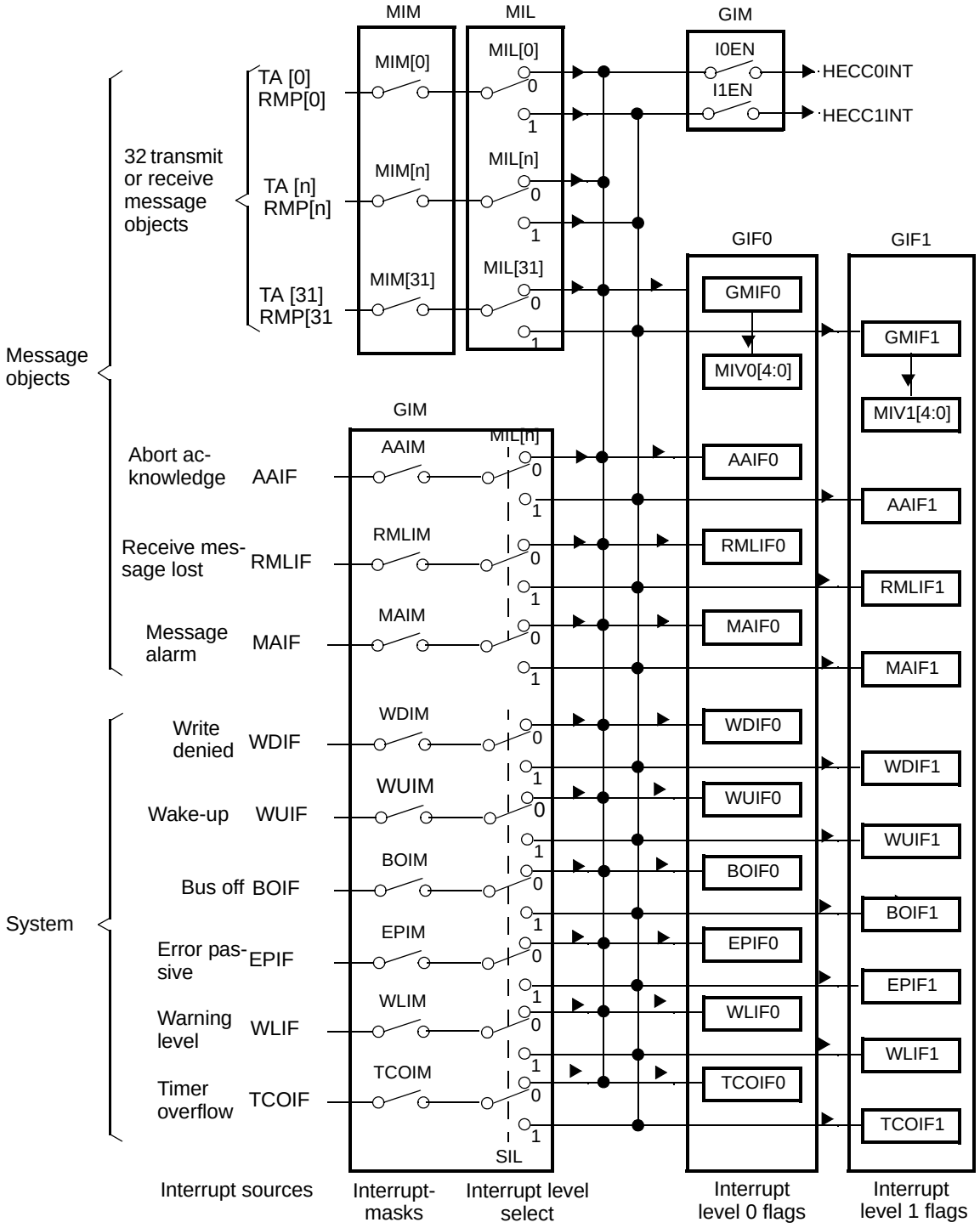


Figure 18-18. HECC Interrupts Scheme



18.7.2 Message Object Interrupt

Each of the 32 message objects in the HECC or the 16 message objects in the SCC may initiate an interrupt on one of the two interrupt output lines 1 or 0. These interrupts can be receive or transmit interrupts depending on the mailbox configuration.

There is one interrupt mask bit (MIM[*n*] in the CANMIM register) and one interrupt level bit (MIL[n]) dedicated to each mailbox. To generate a mailbox interrupt upon a receive/transmit event, the MIM bit has to be set. If a CAN message is received (RMP[*n*]=1 in the CANRMP register) in a receive mailbox or transmitted (TA[*n*] = 1 in the CANTA register) from a transmit mailbox, an interrupt is asserted. If a mailbox is configured as remote request mailbox (MD[*n*] = 1 in the CANMD register, RTR = 1 in the MCF register), an interrupt occurs upon reception of the reply frame. A remote reply mailbox generates an interrupt upon successful transmission of the reply frame (MD[*n*] = 0, AAM = 1).

The setting of the RMP[*n*] bit or the TA[*n*] bit also sets the GMIF0/GMIF1 flag in the CANGIF0/CANGIF1 register if the corresponding interrupt mask bit is set. The GMIF0/GMIF1 flag then generates an interrupt and the corresponding mailbox vector (mailbox number) can be read from the bit field MIV0/MIV1 in the CANGIF0/CANGIF1 register. If more than one mailbox interrupts are pending, the actual value of MIV0/MIV1 reflects the highest priority interrupt vector. The interrupt generated depends on the setting in the MIL register.

The abort acknowledge flag (AA[*n*]) and the abort acknowledge interrupt flag (AAIF) in the CANGIF0/ CANGIF1 register are set when a transmit message is aborted by setting the TRR[*n*] bit in the CANTRR register. An interrupt is asserted upon transmission abortion if the mask bit AAIM in the CANGIM register is set. Clearing the AA[*n*] flag(s) does not reset the AAIF0/AAIF1 flag in the CANGIF0/CANGIF1 registers. The interrupt flag has to be cleared separately. The interrupt line for the abort acknowledge interrupt is selected in accordance with the SIL bit (bit 2 of the CANGIM register).

A lost receive message is notified by setting the receive message lost flag RML[*n*] and the receive message lost interrupt flag RMLIF0/RMLIF1in the CANGIF0/CANGIF1 register. If an interrupt is generated upon the lost receive message event, the receive message lost interrupt mask bit (RMLIM) in the CANGIM register has to be set. Clearing the RML[*n*] flag does not reset the RMLIF0/RMLIF1 flag in the CANGIF0/CANGIF1 registers. The interrupt flag has to be cleared separately. The interrupt line for the receive message lost interrupt is selected in accordance with the SIL bit (bit 2 of the CANGIM register).

Each mailbox of the HECC (in HECC mode only) is linked to a message- object, time-out register (MOTO). If a time-out event occurs (TOS[*n*]=1), a message alarm interrupt is asserted to one of the two interrupt lines if the message alarm interrupt mask bit (MAIM) in the GIM register is set. The interrupt line for message alarm interrupt is selected in accordance with SIL bit (bit 2 of the CANGIM register). Clearing the TOS[n] flag does not reset the MAIF0/MAIF1 flag.

18.8 CAN Power-Down Mode

There are two different power down modes: the global power-down mode, when all clocks are stopped by the CPU, and the local power-down mode, when only the CAN module internal clock is deactivated by the CAN module itself. Therefore, it is possible to have a local power down, where only the clock of the CAN module logic is disabled, or a global power-down mode where the clock for the complete chip is disabled.

18.8.1 Local Power Down

The local power-down mode is requested by writing a 1 to the PDR (bit 11 in the CANMC register) bit. When the module enters the local power-down mode, the status bit PDA (bit 3 in the CANES register) is set.

During local power-down mode, the clock of the CAN base module is turned off. Only the wake-up logic is still active. Any register access is possible as the clock is enabled for every access on its vbusp request. The actual contents of any register can be read back, even during power down.

The module leaves the local power-down mode when the PDR bit is cleared or if any bus activity is detected on the CAN bus line (if the wake-up-on bus activity is enabled).

The automatic wake-up-on bus activity can be enabled or disabled with the configuration bit WUBA of MC register. If there is any activity on the CAN bus line, the module begins its power-up sequence. The module waits until it detects 11 consecutive recessive bits on the CANRX pin and then it goes bus-active.

Note: First Message Received During Power-Down Mode
The first CAN message, which initiates the bus activity, cannot be received. This means that the first message received in power-down and automatic wake-up mode is lost.

After leaving the sleep mode, the PDR and PDA bits are cleared. The CAN error counters remain unchanged.

If the module is transmitting a message when the PDR bit is set, the transmission is continued until a successful transmission, a lost arbitration, or an error condition on the CAN bus line occurs. Then, the PDA bit is activated. Thus, the module causes no error condition on the CAN bus line.

Note:
If CAN goes into standby mode when the Bus-Off condition has occurred, the CCR bit is set automatically upon wake-up. Then, for normal operation to resume, the device must wait for 128x11 recessive bits to occur and for the CCE bit to be set before the CPU can clear the CCR bit and wait for the CCE to go to zero.

To implement the local power-down mode, two separate clocks are used within the CAN module. One clock stays active all the time to ensure power- down operation (i.e., the wake-up logic). The other clock is enabled depending on the setting of the PDA bit or VBUSP access.

18.8.2 Global Power Down

Global power-down mode is requested by the system module (SYS LPM). When the module is able to enter the global power-down mode, all clocks to the CAN module are disabled.

If the module is transmitting a message when SYS LPM is asserted, the wake-up interrupt flag [WUIF0/1 (bit 12 of the CANGIF0/CANGIF1 registers)] is asserted if enabled. The transmission is continued until a successful transmission, a lost arbitration, or an error condition on the CAN bus line occurs; then low-power mode is entered. Thus, the module causes no error condition on the CAN bus line.

During global power-down mode, a dominant signal on the CAN bus may generate a wake-up interrupt, thus enabling the CPU to exit global power-down mode. There is no internal filtering for the CAN bus line. The WUIF flag is set asynchronously to enable the assertion of an interrupt without any running clocks.

18.9 Timer Management Unit

Several functions are implemented in the HECC to control the time when messages are transmitted or should be transmitted. A separate state machine is included in the HECC to handle the time-control functions. This state machine has lower priority when accessing the registers than the CAN state machine has. Therefore, the time-control functions may be delayed by other ongoing actions.

18.9.1 Time Stamp Functions

To get an indication of the time of reception or transmission of a message, a free-running 32-bit timer (LNT) is implemented in the module. Its content is written into the time stamp register of the corresponding mailbox (MOTS) when a received message is stored or a message has been transmitted. (See Section 18.9.3.)

The counter is driven from the bit clock of the CAN bus line. The timer is stopped during the initialization mode or if the module is in sleep or suspend mode. After power-up reset, the free-running counter is cleared.

The most significant bit of the LNT register is cleared by writing a 1 to LNTM (MC[14]). The LNT register may also be cleared when mailbox 16 transmitted or received (depending on the setting of the MD[16] bit) a message successfully. This is enabled by setting the LNTC bit (CANMC[15]). Therefore, it is possible to use mailbox 16 for global time synchronization of the network. The CPU can read and write the counter.

Overflow of the counter can be detected by the LNT-counter-overflow-interrupt flag (bit 16 of the CANGIF0/CANGIF1 registers). An overflow occurs when the highest bit of the LNT counter changes to 1. Thus, the CPU has enough time to handle this situation.

Note: Time Stamp Registers
The time stamp registers are available on the HECC only.

18.9.2 Local Network Time Register (LNT)

Figure 18-19 and Table 18-11 illustrate this register.

Figure 18-19. Local Network Time Register (LNT) [offset = 5Ch] (1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LNT(31–16)															
R/WP-0(2)															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LNT(15–0)															
R/WP-0(2)															

R = Read in all modes; WP = write in privilege mode only; -*n* = Value after reset

1 Relative address = (+) HECC_Offset
2 HECC only, accessable in SCC mode for write and read operations, but with no functional effect.

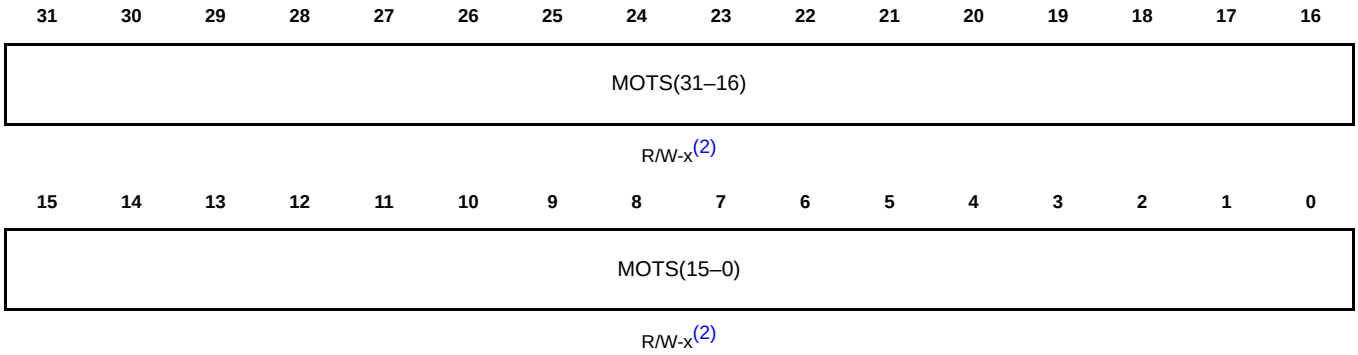
Table 18-11. Local Network Time Register (LNT) Field Descriptions

Bit	Name	Value	Description
31–0	LNT(31–0)	0–FFFF FFFFh	Local network time register. Value of the local network time counter used for the time-stamp and the time-out functions.

18.9.3 Message Object Time Stamp Registers (MOTS)

Figure 18-20 and Table 18-12 illustrate this register.

Figure 18-20. Message Object Time Stamp Registers (MOTS) [offset = 1080h](1)



R = Read; W = Write; x = indeterminate

1 Relative address = + HECC_Mailbox_Offset + 0x1080 + (Object# x 4)
2 HECC only

Table 18-12. Message Object Time Stamp Registers (MOTS) Field Descriptions

Bit	Name	Value	Description
31–0	MOTS(31–0)	0–FFFF FFFFh	Message object time stamp register. *HECC only.* Value of the LNT when the message has been actually received or transmitted.

18.10 Time-Out Functions

To ensure that all messages are sent or received within a predefined period, each mailbox has its own time-out register. If a message has not been sent or received by the time indicated in the time-out register and the corresponding bit TOC[*n*] is set in the CANTOC register, a flag is set in the time-out status register (CANTOS).

For transmit mailboxes the TOS[*n*] flag is cleared when the TOC[*n*] bit is cleared or when the corresponding TRS[*n*] bit is cleared, no matter whether because of successful transmission or because of abortion of the transmit request. For receive mailboxes, the TOS[*n*] flag is cleared when the corresponding TOC[*n*] bit is cleared.

The CPU may also clear the time-out status register flags by writing a 1 into the time-out status register.

The message object time-out registers (MOTO) are implemented as a RAM. The state machine scans all the MOTO registers and compares them to the LNT counter value. If the value in the LNT register is equal to or greater than the value in the time-out register, the corresponding TRS bit (applies to transmit mailboxes only) is set, and the TOC[*n*] bit is set, then the appropriate bit TOS[*n*] is set. Because all the time-out registers are scanned sequentially, there may be a delay before the TOS[*n*] bit is set.

Note: Time-Out Registers
These registers are available on the HECC only. The offset addresses are reserved in the SCC.

18.10.1 Message Object Time-Out Registers (MOTO)

Figure 18-21 and Table 18-13 illustrate this register.

Figure 18-21. Message Object Time-Out Registers (MOTO) [offset = 1100h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOTO(31–16)															
R/W-x															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOTO(15–0)															
R/W-x															

R = Read; W = Write; x = indeterminate

1 Relative address = + HECC_Mailbox_Offset + 0x1100 (Object # x 4)

Table 18-13. Message Object Time-Out Registers (MOTO) Field Descriptions

Bit	Name	Value	Description
31–0	MOTO(31–0)	0–FFFF FFFFh	Message object time-out register. Limit value of the local network time counter (LNT) to actually transmit or receive the message.

