MC68HC908GR8B

Data Sheet

M68HC08
Microcontrollers

MC68HC908GR8B
Rev. 3.0
04/2007

freescale.com

freescale™
semiconductor

MC68HC908GR8B

Data Sheet

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

http://freescale.com/

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
July, 2005	0	Initial release	N/A
June, 2006	1.0	10.5 Clock Generator Module (CGM) — Updated description to remove erroneous information.	104
		19.5 5.0 V DC Electrical Characteristics — Added parameters for DC injection current and input hysteresis.	231
		19.6 3.0 V DC Electrical Characteristics — Added parameters for DC injection current and input hysteresis.	233
September, 2006	2.0	19.5 5.0 V DC Electrical Characteristics — Corrected parameters for I/O ports Hi-Z leakage current and Stop with LVI and TBM enabled.	231
		19.6 3.0 V DC Electrical Characteristics — Corrected parameters for I/O ports Hi-Z leakage current and Stop with LVI and TBM enabled.	233
April, 2007	3.0	Chapter 5 Configuration Register (CONFIG) — Replaced COPCLK with CGMXCLK	73
		Figure 5-2. Configuration Register 1 (CONFIG1) — Corrected the timeout period definitions following this figure.	74
		10.6.2 Stop Mode — Replaced COPCLK with CGMXCLK	105

List of Chapters

Table of Contents

Chapter 1
General Description

Chapter 2
Memory

Chapter 3
Analog-to-Digital Converter (ADC)

Chapter 4
Clock Generator Module (CGM)

Chapter 5
Configuration Register (CONFIG)

Chapter 6
Computer Operating Properly (COP) Module

Chapter 7
Central Processor Unit (CPU)

Chapter 8
External Interrupt (IRQ)

Chapter 9
Keyboard Interrupt Module (KBI)

Chapter 10
Low-Power Modes

Chapter 11
Low-Voltage Inhibit (LVI)

Chapter 12
Input/Output (I/O) Ports

Chapter 13
Serial Communications Interface (SCI) Module

Chapter 14
System Integration Module (SIM)

Chapter 15
Serial Peripheral Interface (SPI) Module

Chapter 16
Timebase Module (TBM)

Chapter 17
Timer Interface Module (TIM1 and TIM2)

Chapter 18
Development Support

Chapter 19
Electrical Specifications

Chapter 20
Ordering Information and Mechanical Specifications

Chapter 1
General Description

1.1 Introduction

The MC68HC908GR8B is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

1.2 Features

For convenience, features have been organized to reflect:

- Standard features
- Features of the CPU08

1.2.1 Standard Features

Features include:

- High-performance M68HC08 architecture optimized for C-compilers
- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families
- 8-MHz internal bus frequency
- Clock generation module supporting 1-MHz to 8-MHz crystals
- FLASH program memory security(1)
- On-chip programming firmware for use with host personal computer which does not require high voltage for entry
- In-system programming (ISP)
- System protection features:
  - Optional computer operating properly (COP) reset
  - Low-voltage detection with reset and trip points for 3.0-V operation
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- Low-power design; fully static with stop and wait modes
- Standard low-power modes of operation:
  - Wait mode
  - Stop mode
- Master reset pin and power-on reset (POR)
- 7680 bytes of on-chip FLASH memory with in-circuit programming capabilities of FLASH program memory.
- 384 bytes of on-chip random-access memory (RAM)
- 544 bytes of FLASH programming routines read-only memory (ROM)

---

1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

- Serial peripheral interface (SPI) module
- Serial communications interface (SCI) module
- One 16-bit, 2-channel timer (TIM1) and one 16-bit, 1-channel timer (TIM2) interface modules with selectable input capture, output compare, and pulse-width modulation (PWM) capability on each channel
- Up to 6-channel, 8-bit successive approximation analog-to-digital converter (ADC) depending on package choice
- BREAK (BRK) module to allow single breakpoint setting during in-circuit debugging
- Internal pullups on $\overline{\text{IRQ}}$ and $\overline{\text{RST}}$ to reduce system cost
- Up to 21 general-purpose input/output (I/O) pins
- Selectable pullups on inputs only on ports A, C, and D. Selection is on an individual port bit basis. During output mode, pullups are disengaged.
- High current 10-mA sink/source capability on all port pins
- Higher current 15-mA sink/source capability on PTC0–PTC1
- Timebase module (TBM) with clock prescaler circuitry for eight user selectable periodic real-time interrupts with optional active clock source during stop mode for periodic wakeup from stop using an external crystal
- User selection of having the oscillator enabled or disabled during stop mode
- 4-bit keyboard wakeup port
- Available packages:
  - 32-pin low-profile quad flat pack (LQFP)
  - 28-pin plastic dual in-line package (DIP)
  - 28-pin small outline integrated circuit (SOIC)
- Specific features of the MC68HC908GR8B in 28-pin DIP and 28-pin SOIC are:
  - Port B is only 4 bits: PTB0–PTB3; 4-channel ADC module
  - No port C bits

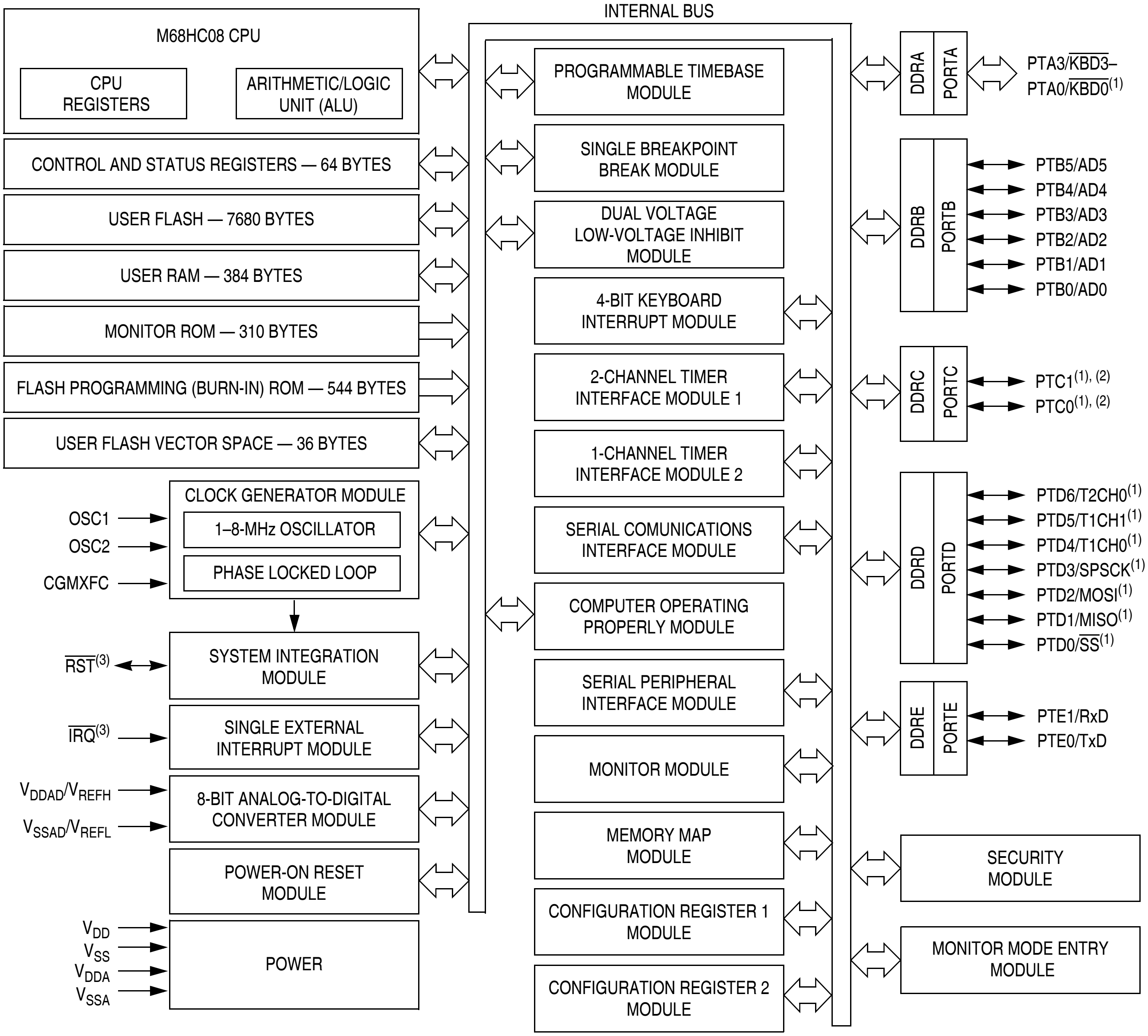
1.2.2 Features of the CPU08

Features of the CPU08 include:

- Enhanced HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 × 8 multiply instruction
- Fast 16/8 divide instruction
- Binary-coded decimal (BCD) instructions
- Optimization for controller applications
- Efficient C language support

1.3 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908GR8B.



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 1-1. MCU Block Diagram

1.4 Pin Assignments

Figure 1-2 illustrations the pin assignments for the 32-pin LQFP. Figure 1-3 illustrate the pin assignments for the 28-pin DIP and 28-pin SOIC.

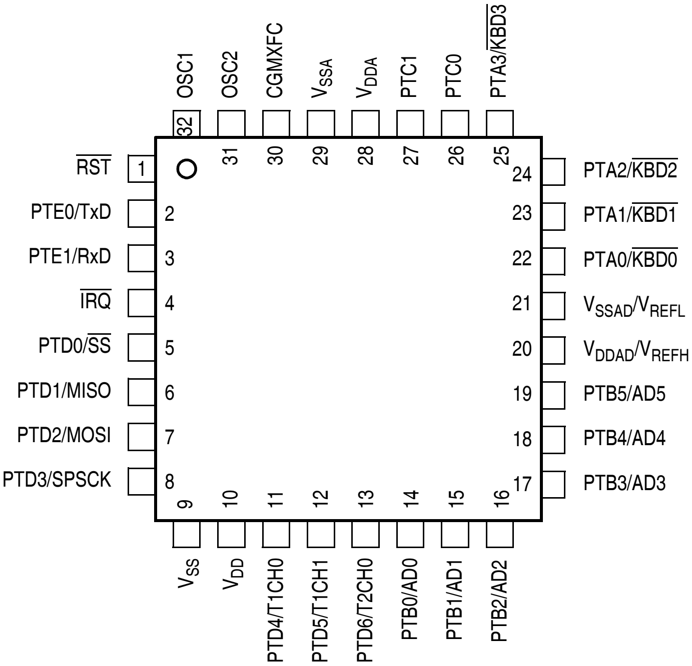
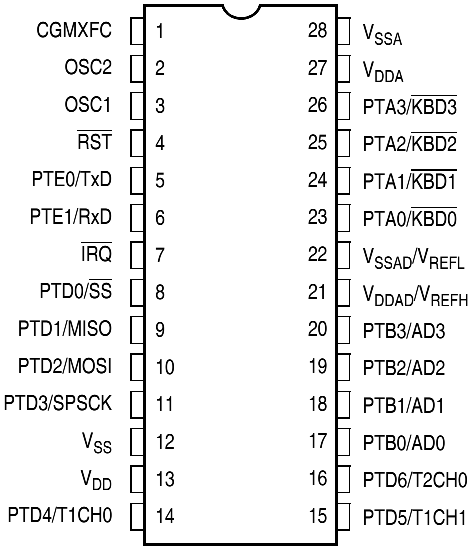


Figure 1-2. 32-Pin LQFP Pin Assignments



NOTE: Ports PTB4, PTB5, PTC0, and PTC1 are available only with the LQFP.

Figure 1-3. 28-Pin DIP and SOIC Pin Assignments

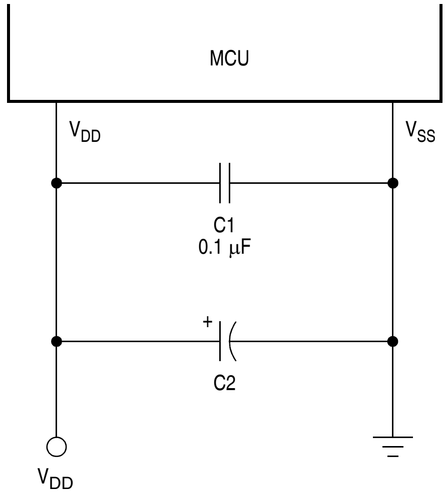
1.5 Pin Functions

Descriptions of the pin functions are provided here.

1.5.1 Power Supply Pins (V_{DD} and V_{SS})

V_{DD} and V_{SS} are the power supply and ground pins. The MCU operates from a single power supply.

Fast signal transitions on MCU pins place high, short-duration current demands on the power supply. To prevent noise problems, take special care to provide power supply bypassing at the MCU as Figure 1-4 shows. Place the C1 bypass capacitor as close to the MCU as possible. Use a high-frequency-response ceramic capacitor for C1. C2 is an optional bulk current bypass capacitor for use in applications that require the port pins to source high current levels.



Note: Component values shown represent typical applications.

Figure 1-4. Power Supply Bypassing

1.5.2 Oscillator Pins (OSC1 and OSC2)

OSC1 and OSC2 are the connections for an external crystal, resonator, or clock circuit. See Chapter 4 Clock Generator Module (CGM).

1.5.3 External Reset Pin ($\overline{\text{RST}}$)

A 0 on the $\overline{\text{RST}}$ pin forces the MCU to a known startup state. $\overline{\text{RST}}$ is bidirectional, allowing a reset of the entire system. It is driven low when any internal reset source is asserted. This pin contains an internal pullup resistor. See Chapter 14 System Integration Module (SIM).

1.5.4 External Interrupt Pin ($\overline{\text{IRQ}}$)

$\overline{\text{IRQ}}$ is an asynchronous external interrupt pin. This pin contains an internal pullup resistor. See Chapter 8 External Interrupt (IRQ).

1.5.5 CGM Power Supply Pins (V_{DDA} and V_{SSA})

V_{DDA} and V_{SSA} are the power supply pins for the analog portion of the clock generator module (CGM). Decoupling of these pins should be as per the digital supply. See Chapter 4 Clock Generator Module (CGM).

1.5.6 External Filter Capacitor Pin (V_{CGMXFC})

CGMXFC is an external filter capacitor connection for the CGM. See Chapter 4 Clock Generator Module (CGM).

1.5.7 ADC Power Supply/Reference Pins (V_{DDAD}/V_{REFH} and V_{SSAD}/V_{REFL})

V_{DDAD} and V_{SSAD} are the power supply pins to the analog-to-digital converter (ADC). V_{REFH} and V_{REFL} are the reference voltage pins for the ADC. V_{REFH} is the high reference supply for the ADC, and by default the V_{DDAD}/V_{REFH} pin should be externally filtered and connected to the same voltage potential as V_{DD}. V_{REFL} is the low reference supply for the ADC, and by default the V_{SSAD}/V_{REFL} pin should be connected to the same voltage potential as V_{SS}. See Chapter 3 Analog-to-Digital Converter (ADC).

1.5.8 Port A Input/Output (I/O) Pins (PTA3/$\overline{\text{KBD3}}$–PTA0/$\overline{\text{KBD0}}$)

PTA3–PTA0 are special-function, bidirectional I/O port pins. Any or all of the port A pins can be programmed to serve as keyboard interrupt pins. See Chapter 12 Input/Output (I/O) Ports and Chapter 9 Keyboard Interrupt Module (KBI).

These port pins also have selectable pullups when configured for input mode. The pullups are disengaged when configured for output mode. The pullups are selectable on an individual port bit basis.

1.5.9 Port B I/O Pins (PTB5/AD5–PTB0/AD0)

PTB5–PTB0 are special-function, bidirectional I/O port pins that can also be used for analog-to-digital converter (ADC) inputs. See Chapter 12 Input/Output (I/O) Ports and Chapter 3 Analog-to-Digital Converter (ADC).

1.5.10 Port C I/O Pins (PTC1 and PTC0)

PTC1 and PTC0 are general-purpose, bidirectional I/O port pins. PTC1–PTC0 are only available on the 32-pin LQFP package. See Chapter 12 Input/Output (I/O) Ports.

These port pins also have selectable pullups when configured for input mode. The pullups are disengaged when configured for output mode. The pullups are selectable on an individual port bit basis.

1.5.11 Port D I/O Pins (PTD6/T2CH0–PTD0/$\overline{\text{SS}}$)

PTD6–PTD0 are special-function, bidirectional I/O port pins. PTD3–PTD0 can be programmed to be serial peripheral interface (SPI) pins, while PTD6–PTD4 can be individually programmed to be timer interface module (TIM1 and TIM2) pins. See Chapter 17 Timer Interface Module (TIM1 and TIM2), Chapter 15 Serial Peripheral Interface (SPI) Module, and Chapter 12 Input/Output (I/O) Ports.

These port pins also have selectable pullups when configured for input mode. The pullups are disengaged when configured for output mode. The pullups are selectable on an individual port bit basis.

1.5.12 Port E I/O Pins (PTE1/RxD and PTE0/TxD)

PTE1 and PTE0 are special-function, bidirectional I/O port pins. These pins can also be programmed to be serial communications interface (SCI) pins. See Chapter 13 Serial Communications Interface (SCI) Module and Chapter 12 Input/Output (I/O) Ports.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (either V_{DD} *or* V_{SS}*). Although the I/O ports of the MC68HC908GR8B do not require termination, termination is recommended to reduce the possibility of static damage.*

Chapter 2
Memory

2.1 Introduction

The CPU08 can address 64 Kbytes of memory space. The memory map, shown in Figure 2-1, includes:

- 7680 bytes of user FLASH memory
- 384 bytes of random-access memory (RAM)
- 544 bytes of FLASH burn-in routines in ROM
- 36 bytes of user-defined vectors
- 310 bytes of monitor ROM

2.2 Unimplemented Memory Locations

Accessing an unimplemented location can cause an illegal address reset. In the memory map (Figure 2-1) and in register figures in this document, unimplemented locations are shaded.

2.3 Reserved Memory Locations

Accessing a reserved location can have unpredictable effects on microcontroller (MCU) operation. In the Figure 2-1 and in register figures in this document, reserved locations are marked with the word Reserved or with the letter R.

2.4 Input/Output (I/O) Section

Most of the control, status, and data registers are in the zero page area of \$0000–\$003F. Additional I/O registers have these addresses:

- \$FE00; break status register, SBSR
- \$FE01; SIM reset status register, SRSR
- \$FE02; reserved
- \$FE03; break flag control register, SBFCR
- \$FE04; interrupt status register 1, INT1
- \$FE05; interrupt status register 2, INT2
- \$FE06; interrupt status register 3, INT3
- \$FE07; reserved
- \$FE08; FLASH control register, FLCR
- \$FE09; break address register high, BRKH
- \$FE0A; break address register low, BRKL
- \$FE0B; break status and control register, BRKSCR
- \$FE0C; LVI status register, LVISR
- \$FF7E; FLASH block protect register, FLBPR

Data registers are shown in Figure 2-2. Table 2-1 is a list of vector locations.

$0000 ↓ $003F	I/O REGISTERS 64 BYTES
$0040 ↓ $01BF	RAM 384 BYTES
$01C0 ↓ $1BFF	UNIMPLEMENTED 6720 BYTES
$1C00 ↓ $1E1F	RESERVED FOR INTEGRATED FLASH BURN-IN ROUTINES 544 BYTES
$1E20 ↓ $DFFF	UNIMPLEMENTED 49,632 BYTES
$E000 ↓ $EDFF $EE00 ↓ $FDFF	MC68HC908GR8B FLASH MEMORY 7680 BYTES
$FE00	SIM BREAK STATUS REGISTER (SBSR)
$FE01	SIM RESET STATUS REGISTER (SRSR)
$FE02	RESERVED
$FE03	SIM BREAK FLAG CONTROL REGISTER (SBFCR)
$FE04	INTERRUPT STATUS REGISTER 1 (INT1)
$FE05	INTERRUPT STATUS REGISTER 2 (INT2)
$FE06	INTERRUPT STATUS REGISTER 3 (INT3)
$FE07	Reserved for FLASH Test Control Register (FLTCR)

Figure 2-1. Memory Map

$FE08	FLASH CONTROL REGISTER (FLCR)
$FE09	BREAK ADDRESS REGISTER HIGH (BRKH)
$FE0A	BREAK ADDRESS REGISTER LOW (BRKL)
$FE0B	BREAK STATUS AND CONTROL REGISTER (BRKSCR)
$FE0C	LVI STATUS REGISTER (LVISR)
$FE0D ↓ $FE0F	RESERVED 3 BYTES
$FE10 ↓ $FE1F	UNIMPLEMENTED 16 BYTES RESERVED FOR COMPATIBILITY WITH MONITOR CODE FOR A-FAMILY PARTS
$FE20 ↓ $FF55	MONITOR ROM 310 BYTES
$FF56 ↓ $FF7D	UNIMPLEMENTED 40 BYTES
$FF7E	FLASH BLOCK PROTECT REGISTER (FLBPR)
$FF7F ↓ $FFDB	UNIMPLEMENTED 93 BYTES
$FFDC ↓ $FFFE	FLASH VECTORS (36 BYTES INCLUDING $FFFF)
$FFFF	LOW BYTE OF RESET VECTOR WHEN READ COP CONTROL REGISTER (COPCTL)

Note: $FFF6–$FFFD
contains
8 security bytes

Figure 2-1. Memory Map (Continued)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA) See page 115.	Read:	0	0	0	0	PTA3	PTA2	PTA1	PTA0
		Write:								
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB) See page 118.	Read:	0	0	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Write:								
		Reset:	Unaffected by reset							
$0002	Port C Data Register (PTC) See page 120.	Read:	0	0	0	0	0	0	PTC1	PTC0
		Write:								
		Reset:	Unaffected by reset							
$0003	Port D Data Register (PTD) See page 122.	Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Write:								
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA) See page 116.	Read:	0	0	0	0	DDRA3	DDRA2	DDRA1	DDRA0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB) See page 118.	Read:	0	0	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC) See page 120.	Read:	0	0	0	0	0	0	DDRC1	DDRC0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD) See page 123.	Read:	0	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE) See page 125.	Read:	0	0	0	0	0	0	PTE1	PTE0
		Write:								
		Reset:	Unaffected by reset							
$0009 ↓ $000B	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000C	Data Direction Register E (DDRE) See page 126.	Read:	0	0	0	0	0	0	DDRE1	DDRE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 1 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000D	Port A Input Pullup Enable Register (PTAPUE) See page 117.	Read:	0	0	0	0	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$000E	Port C Input Pullup Enable Register (PTCPUE) See page 122.	Read:	0	0	0	0	0	0	PTCPUE1	PTCPUE0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	0	0	0	0	0	0	0	0
$000F	Port D Input Pullup Enable Register (PTDPUE) See page 125.	Read:	0	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
		Write:	▨							
		Reset:	0	0	0	0	0	0	0	0
$0010	SPI Control Register (SPCR) See page 189.	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0011	SPI Status and Control Register (SPSCR) See page 190.	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:	▨		▨	▨	▨			
		Reset:	0	0	0	0	1	0	0	0
$0012	SPI Data Register (SPDR) See page 192.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0013	SCI Control Register 1 (SCC1) See page 144.	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2) See page 146.	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3) See page 147.	Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
		Write:	▨							
		Reset:	U	0	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1) See page 149.	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2) See page 151.	Read:	0	0	0	0	0	0	BKF	RPF
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 2 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0018	SCI Data Register (SCDR) See page 152.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR) See page 152.	Read:	▨	▨	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001A	Keyboard Status and Control Register (INTKBSCR) See page 101.	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:	▨	▨	▨	▨	▨	ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001B	Keyboard Interrupt Enable Register (INTKBIER) See page 102.	Read:	▨	▨	▨	▨	KBIE3	KBIE2	KBIE1	KBIE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$001C	Timebase Module Control Register (TBCR) See page 195.	Read:	TBIF	TBR2	TBR1	TBR0	0	TBIE	TBON	R
		Write:	▨				TACK			
		Reset:	0	0	0	0	0	0	0	0
$001D	IRQ Status and Control Register (INTSCR) See page 96.	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:	▨	▨	▨	▨	▨	ACK		
		Reset:	0	0	0	0	0	0	0	0
$001E	Configuration Register 2 (CONFIG2)(1) See page 73.	Read:	0	0	0	0	0	0	OSC-STOPENB	SCIBDSRC
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	0	0	0	0	0	0	0	0
$001F	Configuration Register 1 (CONFIG1)(1) See page 74.	Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVI5OR3	SSREC	STOP	COPD
		Write:			▨	▨	▨			
		Reset:	0	0	0	0	0	0	0	0
$0020	Timer 1 Status and Control Register (T1SC) See page 207.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	▨			
		Reset:	0	0	1	0	0	0	0	0
$0021	Timer 1 Counter Register High (T1CNTH) See page 208.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0022	Timer 1 Counter Register Low (T1CNTL) See page 208.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 3 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0023	Timer 1 Counter Modulo Register High (T1MODH) See page 209.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	Timer 1 Counter Modulo Register Low (T1MODL) See page 209.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	Timer 1 Channel 0 Status and Control Register (T1SC0) See page 212.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	Timer 1 Channel 0 Register High (T1CH0H) See page 212.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0027	Timer 1 Channel 0 Register Low (T1CH0L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0028	Timer 1 Channel 1 Status and Control Register (T1SC1) See page 212.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		▨					
		Reset:	0	0	0	0	0	0	0	0
$0029	Timer 1 Channel 1 Register High (T1CH1H) See page 212.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$002A	Timer 1 Channel 1 Register Low (T1CH1L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$002B	Timer 2 Status and Control Register (T2SC) See page 209.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	▨			
		Reset:	0	0	1	0	0	0	0	0
$002C	Timer 2 Counter Register High (T2CNTH) See page 208.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$002D	Timer 2 Counter Register Low (T2CNTL) See page 208.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented R = Reserved U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$002E	Timer 2 Counter Modulo Register High (T2MODH) See page 209.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$002F	Timer 2 Counter Modulo Register Low (T2MODL) See page 209.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0030	Timer 2 Channel 0 Status and Control Register (T2SC0) See page 209.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0031	Timer 2 Channel 0 Register High (T2CH0H) See page 209.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0032	Timer 2 Channel 0 Register Low (T2CH0L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0033 ↓ $0035	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0036	PLL Control Register (PCTL) See page 64.	Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
		Write:		▨						
		Reset:	0	0	1	0	0	0	0	0
$0037	PLL Bandwidth Control Register (PBWC) See page 65.	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	0	0	0	0	R
		Write:		▨		▨	▨	▨	▨	
		Reset:	0	0	0	0	0	0	0	0
$0038	PLL Multiplier Select High Register (PMSH) See page 66.	Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$0039	PLL Multiplier Select Low Register (PMSL) See page 67.	Read:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003A	PLL VCO Select Range Register (PMRS) See page 67.	Read:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
		Write:								
		Reset:	0	1	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 5 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003B	PLL Reference Divider Select Register (PMDS) See page 68.	Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	1
$003C	ADC Status and Control Register (ADSCR) See page 49.	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:	R							
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register (ADR) See page 51.	Read:	AD7	AD6	AD5	AD4	A3	AD2	AD1	AD0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	Unaffected by reset							
$003E	ADC Clock Register (ADCLK) See page 51.	Read:	ADIV2	ADIV1	ADIV0	ADICLK	0	0	0	0
		Write:					▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$003F	Reserved		R	R	R	R	R	R	R	R
$FE00	SIM Break Status Register (SBSR) See page 218.	Read:	R	R	R	R	R	R	SBSW	R
		Write:							(Note 1)	
		Reset:	0	0	0	0	0	0	0	0