18.10.2 Time Out Control-Register (TOC)

Figure 18-22 and Table 18-14 illustrate this register.

Figure 18-22. Time-Out Control Register (TOC) [offset = 60h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOC(2)															
R/W-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC(2)															
R/W-0															

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + HECC_Offset
2 HECC only, accessable in SCC mode for write and read operations, but with no functional effect.

Table 18-14. Time-Out Control Register (TOC) Field Descriptions

Bit	Name	Value	Description
31–0	TOC(31–0)		Time-out control register.
		0	The time-out function is disabled. The TOS[*n*] flag is never set.
		1	The TOC[*n*] bit has to be set by the CPU to enable the time-out function for mailbox *n*. Before setting the TOC[*n*] bit the corresponding MOTO register should be loaded with the time-out value relative to LNT.

18.10.3 Time Out Status Register (TOS)

Figure 18-23 and Table 18-15 illustrate this register.

Figure 18-23. Time-Out Status Register (TOS) [offset = 64h] (1)

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

R = Read; C = Clear by Writing a 1; -*n* = Value after reset

1 Relative address = + HECC_Offset

Table 18-15. Time-Out Status Register (TOS) Field Descriptions

Bit	Name	Value	Description
31–0	TOS[31:0]		Time-out status register.
		0	No time-out occurred or it is disabled for that mailbox.
		1	The value in the LNT register is larger or equal to the value in the time-out register that corresponds to mailbox *n* and the TOC[*n*] bit is set.

18.11 CAN SCC/HECC Control Registers

The 470 SCC and HECC registers listed in Table 18-16 are used by the CPU to configure and control the CAN controller and the message objects. All CAN registers support 8-, 16-, and 32-bit accesses. The start address for the SCC module is 0xFFF7 DC00 and of the HECC is 0xFFF7DC00.

Table 18-16. Combined SCC/HECC Registers

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
00h[2] CANME Page 1063	ME[31:16]															
	ME[15:0]															
04h CANMD Page 1064	MD[31:16]															
	MD[15:0]															
08h CANTRS Page 1065	TRS[31:16]															
	TRS[15:0]															
0Ch CANTRR Page 1066	TRR[31:16]															
	TRR[15:0]															
10h CANTA Page 1067	TA[31:16]															
	TA[15:0]															
14h CANAA Page 1068	AA[31:16]															
	AA[15:0]															
18h CANRMP Page 1069	RMP[31:16]															
	RMP[15:0]															

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the SCC module memory offset and RAM offset. Object# specifies the number of the message object.
2 CANGAM register available in SCC module/mode only. Reserved in HECC mode.
3 LNT, TOC, TOS registers available in HECC module only. Reserved in SCC

Table 18-16. Combined SCC/HECC Registers (Continued)

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
1Ch CANRML Page 1070	RML[31:16]															
	RML[15:0]															
20h CANRFP Page 1071	RFP[31:16]															
	RFP[15:0]															
24h CANGAM[2] Page 1072	AMI	Reserved				GAM[28:16]										
	GAM[15:0]															
28h CANMC Page 1073	Reserved															
	LNTC	LNTM	SCM	CCR	PDR	DBO	WUBA	CDR	ABO	STM	SRES	MBNR				
2Ch CANBTC Page 1077	Reserved								BRP							
	Reserved					ERM	SJW		SAM	TSEG1				TSEG2		
30h CANES Page 1081	Reserved							FE	BE	SA1	CRCE	SE	ACKE	BO	EP	EW
	Reserved										SMA	CCE	PDA	Res'rvd	RM	TM
34h CANTEC Page 1085	Reserved															
	Reserved								TEC							
38h CANREC Page 1085	Reserved															
	Reserved								REC							

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the SCC module memory offset and RAM offset. Object# specifies the number of the message object.
2 CANGAM register available in SCC module/mode only. Reserved in HECC mode.
3 LNT, TOC, TOS registers available in HECC module only. Reserved in SCC

Table 18-16. Combined SCC/HECC Registers (Continued)

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
3Ch CANGIF0 Page 1086	Reserved														MAIF0	TCOF0
	GMIF0	AAIF0	WDIF0	WUIF0	RMLIF 0	BOIF0	EPIF0	WLIF0	Reserved			MIV0				
40h CANGIM Page 1089	Reserved														MAIM	TCOIM
	Reserv ed	AAIM	WDIM	WUIM	RMLIM	BOIM	EPIM	WLIM	Reserved					SIL	I1EN	I0EN
44h CANGIF1 Page 1086	Reserved														MAIF1	TCOF1
	GMIF1	AAIF1	WDIF1	WUIF1	RMLIF 1	BOIF1	EPIF1	WLIF1	Reserved			MIV1				
48h CANMIM Page 1092	MIM[31:16]															
	MIM[15:0]															
4Ch CANMIL Page 1093	MIL[31:16]															
	MIL[15:0]															
50h CANOPC Page 1094	OPC[31:16]															
	OPC[15:0]															
54h CANTIOC Page 1095	Reserved															
	Reserved												TX FUNC	TXDIR	TXOU T	TXIN
58h CANRIOC Page 1095	Reserved															
	Reserved												RX FUNC	RXDIR	RX OUT	RXIN

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the SCC module memory offset and RAM offset. Object# specifies the number of the message object.
2 CANGAM register available in SCC module/mode only. Reserved in HECC mode.
3 LNT, TOC, TOS registers available in HECC module only. Reserved in SCC

Table 18-16. Combined SCC/HECC Registers (Continued)

Offset Address[1] Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
5Ch LNT[3] Page 1053	LNT[31:16]															
	LNT[15:0]															
60h TOC[3] Page 1057	TOC[31:16]															
	TOC[15:0]															
64h TOS[3] Page 1058	TOS[31:16]															
	TOS[15:0]															
68h CANTIOCE Page 1097	Reserved												TX SRS	TXPSL	TXPUL DIS	TX OPDR
	Reserved												TX FUNC	TXDIR	TXOU T	TXIN
6Ch CANRIOCE Page 1097	Reserved												RX SRS	RXPSL	RXPUL DIS	RX OPDR
	Reserved												RX FUNC	RXDIR	RX OUT	RXIN
70h CANETC Page 1100	Reserved							FEN	BEN	SA1E	CRC EN	SEN	Reserved			
	Reserved					CANETCENA			FIELD_SEL				Reserved			

1 The actual addresses of the message objects are device-specific. See the specific device data sheet to verify the SCC module memory offset and RAM offset. Object# specifies the number of the message object.
2 CANGAM register available in SCC module/mode only. Reserved in HECC mode.
3 LNT, TOC, TOS registers available in HECC module only. Reserved in SCC

18.11.1 Mailbox Enable Register (CANME)

Each mailbox can be enabled or disabled. Figure 18-24 and Table 18-17 illustrate this register.

Figure 18-24. Mailbox Enable Register (CANME) [offset = 00h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ME[31:16]															
R/W-0[2]															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ME[15:0]															
R/W-0															

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-17. Mailbox Enable Register (CANME) Field Descriptions

Bit	Name	Value	Description
31–0	ME[31:0]		Mailbox enable register. After power-up, all bits in CANME are cleared. Disabled mailboxes may be used as additional memory for the CPU.
		0	The corresponding mailbox is disabled.
		1	The corresponding mailbox is enabled for the CAN module. The mailbox must be disabled before writing to the contents of any identifier field. If the corresponding bit in CANME is set, the write access to the identifier of a message object is discarded.

18.11.2 Mailbox Direction Register (CANMD)

Each mailbox can be configured as a transmit or a receive mailbox. Figure 18-25 and Table 18-18 illustrate this register.

Figure 18-25. Mailbox Direction Register (CANMD) [offset = 04h](1)

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

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-18. Mailbox Direction Register (CANMD) Field Descriptions

Bit	Name	Value	Description
31–0	MD[31:0]		Mailbox direction. After power-up, all bits in CANMD are cleared.
		0	The corresponding mailbox is defined as a transmit mailbox.
		1	The corresponding mailbox is defined as a receive mailbox.

18.11.3 Transmission Request Set Register (CANTRS)

When mailbox *n* is ready to be transmitted, the CPU sets the TRS[*n*] bit to 1 to start the transmission.

These bits can be set by the CPU and reset by the CAN module logic. It is also set by the CAN module in case of a remote frame request. These bits are reset in case of a successful transmission or an aborted transmission (if requested). If a mailbox is configured as a receive mailbox, the corresponding bit in CANTRS is ignored. If the TRS[n] bit of a remote request mailbox is set, a remote frame is transmitted. If the CPU tries to set a bit while the CAN module tries to clear it, the bit is set.

Setting CANTRS[*n*] causes the particular message *n* to be transmitted. Several bits can be set simultaneously. Therefore, all messages with the TRS bit set are transmitted in turn, starting with the mailbox having the highest mailbox number (= highest priority).

The bits in CANTRS are set by writing a 1 from the CPU. Writing a 0 has no effect. After power-up, all bits are cleared. Figure 18-26 and Table 18-19 illustrate this register.

Figure 18-26. Transmission Request Set Register (CANTRS) [offset = 08h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRS[31:16]															
RS-0(2)															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRS[15:0]															
RS-0															

R = Read; S = Set; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-19. Transmission Request Set Register (CANTRS) Field Descriptions

Bit	Name	Value	Description
31–0	TRS[31:0]		Transmit request set. A successful transmission initiates a GMIF0/GMIF1 (bit 15 in CANGIF0/ CANGIF1) interrupt if enabled.
		0	No operation occurs.
		1	If the corresponding message object is configured as a transmit mailbox or remote request mailbox, the data or remote message of this mailbox will be transmitted.

18.11.4 Transmission Request Reset Register (CANTRR)

Setting the TRR[*n*] bit of the message object *n* causes a transmission request to be cancelled if it was initiated by the corresponding bit (TRS[*n*]) and is not currently being processed. If the corresponding message is currently being processed, the bit is reset in case of a successful transmission (normal operation) or in case of an aborted transmission due to a lost arbitration or an error condition detected on the CAN bus line. In case of an aborted transmission, the corresponding status bit (AA.31–0) is set and in case of a successful transmission, the status bits [TA(31–0)] is set. The status of the transmission request reset can be read from the TRS(31–0) bits.

These bits can only be set by the CPU and reset by the internal logic. These bits are reset in case of a successful transmission or an aborted transmission. If the CPU tries to set a bit while the CAN tries to clear it, the bit is set.

If a transmission reset is requested for a transmit mailbox with corresponding TRS[*n*] bit cleared, is requested for a disabled mailbox, or is requested for a receive mailbox, then the CAN clears the TRR[*n*] and sets the abort acknowledge flag AA[*n*] to respond to the request.

The bits in CANTRR are set by writing a 1 from the CPU. Figure 18-27 and Table 18-20 illustrate this register.

Figure 18-27. Transmission Request Reset Register (CANTRR) [offset = 0Ch](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRR[31:16]															
RS-0(2)															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRR[15:0]															
RS-0															

R = Read; S = Set; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-20. Transmission Request Reset Register (CANTRR) Field Descriptions

Bit	Name	Value	Description
31–0	TRR[31:0]		Transmit request reset. A successful transmission initiates a GMIF0/GMIF1 (bit 15 in CANGIF0/CANGIF1) interrupt if enabled.
		0	No operation occurs.
		1	Setting TRR[*n*] = 1 causes a transmission request to be cancelled if it was initiated by the corresponding TRS[*n*] bit and is not currently being processed. If the corresponding message is currently being processed, the bit is reset in case of a successful transmission (normal operation), or in case of an aborted transmission caused by a lost arbitration or an error condition detected on the CAN bus line. If the transmission request reset bit TRR[*n*] is set and no TRS[*n*] bit is set, then the CAN resolves the situation by clearing the TRR[*n*] bit and setting the AA[*n*] flag.

18.11.5 Transmission Acknowledge Register (CANTA)

If the message of mailbox *n* was sent successfully, the bit TA[*n*] is set. This also sets the GMIF0/GMIF1 (GIF0.15/GIF1.15) bit if the corresponding interrupt mask bit in the CANMIM register is set. The GMIF0/ GMIF1 bit initiates an interrupt.

The CPU resets the bits in CANTA by writing a 1. This will also clear the interrupt if an interrupt had been generated. Writing a 0 has no effect. If the CPU tries to reset the bit while the CAN tries to set it, the bit is set. After power-up, all bits are cleared. Figure 18-28 and Table 18-21 illustrate this register.

Figure 18-28. Transmission Acknowledge Register (CANTA) [offset = 10h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TA[31:16]															
R/C-0[2]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TA[15:0]															
R/C-0															

R = Read; C = Clear by Writing a 1; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-21. Transmission Acknowledge Register (CANTA) Field Descriptions

Bit	Name	Value	Description
31–0	**TA[31:0]**		Transmit acknowledge.
		0	The message is not sent.
		1	The message of mailbox *n* was sent successfully.

18.11.6 Abort Acknowledge Register (CANAA)

If the transmission of the message in mailbox *n* was aborted, the bit AA[*n*] is set and the AAIF (bit 14 of CANGIF) bit is set, which may generate an interrupt if enabled.

Note: Additional Conditions that Set the AA[*n*] Flag
The AA[*n*] flag is set if a transmission reset is requested and the TRS[*n*] bit of the corresponding transmit mailbox is not set, a receive mailbox or a disabled mailbox is concerned.

The bits in CANAA are reset by writing a 1 from the CPU. Writing a 0 has no effect. If the CPU tries to reset a bit and the CAN tries to set the bit at the same time, the bit is set. After power-up, all bits are cleared. Figure 18-29 and Table 18-22 illustrate this register. (1)

Figure 18-29. Abort Acknowledge Register (CANAA) Field Descriptions [offset = 14h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AA[31:16]															
R/C-0(2)															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AA[15:0]															
R/C-0															

R = Read; C = Clear by Writing a 1; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-22. Abort Acknowledge Register (CANAA) Field Descriptions

Bit	Name	Value	Description
31–0	TA[31:0]		Transmission abort acknowledge.
		0	The transmission is not aborted.
		1	The transmission of the message in mailbox *n* was aborted.

18.11.7 Receive Message Pending Register (CANRMP)

If mailbox *n* contains a received message, RMP[*n*] of this register is set. These bits can only be reset by the CPU and set by the internal logic. A new incoming message will overwrite the stored one if the OPC[*n*] bit is cleared, otherwise the next mailboxes are checked for a matching ID. In this case, the corresponding status bit RML[*n*] is set. The bits in the CANRMP and the CANRML registers are cleared by a write access to the base address of the register CANRMP, with a 1 at the corresponding bit location. If the CPU tries to reset a bit and the CAN tries to set the bit at the same time, the bit is set.

The bits in the CANRMP register may set GMIF0/GMIF1 (bit 15 of CANGIF0/CANGIF1) if the corresponding interrupt mask bit in the CANMIM register is set. The GMIF0/GMIF1 bit initiates an interrupt. Figure 18-30 and Table 18-23 illustrate this register.

Figure 18-30. Receive Message Pending Register (CANRMP) [offset = 18h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMP[31:16]															
R/C-0[2]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RMP[15:0]															
R/C-0															

R = Read; C = Clear by Writing a 1; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-23. Receive Message Pending Register (CANRMP) Field Descriptions

Bit	Name	Value	Description
31–0	**RMP[31:0]**		Receive message pending.
		0	The mailbox does not contain a message.
		1	If mailbox *n* contains a received message, bit RMP[*n*] of this register is set.

18.11.8 Receive Message Lost Register (CANRML)

If an old message has been overwritten by a new one in message object *n*, bit RML[*n*] of this register is set. These bits can only be reset by the CPU, and set by the internal logic. The bits can be cleared by a write access to the base address of the register CANRMP with a 1 at the corresponding bit location. If the CPU tries to reset a bit and the CAN tries to set the bit at the same time, the bit is set. The CANRML register is not changed if the OPC[*n*] (OPC[31:0]) bit is set.

If one or more of the bits in the CANRML register are set, the RMLIF (GIF0[11]/ GIF1[11]) bit is also set. This may initiate an interrupt if the RMLIM (GIM[11]) bit is set. Figure 18-31 and Table 18-24 illustrate this register.

Figure 18-31. Receive Message Lost Register (CANRML) [offset = 1Ch][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RML[31:16]															
R/C-0[2]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RML[15:0]															
R/C-0															

R = Read; Clear by writing '1' to the corresponding bit in RMP register; *-n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-24. Receive Message Lost Register (CANRML) Field Descriptions

Bit	Name	Value	Description
31–0	RML[31:0]		Receive message lost.
		0	No message was lost.
		1	An old unread message has been overwritten by a new one in that mailbox.

18.11.9 Remote Frame Pending Register (CANRFP)

Whenever a remote frame request is received by the CAN module, the corresponding bit RFP[n] in the remote frame pending register is set. If a remote frame is stored in a receive mailbox (AAM = 0, MD = 1), the CPU has to initiate the reply frame transmission and has to reset the RFP[n] flag. If a mailbox configured in auto-answer mode (AAM = 1, MD = 0) receives a remote frame, the CAN module clears the RFP[n] flag after the reply frame has been successfully transmitted.

To prevent an auto-answer mailbox from replying to a remote frame request, the CPU has to clear the RFP[n] flag and the TRS[*n*] bit by setting the corresponding transmission request reset bit TRR[*n*]. The AAM bit may also be cleared by the CPU to stop the module from sending the message.

If the CPU tries to reset a bit and the CAN module tries to set the bit at the same time, the bit is not set. The CPU cannot interrupt an ongoing transfer. Figure 18-32 and Table 18-25 illustrate this register.

Figure 18-32. Remote Frame Pending Register (CANRFP) [offset = 20h][(1)]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFP[31:16]															
R/C-0[(2)]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFP[15:0]															
R/C-0															

R = Read; C = Clear by Writing a 1; -*n* Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-25. Remote Frame Pending Register (CANRFP) Field Descriptions

Bit	Name	Value	Description
31–0	RFP[31:0]		Remote frame pending.
		0	No remote frame request was received.
		1	A remote frame request was received by the module.

18.11.10Global Acceptance Mask Register (CANGAM)

The global acceptance mask is used by the SCC or by the HECC in SCC-compatible mode. The global acceptance mask is used for the mailboxes 6–15 if the AME bit (bit 30 of the MID mailbox register) of the corresponding mailbox is set. A received message will only be stored in the first mailbox with a matching identifier.

The global acceptance mask is used for the mailboxes 6–15 of the SCC. Figure 18-33 and Table 18-26 illustrate this register.

Figure 18-33. Global Acceptance Mask Register (CANGAM) [offset = 24h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AMI	Reserved		GAM[28:16]												
R/W-0	R-0		R/W-0												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GAM[15:0]															
R/W-0															

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + SCC_Offset

Table 18-26. Global Acceptance Mask Register (CANGAM) Field Descriptions

Bit	Name	Value	Description
31	AMI		Acceptance mask identifier extension.
		0	The identifier extension bit stored in the mailbox determines which messages shall be received.
		1	Standard and extended frames can be received. In case of an extended frame, all 29 bits of the identifier are stored in the mailbox and all 29 bits of global acceptance mask register are used for the filter. In case of a standard frame, only the first eleven bits (bit 28–18) of the identifier and the global acceptance mask are used.
30–29	Reserved		Reads return 0 and writes have no effect.
28–0	GAM[28:0]		Global acceptance mask. These bits allow any identifier bits of an incoming message to be masked.
		0	The received identifier bit value must match the corresponding identifier bit of the MID register.
		1	A 0 or a 1 (don't care) will be accepted for the corresponding bit of the received identifier.

18.11.11 Master Control Register (CANMC)

This register is used to control the settings of the CAN module.

Note: Bits Protected in User Mode
Some bits of the CANMC register cannot be changed in user mode.

Figure 18-34 and Table 18-27 illustrate this register.

Figure 18-34. Master Control Register (CANMC) [offset = 28h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LNTC	LNTM	SCM	CCR	PDR	DBO	WUBA	CDR	ABO	STM	SRES	MBNR[4:0]				
R/WP-0(2)	SP-x(2)	R/WP-0(2)	R/WP-1	R/WP-0	R/WP-0	R/WP-0	R/W-0	R/WP-0	R/WP-0	R/SP-0	R/W-0(2)				

Read = Read in all modes; WP = Write in privilege mode only; SP = Set in privilege mode only; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC Offset
2 HECC only, reserved in the SCC

Table 18-27. Master Control Register (CANMC) Field Descriptions

Bit	Name	Value	Description
31–16	Reserved		Reads return 0 and writes have no effect.
15	LNTC		Local network time clear. *HECC only, reserved in the SCC.* This bit is protected in user mode.
		0	The LNT register is not reset.
		1	The LNT register is reset to 0 after a successful transmission or reception of mailbox 16.
14	LNTM		Local network time MSB clear. *HECC only, reserved in the SCC.* This bit is protected in user mode.
		0	The LNT register is not changed.
		1	The MSB (most significant bit) of the LNT register is reset to 0. The LNTM bit is reset after one clock cycle by the internal logic.

Table 18-27. Master Control Register (CANMC) Field Descriptions (Continued)

Bit	Name	Value	Description
13	SCM		SCC-compatibility mode. *HECC only, reserved in the SCC.* This bit is protected in user mode. If this bit is cleared (SCM = 0), the HECC behaves like the SCC, and only mailboxes 15–0 can be used. You must refer to the SCC specification to operate the HECC in this mode. Since this mode is selected by default, all software developed for the SCC runs without any change to the HECC module.
		0	The HECC is in SCC-compatibility mode.
		1	The HECC is in normal mode.
12	CCR		Change configuration request. This bit is protected in user mode.
		0	The CPU requests normal operation. This can only be done after the configuration register CANBTC is set to the allowed values.
		1	The CPU requests write access to the configuration register CANBTC of the SCC. After setting this bit, the CPU must wait until the CCE flag of CANES register is at 1, before proceeding to the CANBTC register (see section 18.11.12). This could also mean that the CAN module is in Bus Off state (see BO flag of CANES register, section 18.11.13). When CCR is set no new transmit requests can be issued.
11	PDR		Power-down-mode request. This bit is protected in user mode.
		0	The CAN module power-down mode is not requested (normal operation).
		1	The CAN module power-down mode is requested.
10	DBO		Data byte order. This bit selects the byte order of the message data field (see section 18.5.3.7).This bit is protected in user mode.
		0	The data is received or transmitted MSB (most significant byte) first.
		1	The data is received or transmitted LSB (least significant byte) first.
9	WUBA		Wake up on bus activity. This bit is protected in user mode.
		0	The module leaves the power-down mode only after writing a 0 to the PDR bit.
		1	The module leaves the power-down mode after detecting any bus activity.

Table 18-27. Master Control Register (CANMC) Field Descriptions (Continued)

Bit	Name	Value	Description
8	CDR		Change data field request. This bit allows fast data message updates.
		0	The CPU requests normal operation.
		1	The CPU requests write access to the data field of the mailbox specified by the MBNR(4–0) field (bits 4–0 of CANMC). The CPU must clear the CDR bit after accessing the mailbox. The module does not transmit that mailbox content while the CDR is set. This is checked by the state machine before and after it reads the data from the mailbox to store it in the transmit buffer. Update data with CDR mechanism. Note: When using the CDR bit to update the data field of a mailbox, the SCC and the HECC behave differently. The SCC always considers new data after an arbitration loss or bus error, whereas the HECC tries to send the former data if the message was loaded into the internal transmit buffer before the data update. If the software wants to force the HECC to transmit updated data as soon as possible, it has to perform a dummy write to the TRS register (e.g., TRS = 0x00000000) after the data update is finished (CDR bit cleared).
7	ABO		Auto bus on. This bit is protected in user mode.
		0	The CCR (MC.12) bit is set when the Bus-Off state is detected. The CCR bit should be cleared before resuming any CAN communication. Bus-Off state is exited after 128 x 11 recessive bits.
		1	After Bus-Off state, the module will go back automatically into Bus-On state after 128 × 11 recessive bits. This bit is protected in user mode.
6	STM		Self-test mode This bit is protected in user mode.
		0	The module is in normal mode.
		1	The module is in self-test mode. In this mode, the CAN module generates its own acknowledge (ACK) signal, thus enabling operation without a bus connected to the module. The message is not sent, but is read back and stored in the appropriate mailbox. **Note: In self test mode, with the acceptance mask enabled, the message ID will not be updated when a message is received.**

Table 18-27. Master Control Register (CANMC) Field Descriptions (Continued)

Bit	Name	Value	Description
5	SRES		Software reset. This bit can only be written to and is always read as 0.
		0	No effect occurs.
		1	A write access to this register causes a software reset of the module, that is, all parameters, except the following bits, are reset to their default values. • CANBTC[23:16] and [10:0] • CANMC[15:9] and [7:6] • CANTIOC[3] • CANRIOC[3] The mailbox contents and the error counters are not modified. Pending and ongoing transmissions are canceled without disturbing the communication.
4–0	MBNR 4:0	0–F (HECC) 0–7 (SCC)	Mailbox number. **Note: The bit MBNR[4] is for HECC only, and is reserved in the SCC.** This is the number of the mailbox, for which the CPU requests a write access to the data field. This field is used in conjunction with the CDR bit.

18.11.12Bit-Timing Configuration Register (CANBTC)

The CANBTC register is used to configure the CAN node with the appropriate network-timing parameters. This register must be programmed before using the CAN module.

Note:
This register is write-protected in user mode and can only be written in initialization mode. (See section 18.6.2.)

Note: Forbidden Configuration Values
To avoid unpredictable behavior of the CAN module, the CANBTC register should never be programmed with values not allowed by the CAN protocol specification and by the bit timing rules listed in section 18.6.2.

Figure 18-35 and Table 18-28 illustrate this register.

Figure 18-35. Bit-Timing Configuration Register (CANBTC) [offset = 2Ch](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								BRP							
R-0								R/WPI-0							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					ERM	SJW		SAM	TSEG1				TSEG2		
R-0					R/WPI-0	R/WPI-0		R/WPI-0	R/WPI-0				R/WPI-0		

R = Read; WPI = Write in privilege mode during initialization mode only; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset

Table 18-28. Bit-Timing Configuration Register (CANBTC) Field Descriptions

Bit	Name	Value	Description
31–24	Reserved		Reads return 0 and writes have no effect.
23–16	BRP[7:0]		**Note: The CPK clock frequency is the CAN module system clock, if the dual clock option is disabled (default). If the dual clock option is enabled, the CPK clock is supplied with a separate clock source. For details please refer to the specific device documentation.** **Note: BRP is programmable from 1–256.** The value programmed into the CANBTC register must be decremented by 1, because of the 8-bit size of the binary BRP value (BRP[7:0]): $BRP_{BTC} = BRP_{CALC} - 1$ BRP_{CALC} Result of bit-timing calculation. $1 \leq BRP_{CALC} \leq 256$ BRP_{BTC} Value to be programmed into the CANBTC register. $0 \leq BRP_{BTC} \leq 255$
15–11	Reserved		Reads return 0 and writes have no effect.
10	ERM		Edge resynchronization mode. This bit selects the synchronization mode with the receive data stream on the CAN bus.
		0	The CAN module will resynchronize on the falling edge only.
		1	The CAN module will resynchronize on both rising and falling edges.

Table 18-28. Bit-Timing Configuration Register (CANBTC) Field Descriptions (Continued)

Bit	Name	Value	Description
9–8	SJW(1–0)	0–3	Synchronization jump width. The parameter SJW indicates by how many units of TQ a bit is allowed to be lengthened or shortened when resynchronizing with the receive data stream on the CAN bus. The synchronization is performed either with the falling edge (ERM = 0) or with both edges (ERM = 1) of the bus signal. The synchronization jump width, SJW, is calculated as follows: *SJW = 1 + SJW.0 + (2 × SJW.1)* SJW is programmable from 1–4 TQ. The maximum value of SJW is determined by the minimum value of $TSEG2_{CALC}$ and 4TQ: $SJW_{CALC(max)} \leq min[4\ TQ, TSEG2_{CALC}]$ The value programmed into the CANBTC register has to be decremented by 1, because of the 2-bit size of the binary SJW value (SJW.1:0): $SJW_{BTC} = SJW_{CALC} - 1$ SJW_{CALC} Result of the calculation. $1 \leq SJW_{CALC} \leq 4$ SJW_{BTC} Value to be programmed into the CANBTC register. $0 \leq SJW_{BTC} \leq 3$
7	SAM		Sample point setting. This parameter sets the number of samples used by the CAN module to determine the actual level of the CAN bus. When the SAM bit is set, the level determined by the CAN bus corresponds to the result from the majority decision of the last three values. The sample points are at the sample point and twice before with a distance of 1/2 TQ.
		0	The CAN module samples only once at the sampling point.
		1	The CAN module will sample three times and make a majority decision. The triple sample mode shall be selected only for bit rate prescale values greater than 4 ($BRP_{CALC} > 4$).

Table 18-28. Bit-Timing Configuration Register (CANBTC) Field Descriptions (Continued)

Bit	Name	Value	Description
6–3	TSEG1(3–0)		Time segment 1. This parameter specifies the length of the TSEG1 segment in TQ units. The value of TSEG1 is calculated as follows: TSEG1 = 1 + TSEG1.0 + (2 *TSEG1.1) + (4 *TSEG1.2) + (8 * TSEG1.3) TSEG1 combines PROP_SEG and PHASE_SEG1 segments: TSEG1 = PROP_SEG + PHASE_SEG1 where PROP_SEG and PHASE_SEG1 are the length of these two segments in TQ units. $TSEG1_{CALC}$ value is programmable in the range of 1 TQ to 16 TQ and has to fulfill the following timing rule: TSEG1 must be greater than or equal to TSEG2 and IPT (for more details about IPT refer to Section 18.6.2). The value programmed into the CANBTC register has to be decremented by 1, because of the 4-bit size of the binary TSEG1 value (TSEG1.3:0): $TSEG1_{BTC} = TSEG1_{CALC} - 1$ $TSEG1_{CALC}$ Result of the calculation. $2 \leq TSEG1_{CALC} \leq 16$ $TSEG1_{BTC}$ Value to be programmed into the CANBTC register. $1 \leq TSEG1_{BTC} \leq 15$
2–0	TSEG2[2:0]		Time Segment 2. TSEG2 defines the length of PHASE_SEG2 segment in TQ units and is calculated as follow: TSEG2 = 1 + TSEG2.0 + (2 ¥ TSEG2.1) + (4 ¥ TSEG2.2) $TSEG2_{CALC}$ is programmable in the range of 1 TQ to 8 TQ and has to fulfill the following timing rule: TSEG2 must be smaller than or equal to TSEG1 and must be greater or equal to IPT (for more details about IPT refer to Section 18.6.2). The value programmed into the CANBTC register has to be decremented by 1, because of the 3-bit size of the binary TSEG2 value (TSEG2.3:0): $TSEG2_{BTC} = TSEG2_{CALC} - 1$ $TSEG2_{CALC}$ Result of the calculation. $TSEG2_{BTC}$ Value to be programmed into the CANBTC

18.11.13Error and Status Register (CANES)

The error and status register contains information about the actual status of the CAN module and displays bus error flags as well as error status flags.

The way of storing the bus error flags (FE, BE, CRCE, SE, ACKE) and the error status flags (BO, EP, EW) in the ES register is subject to a special mechanism. If the first of these errors occur, the corresponding flag is set in the CANES register. All other error flags, occuring later, will be frozen in the background and are not flagged in the CANES register. In order to update the CANES register to the frozen flags, the error flag that is set has to be acknowledged by writing a 1 to it.

This allows the software to distinguish between the first error and all following. Figure 18-36 and Table 18-29 illustrate this register.