1. Writing a 0 clears SBSW.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE01	SIM Reset Status Register (SRSR) See page 170.	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		POR:	1	0	0	0	0	0	0	0
$FE02	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE03	SIM Break Flag Control Register (SBFCR) See page 218.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE04	Interrupt Status Register 1 (INT1) See page 165.	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2) See page 166.	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented　R = Reserved　U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 6 of 7)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE06	Interrupt Status Register 3 (INT3) See page 166.	Read:	0	0	0	0	0	0	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE07	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE08	FLASH Control Register (FLCR) See page 37.	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$FE09	Break Address Register High (BRKH) See page 217.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Break Address Register Low (BRKL) See page 217.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0B	Break Status and Control Register (BRKSCR) See page 217.	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:			▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$FE0C	LVI Status Register (LVISR) See page 111.	Read:	LVIOUT	0	0	0	0	0	0	0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$FF7E	FLASH Block Protect Register (FLBPR)[1] See page 42.	Read:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
		Write:								
		Reset:	Unaffected by reset							

[1] Non-volatile FLASH register

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FFFF	COP Control Register (COPCTL) See page 79.	Read:	Low byte of reset vector							
		Write:	Writing clears COP counter (any value)							
		Reset:	Unaffected by reset							

▨ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 7 of 7)

Table 2-1. Vector Addresses

Vector Priority	Vector	Address	Vector
Lowest	IF16	$FFDC	Timebase Vector (High)
		$FFDD	Timebase Vector (Low)
	IF15	$FFDE	ADC Conversion Complete Vector (High)
		$FFDF	ADC Conversion Complete Vector (Low)
	IF14	$FFE0	Keyboard Vector (High)
		$FFE1	Keyboard Vector (Low)
	IF13	$FFE2	SCI Transmit Vector (High)
		$FFE3	SCI Transmit Vector (Low)
	IF12	$FFE4	SCI Receive Vector (High)
		$FFE5	SCI Receive Vector (Low)
	IF11	$FFE6	SCI Error Vector (High)
		$FFE7	SCI Error Vector (Low)
	IF10	$FFE8	SPI Transmit Vector (High)
		$FFE9	SPI Transmit Vector (Low)
	IF9	$FFEA	SPI Receive Vector (High)
		$FFEB	SPI Receive Vector (Low)
	IF8	$FFEC	TIM2 Overflow Vector (High)
		$FFED	TIM2 Overflow Vector (Low)
	IF7	$FFEE	Reserved
		$FFEF	Reserved
	IF6	$FFF0	TIM2 Channel 0 Vector (High)
		$FFF1	TIM2 Channel 0 Vector (Low)
	IF5	$FFF2	TIM1 Overflow Vector (High)
		$FFF3	TIM1 Overflow Vector (Low)
	IF4	$FFF4	TIM1 Channel 1 Vector (High)
		$FFF5	TIM1 Channel 1 Vector (Low)
	IF3	$FFF6	TIM1 Channel 0 Vector (High)
		$FFF7	TIM1 Channel 0 Vector (Low)
	IF2	$FFF8	PLL Vector (High)
		$FFF9	PLL Vector (Low)
	IF1	$FFFA	$\overline{\mathrm{IRQ}}$ Vector (High)
		$FFFB	$\overline{\mathrm{IRQ}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
	—	$FFFE	Reset Vector (High)
Highest		$FFFF	Reset Vector (Low)

2.5 Random-Access Memory (RAM)

Addresses $0040 through $01BF are RAM locations. The location of the stack RAM is programmable. The 16-bit stack pointer allows the stack to be anywhere in the 64-Kbyte memory space.

NOTE

For correct operation, the stack pointer must point only to RAM locations.

Within page zero are 192 bytes of RAM. Because the location of the stack RAM is programmable, all page zero RAM locations can be used for I/O control and user data or code. When the stack pointer is moved from its reset location at $00FF out of page zero, direct addressing mode instructions can efficiently access all page zero RAM locations. Page zero RAM, therefore, provides ideal locations for frequently accessed global variables.

Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers.

NOTE

For M6805 compatibility, the H register is not stacked.

During a subroutine call, the CPU uses two bytes of the stack to store the return address. The stack pointer decrements during pushes and increments during pulls.

NOTE

Be careful when using nested subroutines. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.

2.6 FLASH Memory (FLASH)

This subsection describes the operation of the embedded FLASH memory. This memory can be read, programmed, and erased from a single external supply. The program, erase, and read operations are enabled through the use of an internal charge pump.

2.6.1 Functional Description

The FLASH memory is an array of 7,680 bytes with an additional 36 bytes of user vectors and one byte of block protection. *An erased bit reads as 1 and a programmed bit reads as a 0*. Memory in the FLASH array is organized into two rows per page basis. The page size is 64 bytes per page and the row size is 32 bytes per row. Hence the minimum erase page size is 64 bytes and the minimum program row size is 32 bytes. Program and erase operation operations are facilitated through control bits in FLASH control register (FLCR). Details for these operations appear later in this section.

The address ranges for the user memory and vectors are:

- $E000–$FDFF; user memory
- $FE08; FLASH control register
- $FF7E; FLASH block protect register
- $FFDC–$FFFF; these locations are reserved for user-defined interrupt and reset vectors

NOTE

A security feature prevents viewing of the FLASH contents.(1)

---

1. No security feature is absolutely secure. However, Freescale’s strategy is to make reading or copying the FLASH difficult for unauthorized users.

2.6.2 FLASH Control Register

The FLASH control register (FLCR) controls FLASH program and erase operations.

Address: $FE08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 2-3. FLASH Control Register (FLCR)

HVEN — High-Voltage Enable Bit

This read/write bit enables the charge pump to drive high voltages for program and erase operations in the array. HVEN can only be set if either PGM = 1 or ERASE = 1 and the proper sequence for program or erase is followed.

1 = High voltage enabled to array and charge pump on
0 = High voltage disabled to array and charge pump off

MASS — Mass Erase Control Bit

Setting this read/write bit configures the FLASH array for mass erase operation.

1 = MASS erase operation selected
0 = PAGE erase operation selected

ERASE — Erase Control Bit

This read/write bit configures the memory for erase operation. ERASE is interlocked with the PGM bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Erase operation selected
0 = Erase operation unselected

PGM — Program Control Bit

This read/write bit configures the memory for program operation. PGM is interlocked with the ERASE bit such that both bits cannot be equal to 1 or set to 1 at the same time.

1 = Program operation selected
0 = Program operation unselected

2.6.3 FLASH Page Erase Operation

Use this step-by-step procedure to erase a page (64 bytes) of FLASH memory. A page consists of 64 consecutive bytes starting from addresses $XX00, $XX40, $XX80, or $XXC0. The 36-byte user interrupt vectors area also forms a page. Any FLASH memory page can be erased alone.

1. Set the ERASE bit, and clear the MASS bit in the FLASH control register.
2. Read the FLASH block protect register.
3. Write any data to any FLASH location within the page address range of the block to be erased.
4. Wait for a time, t_{NVS} (minimum 10 μs)
5. Set the HVEN bit.
6. Wait for a time, t_{Erase} (minimum 1 ms or 4 ms)
7. Clear the ERASE bit.
8. Wait for a time, t_{NVH} (minimum 5 μs)
9. Clear the HVEN bit.
10. After a time, t_{RCV} (typical 1 μs), the memory can be accessed in read mode again.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from FLASH memory. While these operations must be performed in the order shown, other unrelated operations may occur between the steps.

In applications that need more than 1000 program/erase cycles, use the 4-ms page erase specification to get improved long-term reliability. Any application can use this 4-ms page erase specification. However, in applications where a FLASH location will be erased and reprogrammed less than 1000 times, and speed is important, use the 1-ms page erase specification to get a shorter cycle time.

2.6.4 FLASH Mass Erase Operation

Use this step-by-step procedure to erase entire FLASH memory:

1. Set both the ERASE bit, and the MASS bit in the FLASH control register.
2. Read the FLASH block protect register.
3. Write any data to any FLASH address(1) within the FLASH memory address range.
4. Wait for a time, t_{NVS} (minimum 10 μs)
5. Set the HVEN bit.
6. Wait for a time, t_{MErase} (minimum 4 ms)
7. Clear the ERASE and MASS bits.

NOTE

Mass erase is disabled whenever any block is protected (FLBPR does not equal $FF).

8. Wait for a time, t_{NVHL} (minimum 100 μs)
9. Clear the HVEN bit.
10. After a time, t_{RCV} (typical 1 μs), the memory can be accessed in read mode again.

---

1. When in monitor mode, with security sequence failed (see 18.3.2 Security), write to the FLASH block protect register instead of any FLASH address.

NOTE

Programming and erasing of FLASH locations cannot be performed by code being executed from FLASH memory. While these operations must be performed in the order shown, other unrelated operations may occur between the steps.

2.6.5 FLASH Program/Read Operation

Programming of the FLASH memory is done on a row basis. A row consists of 32 consecutive bytes starting from addresses $XX00, $XX20, $XX40, $XX60, $XX80, $XXA0, $XXC0, and $XXE0.

During the programming cycle, make sure that all addresses being written to fit within one of the ranges specified above. Attempts to program addresses in different row ranges in one programming cycle will fail. Use this step-by-step procedure to program a row of FLASH memory (Figure 2-4 is a flowchart representation).

NOTE

Only bytes which are currently $FF may be programmed.

1. Set the PGM bit. This configures the memory for program operation and enables the latching of address and data for programming.
2. Read the FLASH block protect register.
3. Write any data to any FLASH address within the row address range desired.
4. Wait for a time, t_{NVS} (minimum 10 µs).
5. Set the HVEN bit.
6. Wait for a time, t_{PGS} (minimum 5 µs).
7. Write data to the FLASH address to be programmed.
8. Wait for a time, t_{PROG} (minimum 30 µs).
9. Repeat step 7 and 8 until all the bytes within the row are programmed.
10. Clear the PGM bit.(1)
11. Wait for a time, t_{NVH} (minimum 5 µs).
12. Clear the HVEN bit.
13. After time, t_{RCV} (typical 1 µs), the memory can be accessed in read mode again.

This program sequence is repeated throughout the memory until all data is programmed.

NOTE

Programming and erasing of FLASH locations can not be performed by code being executed from the same FLASH array.