Figure 18-36. Error and Status Register (CANES) [offset = 30h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							FE	BE	SA1	CRCE	SE	ACKE	BO	EP	EW
R-0							R/C-0	R/C-0	R/C-1	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										SMA	CCE	PDA	Reserved	RM	TM
R-0										R-0	R-1	R-0	R-0	R-0	R-0

R = Read; C = Clear by Writing a 1; -*n* Value after reset

1 Relative address = + SCC/HECC_Offset

Table 18-29. Error and Status Register (CANES) Field Descriptions

Bit	Name	Value	Description
31–25	Reserved		Reads return 0 and writes have no effect.
24	FE		Form error flag.
		0	No form error has been detected.
		1	A form error occurred on the bus. This means that one or more of the fixed-form bit fields had the wrong level on the bus.
23	BE		Bit error flag.
		0	No bit error has been detected.
		1	The received bit does not match the transmitted bit outside of the arbitration field or during transmission of the arbitration field, a dominant bit was sent but a recessive bit was received.
22	SA1		Stuck at dominant error. The SA1 bit is always at 1 after a hardware reset, a software reset, or a *Bus-Off* condition. This bit is cleared when a recessive bit is detected on the bus or when a 1 is written.
		0	The CAN module detected a recessive bit.
		1	The CAN module never detected a recessive bit.

Table 18-29. Error and Status Register (CANES) Field Descriptions (Continued)

Bit	Name	Value	Description
21	CRCE		CRC error.
		0	The CAN module never received a wrong CRC.
		1	The CAN module received a wrong CRC.
20	SE		Stuff error.
		0	No stuff bit error occurred.
		1	A stuff bit error occurred.
19	ACKE		Acknowledge error.
		0	All messages have been correctly acknowledged.
		1	The CAN module received no acknowledge.
18	BO		Bus-off state. The CAN module is in *Bus-Off* state.
		0	Normal operation
		1	There is an abnormal rate of errors on the CAN bus. This condition occurs when the transmit error counter (CANTEC) has reached the limit of 256. During *Bus Off,* no messages can be received or transmitted. The Bus-Off state is exited automatically after 128 x 11 recessive bits. After leaving Bus-Off, the error counters are cleared. When ABO = 0 (CANMC[7] = 0), then the CCR bit (CANMC[12]) should be cleared before resuming any CAN communication.
17	EP		Error passive state.
		0	The CAN module is in error-active mode.
		1	The CAN module is in error-passive mode.

Table 18-29. Error and Status Register (CANES) Field Descriptions (Continued)

Bit	Name	Value	Description
16	EW		Warning status.
		0	Both error counters (CANREC and CANTEC) are less than 96.
		1	One of the two error counters (CANREC or CANTEC) has reached the warning level of 96.
15–6	Reserved		Reads return 0 and writes have no effect.
5	SMA		Suspend mode acknowledge. This bit is set after a latency of one clock cycle—up to the length of one frame—after the suspend mode was activated. The suspend mode is activated with the debugger tool when the circuit is not in run mode. During the suspend mode, the CAN module is frozen and cannot receive or transmit any frame. However, if the CAN module is transmitting or receiving a frame when the suspend mode is activated, the module will enter suspend mode only at the end of the frame.
		0	The module is not in suspend mode.
		1	The module has entered suspend mode.
4	CCE		Change configuration enable. This bit displays the configuration access right. (See section 18.6.) After setting CCR, CCE will be set after finishing the current bus activity, which means that this bit is set after a latency of one clock cycle up to the length of one frame. If CCR is set when in bus-off state, CCE will not be set until 128 groups of 11 recessive bits occur.
		0	The CPU is denied write access to the configuration registers.
		1	The CPU has write access to the configuration registers. The CAN transmission is turned off.
3	PDA		Power-down mode acknowledge.
		0	Normal operation
		1	The CAN module has entered the power-down mode.
2	Reserved		Reads return 0 and writes have no effect.
1	RM		Receive mode. The CAN protocol kernel (CPK) is in receive mode. This bit reflects what the CPK is actually doing regardless of mailbox configuration.
		0	The CAN protocol kernel is not receiving a message.
		1	The CAN protocol kernel is receiving a message.

Table 18-29. Error and Status Register (CANES) Field Descriptions (Continued)

Bit	Name	Value	Description
0	TM		Transmit mode. The CPK is in transmit mode. This bit reflects what the CPK is actually doing regardless of mailbox configuration.
		0	The CAN protocol kernel is not transmitting a message.
		1	The CAN protocol kernel is transmitting a message.

18.11.14Error Counter Registers (CANTEC/CANREC)

The CAN module contains two error counters: the receive error counter (CANREC) and the transmit error counter (CANTEC). The values of both counters can be read via the CPU interface. These counters are incremented or decremented according to the CAN protocol specification version 2.0. Figure 18-37 illustrates the CANTEC register; Figure 18-38 illustrates the CANTEC register.

Figure 18-37. Transmit Error Counter Register (CANTEC) [offset = 34h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TEC							
R-0								R-0							

R = Read; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset

Figure 18-38. Receive Error Counter Register (CANREC) [offset = 38h] (1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								REC							
R-0								R-0							

R = Read; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset

After reaching or exceeding the error passive limit (128), the receive error counter will not be increased anymore. When a message was received correctly, the counter is set again to a value between 119 and 127 (compare with the CAN specification). After reaching the bus-off state, the transmit error counter is undefined while the receive error counter changes its function.

After reaching the bus-off state, the receive error counter is cleared. It will then be incremented after every 11 consecutive recessive bits on the bus. These 11 bits correspond to the gap between two frames on the bus. If the counter reaches 128, the module automatically changes back to the bus-on status if this feature is enabled (ABO, bit 7 in the CANMC register). All internal flags are reset and the error counters are cleared. After leaving initialization mode, the error counters are cleared.

18.11.15Global Interrupt Flag Registers (CANGIF0/CANGIF1)

These registers allow the CPU to identify the interrupt source. Figure 18-39, Figure 18-40, and Table 18-30 illustrate this register.

Figure 18-39. Global Interrupt Flag 0 Register (CANGIF0) [offset = 3Ch](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														MAIF0	TCOIF0
R-0														R/C-0(2)	R/C-0(2)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GMIF0	AAIF0	WDIF0	WUIF0	RMLIF0	BOIF0	EPIF0	WLIF0	Reserved			MIV0.4	MIV0.3:0			
R-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R-0			R-0(2)	R-0			

R = Read; C = Clear by Writing a 1; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Figure 18-40. Global Interrupt Flag 1 Register (CANGIF1) [offset = 44h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														MAIF1	TCOIF1
R-0														R/C-0(2)	R/C-0(2)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GMIF1	AAIF1	WDIF1	WUIF1	RMLIF1	BOIF1	EPIF1	WLIF1	Reserved			MIV1.4	MIV1.3:0			
R-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R-0			R-0(2)	R-0			

R = Read; C = Clear by Writing a 1; -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-30. Global Interrupt Flag Registers (CANGIF0/CANGIF1) Field Descriptions

Bit	Name	Value	Description
31–18	**Reserved**		Reads return 0 and writes have no effect.
17	MAIF0/MAIF1		Message Alarm Flag. *HECC only, reserved in the SCC.*
		0	No time out for the mailboxes occurred.
		1	One of the mailboxes did not transmit or receive a message within the specified time frame.

Table 18-30. Global Interrupt Flag Registers (CANGIF0/CANGIF1) Field Descriptions (Continued)

Bit	Name	Value	Description
16	TCOIF0/TCOIF1		Local network time counter overflow flag. *HECC only, reserved in the SCC.*
		0	The MSB of the LNT register is 0.
		1	The MSB of the LNT register is 1.
15	GMIF0/GMIF1		Global mailbox interrupt flag.
		0	No message has been transmitted or received.
		1	One of the mailboxes transmitted or received a message successfully.
14	AAIF0/AAIF1		Abort-acknowledge interrupt flag.
		0	No transmission has been aborted.
		1	A transmission request has been aborted.
13	WDIF0/WDIF1		Write-denied interrupt flag.
		0	The CPU's write access to the mailbox was successful.
		1	The CPU's write access to a mailbox was not successful.
12	WUIF0/WUIF1		Wake-up interrupt flag.
		0	The module is still in sleep mode or normal operation
		1	During local power down, this flag indicates that the module has left sleep mode. During global power-down, this flag indicates that there is activity on the CAN bus lines. This flag is also set when a global power- down is requested by the CPU, but the CAN module is not ready to enter power-down mode yet.
11	RMLIF0/RMLIF1		Receive-message-lost interrupt flag.
		0	No message has been lost.
		1	For at least one of the receive mailboxes, an overflow condition has occurred.
10	BOIF0/BOIF1		Bus-off interrupt flag.
		0	The CAN module is still in bus-on mode.
		1	The CAN module has entered bus-off mode.
9	EPIF0/EPIF1		Error-passive interrupt flag.
		0	The CAN module is not in error-passive mode.
		1	The CAN module has entered error-passive mode.

Table 18-30. Global Interrupt Flag Registers (CANGIF0/CANGIF1) Field Descriptions (Continued)

Bit	Name	Value	Description
8	WLIF0/WLIF1		Warning level interrupt flag.
		0	None of the error counters has reached the warning level.
		1	At least one of the error counters has reached the warning level.
7–5	Reserved		Reads return 0 and writes have no effect.
4–0	MIV0(4–0)/ MIV1(4–0)	0–F **(HECC)** 0–7 (SCC)	Message object interrupt vector. This vector indicates the number of the message object that activated the global mailbox interrupt flag [GMIF0/ GMIF1 (bit 15 in CANGIF0/CANGIF1)]. It keeps that vector until the appropriate TA[*n*] bit or RMP[*n*] bit is cleared or when a higher priority mailbox interrupt occurred. Then, the highest interrupt vector is displayed, with mailbox 31 having the highest priority in the HECC. In the SCC or in SCC-compatible mode, mailbox 15 has the highest priority. When the GMIF0/GMIF1 flag is 0, the corresponding MIV0/MIV1 vector is undefined. Note: MIV0[4] and MIV1[4] are not available in the SCC. Note: Bit 4 (MIVO.4 and MIV1.4), as previously described, applies to the HECC only. These bits are reserved and read as 0 in the SCC.

18.11.16Global Interrupt Mask Register (CANGIM)

This register allows the various interrupts to be enabled. If a bit is set, the corresponding interrupt is enabled. This register is write-protected in user mode. Figure 18-41 and Table 18-31 illustrate this register.

Figure 18-41. Global Interrupt Mask Register (CANGIM) [offset = 40h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														MAIM	TCOIM
R-0														R/WP-0[2]	R/WP-0[2]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsr'vd	AAIM	WDIM	WUIM	RMLIM	BOIM	EPIM	WLIM	Reserved					SIL	I1EN	I0EN
R-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R/WP-0	R-0					R/WP-0	R/WP-0	R/WP-0

R = Read only, RWP =Read in all modes, Write in privilege mode only, -*n* = Value after reset, x = Indeterminate

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-31. Global Interrupt Mask Register (CANGIM) Field Descriptions

Bit	Name	Value	Description
31–18	**Reserved**		Reads return 0 and writes have no effect.
17	MAIM		Message alarm interrupt mask. *HECC only, reserved in the SCC.*
		0	The MAIF interrupt is disabled.
		1	The MAIF interrupt is enabled.
16	TCOIM		Local network time counter overflow interrupt mask. *HECC only, reserved in the SCC.*
		0	The TCOIF interrupt is disabled.
		1	The TCOIF interrupt is enabled.
15	Reserved		Reads return 0 and writes have no effect.

Table 18-31. Global Interrupt Mask Register (CANGIM) Field Descriptions (Continued)

Bit	Name	Value	Description
14	AAIM		Abort acknowledge interrupt mask.
		0	The AAIF interrupt is disabled.
		1	The AAIF interrupt is enabled.
13	WDIM		Write denied interrupt mask.
		0	The WDIF interrupt is disabled.
		1	The WDIF interrupt is enabled.
12	WUIM		Wake-up interrupt mask.
		0	The WUIF interrupt is disabled.
		1	The WUIF interrupt is enabled.
11	RMLIM		Receive message lost interrupt mask.
		0	The RMLIF interrupt is disabled.
		1	The RMLIF interrupt is enabled.
10	BOIM		Bus-off interrupt mask.
		0	The BOIF interrupt is disabled.
		1	The BOIF interrupt is enabled.
9	EPIM		Error passive interrupt mask.
		0	The EPIF interrupt is disabled.
		1	The EPIF interrupt is enabled.
8	WLIM		Warning level interrupt mask.
		0	The WLIF interrupt is disabled.
		1	The WLIF interrupt is enabled.
7–3	Reserved		Reads return 0 and writes have no effect.

Table 18-31. Global Interrupt Mask Register (CANGIM) Field Descriptions (Continued)

Bit	Name	Value	Description
2	SIL		System interrupt level. This bit defines the interrupt level for TCOIF, WDIF, WUIF, BOIF, EPIF, AAIF, RMLIF, and WLIF.
		0	All global interrupts are mapped to the HECC0INT/SCC0INT interrupt line.
		1	All system interrupts are mapped to the HECC1INT/SCC1INT interrupt line.
1	I1EN		Interrupt line 1 enable.
		0	The HECC1INT/SCC1INT interrupt line is globally disabled.
		1	This bit globally enables all interrupts for the HECC1INT/SCC1INT interrupt line if the corresponding masks are set.
0	I0EN		Interrupt line 0 enable.
		0	The HECC0INT/SCC0INT interrupt line is globally disabled.
		1	All interrupts for the HECC0INT/SCC0INT interrupt line are globally enabled if the corresponding masks are set.

18.11.17Mailbox Interrupt Mask Register (CANMIM)

There is one interrupt flag available for each mailbox. This may be a receive or a transmit interrupt depending on the configuration register. After power up, all interrupt mask bits are cleared and all interrupts are disabled.

This register is write-protected in user mode. Figure 18-42 and Table 18-32 illustrate this register.

Figure 18-42. Mailbox Interrupt Mask Register (CANMIM) [offset = 48h](1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MIM[31:16]															
R/WP-0(2)															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MIM[15:0]															
R/WP-0															

Read = Read in all modes; WP = Write in privilege mode only; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-32. Mailbox Interrupt Mask Register (CANMIM) Field Descriptions

Bit	Name	Value	Description
31–0	**MIM[31:0]**		Mailbox interrupt mask. These bits allow any mailbox interrupt to be masked individually.
		0	The mailbox interrupt is disabled.
		1	The mailbox interrupt is enabled. An interrupt is generated if a message has been transmitted successfully (in case of a transmit mailbox) or if a message has been received without any error (in case of a receive mailbox).

18.11.18Mailbox Interrupt Level Register (CANMIL)

Each of the message objects may initiate an interrupt on one of the two interrupt lines. Depending on the setting in the mailbox interrupt level register (CANMIL), the interrupt is generated on HECC0INT/SCC0INT (MIL[*n*] = 0) or on line HECC1INT/SCC1INT (MIL[*n*] = 1). Figure 18-43 and Table 18-33 illustrate this register.

Figure 18-43. Mailbox Interrupt Level Register (CANMIL) [offset = 4Ch][(1)]

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

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-33. Mailbox Interrupt Level Register (CANMIL) Field Descriptions

Bit	Name	Value	Description
31–0	MIL[31:0]		Mailbox interrupt level. These bits allow any mailbox interrupt level to be selected individually.
		0	The mailbox interrupt is generated on interrupt line 0.
		1	The mailbox interrupt is generated on interrupt line 1.

18.11.19 Overwrite Protection Control Register (CANOPC)

If there is an overflow condition for mailbox *n* (RMP[*n*] is set to 1 and a new receive message would fit for mailbox *n*), the new message is stored depending on the settings in the CANOPC register. If the corresponding bit OPC[*n*] is set to 1, the old message is protected against being overwritten by the new message; thus, the next mailboxes are checked for a matching ID. If no other mailbox is found, the message is lost without further notification. If the bit OPC[*n*] is cleared to 0, the old message is overwritten by the new one. This will be notified by setting the receive message lost bit RML[*n*]. Figure 18-44 and Table 18-34 illustrate this register.

Figure 18-44. Overwrite Protection Control Register (CANOPC) [offset = 50h](1)

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

R = Read; W = Write; -*n* = Value after reset

1 Relative address = + SCC/HECC_Offset
2 HECC only, reserved in the SCC

Table 18-34. Overwrite Protection Control Register (CANOPC) Field Descriptions

Bit	Name	Value	Description
31–0	OPC[31:0]		Overwrite protection control.
		0	The old message may be overwritten by a new one.
		1	An old message stored in that mailbox is protected against being overwritten by a new message.

18.11.20I/O Control Registers (CANTIOC, CANRIOC)

The CANTX and CANRX pins may be configured as normal I/O pins if the HECC/SCC is not used. This is done using the CANTIOC and CANRIOC registers. Figure 18-45 and Table 18-35 illustrate the CANTIOC register, and Figure 18-46 and Table 18-36 illustrate the CANRIOC register.

Figure 18-45. I/O Control Register (CANTIOC) [offset = 54h] (1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TX FUNC	TXDIR	TXOUT	TXIN
R-0												R/WP-0	R/W-0	R/W-0	R-x

R = Read only, R = Read; W = Write; WP = Write in privilege mode only; *-n* = Value after reset, x = indeterminate

1 Relative address = + SCC/HECC_Offset

Table 18-35. I/O Control Register (CANTIOC) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return 0 and writes have no effect.
3	TXFUNC		External pin I/O enable flag. This bit is protected in user mode. Note: When the CANTX pin is configured for CAN functions, the flags TXDIR and TXOUT have no meaning and can be set either to 0 or 1. The TXIN flag always shows the actual value of the CANTX pin.
		0	The CANTX pin is used as a normal I/O pin; the CAN protocol kernel (CPK) output is unconnected.
		1	The CANTX pin is used for the CAN transmit functions.
2	TXDIR		Transmit pin direction flag.
		0	The CANTX pin is used as an input pin if CANTX is configured as an I/O pin (TXFUNC = 0).
		1	The CANTX pin is used as an output pin if CANTX is configured as an I/O pin (TXFUNC = 0).
1	TXOUT	0–1	Output value for CANTX pin. This value is output on the CANTX pin if CANTX is configured as an I/O pin (TXFUNC = 0) and as an output (TXDIR = 1).
0	TXIN		Transmit in. This bit reflects the value on the CANTX pin.
		0	Logic low (0)
		1	Logic high (1)

Figure 18-46. RX I/O Control Register (CANRIOC) [offset = 58h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												RX FUNC	RX DIR	RX OUT	RXIN
R-0												R/WP-0	R/W-0	R/W-0	R-x

R = Read; W = Write; WP = Write in privilege mode only; -*n* = Value after reset, x = indeterminate

1 Relative address = + SCC/HECC_Offset

Table 18-36. RX I/O Control Register (CANRIOC) Field Descriptions

Bit	Name	Value	Description
31–4	**Reserved**		Reads return 0 and writes have no effect.
3	RXFUNC		External pin I/O enable flag. This bit is protected in user mode. Note: When the CANRX pin is configured for CAN functions, the flags RXDIR and RXOUT have no meaning and can be set either to 0 or 1. The RXIN flag always shows the actual value of the CANRX pin.
		0	The CANRX pin is used as a normal I/O pin. The RX signal is still internally connected to the CPK.
		1	The CANRX pin is used for the CAN receive functions.
2	RXDIR		Receive pin direction flag.
		0	The CANRX pin is used as an input pin if CANRX is configured as an I/O pin (RXFUNC = 0).
		1	The CANRX pin is used as an output pin if CANRX is configured as an I/O pin (RXFUNC = 0).
1	RXOUT	0–1	Output value for CANRX pin. This value is output on the CANRX pin if CANRX is configured as an I/O pin (RXFUNC = 0) and as an output (RXDIR = 1).
0	RXIN		Input value for the CANRX pin. This bit reflects the value on the CANRX pin.
		0	Logic low (0)
		1	Logic high (1)

18.11.21 Enhanced I/O Control Registers (CANTIOCE, CANRIOCE)

The CANTX and CANRX pins may be configured as normal I/O pins if the HECC/SCC is not used. This can be done using the CANTIOC and CANRIOC registers or the CANTIOCE and CANRIOCE if enhanced I/O pin functionallity is required. Figure 18-47 and Table 18-37 illustrate the CANTIOCE register, and Figure 18-48 and Table 18-38 illustrate the CANRIOCE register.

Figure 18-47. Enhanced I/O Control Register (CANTIOCE) [offset = 68h][1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													TXPSL	TXPUL-DIS	Reserved
R-0												R/W-0	R/W-d	R/W-d	R/W-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												TX FUNC	TXDIR	TX OUT	TXIN
R-0												R/WP-0	R/W-0	R/W-0	R-x

R = Read; W = Write; WP = Write in privilege mode only; *-n* = Value after reset, x = indeterminate, d = device dependent (see device datasheet for details)

1 Relative address = + SCC/HECC_Offset

Table 18-37. Enhanced I/O Control Register (CANTIOCE) Field Descriptions

Bit	Name	Value	Description
31–19	**Reserved**		Reads return 0 and writes have no effect.
18	TXPSL		CANTX pull select.
		0	Input buffer for TX is disabled if PULLDIS = 1.
		1	Input buffer for TX is enabled if PULLDIS = 1.
17	TXPULDIS		CANTX pull disable.
		0	Pull functionality is enabled.
		1	Pull functionality is disabled.
16–4	Reserved		Reads return 0 and writes have no effect.
3	TXFUNC		External pin I/O enable flag (mirrored bit from CANTIOC register). This bit is protected in user mode. Note: When the CANTX pin is configured for CAN functions, the flags TXDIR and TXOUT have no meaning and can be set either to 0 or 1. The TXIN flag always shows the actual value of the CANTX pin.
		0	The CANTX pin is used as a normal I/O pin; the CAN protocol kernel (CPK) output is unconnected.
		1	The CANTX pin is used for the CAN transmit functions.
2	TXDIR		Transmit pin direction flag (mirrored bit from CANTIOC register).

Table 18-37. Enhanced I/O Control Register (CANTIOCE) Field Descriptions (Continued)

Bit	Name	Value	Description
		0	The CANTX pin is used as an input pin if CANTX is configured as an I/O pin (TXFUNC = 0).
		1	The CANTX pin is used as an output pin if CANTX is configured as an I/O pin (TXFUNC = 0).
1	TXOUT	0–1	Output value for CANTX pin (mirrored bit from CANTIOC register). This value is output on the CANTX pin if CANTX is configured as an I/O pin (TXFUNC = 0) and as an output (TXDIR = 1).
0	TXIN		Input value for the CANTX pin (mirrored bit from CANTIOC register) This value reflects the value on the CANTX pin.
		0	Logic low (0)
		1	Logic high (1)

Figure 18-48. Enhanced I/O Control Register (CANRIOCE) [offset = 6Ch][(1)]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												RX SRS	RXPSL	RXPUL-DIS	RX OPDR
R-0												R/W-0	R/W-d	R/W-d	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												RX FUNC	RXDIR	RX OUT	RXIN
R-0												R/WP-0	R/W-0	R/W-0	R-x

R = Read; W = Write; WP = Write in privilege mode only; *-n* = Value after reset, x = indeterminate, d = device dependant (see device datasheet for details)

1 Relative address = + SCC/HECC_Offset

Table 18-38. Enhanced I/O Control Register (CANRIOCE) Field Descriptions

Bit	Name	Value	Description
31–20	**Reserved**		Reads return 0 and writes have no effect.
19	RXSRS		External pin I/O slew rate select.
		0	The normal output buffer is enabled.
		1	The slow output buffer is enabled.
18	RXPSL		CANRX pull select.
		0	Input buffer for RX is disabled if PULLDIS = 1.
		1	Input buffer for RX is enabled if PULLDIS = 1.
17	RXPULDIS		CANRX pull disable

Table 18-38. Enhanced I/O Control Register (CANRIOCE) Field Descriptions

Bit	Name	Value	Description
		0	Pull functionality is enabled.
		1	Pull functionality is disabled.
16	RXOPDR		CANRX open drain
		0	Open drain functionality is disabled.
		1	Open drain functionality is enabled.
15–4	**Reserved**		Reads return 0 and writes have no effect.
3	RXFUNC		External pin I/O enable flag (mirrored bit from CANRIOC register). This bit is protected in user mode. Note: When the CANRX pin is configured for CAN functions, the flags RXDIR and RXOUT have no meaning and can be set either to 0 or 1. The RXIN flag always shows the actual value of the CANRX pin.
		0	The CANRX pin is used as a normal I/O pin; the CAN protocol kernel (CPK) output is unconnected.
		1	The CANRX pin is used for the CAN transmit functions.
2	RXDIR		Transmit pin direction flag (mirrored bit from CANTIOC register).
		0	The CANRX pin is used as an input pin if CANRX is configured as an I/O pin (RXFUNC = 0).
		1	The CANRX pin is used as an output pin if CANTX is configured as an I/O pin (RXFUNC = 0).
1	RXOUT	0–1	Output value for CANRX pin (mirrored bit from CANRIOC register). This value is output on the CANRX pin if CANRX is configured as an I/O pin (RXFUNC = 0) and as an output (RXDIR = 1).
0	RXIN		Input value for CANRX pin (mirrored bit from CANTIOC register).
		0	Logic low (0)
		1	Logic high (1)

18.11.22Error Test Control Register (CANETC)

All the bits of the CANETC register are functional in error test mode only. When error test mode is enabled, the CAN module will be configured in loopback mode automatically, except when BEN is set and the CAN module is in receive mode or when CRCEN is set. In these cases, loopback mode will be disabled automatically and you should set the CAN module in receive mode. Figure 18-49 and Table 18-39 illustrate this register.

Figure 18-49. Error Test Control Register (CANETC) [offset = 70)[1]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							FEN	BEN	SA1E	CRC EN	SEN	Resefved			
R-0							R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				CANETCENA[3:0]				FIELD_SEL[3:0]				Reserved			
R-0				R/WP-0101				R/W-0				R-0			

Read = Read in all modes; WP = Write in privilege mode only; -*n* = Value after reset; x = indeterminate

1 Relative address = + SCC/HECC_Offset

Table 18-39. Error Test Control Register (CANETC) Field Descriptions

Bit	Name	Value	Description
31–25	**Reserved**		Reads return 0 and writes have no effect.
24	FEN		Form error enable
		0	No form error will be generated.
		1	A form error will be created. The bit received during the fixed format field is toggled and passed to the error checking circuitry A form error can be verified in the following fields: - CRC delimiter field, - Acknowledge delimiter field - End-of-frame field The corresponding field is selected using the FIELD_SEL bits in the CANETC register.

Table 18-39. Error Test Control Register (CANETC) Field Descriptions (Continued)

Bit	Name	Value	Description
23	BEN		Bit error enable
		0	No bit error will be generated.
		1	A bit error will be created. The received bit is XORed with 1 and passed to the bit monitor circuitry. A bit error (during transmit mode) can be verified in the following fields: CRC field, Data field Data length control field During receive mode, bit error is verified in ACK slot field. In this mode, the CAN module will not be in loopback mode. It should be set in receive mode by the user. The corresponding field is selected using the FIELD_SEL bits in the CANETC register.
22	SA1E		Stuck at dominant error enable.
		0	No stuck at dominant (SA1) error will be generated.
		1	The received bit is ANDed with 0 to generate a stuck at dominant (SA1) error.
21	CRCEN		CRC error enable.
		0	No CRC error will be generated.
		1	The CRC bits received as well as the fields used for CRC calculation are masked and passed to the CRC check circuitry. A CRC error can be verified in the following fields: STDID field, EXTID field, Data field CRC field For verifying CRC errors, the CAN module will not be in loopback mode and should be set in receive mode by the user. The corresponding field is selected using the FIELD_SEL bits in the CANETC register.

Table 18-39. Error Test Control Register (CANETC) Field Descriptions (Continued)

Bit	Name	Value	Description
20	SEN		Stuff error enable.
		0	No stuff error will be generated.
		1	The stuff bits received are toggled and passed to the error monitor circuitry. A stuff error can be verified in the following fields: STDID field EXTID field Data field The corresponding field is selected using the FIELD_SEL bits in the CANETC register.
19–12	**Reserved**		Reads return 0 and writes have no effect.
11-8	CANETCENA		Test register enable key (0101).
		1010	CANETC is enabled.
		All other values	CANETC is disabled.
7-4	FIELD_SEL		CAN field selects.
		0000	No field is selected.
		0001	The STDID field is selected.
		0010	The DATALENGTHCODE field is selected.
		0011	The DATA field is selected.
		0100	The CRC field is selected.
		0101	The CRC delimiter field is selected.
		0110	The ACKSLOT field is selected.
		0111	The ACKDEL field is selected.
		1000	The ENDOFFRAME field is selected.
		1001	The EXTID field is selected.
3–0	Reserved		Reads return 0 and writes have no effect.

18.12 CAN Operation Examples

This section provides examples on how to use the HECC and the SCC in specific applications.

18.12.1 Configuration of SCC or HECC

The following steps must be performed to configure the SCC/HECC for operation:

Note: Protected Registers Access.
This sequence must be done in TMS470Px privilege mode.

1. Set the CANTX and the CANRX pins to CAN functions:
   Write CANTIOC[3:0] = 0x08
   Write CANRIOC[3:0] = 0x08
2. After a reset, bit CCR (bit 12 of CANMC) and bit CCE (bit 4 of CANES) are set to 1. This allows the user to configure the bit-timing configuration register (CANBTC).
   If the CCE bit is set (CANES[4] = 1), proceed to next step; otherwise, set the CCR bit (CANMC[12] = 1) and wait until CCE bit is set (CANES[4] = 1).
3. Program the CANBTC register with the appropriate timing values. Make sure that the values TSEG1 and TSEG2 are not 0. If they are 0, the module does not leave the initialization mode. For example:
   Write BTC = 0x1021A
4. For the SCC, program the acceptance masks now. For example:
   Write LAM(3) = 0x3c0000
5. Program the master control register (CANMC) as follows:
   Clear CCR (CANMC[12]) = 0
   Clear PDR (CANMC[11]) = 0
   Clear DBO (CANMC[10]) = 0
   Clear WUBA (CANMC[9])= 0
   Clear CDR (CANMC[8]) = 0
   Clear ABO (CANMC[7]) = 0
   Clear STM (CANMC[6]) = 0
   Clear SRES (CANMC[5]) = 0
   Clear MBNR (CANMC[4–0]) = 0
6. Verify the CCE bit is cleared (ES[4] = 0), indicating that the CAN module has been configured.

This completes the setup for the basic functionality.

18.12.2 Transmit Mailbox

18.12.2.1 Configuring a Mailbox for Transmit

To transmit a message, the following steps need to be performed (in this example, for object 1):

1. Clear the appropriate bit in the CANTRS register to 0:
   Clear TRS[1] = 0 (Writing a 0 to TRS has no effect; instead, set TRR[1] and wait until TRS[1] clears.)
2. Disable the object by clearing the corresponding bit in the mailbox enable (CANME) register.
   Clear CANME[1] = 0

3. Load the message identifier (MID) register of the object. Clear the AME (bit 30) and AAM (bit 29) bits for a normal send mailbox (MID[30] = 0 and MID[29] = 0). This register is usually not modified during operation. It can only be modified when the mailbox is disabled. For example:

   Write MID(1) = 0x15ac0000

   Write the data length into the DLC field of the message control field register (MCF.3:0). The RTR flag is usually cleared (MCF.4 = 0). The CANMCF register is usually not modified during operation and can only be modified when the object is disabled. For example:

   MCF(1) = 2

   Set the mailbox direction by clearing the corresponding bit in the CANMD register.

   Clear CANMD[1] = 0

4. Set the mailbox enable by setting the corresponding bit in the CANME register

   Set CANME[1] = 1

This configures object 1 for transmit mode.

18.12.2.2 Transmitting a Message

To start a transmission (in this example, for object 1):

1. Write the message data into the mailbox data field.

   Since DBO (CANMC[10]) is set to zero in the configuration section and MCF(1) is set to 2, the data are stored in the 2 most significant bytes of MDL(1).