NOTE

While these operations must be performed in the order shown, other unrelated operations may occur between the steps. Care must be taken within the FLASH array memory space such as the COP control register (COPCTL) at $FFFF.

NOTE

It is highly recommended that interrupts be disabled during program/ erase operations.

NOTE

Do not exceed t_{PROG} maximum or t_{HV} maximum. t_{HV} is defined as the cumulative high voltage programming time to the same row before next erase. t_{HV} must satisfy this condition:

$$t_{NVS} + t_{NVH} + t_{PGS} + (t_{PROG} \times 32) \leq t_{HV} \text{ maximum}$$

Refer to 19.17 Memory Characteristics.

NOTE

The time between programming the FLASH address change (step 7 to step 7), or the time between the last FLASH programmed to clearing the PGM bit (step 7 to step 10) must not exceed the maximum programming time, t_{PROG} maximum.

CAUTION

Be cautious when programming the FLASH array to ensure that non-FLASH locations are not used as the address that is written to when selecting either the desired row address range in step 3 of the algorithm or the byte to be programmed in step 7 of the algorithm. This applies particularly to $FFD4–$FFDF.

2.6.6 FLASH Block Protection

Due to the ability of the on-board charge pump to erase and program the FLASH memory in the target application, provision is made for protecting a block of memory from unintentional erase or program operations due to system malfunction. This protection is done by using of a FLASH block protect register (FLBPR). The FLBPR determines the range of the FLASH memory which is to be protected. The range of the protected area starts from a location defined by FLBPR and ends at the bottom of the FLASH memory ($FFFF). When the memory is protected, the HVEN bit cannot be set in either ERASE or PROGRAM operations.

NOTE

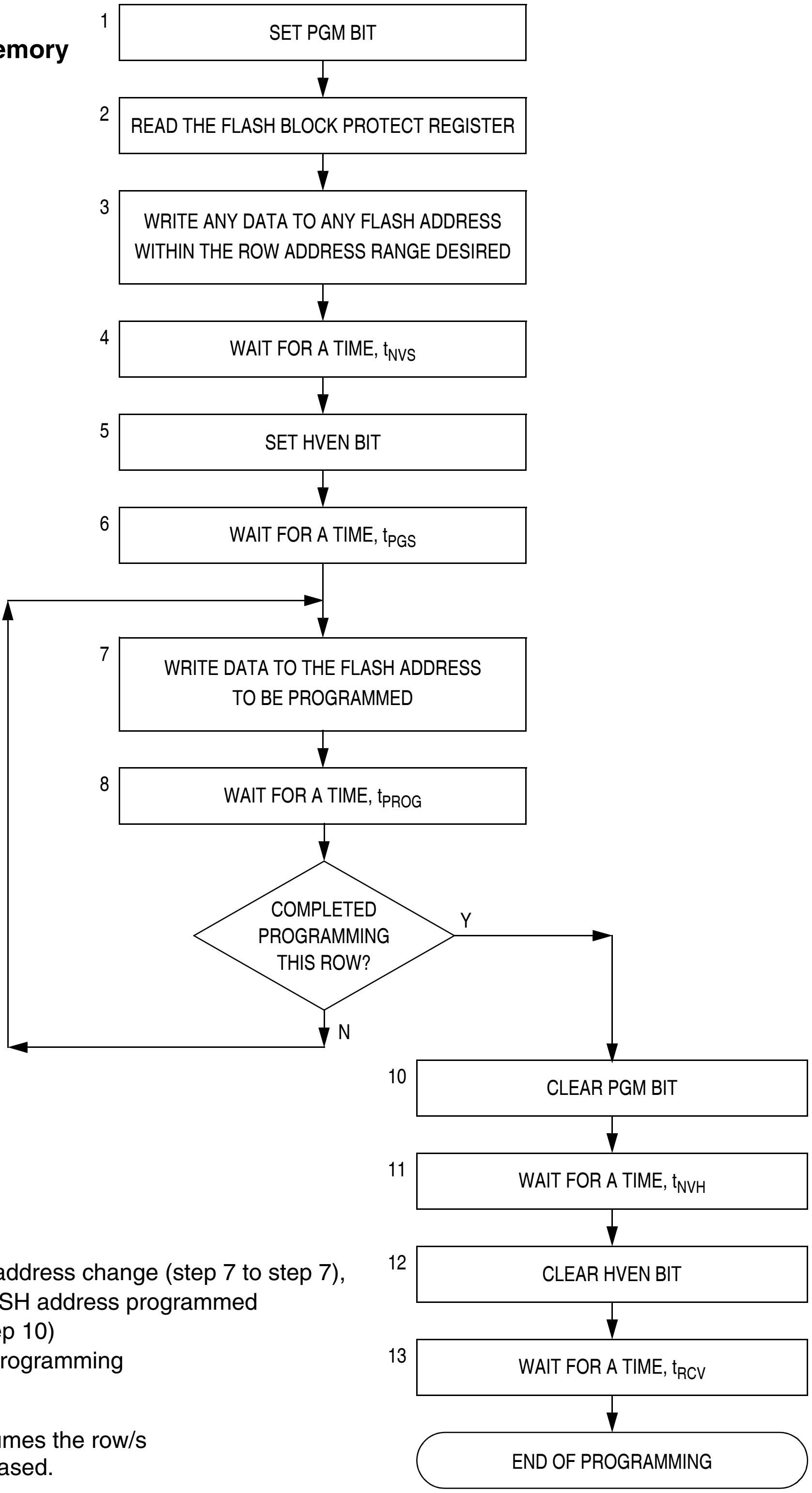
In performing a program or erase operation, the FLASH block protect register must be read after setting the PGM or ERASE bit and before asserting the HVEN bit

When the FLBPR is program with all 0's, the entire memory is protected from being programmed and erased. When all the bits are erased (all 1's), the entire memory is accessible for program and erase.

When bits within the FLBPR are programmed, they lock a block of memory, address ranges as shown in 2.6.7 FLASH Block Protect Register. Once the FLBPR is programmed with a value other than $FF or $FE, any erase or program of the FLBPR or the protected block of FLASH memory is prohibited. Mass erase is disabled whenever any block is protected (FLBPR does not equal $FF). The presence of a V_{TST} on the $\overline{\text{IRQ}}$ pin will bypass the block protection so that all of the memory included in the block protect register is open for program and erase operations.

NOTE

The FLASH block protect register is not protected with special hardware or software. Therefore, if this page is not protected by FLBPR the register is erased by either a page or mass erase operation.



Note:

The time between each FLASH address change (step 7 to step 7), or the time between the last FLASH address programmed to clearing PGM bit (step 7 to step 10) must not exceed the maximum programming time, t_{PROG} max.

This row program algorithm assumes the row/s to be programmed are initially erased.

Figure 2-4. FLASH Programming Flowchart

2.6.7 FLASH Block Protect Register

The FLASH block protect register (FLBPR) is implemented as a byte within the FLASH memory, and therefore can only be written during a programming sequence of the FLASH memory. The value in this register determines the starting location of the protected range within the FLASH memory.

Address: $FF7E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
Reset:	Unaffected by reset. Initial value from factory is 1.							

Write to this register is by a programming sequence to the FLASH memory.

Figure 2-5. FLASH Block Protect Register (FLBPR)

BPR[7:0] — FLASH Block Protect Bits

These eight bits represent bits [13:6] of a 16-bit memory address.
Bit 15 and Bit 14 are 1s and bits [5:0] are 0s.

The resultant 16-bit address is used for specifying the start address of the FLASH memory for block protection. The FLASH is protected from this start address to the end of FLASH memory, at $FFFF. With this mechanism, the protect start address can be $XX00, $XX40, $XX80, and $XXC0 (64 bytes page boundaries) within the FLASH memory.

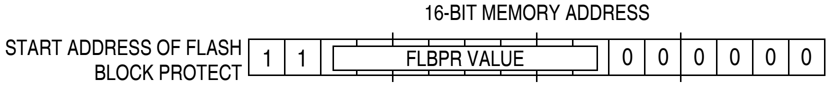


Figure 2-6. FLASH Block Protect Start Address

Table 2-2. Examples of Protect Address Ranges

BPR[7:0]	Addresses of Protect Range
$00	The entire FLASH memory is protected.
$81 (**1000 0001**)	$E040 (11**10 0000 01**00 0000) — $FFFF
$82 (**1000 0010**)	$E080 (11**10 0000 10**00 0000) — $FFFF
and so on...	
$FC (**1111 1100**)	$FF00 (11**11 1111 00**00 0000) — FFFF
$FD (**1111 1101**)	$FF40 (11**11 1111 01**00 0000) — $FFFF FLBPR and vectors are protected
$FE (**1111 1110**)	$FF80 (11**11 1111 10**00 0000) — FFFF Vectors are protected
$FF	The entire FLASH memory is not protected.

2.6.8 Wait Mode

Putting the MCU into wait mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly, but there will not be any memory activity since the CPU is inactive.

The WAIT instruction should not be executed while performing a program or erase operation on the FLASH, otherwise the operation will discontinue, and the FLASH will be on standby mode.

2.6.9 Stop Mode

Putting the MCU into stop mode while the FLASH is in read mode does not affect the operation of the FLASH memory directly, but there will not be any memory activity since the CPU is inactive.

The STOP instruction should not be executed while performing a program or erase operation on the FLASH, otherwise the operation will discontinue, and the FLASH will be on standby mode

NOTE

Standby mode is the power saving mode of the FLASH module in which all internal control signals to the FLASH are inactive and the current consumption of the FLASH is at a minimum.

Chapter 3
Analog-to-Digital Converter (ADC)

3.1 Introduction

This section describes the 8-bit analog-to-digital converter (ADC).

3.2 Features

Features of the ADC module include:

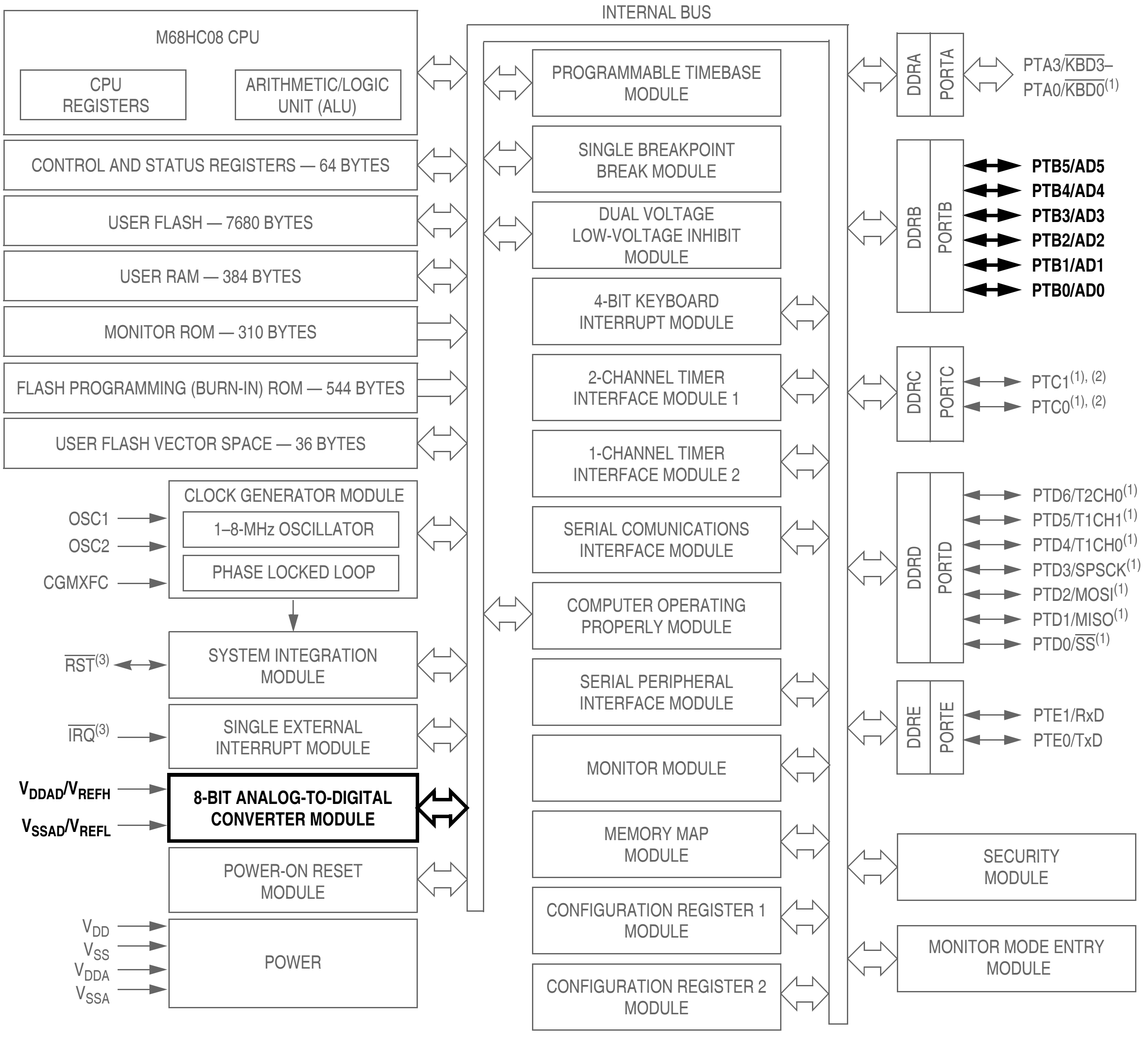
- Six channels with multiplexed input
- Linear successive approximation with monotonicity
- 8-bit resolution
- Single or continuous conversion
- Conversion complete flag or conversion complete interrupt
- Selectable ADC clock

3.3 Functional Description

The ADC provides six pins for sampling external sources at pins PTB5/AD5–PTB0/KBD0. An analog multiplexer allows the single ADC converter to select one of six ADC channels as ADC voltage in (V_{ADIN}). V_{ADIN} is converted by the successive approximation register-based analog-to-digital converter. When the conversion is completed, ADC places the result in the ADC data register and sets a flag or generates an interrupt. See Figure 3-2.

3.3.1 ADC Port I/O Pins

PTB5/AD5–PTB0/KBD0 are general-purpose I/O (input/output) pins that share with the ADC channels. The channel select bits define which ADC channel/port pin will be used as the input signal. The ADC overrides the port I/O logic by forcing that pin as input to the ADC. The remaining ADC channels/port pins are controlled by the port I/O logic and can be used as general-purpose I/O. Writes to the port register or data direction register (DDR) will not have any affect on the port pin that is selected by the ADC. Read of a port pin in use by the ADC will return a 0.



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 3-1. Block Diagram Highlighting ADC Block and Pins

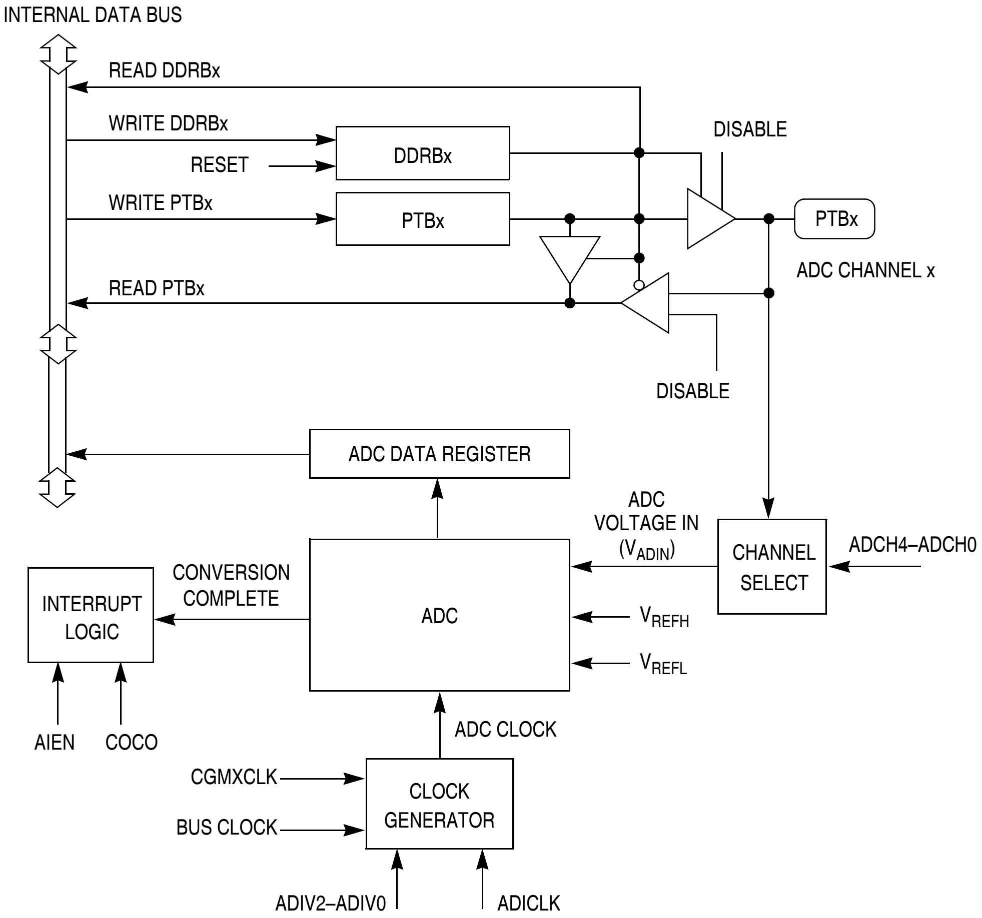


Figure 3-2. ADC Block Diagram

3.3.2 Voltage Conversion

When the input voltage to the ADC equals V_{REFH}, the ADC converts the signal to \$FF (full scale). If the input voltage equals V_{REFL}, the ADC converts it to \$00. Input voltages between V_{REFH} and V_{REFL} are a straight-line linear conversion.

NOTE

The ADC input voltage must always be greater than V_{SSA} and less than V_{DDA}. V_{REFH} must always be greater than or equal to V_{REFL}.

NOTE

Connect the V_{DDA} pin to the same voltage potential as the V_{DD} pin, and connect the V_{SSA} pin to the same voltage potential as the V_{SS} pin. The V_{DDA} pin should be routed carefully for maximum noise immunity.

3.3.3 Conversion Time

Conversion starts after a write to the ADC status and control register (ADSCR). One conversion will take between 16 and 17 ADC clock cycles. The ADIVx and ADICLK bits should be set to provide a 1-MHz ADC clock frequency.

$$\text{Conversion time} = \frac{\text{16 to 17 ADC cycles}}{\text{ADC frequency}}$$

$$\text{Number of bus cycles} = \text{conversion time} \times \text{bus frequency}$$

3.3.4 Conversion

In continuous conversion mode, the ADC data register will be filled with new data after each conversion. Data from the previous conversion will be overwritten whether that data has been read or not. Conversions will continue until the ADCO bit is cleared. The COCO bit is set after the first conversion and will stay set until the next read of the ADC data register.

In single conversion mode, conversion begins with a write to the ADSCR. Only one conversion occurs between writes to the ADSCR.

When a conversion is in process and the ADSCR is written, the current conversion data should be discarded to prevent an incorrect reading.

3.3.5 Accuracy and Precision

The conversion process is monotonic and has no missing codes.

3.4 Interrupts

When the AIEN bit is set, the ADC module is capable of generating CPU interrupts after each ADC conversion. A CPU interrupt is generated if the COCO bit is at 0. The COCO bit is not used as a conversion complete flag when interrupts are enabled.

3.5 Low-Power Modes

The WAIT and STOP instruction can put the MCU in low power-consumption standby modes.

3.5.1 Wait Mode

The ADC continues normal operation during wait mode. Any enabled CPU interrupt request from the ADC can bring the MCU out of wait mode. If the ADC is not required to bring the MCU out of wait mode, power down the ADC by setting ADCH4–ADCH0 bits in the ADC status and control register before executing the WAIT instruction.

3.5.2 Stop Mode

The ADC module is inactive after the execution of a STOP instruction. Any pending conversion is aborted. ADC conversions resume when the MCU exits stop mode after an external interrupt. Allow one conversion cycle to stabilize the analog circuitry.

3.6 I/O Signals

The ADC module has six pins shared with port B, PTB5/AD5–PTB0/KBD0.

3.6.1 ADC Analog Ground Pin (V_{SSAD})/ADC Voltage Reference Low Pin (V_{REFL})

The ADC analog portion uses V_{SSAD} as its ground pin and V_{REFL} as its lower voltage reference pin. Connect the V_{SSAD}/V_{REFL} pin to the same voltage potential as V_{SS}.

NOTE

For maximum noise immunity, route V_{SSAD}/V_{REFL} *carefully and place bypass capacitors as close as possible to the package. Routing* V_{REFH} *close and parallel to* V_{REFL} *may improve common mode noise rejection.*

3.6.2 ADC Analog Power Pin (V_{DDAD})/ADC Voltage Reference High Pin (V_{REFH})

The ADC analog portion uses V_{DDAD} as its power pin and V_{REFH} as its upper voltage reference pin. Connect the V_{DDAD}/ V_{REFH} pin to the same voltage potential as V_{DD}.

NOTE

For maximum noise immunity, route V_{DDAD}/V_{REFH} *carefully and place bypass capacitors as close as possible to the package. Routing* V_{REFH} *close and parallel to* V_{REFL} *may improve common mode noise rejection.*

3.6.3 ADC Voltage In (V_{ADIN})

V_{ADIN} is the input voltage signal from one of the six ADC channels to the ADC module.

3.7 I/O Registers

These I/O registers control and monitor ADC operation:

- ADC status and control register (ADSCR)
- ADC data register (ADR)
- ADC clock register (ADCLK)

3.7.1 ADC Status and Control Register

Function of the ADC status and control register (ADSCR) is described here.

Address: $003C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
Write:	R							
Reset:	0	0	0	1	1	1	1	1

R = Reserved

Figure 3-3. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete Bit

In non-interrupt mode (AIEN = 0), COCO is a read-only bit that is set at the end of each conversion. COCO will stay set until cleared by a read of the ADC data register. Reset clears this bit.

In interrupt mode (AIEN = 1), COCO is a read-only bit that is not set at the end of a conversion. It always reads as a 0.

1 = Conversion completed (AIEN = 0)
0 = Conversion not completed (AIEN = 0) or CPU interrupt enabled (AIEN = 1)

NOTE

The write function of the COCO bit is reserved. When writing to the ADSCR register, always have a 0 in the COCO bit position.

AIEN — ADC Interrupt Enable Bit

When this bit is set, an interrupt is generated at the end of an ADC conversion. The interrupt signal is cleared when the data register is read or the status/control register is written. Reset clears the AIEN bit.

1 = ADC interrupt enabled
0 = ADC interrupt disabled

ADCO — ADC Continuous Conversion Bit

When set, the ADC will convert samples continuously and update the ADR register at the end of each conversion. Only one conversion is completed between writes to the ADSCR when this bit is cleared. Reset clears the ADCO bit.

1 = Continuous ADC conversion
0 = One ADC conversion

ADCH4–ADCH0 — ADC Channel Select Bits

ADCH4–ADCH0 form a 5-bit field which is used to select one of 16 ADC channels. Only six channels, AD5–AD0, are available on this MCU. The channels are detailed in Table 3-1. Care should be taken when using a port pin as both an analog and digital input simultaneously to prevent switching noise from corrupting the analog signal. See Table 3-1.

The ADC subsystem is turned off when the channel select bits are all set to 1. This feature allows for reduced power consumption for the MCU when the ADC is not being used.

NOTE

Recovery from the disabled state requires one conversion cycle to stabilize.

The voltage levels supplied from internal reference nodes, as specified in Table 3-1, are used to verify the operation of the ADC converter both in production test and for user applications.

Table 3-1. Mux Channel Select(1)

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	Input Select
0	0	0	0	0	PTB0/KBD0
0	0	0	0	1	PTB1/AD1
0	0	0	1	0	PTB1/AD2
0	0	0	1	1	PTB2/AD3
0	0	1	0	0	PTB4/AD4
0	0	1	0	1	PTB5/AD5
0 ↓ 1	0 ↓ 1	1 ↓ 1	1 ↓ 0	0 ↓ 0	Reserved
1	1	1	0	1	V_{REFH}
1	1	1	1	0	V_{REFL}
1	1	1	1	1	ADC power off

1. If any unused channels are selected, the resulting ADC conversion will be unknown or reserved.

3.7.2 ADC Data Register

One 8-bit result register, ADC data register (ADR), is provided. This register is updated each time an ADC conversion completes.

Address: $003D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-4. ADC Data Register (ADR)

3.7.3 ADC Clock Register

The ADC clock register (ADCLK) selects the clock frequency for the ADC.

Address: $003E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ADIV2	ADIV1	ADIV0	ADICLK	0	0	0	0
Write:					▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-5. ADC Clock Register (ADCLK)

ADIV2–ADIV0 — ADC Clock Prescaler Bits

ADIV2–ADIV0 form a 3-bit field which selects the divide ratio used by the ADC to generate the internal ADC clock. Table 3-2 shows the available clock configurations. The ADC clock should be set to approximately 1 MHz.