   Write MDL(1) = xxxx0000h

2. Set the corresponding flag in the transmit request register (CANTRS.1 = 1) to start the transmission of the message. The CAN module now handles the complete transmission of the CAN message.
3. Wait until the transmit-acknowledge flag of the corresponding mailbox is set (TA.1 = 1). After a successful transmission, this flag is set by the CAN module.
4. The TRS flag is reset to 0 by the module after a successful or aborted transmission (TRS.1 = 0).
5. The transmit acknowledge must be cleared for the next transmission.

   Set TA.1 = 1

   Wait until read TA.1 is 0

6. To transmit another message in the same message object, the mailbox RAM data must be updated. Setting the TRS.1 flag starts the next transmission.

18.12.3 Receive Mailbox

18.12.3.1 Configuring Mailboxes for Receive

To configure a mailbox to receive messages, the following steps must be performed (in this example, mailbox 3):

1. Disable the mailbox by clearing the corresponding bit in the mailbox enable (CANME) register.

   Clear ME[3] = 0

2. Write the selected identifier into the corresponding MID register. The identifier extension bit must be configured to fit the expected identifier. If the acceptance mask is used, the acceptance mask enable (AME) bit must be set (MID[30] = 1). For example:

   Write MID[3] = 0x4f780000

3. For HECC only: If the AME bit is set to 1, the corresponding acceptance mask must be programmed.

   For the SCC: (See section 18.12.1.) For example:

   Write LAM[3] = 0x03c0000.

4. Configure the mailbox as a receive mailbox by setting the corresponding flag in the mailbox direction register (MD[3] = 1). Make sure no other bits in this register are affected by this operation.
5. If data in the mailbox is to be protected, the overwrite protection control register (CANOPC) should be programmed now. This protection is useful if no message must be lost. If OPC is set, the software has to make sure that an additional mailbox (buffer mailbox) is configured to store 'overflow' messages. Otherwise messages can be lost without notification.

   Write OPC[3] = 1
6. Enable the mailbox by setting the appropriate flag in the mailbox enable register (CANME). This should be done by reading CANME, and writing back (CANME |= 0x0008) to make sure no other flag has changed accidentally.

The object is now configured for the receive mode. Any incoming message for that object is handled automatically.

18.12.3.2 Receiving a Message

This example uses message object 3. When a message is received, the corresponding flag in the receive message pending register (CANRMP) is set to 1 and an interrupt may be initiated. The CPU may then read the message from the mailbox RAM. Before the CPU reads the message from the mailbox, it should first clear the RMP bit (RMP[3] = 1). The CPU should also check the receive message lost flag RML[3] = 1. Depending on the application, the CPU has to decide how to handle this situation.

After reading the data, the CPU needs to check that the RMP bit has not been set again by the module. If the RMP bit has been set to 1, the data may have been corrupted. The CPU needs to read the data again because a new message was received while the CPU was reading the old one.

18.12.3.3 Handling of Overload Situations

If the CPU is not able to handle important messages fast enough, it may be advisable to configure more than one mailbox for that identifier. Here is an example where the objects 3, 4, and 5 have the same identifier and share the same mask. For the SCC, the mask is LAM(3). For the HECC, each object has its own LAM: LAM(3), LAM(4), and LAM(5), all of which need to be programmed with the same value.

To make sure that no message is lost, objects 4 and 5 set the OPC flag, which will prevent unread messages from being overwritten. If the CAN module needs to store a received message, it will first check mailbox 5. If the mailbox is empty, the message is stored there. If the RMP flag of object 5 is set (mailbox occupied), the CAN module will check the condition of mailbox 4. If that mailbox is also busy, the module will check in mailbox 3 and store the message there since the OPC flag is not set for mailbox 3. It also sets the RML flag of object 3, which may initiate an interrupt.

It also may be advisable to have object 4 generate an interrupt telling the CPU to read mailboxes 4 and 5 at once. This technique is also useful for messages that require more than 8 bytes of data (i.e., more than one message). In this case, all data needed for the message can be collected in the mailboxes and be read at once.

18.12.4 Handling of Remote Frame Mailboxes

There are two functions for remote frame handling. One is a request by the module for data from another node, the other is a request by another node for data that the module needs to answer.

18.12.4.1 Requesting Data From Another Node

To request data from another node, the object is configured as receive mailbox. (See section 18.12.3.1.) Using object 3 for this example, the CPU needs to do the following:

1. Set the RTR bit in the message control field register (CANMCF) to 1.

   Write MCF[3] = 0x12
2. Write the correct identifier into the message identifier register (MID).

   Write MID[3] = 0x4f780000

3. Set the TRS flag for that mailbox. Since the mailbox is configured as receive, it will only send a remote request message to the other node.

   Set TRS[3] = 1

4. Now, the module stores the answer in that mailbox and sets the RMP bit when it is received. This action may initiate an interrupt. Also, make sure no other mailbox has the same ID.

   Wait for RMP[3] = 1

5. Now read the received message as explained in Section 18.12.3.2 on page 1105.

18.12.4.2 Answering a Remote Request

1. The object needs to be configured as a transmit mailbox, as explained in Section 18.12.2.1 on page 1103, steps 1 to 4.
2. The auto answer mode (AAM) (MID[29]) bit must be set in the MID register before the mailbox is enabled.

   MID[1] = 0x35ac0000

3. The data field must be updated.

   MIL[1] = xxxx0000h

4. Now the mailbox can be enabled by setting the ME flag to 1.

   ME[1] = 1

5. When a remote request is received from another node, the TRS flag is set automatically and the data is transmitted to that node. The identifier of the received message and the transmitted message are the same.
6. After transmission of the data, the TA flag is set. The CPU may then update the data.

   Wait for TA[1] = 1

18.12.4.3 Updating the Data Field

To update the data of an object that is configured in auto answer mode, the following steps need to be performed. This sequence can also be used to update the data of an object configured in normal transmission with TRS flag set.

1. Set the change data request (CDR) (bit 8 of CANMC) bit and the mailbox number (MBNR) of that object in the master control register (CANMC). This tells the CAN module that the CPU wants to change the data field. For example, for object 1:

   Write MC = 0x0000101

2. Write the message data into the mailbox data register. For example:

   Write MDL[1] = xxxx0000h

3. Clear the CDR bit (MC.8) to enable the object.

   Write MC = 0x00000000

18.12.5 Interrupt Handling

The CPU is interrupted by asserting one of the two interrupt lines. After handling the interrupt, which should generally also clear the interrupt source, the interrupt flag must be cleared by the CPU. To do this, the interrupt flag must be cleared in the CANGIF0 or CANGIF1 register by writing a 1 to the interrupt flag. This also releases the interrupt line if no other interrupt is pending.

18.12.5.1 Configuring for Interrupt Handling

To configure for interrupt handling, the mailbox interrupt level register (CANMIL), the mailbox interrupt mask register (CANMIM), and the global interrupt mask register (CANGIM) need to be configured. The steps to do this are described below:

1. Write the CANMIL register. This defines whether a successful transmission will assert interrupt line 0 or 1. For example, CANMIL = 0xFFFFFFFF will set all mailbox interrupts to level 1.
2. Configure the mailbox interrupt mask register (CANMIM) to mask out the mailboxes that should not cause an interrupt. This register could be set to 0xFFFFFFFF, which enables all mailbox interrupts. Mailboxes that are not used will not cause any interrupts anyhow.
3. Now configure the CANGIM register. The flags AAIM, WDIM, WUIM, BOIM, EPIM, and WLIM (GIM.14-9) should always be set (enabling these interrupts). In addition, the SIL (bit 2 of CANGIM) bit may be set to have the global interrupts on another level than the mailbox interrupts. Both the I1EN (bit 1 of CANGIM) and I0EN (bit 0 of CANGIM) flags should be set to enable both interrupt lines. The flag RMLIM (bit 11 of CANGIM) may also be set depending on the load of the CPU.

This configuration puts all mailbox interrupts on line 1 and all system interrupts on line 0. Thus, the CPU can handle all system interrupts (which are always serious) with high priority, and the mailbox interrupts (on the other line) with a lower priority. All messages with a high priority can also be directed to the interrupt line 0.

18.12.5.2 Handling Mailbox Interrupts

There are three interrupt flags for mailbox interrupts. These are listed below:

GMIF0/GMIF1: One of the objects has received or transmitted a message. The number of the mailbox is in MIV0/MIV1(bits 4–0 of CANGIF0/CANGIF1). The normal handling routine is as follows:

1. Do a half-word read on the GIF register that caused the interrupt. If the value is negative, a mailbox caused the interrupt. Otherwise, check the AAIF0/AAIF1 (bit 14 of CANGIF0/CANGIF1) bit (abort-acknowledge interrupt flag) or the RMLIF0/RMLIF1 (bit 11 of CANGIF0/GIF1) bit (receive-message-lost interrupt flag). Otherwise, a system interrupt has occurred. In this case, each of the system-interrupt flags must be checked.
2. If the RMLIF (CANGIF0[11]) flag caused the interrupt, the message in one of the mailboxes has been overwritten by a new one. This should not happen in normal operation. The CPU needs to clear that flag by writing a 1 to it. The CPU must check the receive-message-lost register (RML) to find out which mailbox caused that interrupt. Depending on the application, the CPU has to decide what to do next. This interrupt comes together with an GMIF0/GMIF1 interrupt.
3. If the AAIF (bit 14 of CANGIF) flag caused the interrupt, a send transmission operation was aborted by the CPU. The CPU should check the abort acknowledge register (AA[31:0]) to find out which mailbox caused the interrupt and send that message again if requested. The flag must be cleared by writing a 1 to it.
4. If the GMIF0/GMIF1 (bit 15 of CANGIF0/CANGIF1) flag caused the interrupt, the mailbox number that caused the interrupt can be read from the MIV0/MIV1 (bits 4–0 of CANGIF0/CANGIF1) field. This vector may be used to jump to a location where that mailbox is handled. If it is a receive mailbox, the CPU should read the data as described above and clear the RMP[31:0] flag by writing a 1 to it. If it is a send mailbox, no further action is required, unless the CPU needs to send more data. In this case, the normal send procedure as described above is necessary. The CPU needs to clear the transmit acknowledge bit (TA[31:0]) by writing a 1 to it.

Memory Protection Unit (MPU)

This document describes the behavior and programming of the TMS470Px memory protection unit (MPU). The MPU works with the ARM7TDMI CPU.

Topic **Page**

19.1 General Description

The basic functionality of the memory protection unit (MPU) is to protect various regions of the memory space through the ARM system control coprocessor or the CP15 registers. A total of up to 4 data/unified protection regions can be defined by writing to the CP15's Protection Unit registers. Some of the main features of MPU are as follows:

- Provide extensive protection to memory regions ranging from 64 bytes to 4 GB in size.
- Compatible with ARM PMSAv6.
- A total of up to eight data/unified regions, each defined by the base address and memory size, programmable in CP15 registers
- Programmable access permission (AP) for each of the protection regions, used to determine if memory accesses to those regions should be allowed in user or privileged mode.
- Each region programmable to allow various combinations of read/write permissions to memory locations in both user and privileged mode.
- Memory that can be partitioned into executable or nonexecutable regions. Only instruction accesses to executable locations in the memory will be allowed.
- Ability to configure protection regions with overlapping locations to allow more flexibility in programming protection attributes for various memory regions.
- Protection bypassed in suspend mode to allow for debugging.

19.2 ARM Coprocessor Interface

The functionality of MPU is controlled by programming some of the registers in the ARM's coprocessor CP15, also known as the system control coprocessor. Instructions MRC (move from coprocessor register to CPU's register) and MCR (move from CPU's register to the coprocessor register) can write to the system control CP15 registers.

Note:
The MCR/MRC instructions can only be executed in privileged mode.

19.3 Operation

The MPU operates as described in the following sections.

19.3.1 Functionality

The MPU generates the necessary protection against illegal accesses.

- Valid region match: Based on the values programmed in the CP15 registers, the incoming address is matched against the address range of all the protection regions to determine if the access is to a valid location. The same address may be valid in more than one protection region.
- Access permission arbitration: If the address is valid in more than one region, then arbitration (using a fixed priority scheme) is performed to determine the access permissions for this access. For more details refer to Section 19.3.3.
- Abort logic: If access permissions are violated or if the access is in an invalid region, the MPU generates an illegal access to cancel the transaction. In addition, the MPU generates an abort signal for the access in the data phase.

19.3.2 Enabling the MPU

By default the protection unit is disabled after reset. The MPU can be enabled or disabled by writing to bit 0 of the CPGCTRL register (Section 19.4.3). On reset, this bit is cleared and as a result the MPU is disabled by default. While the protection unit is disabled, all memory accesses are treated as non-aborting.

Before the MPU is enabled, care needs to be taken to ensure that at least one valid protection region is specified and its access permission fields are defined. In addition to that, the code that enables the MPU must be placed in this valid protection region.

19.3.3 Programing Protection Regions

The MPU can have up to eight protection regions. The number of regions defined in the device can be read from the CPMPUTYPE register, bits[10:8]. Each protection region can be enabled individually using the MPUEN bit in each of the CPMPUREGSIZE registers (Section 19.4.8). The sizes and the starting addresses of the protection regions (instruction regions) are defined using the CPMPUREGSIZE and CPMPUBASEADDR registers (Section 19.4.8 and Section 19.4.7) respectively. The access permissions for all the regions mentioned above is programmed in the CPMPUDAP registers (Section 19.4.9).

Before programming the MPU regions for each task, first disable the MPU by setting the MPUEN bit in the CPGCTRL register to 0 (Section 19.4.9). The base address and the region size of the different protection regions can then be programmed. The MPU is reenabled by setting the MPUEN bit back to 1.

Note:
When a write to the CPGCTRL register reaches the execute stage of the pipeline, the MPU will be enabled.

The size of the region is a power of 2 and can range from 64 bytes to 4 GB. The base address must be a multiple of the region size. For example, 0x11110000 can be the base address of the region if the region size is defined as 64 KB (0x10000). If the region size is defined as 128 KB (0x20000), the base address cannot be 0x11110000.

Access permissions (AP) are defined for each of the protection regions using the AP bits and an XN bit in the CPMPUDAP registers (Section 19.4.9). Table 19-1 shows the allowable data/instruction accesses for all combinations of the AP bits that can programmed using the CPMPUDAP register:

Table 19-1. Region Access Permissions

Value	Privileged	User	Description
00	No Access	No Access	All accesses generate region permission abort
01	Read/Write	No Access	Privileged accesses only
10	Read/Write	Read Only	User mode writes generate region permission abort
11	Read/Write	Read/Write	User and Privileged accesses allowed

Separate access permissions are supported for instruction accesses. This allows an area of memory to be marked executable or nonexecutable by writing to the XN bit of the CPMPUDAP register. For instruction accesses to be executed from a memory region, the region must have data access (determined by the AP bits) and the XN bit must be 0x0. For example, programming the AP bits to 0b10 and the XN bit to 0b0 will allow user mode reads and privileged mode reads and writes.