Table 3-2. ADC Clock Divide Ratio

ADIV2	ADIV1	ADIV0	ADC Clock Rate
0	0	0	ADC input clock ÷ 1
0	0	1	ADC input clock ÷ 2
0	1	0	ADC input clock ÷ 4
0	1	1	ADC input clock ÷ 8
1	X(1)	X(1)	ADC input clock ÷ 16

1. X = Don't care

ADICLK — ADC Input Clock Select Bit

ADICLK selects either the bus clock or the oscillator output clock (CGMXCLK) as the input clock source to generate the internal ADC clock. Reset selects CGMXCLK as the ADC clock source.

1 = Internal bus clock
0 = Oscillator output clock (CGMXCLK)

The ADC requires a clock rate of approximately 1 MHz for correct operation. If the selected clock source is not fast enough, the ADC will generate incorrect conversions. See 19.12 ADC Characteristics.

$$f_{ADIC} = \frac{f_{CGMXCLK} \text{ or bus frequency}}{\text{ADIV[2:0]}} \cong 1 \text{ MHz}$$

Chapter 4
Clock Generator Module (CGM)

4.1 Introduction

This section describes the clock generator module. The CGM generates the crystal clock signal, CGMXCLK, which operates at the frequency of the crystal. The CGM also generates the base clock signal, CGMOUT, which is based on either the crystal clock divided by two or the phase-locked loop (PLL) clock, CGMVCLK, divided by two. In user mode, CGMOUT is the clock from which the SIM derives the system clocks, including the bus clock, which is at a frequency of CGMOUT/2. The PLL is a fully functional frequency generator designed for use with crystals or ceramic resonators. The PLL can generate an 8-MHz bus frequency using a 1–8 MHz crystal or external clock source.

4.2 Features

Features of the CGM include:

- Phase-locked loop with output frequency in integer multiples of an integer dividend of the crystal reference
- High-frequency crystal operation with low-power operation and high-output frequency resolution
- Programmable hardware voltage-controlled oscillator (VCO) for low-jitter operation
- Automatic bandwidth control mode for low-jitter operation
- Automatic frequency lock detector
- CPU interrupt on entry or exit from locked condition
- Configuration register bit to allow oscillator operation during stop mode

4.3 Functional Description

The CGM consists of three major submodules:

- Crystal oscillator circuit — The crystal oscillator circuit generates the constant crystal frequency clock, CGMXCLK.
- Phase-locked loop (PLL) — The PLL generates the programmable VCO frequency clock, CGMVCLK.
- Base clock selector circuit — This software-controlled circuit selects either CGMXCLK divided by two or the VCO clock, CGMVCLK, divided by two as the base clock, CGMOUT. The SIM derives the system clocks from either CGMOUT or CGMXCLK.

Figure 4-1 shows the structure of the CGM.

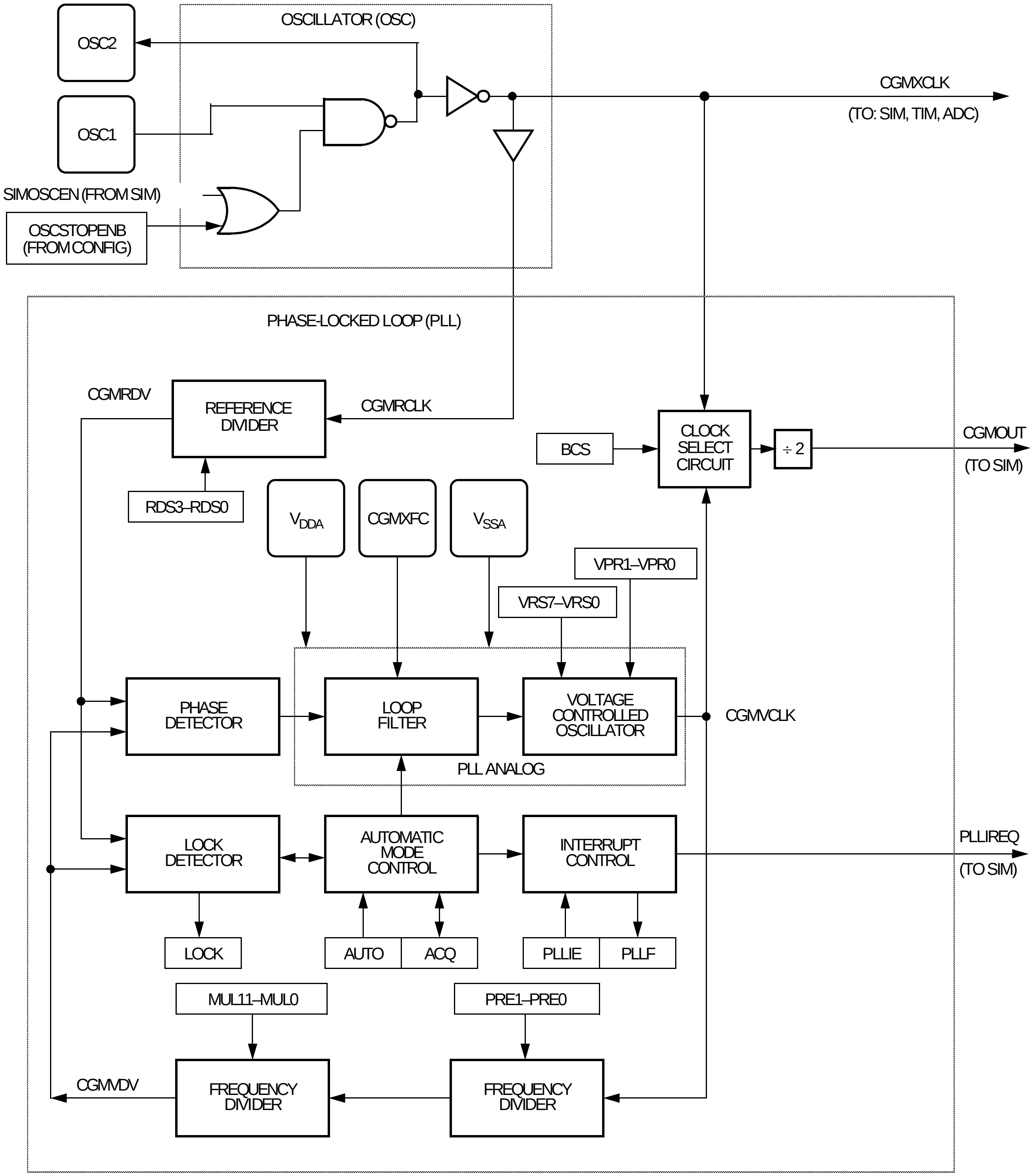


Figure 4-1. CGM Block Diagram

4.3.1 Crystal Oscillator Circuit

The crystal oscillator circuit consists of an inverting amplifier and an external crystal. The OSC1 pin is the input to the amplifier and the OSC2 pin is the output. The SIMOSCEN signal from the system integration module (SIM) or the OSCSTOPENB bit in the CONFIG register enable the crystal oscillator circuit.

The CGMXCLK signal is the output of the crystal oscillator circuit and runs at a rate equal to the crystal frequency. CGMXCLK is then buffered to produce CGMRCLK, the PLL reference clock.

CGMXCLK can be used by other modules which require precise timing for operation. The duty cycle of CGMXCLK is not guaranteed to be 50% and depends on external factors, including the crystal and related external components. An externally generated clock also can feed the OSC1 pin of the crystal oscillator circuit. Connect the external clock to the OSC1 pin and let the OSC2 pin float.

4.3.2 Phase-Locked Loop Circuit (PLL)

The PLL is a frequency generator that can operate in either acquisition mode or tracking mode, depending on the accuracy of the output frequency. The PLL can change between acquisition and tracking modes either automatically or manually.

4.3.3 PLL Circuits

The PLL consists of these circuits:

- Voltage-controlled oscillator (VCO)
- Modulo VCO frequency divider
- Phase detector
- Loop filter
- Lock detector

The operating range of the VCO is programmable for a wide range of frequencies and for maximum immunity to external noise, including supply and CGM/XFC noise. The VCO frequency is bound to a range from roughly one-half to twice the center-of-range frequency, f_{VRS}. Modulating the voltage on the CGM/XFC pin changes the frequency within this range. By design, f_{VRS} is equal to the nominal center-of-range frequency, f_{NOM}, (38.4 kHz) times a linear factor, L, and a power-of-two factor, E, or $(L \times 2^E)f_{NOM}$.

CGMRCLK is the PLL reference clock, a buffered version of CGMXCLK. CGMRCLK runs at a frequency, f_{RCLK}. The VCO's output clock, CGMVCLK, running at a frequency, f_{VCLK}, is fed back through a programmable prescale divider and a programmable modulo divider. The prescaler divides the VCO clock by a power-of-two factor P and the modulo divider reduces the VCO clock by a factor, N. The dividers' output is the VCO feedback clock, CGMVDV, running at a frequency, $f_{VDV} = f_{VCLK}/(N \times 2^P)$. (See 4.3.6 Programming the PLL for more information.)

The phase detector then compares the VCO feedback clock, CGMVDV, with the final reference clock, CGMRDV. A correction pulse is generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the external capacitor connected to CGM/XFC based on the width and direction of the correction pulse. The filter can make fast or slow corrections depending on its mode, described in 4.3.4 Acquisition and Tracking Modes. The value of the external capacitor and the reference frequency determine the speed of the corrections and the stability of the PLL.

The lock detector compares the frequencies of the VCO feedback clock, CGMVDV, and the reference clock, CGMRCLK. Therefore, the speed of the lock detector is directly proportional to the reference frequency, f_{RCLK}. The circuit determines the mode of the PLL and the lock condition based on this comparison.

4.3.4 Acquisition and Tracking Modes

The PLL filter is manually or automatically configurable into one of two operating modes:

- Acquisition mode — In acquisition mode, the filter can make large frequency corrections to the VCO. This mode is used at PLL startup or when the PLL has suffered a severe noise hit and the VCO frequency is far off the desired frequency. When in acquisition mode, the $\overline{\text{ACQ}}$ bit is clear in the PLL bandwidth control register. (See 4.5.2 PLL Bandwidth Control Register.)
- Tracking mode — In tracking mode, the filter makes only small corrections to the frequency of the VCO. PLL jitter is much lower in tracking mode, but the response to noise is also slower. The PLL enters tracking mode when the VCO frequency is nearly correct, such as when the PLL is selected as the base clock source. (See 4.3.8 Base Clock Selector Circuit.) The PLL is automatically in tracking mode when not in acquisition mode or when the $\overline{\text{ACQ}}$ bit is set.

4.3.5 Manual and Automatic PLL Bandwidth Modes

The PLL can change the bandwidth or operational mode of the loop filter manually or automatically. Automatic mode is recommended for most users.

In automatic bandwidth control mode (AUTO = 1), the lock detector automatically switches between acquisition and tracking modes. Automatic bandwidth control mode also is used to determine when the VCO clock, CGMVCLK, is safe to use as the source for the base clock, CGMOUT. (See 4.5.2 PLL Bandwidth Control Register.) If PLL interrupts are enabled, the software can wait for a PLL interrupt request and then check the LOCK bit. If interrupts are disabled, software can poll the LOCK bit continuously (during PLL startup, usually) or at periodic intervals. In either case, when the LOCK bit is set, the VCO clock is safe to use as the source for the base clock. (See 4.3.8 Base Clock Selector Circuit.) If the VCO is selected as the source for the base clock and the LOCK bit is clear, the PLL has suffered a severe noise hit and the software must take appropriate action, depending on the application. (See 4.6 Interrupts for information and precautions on using interrupts.)

The following conditions apply when the PLL is in automatic bandwidth control mode:

- The $\overline{\text{ACQ}}$ bit (see 4.5.2 PLL Bandwidth Control Register) is a read-only indicator of the mode of the filter. (See 4.3.4 Acquisition and Tracking Modes.)
- The $\overline{\text{ACQ}}$ bit is set when the VCO frequency is within a certain tolerance and is cleared when the VCO frequency is out of a certain tolerance. (See 4.8 Acquisition/Lock Time Specifications for more information.)
- The LOCK bit is a read-only indicator of the locked state of the PLL.
- The LOCK bit is set when the VCO frequency is within a certain tolerance and is cleared when the VCO frequency is out of a certain tolerance. (See 4.8 Acquisition/Lock Time Specifications for more information.)
- CPU interrupts can occur if enabled (PLLIE = 1) when the PLL's lock condition changes, toggling the LOCK bit. (See 4.5.1 PLL Control Register.)

The PLL also may operate in manual mode (AUTO = 0). Manual mode is used by systems that do not require an indicator of the lock condition for proper operation. Such systems typically operate well below f_{BUSMAX}.

The following conditions apply when in manual mode:

- $\overline{\text{ACQ}}$ is a writable control bit that controls the mode of the filter. Before turning on the PLL in manual mode, the $\overline{\text{ACQ}}$ bit must be clear.
- Before entering tracking mode ($\overline{\text{ACQ}}$ = 1), software must wait a given time, t_{ACQ} (see 4.8 Acquisition/Lock Time Specifications), after turning on the PLL by setting PLLON in the PLL control register (PCTL).
- Software must wait a given time, t_{AL}, after entering tracking mode before selecting the PLL as the clock source to CGMOUT (BCS = 1).
- The LOCK bit is disabled.
- CPU interrupts from the CGM are disabled.

4.3.6 Programming the PLL

The following procedure shows how to program the PLL.

NOTE

The round function in the following equations means that the real number should be rounded to the nearest integer number.

1. Choose the desired bus frequency, f_{BUSDES}.
2. Calculate the desired VCO frequency (four times the desired bus frequency).

$$f_{VCLKDES} = 4 \times f_{BUSDES}$$

3. Choose a practical PLL (crystal) reference frequency, f_{RCLK}, and the reference divider, R. Typically, the reference crystal is 4 MHz and R = 1.

   Frequency errors to the PLL are corrected at a rate of f_{RCLK}/R. For stability and lock time reduction, this rate must be as fast as possible. The VCO frequency must be an integer multiple of this rate. The relationship between the VCO frequency, f_{VCLK}, and the reference frequency, f_{RCLK}, is

$$f_{VCLK} = \frac{2^P N}{R}(f_{RCLK})$$

   P, the power of two multiplier, and N, the range multiplier, are integers.

   In cases where desired bus frequency has some tolerance, choose f_{RCLK} to a value determined either by other module requirements (such as modules which are clocked by CGMXCLK), cost requirements, or ideally, as high as the specified range allows. See Chapter 19 Electrical Specifications. Choose the reference divider, R = 1. After choosing N and P, the actual bus frequency can be determined using equation in 2 above.

   When the tolerance on the bus frequency is tight, choose f_{RCLK} to an integer divisor of f_{BUSDES}, and R = 1. If f_{RCLK} cannot meet this requirement, use the following equation to solve for R with practical choices of f_{RCLK}, and choose the f_{RCLK} that gives the lowest R.

$$R = \text{round}\left[R_{MAX} \times \left\{\left(\frac{f_{VCLKDES}}{f_{RCLK}}\right) - \text{integer}\left(\frac{f_{VCLKDES}}{f_{RCLK}}\right)\right\}\right]$$

4. Select a VCO frequency multiplier, N.

$$N = \mathrm{round}\left(\frac{R \times f_{VCLKDES}}{f_{RCLK}}\right)$$

5. If N is < N_{max}, use P = 0. If N > N_{max}, choose P using this table:

Current N Value	P
$0 < N \le N_{max}$	0
$N_{max} < N \le N_{max} \times 2$	1
$N_{max} \times 2 < N \le N_{max} \times 4$	2
$N_{max} \times 4 < N \le N_{max} \times 8$	3

Then recalculate N:

$$N = \mathrm{round}\left(\frac{R \times f_{VCLKDES}}{f_{RCLK} \times 2^P}\right)$$

6. Calculate and verify the adequacy of the VCO and bus frequencies f_{VCLK} and f_{BUS}.

$$f_{VCLK} = (2^P \times N/R) \times f_{RCLK}$$

$$f_{BUS} = (f_{VCLK})/4$$

7. Select the VCO's power-of-two range multiplier E, according to this table:

Frequency Range	E
$0 < f_{VCLK} < 8MHz$	0
$8MHz \le f_{VCLK} < 16MHz$	1
$16MHz \le f_{VCLK} < 32MHz$	2

NOTE: Do not program E to a value of 3.

8. Select a VCO linear range multiplier, L, where f_{NOM} = 38.4 kHz

$$L = \mathrm{round}\left(\frac{f_{VCLK}}{2^E \times f_{NOM}}\right)$$

9. Calculate and verify the adequacy of the VCO programmed center-of-range frequency, f_{VRS}. The center-of-range frequency is the midpoint between the minimum and maximum frequencies attainable by the PLL.

$$f_{VRS} = (L \times 2^E) f_{NOM}$$

For proper operation,

$$|f_{VRS} - f_{VCLK}| \le \frac{f_{NOM} \times 2^E}{2}$$

10. Verify the choice of P, R, N, E, and L by comparing f_{VCLK} to f_{VRS} and $f_{VCLKDES}$. For proper operation, f_{VCLK} must be within the application's tolerance of $f_{VCLKDES}$, and f_{VRS} must be as close as possible to f_{VCLK}.

NOTE

Exceeding the recommended maximum bus frequency or VCO frequency can crash the MCU.

11. Program the PLL registers accordingly:
    a. In the PRE bits of the PLL control register (PCTL), program the binary equivalent of P.
    b. In the VPR bits of the PLL control register (PCTL), program the binary equivalent of E.
    c. In the PLL multiplier select register low (PMSL) and the PLL multiplier select register high (PMSH), program the binary equivalent of N.
    d. In the PLL VCO range select register (PMRS), program the binary coded equivalent of L.
    e. In the PLL reference divider select register (PMDS), program the binary coded equivalent of R.

Table 4-1 provides numeric examples (numbers are in hexadecimal notation):

Table 4-1. Numeric Example

f_{BUS} (MHz)	f_{RCLK} (MHz)	PCTL		PMSH,L	PMRS	PMDS
		P	E	N	L	R
2.0	8.0	0	0	008	D0	8
4.0	8.0	0	1	010	D0	8
8.0	8.0	0	2	020	D0	8
2.0	4.0	0	0	002	D0	1
4.0	4.0	0	1	004	D0	1
8.0	4.0	0	2	008	D0	1
2.4576	4.9152	0	1	002	80	1
4.9152	4.9152	0	2	004	80	1
7.3728	4.9152	0	2	006	C0	1
2.0	8.0	0	0	001	D0	1
4.0	8.0	0	1	002	D0	1
8.0	8.0	0	2	004	D0	1

4.3.7 Special Programming Exceptions

The programming method described in 4.3.6 Programming the PLL does not account for three possible exceptions. A value of 0 for R, N, or L is meaningless when used in the equations given. To account for these exceptions:

- A 0 value for R or N is interpreted exactly the same as a value of 1.
- A 0 value for L disables the PLL and prevents its selection as the source for the base clock.

(See 4.3.8 Base Clock Selector Circuit.)

4.3.8 Base Clock Selector Circuit

This circuit is used to select either the crystal clock, CGMXCLK, or the VCO clock, CGMVCLK, as the source of the base clock, CGMOUT. The two input clocks go through a transition control circuit that waits up to three CGMXCLK cycles and three CGMVCLK cycles to change from one clock source to the other. During this time, CGMOUT is held in stasis. The output of the transition control circuit is then divided by two to correct the duty cycle. Therefore, the bus clock frequency, which is one-half of the base clock frequency, is one-fourth the frequency of the selected clock (CGMXCLK or CGMVCLK).

The BCS bit in the PLL control register (PCTL) selects which clock drives CGMOUT. The VCO clock cannot be selected as the base clock source if the PLL is not turned on. The PLL cannot be turned off if the VCO clock is selected. The PLL cannot be turned on or off simultaneously with the selection or deselection of the VCO clock. The VCO clock also cannot be selected as the base clock source if the factor L is programmed to a 0. This value would set up a condition inconsistent with the operation of the PLL, so that the PLL would be disabled and the crystal clock would be forced as the source of the base clock.

4.3.9 CGM External Connections

In its typical configuration, the CGM requires up to nine external components. Five of these are for the crystal oscillator and two or four are for the PLL.

The crystal oscillator is normally connected in a Pierce oscillator configuration, as shown in Figure 4-2. Figure 4-2 shows only the logical representation of the internal components and may not represent actual circuitry. The oscillator configuration uses five components:

- Crystal, X_1
- Fixed capacitor, C_1
- Tuning capacitor, C_2 (can also be a fixed capacitor)
- Feedback resistor, R_B
- Series resistor, R_S

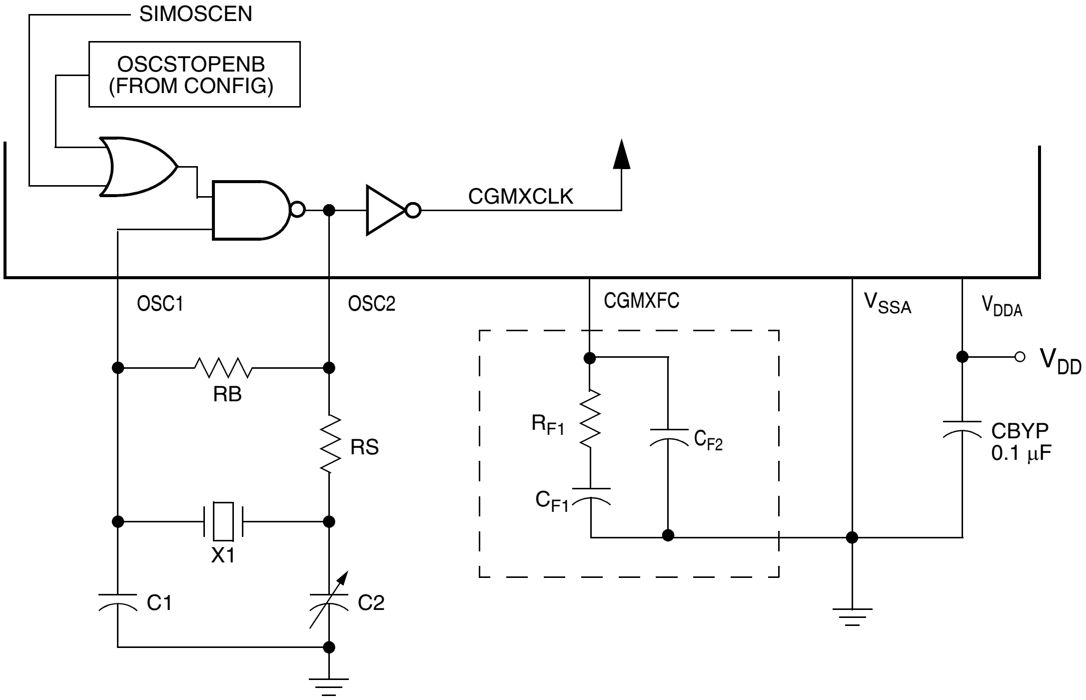
The series resistor (R_S) is included in the diagram to follow strict Pierce oscillator guidelines. Refer to the crystal manufacturer's data for more information regarding values for C1 and C2.

Figure 4-2 also shows the external components for the PLL:

- Bypass capacitor, C_{BYP}
- Filter network

Routing should be done with great care to minimize signal cross talk and noise.

See 19.15 Clock Generation Module Characteristics for capacitor and resistor values.



Note: Filter network in box can be replaced with C_F but will degrade stability.

Figure 4-2. CGM External Connections

4.4 I/O Signals

The following paragraphs describe the CGM I/O signals.

4.4.1 Crystal Amplifier Input Pin (OSC1)

The OSC1 pin is an input to the crystal oscillator amplifier.

4.4.2 Crystal Amplifier Output Pin (OSC2)

The OSC2 pin is the output of the crystal oscillator inverting amplifier.

4.4.3 External Filter Capacitor Pin (CGMXFC)

The CGMXFC pin is required by the loop filter to filter out phase corrections. An external filter network is connected to this pin. (See Figure 4-2.)

NOTE

To prevent noise problems, the filter network should be placed as close to the CGMXFC pin as possible, with minimum routing distances and no routing of other signals across the network.

4.4.4 PLL Analog Power Pin (V_{DDA})

V_{DDA} is a power pin used by the analog portions of the PLL. Connect the V_{DDA} pin to the same voltage potential as the V_{DD} pin.

NOTE

Route V_{DDA} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.

4.4.5 PLL Analog Ground Pin (V_{SSA})

V_{SSA} is a ground pin used by the analog portions of the PLL. Connect the V_{SSA} pin to the same voltage potential as the V_{SS} pin.

NOTE

Route V_{SSA} carefully for maximum noise immunity and place bypass capacitors as close as possible to the package.

4.4.6 Oscillator Enable Signal (SIMOSCEN)

The SIMOSCEN signal comes from the system integration module (SIM) and enables the oscillator and PLL.

4.4.7 Oscillator Stop Mode Enable Bit (OSCSTOPENB)

OSCSTOPENB is a bit in the CONFIG register that enables the oscillator to continue operating during stop mode. If this bit is set, the Oscillator continues running during stop mode. If this bit is not set (default), the oscillator is controlled by the SIMOSCEN signal which will disable the oscillator during stop mode.