Table 19-2. Instruction Access Permissions

Value	User
0	All Instruction fetches allowed
1	No Instruction fetches allowed

Protection regions can be programmed such that they overlap with each other. When there is an access to a location with overlapping attributes, a fixed priority scheme is applied to determine what protection attributes are assigned to the memory access; see Figure 19-1. If the address lies within more than one protection region, the attributes of the highest-numbered region apply for that location. See the example below and Figure 19-1:

- Region 0 is programmed with base address of 0x00000000, with a size of 8 KB, and with access permission (AP) set as 0b01 (privileged mode access only). Execute the following instructions to program region 0:

  MOV R2, #0x00000000

  MCR p15, #0, R2, c6, c2, #0; Prog Region Control Register as 0

  MCR p15, #0, R2, c6, c1, #0; Prog Base Address as 0x0000

  MOV R2, #0x00000019

  MCR p15, #0, R2, c6, c1, #2; Program Region size as 8KB

  MOV R2, #0x00001100

  MCR p15, #0, R2, c6, c1, #4; Program Access Permission as 0b01

- Region 1 is programmed with base address of 0x00001000, with a size of 1 KB and with AP set as 0b11 (full access in both privileged and user mode). Execute the following instructions to program the region 1:

  MOV R2, #0x00000001

  MCR p15, #0, R2, c6, c2, #0; Prog Region Control Register as 1

  MOV R2, #0x00001000

  MCR p15, #0, R2, c6, c1, #0; Prog Base Address as 0x1000

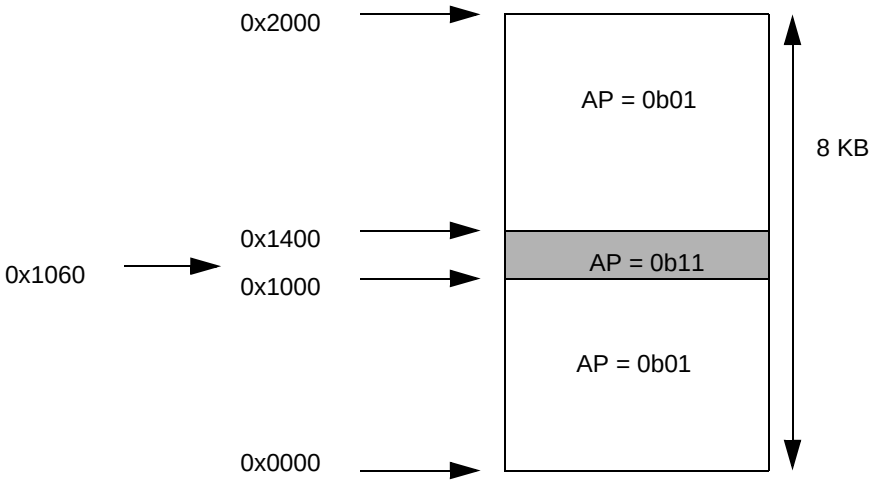
  MOV R2, #0x00000013

  MCR p15, #0, R2, c6, c1, #2; Program Region size as 1KB

  MOV R2, #0x00001300

  MCR p15, #0, R2, c6, c1, #4; Program Access Permission as 0b11

Figure 19-1. Overlapping Protection Attributes



Address 0x1060 falls within protection region 0 and protection region 1 as shown in Figure 19-1. If a user mode access occurs at this location, this access will be allowed to go through because the AP bits are set to 0b11 for the highest numbered region in the overlapping location. AP bits set to 0b11 allows access in both user and privileged modes. User mode access to a location below 0x1000, however, will generate an illegal access exception because AP bits for this region is programmed to allow only privileged mode accesses.

19.3.4 Types of Memory Access Aborts

During a memory access, the protection unit generates the following exceptions if the protection checks by the MPU are violated:

- **Out-of-Range Abort:** This is generated when the CPU is requesting a memory access from a location that is not in a valid region (i.e., the address of the access does not lie within any of the protection regions). This is also referred to as a background fault in the ARM PMSAv6. When the CPU requests an access to an illegal address location, the MPU generates an abort flag to indicate a protection breach in an access. In addition, for a background fault, the MPU sets the status bits to 0b0000 in the CP15 register CPMPUFSR (Section 19.4.4) for data/instruction access violations.
- **Region Permission Abort:** This abort is generated when the CPU requests a memory access (data or instruction) from a location that is in a valid region (i.e., the address for the access lies within any of the protection regions) but does not have the valid access permissions, or when there is an attempt to execute an instruction from a nonexecutable location. This is referred to as a permission fault in the ARM PMSAv6. The MPU generates an abort flag to the CPU to indicate access permission violations when the CPU requests an access to an legal address location. Some examples of such violations are:
  - The application attempts to write to a memory location that is designated as read only
  - The application accessing, in the user mode, a location that is designated as accessible only in privileged mode
  - When there is an instruction access from a location that has the XN bit for that region set to 0b1 an abort is generated. In addition, the MPU also sets the status bits to 0b1101 in the CP15 register CPMPUFSR (Section 19.4.4) for data/instruction access violations.
- **External Abort:** Aborts generated externally of the memory protection unit are referred to as external aborts. Aborts generated by other modules on the device are passed into the MPU, which is responsible for communicating this abort signal to the CPU. The MPU is the focal point for communicating the aborts to the ARM CPU and handles the external aborts. When an external abort occurs, the CP15 register CPMPUFSR bits (Section 19.4.4) are set to 0b1000.

In addition to logging the abort status, the MPU logs the address that caused an abort in the CPMPUDFAR register (Section 19.4.5) or the CPMPUIFAR register (Section 19.4.6), depending on whether it was an instruction or data access that was aborted.

Note:
Only the first address that caused the initial abort is logged in the CPMPUDFAR/ CPMPUIFAR registers. The abort status bits need to be cleared before a second abort occurs so that the details of the second abort can be logged in the CPMPUFSR, CPMPUDFAR, and CPMPUIFAR registers (Section 19.4.4, Section 19.4.5, and Section 19.4.6).

Note:
The default value of the CPMPUFSR register is 0b0000. Therefore, a read from the MPUFSR bits is only valid when an abort has been generated by the MPU. A read at any other time does not yield relevant information.

19.4 MPU Control Registers

This section describes all of the CP15 registers, including the MPU registers. Access to these registers is set using the MCR/MRC instructions in privileged mode, in the format shown below:

MRC/MCR{cond} <opcode1>,<Rd>, <CRn>, <CRm>, <opcode2>

The fields CRn, CRm, and opcode2 are used to specify the address of the MPU registers as summarized in Table 19-3.

Table 19-3. MPU Control Register Map

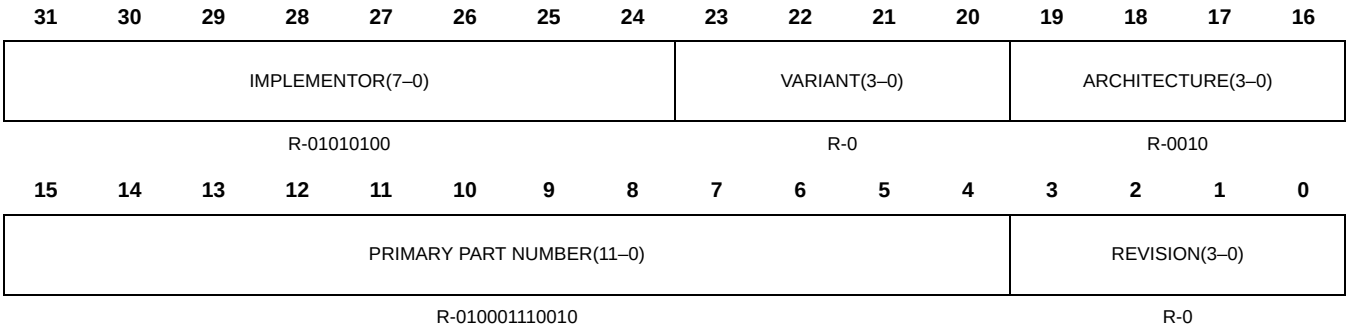
CP15 Definition (CRn,CRm, opcode_2)	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
c0, c0, 0 CPID Page 1117	IMPLEMENTOR(7–0)								VARIANT(3–0)				ARCHITECTURE(3–0)			
	PRIMARY PART NUMBER(11–0)												REVISION(3–0)			
c0, c0, 4 CPMPUTYPE Page 1118	Reserved								Reserved							
	Reserved					NUMBER OF REGIONS(2–0)			Reserved							MPU TYPE
c1, c0, 0 CPGCTRL Page 1119	Reserved							VEC INT	Reserved							
	Reserved								ENDI-AN	Reserved						MPU EN
c5, c0, 0 CPMPUFSR Page 1121	Reserved															
	Reserved												MPU FAULT STATUS(3–0)			
c6, c0, 0 CPMPUDFAR Page 1122	DATA ACCESS FAULT ADDRESS(31–16)															
	DATA ACCESS FAULT ADDRESS(15–0)															
c6, c0, 2 CPMPUIFAR Page 1123	INSTRUCTION ACCESS FAULT ADDRESS(31–16)															
	INSTRUCTION ACCESS FAULT ADDRESS(15–0)															
c6, c1, 0 CPMPU BASEADDR Page 1124	BASE ADDRESS(25–10)															
	BASE ADDRESS(9–0)										Reserved					

CP15 Definition (CRn,CRm, opcode_2)	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
c6, c1, 2 CPMPU REGSIZE Page 1125	Reserved															
	Reserved										SIZE(4–0)					EN
c6, c1, 4 CPMPUAP Page 1127	Reserved															
	Reserved				XN	Reserved			0		AP(1–0)			Reserved		
c6, c2, 0 CPMPU REGCTRL Page 1129	Reserved															
	Reserved															REGION NUMBER(2–0)
c13, c0, 1 CP PROCESSID Page 1130	PROCESS ID(31–16)															
	PROCESS ID(15–0)															

19.4.1 CP15 Identification Register (CIPD)

Figure 19-2 and Table 19-4 illustrate this register.

Figure 19-2. CP15 Identification Register (CIPD) [offset = C0, C0, 0]



R = Read; W = Write; *-n* = Value after reset

.

Table 19-4. CP15 Identification Register (CIPD) Field Descriptions

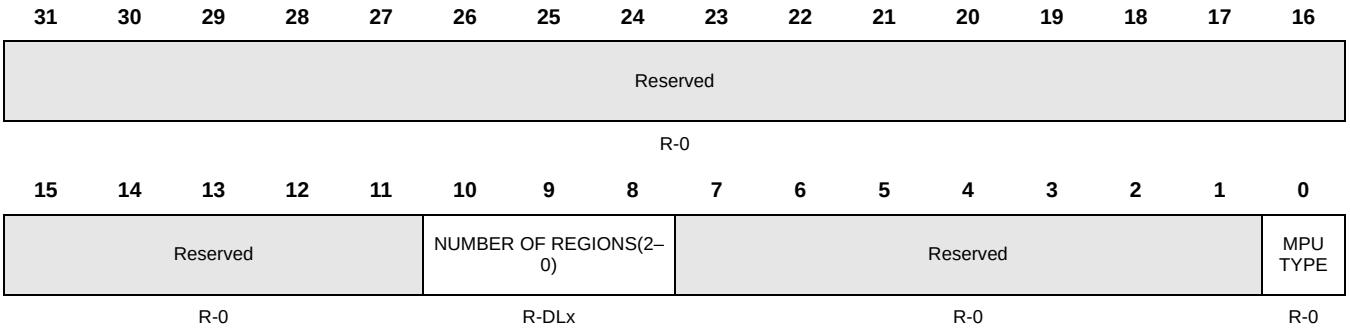
Bit	Name	Value	Description
31–24	IMPLEMENTOR(7–0)	1010100	These bits define the implementor ID for TI and are always set to 0x54 for TI.
23–20	VARIANT(3–0)	0–Fh	These bits return the variant number when read. These bits will be set to 0x0 unless specified otherwise in the datasheet.
19–16	ARCHITECTURE(3–0)	0010	These bits define the architecture of the CPU used in the design. These bits are set to the value shown below: The architecture of the CPU is V4T.
15–4	PRIMARY PART NUMBER(11–0)	010001110010	These bits define the part number of the device. ARM7: This implementation of the ARM MPU follows the PMSAv6 architecture, even though the ARM7TDMI is a V4T implementation. Hence, the primary part number is different from the typical ARM7-based design.
3–0	REVISION NUMBER(3–0)	0	These bits define the revision number for the device and are set to 0x0.

Note:
MRC{cond} p15, 0, Rd, c0, c0, 0;Read instruction

19.4.2 CP15 MPU Type Definition Register (CPMPUTYPE)

Figure 19-3 and Table 19-5 illustrate this register.

Figure 19-3. CP15 MPU Type Definition Register (CPMPUTYPE) [offset = C0, C0, 4]



R = Read; W = Write; *-n* = Value after reset; U = Undefined

.

Table 19-5. CP15 MPU Type Definition Register (CPMPUTYPE) Field Descriptions

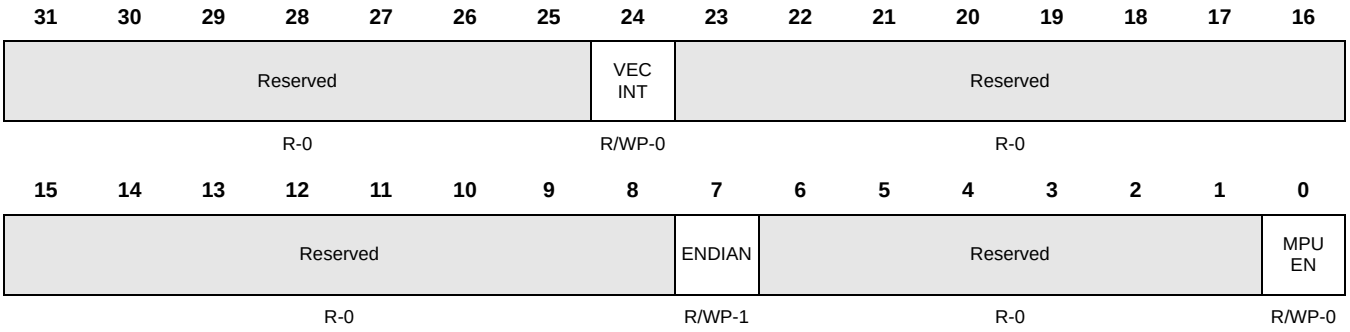
Bit	Name	Value	Description
31–11	Reserved		Reads return zeros and writes have no effect.
10–8	NUMBER OF REGIONS(2–0)	0–8h	Number of data protection regions: These bits define the number of data regions implemented in the MPU. These bits are set to a value representing the number of unified regions in the current version of the MPU as specified in the device-specific datasheet.
7–1	Reserved		Reads return zeros and writes have no effect.
0	MPU TYPE		MPU type: This bit defines the type of MPU that is implemented.
		0	The MPU is of a unified type.
		1	The MPU is not of a unified type.
			Note: ARM7, which has one interface for instructions and data, supports a unified type memory model to process both instruction addresses and data addresses.

Note:
MRC{cond} p15, 0, Rd, c0, c0, 4;Read instruction

19.4.3 CP15 Global Control Register (CPGCTRL)

Figure 19-4 and Table 19-6 illustrate this register.

Figure 19-4. CP15 Global Control Register (CPGCTRL) [offset = C1, C0, 0]



R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-6. CP15 Global Control Register (CPGCTRL) Field Descriptions

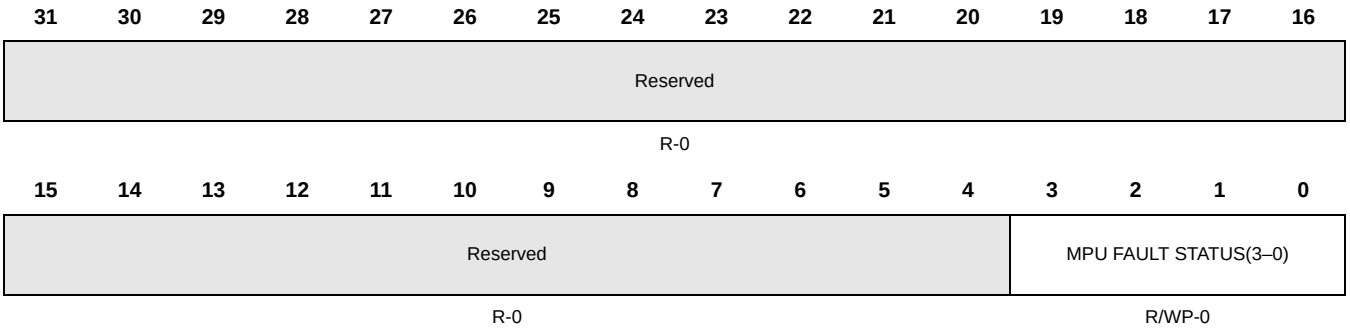
Bit	Name	Value	Description
31–25	Reserved		Reads return zeros and writes have no effect.
24	VEC INT		Vector interrupt enable bit: This bit is used inside the VIM module to configure the hardware vectored interrupts.
		0	The vector interrupt is disabled.
		1	The vector interrupt is enabled.
			Note: This bit is present to comply with ARM7 architecture. This bit has no specific functionality.
23–8	Reserved		Reads return zeros and writes have no effect.
7	ENDIAN		This bit defines the endianess of the system.
		0	The little endian is in effect.
		1	The big endian is in effect.
			Note: All TMSx470 devices should be configured to big endian mode.
6–1	Reserved		Reads return zeros and writes have no effect.
0	MPU EN		MPU enable: This bit is used to enable the MPU.
		0	The MPU is disabled.
		1	The MPU is enabled.

Note:
MRC{cond} p15, 0, Rd, c1, c0, 0;Read instruction
MCR{cond} p15, 0, Rd, c1, c0, 0;Write instruction

19.4.4 CP15 MPU Fault Status Register (CPMPUFSR)

Figure 19-5 and Table 19-7 illustrate this register.

Figure 19-5. CP15 MPU Fault Status Register (CPMPUFSR) [offset = C5, C0, 0]



R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-7. CP15 MPU Fault Status Register (CPMPUFSR) Field Descriptions

Bit	Name	Value	Description
31–4	Reserved		Reads return zeros and writes have no effect.
3–0	MPU FAULT STATUS(3–0)		Data access fault status field: These bits are used to specify the type of violation. The valid values for the status violations are listed below:
		1000	An external abort occurred.
		1101	A region permission abort (permission fault) occurred.
		0000	An out-of-region abort (background fault) occurred.
			Note: The default value of this register at reset is 0x0000. Note that the background fault in MPU has the same value as the default value. Therefore, it is important to understand that a read to the fault status register is of significance only when an ABORT has been generated by MPU.

Note:
MRC{cond} p15, 0, Rd, c5, c0, 0;Read instruction
MCR{cond} p15, 0, Rd, c5, c0, 0;Write instruction

19.4.5 CP15 MPU Data Fault Address Register (CPMPUDFAR)

Figure 19-6 and Table 19-8 illustrate this register.

Figure 19-6. MPU Data Fault Address Register (CPMPUDFAR) [offset = C6, C0, 0]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA ACCESS FAULT ADDRESS(31–16)															
R/WP-0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA ACCESS FAULT ADDRESS(15–0)															
R/WP-0															

R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-8. MPU Data Fault Address Register (CPMPUDFAR) Field Descriptions

Bit	Name	Value	Description
31–0	DATA ACCESS FAULT ADDRESS(31–0)	0–FFFF FFFFh	This register stores the value of the **first** address location that was violated during a data access. If there are back-to-back aborts generated by the MPU, then only the address of the first aborted location is stored in this register. The value in this register has significance only when a data ABORT is generated by the MPU.

Note:
MRC{cond} p15, 0, Rd, c6, c0, 0;Read instruction
MCR{cond} p15, 0, Rd, c6, c0, 0;Write instruction

19.4.6 CP15 MPU Instruction Fault Address Register (CPMPUIFAR)

Figure 19-7 and Table 19-9 illustrate this register.

Figure 19-7. CP15 MPU Instruction Fault Address Register (CPMPUIFAR) [offset = C6, C0, 2]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INSTRUCTION ACCESS FAULT ADDRESS(31–16)															
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSTRUCTION ACCESS FAULT ADDRESS(15–0)															
R/WP-0															

R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-9. CP15 MPU Instruction Fault Address Register (CPMPUIFAR) Field Descriptions

Bit	Name	Value	Description
31–0	INSTRUCTION ACCESS FAULT ADDRESS(31–0)	0–FFFF FFFFh	This register stores the address of the **first** instruction access that was aborted by MPU. If there are back-to-back aborts generated by the MPU, then only the address of the first aborted location is stored in this register. The value in this register is significant only when the MPU generates an ABORT.

Note:
MRC{cond} p15, 0, Rd, c6, c0, 2;Read instruction
MCR{cond} p15, 0, Rd, c6, c0, 2;Write instruction

19.4.7 CP15 MPU Base Address Register (CPMPUBASEADDR)

Figure 19-8 and Table 19-10 illustrate this register.

Figure 19-8. CP15 MPU Base Address Register (CPMPUBASEADDR) [offset = C6, C1, 0]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE ADDRESS(25–10)															
R/WP-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE ADDRESS(9–0)										Reserved					
R/WP-0										R-0					

R = Read; WP = Write in privileged mode only; *-n* = Value after reset

Table 19-10. CP15 MPU Base Address Register (CPMPUBASEADDR) Field Descriptions

Bit	Name	Value	Description
31–6	BASE ADDRESS(25–0)	0–3FF FFFFh	Base address: These bits specify the address of the first byte for each of the protection regions (selected by the CPMPUREGCTRL register; Section 19.4.10). The base address is always required to be a multiple of the region size. Since the minimum region size is 64 bytes, bits (5–0) are always 0 and therefore need not be programmed. Address accessed[31:0] = {Base Address[25:0], 000000}. In addition to the requirement that the base address must always be a multiple of the region size, there may be situations where bits other than (5–0) may need to be programmed to 0. Refer to Table 19-12 for details.
5-0	Reserved		Reads return zeros and writes have no effect.

Note:
MRC{cond} p15, 0, Rd, c6, c1, 0;Read instruction
MCR{cond} p15, 0, Rd, c6, c1, 0;Write instruction

19.4.8 CP15 MPU Region Size Register (CPMPUREGSIZE)

Figure 19-9 and Table 19-11 illustrate this register.

Figure 19-9. CP15 MPU Region Size Register (CPMPUREGSIZE) [offset = C6, C1, 2]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										SIZE(4–0)					EN
R-0										R/WP-0					R/WP-0

R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-11. CP15 MPU Region Size Register (CPMPUREGSIZE) Field Descriptions

Bit	Name	Value	Description
31–6	Reserved		Reads return zeros and writes have no effect.
5–1	SIZE(4–0)	0–1Fh	MPU region size: These bits are used to program the size of all the protection regions in the MPU (selected by the CPMPUREGCTRL register; Section 19.4.10). Table 19-12 describes the encoding of the five bits for the memory protection region.
0	EN		MPU channel enable: This bit is used to enable or disable the associated protection region:
		0	The protection region is disabled.
		1	The protection region is enabled.

Table 19-12. TSIZE Bits Coding

Size Field	Region Size	Base Address Constraints (CPMPUBASEADDR)
00000-00100	Not Defined	-
00101	64 Bytes	None
00110	128 Bytes	Bit[6] must be 0.
00111	256 Bytes	Bit[7:6] must be 0.
01000	512 Bytes	Bit[8:6] must be 0.
01001	1 KB	Bit[9:6] must be 0.
01010	2 KB	Bit[10:6] must be 0.
01011	4 KB	Bit[11:6] must be 0.
01100	8 KB	Bit[12:6] must be 0.
01101	16 KB	Bit[13:6] must be 0.
01110	32 KB	Bit[14:6] must be 0.
01111	64 KB	Bit[15:6] must be 0.

Size Field	Region Size	Base Address Constraints (CPMPUBASEADDR)
10000	128 KB	Bit[16:6] must be 0.
10001	256 KB	Bit[17:6] must be 0.
10010	512 KB	Bit[18:6] must be 0.
10011	1 MB	Bit[19:6] must be 0.
10100	2 MB	Bit[20:6] must be 0.
10101	4 MB	Bit[21:6] must be 0.
10110	8 MB	Bit[22:6] must be 0.
10111	16 MB	Bit[23:6] must be 0.
11000	32 MB	Bit[24:6] must be 0.
11001	64 MB	Bit[25:6] must be 0.
11010	128 MB	Bit[26:6] must be 0.
11011	256 MB	Bit[27:6] must be 0.
11100	512 MB	Bit[28:6] must be 0.
11101	1 GB	Bit[29:6] must be 0.
11110	2 GB	Bit[30:6] must be 0.
11111	4 GB	Bit[31:6] must be 0.

Note:
MRC{cond} p15, 0, Rd, c6, c1, 2;Read instruction
MCR{cond} p15, 0, Rd, c6, c1, 2;Write instruction

19.4.9 CP15 MPU Data Region Access Permission Register (CPMPUDAP)

Figure 19-10 and Table 19-13 illustrate this register.

Figure 19-10. CP15 MPU Data Region Access Permission Register (CPMPUDAP) [offset = C6, C1, 4]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			XN	Reserved	0	AP(1–0)		Reserved							
R-0			R/W-0	R-0	R-0	R/W-0		R-0							

R = Read; W = Write; *-n* = Value after reset; U = Undefined

.

Table 19-13. CP15 MPU Data Region Access Permission Register (CPMPUDAP) Field Descriptions

Bit	Name	Value	Description
31–13	Reserved		Reads return zeros and writes have no effect.
12	XN		Executable permission: This bit specifies the executable permissions for the region specified by the CPMPUREGCTRL register (Section 19.4.10).
		0	This is an executable memory region.
		1	This is a non-executable memory region.
11	Reserved		Reads return zeros and writes have no effect.
10	0		This bit is always 0 as we do not have imprecise aborts in ARM7. Read from this bit is always 0 and writes have no effect.
9–8	AP(1–0)		Access permission: The access permission (AP) for a region specified by the CPMPUREGCTRL register are used to determine if an access is allowable to that location. The reset value for the AP bits are 0x00. The AP bits are decoded as follows:
		Value	**Privileged Mode** / **User Mode**
		00	No Access / No Access
		01	Read/Write / No Access
		10	Read/Write / Read Only
		11	Read/Write / Read/Write
7–0	Reserved		Reads return zeros and writes have no effect.

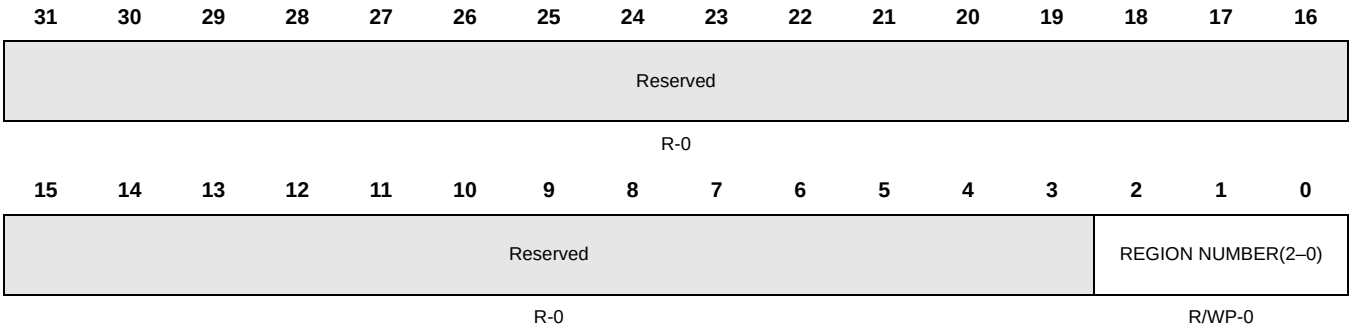
Note:
MRC{cond} p15, 0, Rd, c6, c1, 4;Read instruction
MCR{cond} p15, 0, Rd, c6, c1, 4;Write instruction

Note:
Bits 5:0 are defined as TEX, S, C, and B bits in the PMSAv6 specification. However, these bits have no significance in the ARM7 architecture for which this MPU is specified.

19.4.10 CP15 MPU Region Control Register (CPMPUREGCTRL)

Figure 19-11 and Table 19-14 illustrate this register.

Figure 19-11. CP15 MPU Region Control Register (CPMPUREGCTRL) [offset = C6, C2, 0]



R = Read; WP = Write in privileged mode only; *-n* = Value after reset; U = Undefined

.

Table 19-14. CP15 MPU Region Control Register (CPMPUREGCTRL) Field Descriptions

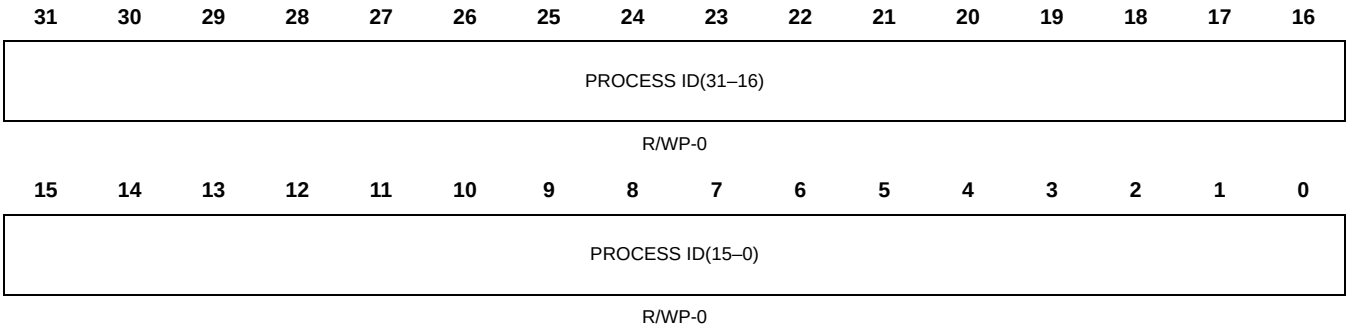
Bit	Name	Value	Description
31–3	Reserved		Reads return zeros and writes have no effect.
2–0	Region Number(2–0)	0–7h	These bits are used to define the region that is being accessed when there is a read/write request to the CPMPUBASEADDR (Section 19.4.7), CPMPUREGSIZE (Section 19.4.8), and CPMPUDAP (Section 19.4.9) registers. The value in this register is decoded to provide information on which protection region is accessed during a read/write to the above registers.

Note:
MRC{cond} p15, 0, Rd, c6, c2, 0;Read instruction
MCR{cond} p15, 0, Rd, c6, c2, 0;Write instruction

19.4.11 CP15 Process ID Register (CPPROCESSID)

Figure 19-12 and Table 19-15 illustrate this register.

Figure 19-12. CP15 Process ID Register (CPPROCESSID) [offset = C13, C0, 1]



R = Read; WP = Write in privileged mode only; *-n* = Value after reset

.

Table 19-15. CP15 Process ID Register (CPPROCESSID) Field Descriptions

Bit	Name	Value	Description
31–0	Process ID(31–0)	0–FFFF FFFFh	This register allows the CPU to log the process that is currently running, thereby allowing the identification of the process that was aborted by the MPU.

Note:
MRC{cond} p15, 0, Rd, c13, c0, 1;Read instruction
MCR{cond} p15, 0, Rd, c13, c0, 1;Write instruction

Revision History

This section lists all of the significant content changes to this manual.

20.1 Revision History

Table 20-1 shows the revision history of this document.

Table 20-1. Revision History

Date	Version	Description
June 2008	Revision A	A distinct SPI chapter was added in addition to the MibSPI chapter.
June 2008	Revision A	The SYSESR and ITIFLAG registers were corrected to include all reset status flags and illegal transaction flags included in the device.
June 2008	Revision A	The HET loopback mode description was added.
June 2008	Revision A	The IAHBILLADDR and DAHBILLADDR register definitions were added to the documentation in support of SYSESR and ITIFLAG register corrections.
June 2008	Revision A	The DIEIDH and DIEIDL registers were updated to reserved register space as these registers are not populated in F05 devices.
June 2008	Revision A	The definition of Pull Selection register function was clarified throughout the document.
June 2008	Revision A	The SCC/HECC chapter was updated to clarify the documents description of the SCC/HECC operation.
June 2008	Revision A	The Serial Communication Interface (SCI)/Buffered Local Interconnect Network (BLIN) Module chapter was updated to describe certain behaviors associated with HW clearing of error flags and missed interrupts.
April 2011	Revision B	Added a note to the Real Time Interrupt Clock Domain section regarding limitations of the ratio of RTI CLK to VCLK when a source other than VCLK is configured for RTICLK.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com	Wireless	www.ti.com/wireless-apps
RF/IF and ZigBee® Solutions	www.ti.com/lprf		
	TI E2E Community Home Page		e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265