4.4.8 Crystal Output Frequency Signal (CGMXCLK)

CGMXCLK is the crystal oscillator output signal. It runs at the full speed of the crystal (f_{XCLK}) and comes directly from the crystal oscillator circuit. Figure 4-2 shows only the logical relation of CGMXCLK to OSC1 and OSC2 and may not represent the actual circuitry. The duty cycle of CGMXCLK is unknown and may depend on the crystal and other external factors. Also, the frequency and amplitude of CGMXCLK can be unstable at startup.

4.4.9 CGM Base Clock Output (CGMOUT)

CGMOUT is the clock output of the CGM. This signal goes to the SIM, which generates the MCU clocks. CGMOUT is a 50 percent duty cycle clock running at twice the bus frequency. CGMOUT is software programmable to be either the oscillator output, CGMXCLK, divided by two or the VCO clock, CGMVCLK, divided by two.

4.4.10 CGM CPU Interrupt (CGMINT)

CGMINT is the interrupt signal generated by the PLL lock detector.

4.5 CGM Registers

These registers control and monitor operation of the CGM:

- PLL control register (PCTL)
  (See 4.5.1 PLL Control Register.)
- PLL bandwidth control register (PBWC)
  (See 4.5.2 PLL Bandwidth Control Register.)
- PLL multiplier select register high (PMSH)
  (See 4.5.3 PLL Multiplier Select Register High.)
- PLL multiplier select register low (PMSL)
  (See 4.5.4 PLL Multiplier Select Register Low.)
- PLL VCO range select register (PMRS)
  (See 4.5.5 PLL VCO Range Select Register.)
- PLL reference divider select register (PMDS)
  (See 4.5.6 PLL Reference Divider Select Register.)

Figure 4-3 is a summary of the CGM registers.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0036	PLL Control Register (PCTL)	Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
		Write:		▨						
		Reset:	0	0	1	0	0	0	0	0
$0037	PLL Bandwidth Control Register (PBWC)	Read:	AUTO	LOCK	$\overline{ACQ}$	0	0	0	0	R
		Write:		▨		▨	▨	▨	▨	
		Reset:	0	0	0	0	0	0	0	0
$0038	PLL Multiplier Select High Register (PMSH)	Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$0039	PLL Multiplier Select Low Register (PMSL)	Read:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003A	PLL VCO Select Range Register (PMRS)	Read:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
		Write:								
		Reset:	0	1	0	0	0	0	0	0
$003B	PLL Reference Divider Select Register (PMDS)	Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	1

▨ = Unimplemented   R = Reserved

NOTES:
1. When AUTO = 0, PLLIE is forced clear and is read-only.
2. When AUTO = 0, PLLF and LOCK read as clear.
3. When AUTO = 1, $\overline{ACQ}$ is read-only.
4. When PLLON = 0 or VRS7:VRS0 = $0, BCS is forced clear and is read-only.
5. When PLLON = 1, the PLL programming register is read-only.
6. When BCS = 1, PLLON is forced set and is read-only.

Figure 4-3. CGM I/O Register Summary

4.5.1 PLL Control Register

The PLL control register (PCTL) contains the interrupt enable and flag bits, the on/off switch, the base clock selector bit, the prescaler bits, and the VCO power-of-two range selector bits.

Address: $0036

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PLLIE	PLLF	PLLON	BCS	PRE1	PRE0	VPR1	VPR0
Write:		▨						
Reset:	0	0	1	0	0	0	0	0

▨ = Unimplemented

Figure 4-4. PLL Control Register (PCTL)

PLLIE — PLL Interrupt Enable Bit

This read/write bit enables the PLL to generate an interrupt request when the LOCK bit toggles, setting the PLL flag, PLLF. When the AUTO bit in the PLL bandwidth control register (PBWC) is clear, PLLIE cannot be written and reads as logic 0. Reset clears the PLLIE bit.

1 = PLL interrupts enabled
0 = PLL interrupts disabled

PLLF — PLL Interrupt Flag Bit

This read-only bit is set whenever the LOCK bit toggles. PLLF generates an interrupt request if the PLLIE bit also is set. PLLF always reads as logic 0 when the AUTO bit in the PLL bandwidth control register (PBWC) is clear. Clear the PLLF bit by reading the PLL control register. Reset clears the PLLF bit.

1 = Change in lock condition
0 = No change in lock condition

NOTE

Do not inadvertently clear the PLLF bit. Any read or read-modify-write operation on the PLL control register clears the PLLF bit.

PLLON — PLL On Bit

This read/write bit activates the PLL and enables the VCO clock, CGMVCLK. PLLON cannot be cleared if the VCO clock is driving the base clock, CGMOUT (BCS = 1). (See 4.3.8 Base Clock Selector Circuit.) Reset sets this bit so that the loop can stabilize as the MCU is powering up.

1 = PLL on
0 = PLL off

BCS — Base Clock Select Bit

This read/write bit selects either the crystal oscillator output, CGMXCLK, or the VCO clock, CGMVCLK, as the source of the CGM output, CGMOUT. CGMOUT frequency is one-half the frequency of the selected clock. BCS cannot be set while the PLLON bit is clear. After toggling BCS, it may take up to three CGMXCLK and three CGMVCLK cycles to complete the transition from one source clock to the other. During the transition, CGMOUT is held in stasis. (See 4.3.8 Base Clock Selector Circuit.) Reset clears the BCS bit.

1 = CGMVCLK divided by two drives CGMOUT
0 = CGMXCLK divided by two drives CGMOUT

NOTE

PLLON and BCS have built-in protection that prevents the base clock selector circuit from selecting the VCO clock as the source of the base clock *if the PLL is off. Therefore, PLLON cannot be cleared when BCS is set, and BCS cannot be set when PLLON is clear. If the PLL is off (PLLON = 0), selecting CGMVCLK requires two writes to the PLL control register. (See 4.3.8 Base Clock Selector Circuit.)*

PRE1 and PRE0 — Prescaler Program Bits

These read/write bits control a prescaler that selects the prescaler power-of-two multiplier, P. (See 4.3.3 PLL Circuits and 4.3.6 Programming the PLL.) PRE1 and PRE0 cannot be written when the PLLON bit is set. Reset clears these bits.

Table 4-2. PRE 1 and PRE0 Programming

PRE1 and PRE0	P	Prescaler Multiplier
00	0	1
01	1	2
10	2	4
11	3	8

VPR1 and 0 — VCO Power-of-Two Range Select Bits

These read/write bits control the VCO's hardware power-of-two range multiplier E that, in conjunction with L (See 4.3.3 PLL Circuits, 4.3.6 Programming the PLL, and 4.5.5 PLL VCO Range Select Register.) controls the hardware center-of-range frequency, f_{VRS}. VPR1:VPR0 cannot be written when the PLLON bit is set. Reset clears these bits.

Table 4-3. VPR1 and VPR0 Programming

VPR1 and VPR0	E	VCO Power-of-Two Range Multiplier
00	0	1
01	1	2
10	2	4
11	3(1)	8

1. Do not program E to a value of 3.

4.5.2 PLL Bandwidth Control Register

The PLL bandwidth control register (PBWC):

- Selects automatic or manual (software-controlled) bandwidth control mode
- Indicates when the PLL is locked
- In automatic bandwidth control mode, indicates when the PLL is in acquisition or tracking mode
- In manual operation, forces the PLL into acquisition or tracking mode

Address: $0037

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	0	0	0	0	R
Write:		▨		▨	▨	▨	▨	
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved

Figure 4-5. PLL Bandwidth Control Register (PBWC)

AUTO — Automatic Bandwidth Control Bit

This read/write bit selects automatic or manual bandwidth control. When initializing the PLL for manual operation (AUTO = 0), clear the $\overline{\text{ACQ}}$ bit before turning on the PLL. Reset clears the AUTO bit.

1 = Automatic bandwidth control
0 = Manual bandwidth control

LOCK — Lock Indicator Bit

When the AUTO bit is set, LOCK is a read-only bit that becomes set when the VCO clock, CGMVCLK, is locked (running at the programmed frequency). When the AUTO bit is clear, LOCK reads as logic 0 and has no meaning. The write one function of this bit is reserved for test, so this bit must ***always*** be written a 0. Reset clears the LOCK bit.

1 = VCO frequency correct or locked
0 = VCO frequency incorrect or unlocked

$\overline{\text{ACQ}}$ — Acquisition Mode Bit

When the AUTO bit is set, $\overline{\text{ACQ}}$ is a read-only bit that indicates whether the PLL is in acquisition mode or tracking mode. When the AUTO bit is clear, $\overline{\text{ACQ}}$ is a read/write bit that controls whether the PLL is in acquisition or tracking mode.

In automatic bandwidth control mode (AUTO = 1), the last-written value from manual operation is stored in a temporary location and is recovered when manual operation resumes. Reset clears this bit, enabling acquisition mode.

1 = Tracking mode
0 = Acquisition mode

4.5.3 PLL Multiplier Select Register High

The PLL multiplier select register high (PMSH) contains the programming information for the high byte of the modulo feedback divider.

Address: $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	MUL11	MUL10	MUL9	MUL8
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 4-6. PLL Multiplier Select Register High (PMSH)

MUL11–MUL8 — Multiplier Select Bits

These read/write bits control the high byte of the modulo feedback divider that selects the VCO frequency multiplier N. (See 4.3.3 PLL Circuits and 4.3.6 Programming the PLL.) A value of $0000 in the multiplier select registers configures the modulo feedback divider the same as a value of $0001. Reset initializes the registers to $0040 for a default multiply value of 64.

NOTE

The multiplier select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).

PMSH[7:4] — Unimplemented Bits

These bits have no function and always read as 0s.

4.5.4 PLL Multiplier Select Register Low

The PLL multiplier select register low (PMSL) contains the programming information for the low byte of the modulo feedback divider.

Address: $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MUL7	MUL6	MUL5	MUL4	MUL3	MUL2	MUL1	MUL0
Reset:	0	1	0	0	0	0	0	0

Figure 4-7. PLL Multiplier Select Register Low (PMSL)

NOTE

For applications using 1–8 MHz reference frequencies, this register must be reprogrammed before enabling the PLL. The reset value of this register will cause applications using 1–8 MHz reference frequencies to become unstable if the PLL is enabled without programming an appropriate value. The programmed value must not allow the VCO clock to exceed 32 MHz. See 4.3.6 Programming the PLL for detailed instructions on choosing the proper value for PMSL.

MUL7–MUL0 — Multiplier Select Bits

These read/write bits control the low byte of the modulo feedback divider that selects the VCO frequency multiplier, N. (See 4.3.3 PLL Circuits and 4.3.6 Programming the PLL.) MUL7–MUL0 cannot be written when the PLLON bit in the PCTL is set. A value of $0000 in the multiplier select registers configures the modulo feedback divider the same as a value of $0001. Reset initializes the register to $40 for a default multiply value of 64.

NOTE

The multiplier select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).

4.5.5 PLL VCO Range Select Register

The PLL VCO range select register (PMRS) contains the programming information required for the hardware configuration of the VCO.

Address: $003A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	VRS7	VRS6	VRS5	VRS4	VRS3	VRS2	VRS1	VRS0
Reset:	0	1	0	0	0	0	0	0

Figure 4-8. PLL VCO Range Select Register (PMRS)

VRS7–VRS0 — VCO Range Select Bits

These read/write bits control the hardware center-of-range linear multiplier L which, in conjunction with E (see 4.3.3 PLL Circuits, 4.3.6 Programming the PLL, and 4.5.1 PLL Control Register), controls the hardware center-of-range frequency, f_{VRS}. VRS7–VRS0 cannot be written when the PLLON bit in the PCTL is set. (See 4.3.7 Special Programming Exceptions.) A value of $00 in the VCO range select register disables the PLL and clears the BCS bit in the PLL control register (PCTL). (See 4.3.8 Base Clock Selector Circuit and 4.3.7 Special Programming Exceptions.). Reset initializes the register to $40 for a default range multiply value of 64.

NOTE

The VCO range select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1) and such that the VCO clock cannot be selected as the source of the base clock (BCS = 1) if the VCO range select bits are all clear.

The PLL VCO range select register must be programmed correctly. Incorrect programming can result in failure of the PLL to achieve lock.

4.5.6 PLL Reference Divider Select Register

The PLL reference divider select register (PMDS) contains the programming information for the modulo reference divider.

Address: $003B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	RDS3	RDS2	RDS1	RDS0
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	1

▨ = Unimplemented

Figure 4-9. PLL Reference Divider Select Register (PMDS)

RDS3–RDS0 — Reference Divider Select Bits

These read/write bits control the modulo reference divider that selects the reference division factor, R. (See 4.3.3 PLL Circuits and 4.3.6 Programming the PLL.) RDS7–RDS0 cannot be written when the PLLON bit in the PCTL is set. A value of $00 in the reference divider select register configures the reference divider the same as a value of $01. (See 4.3.7 Special Programming Exceptions.) Reset initializes the register to $01 for a default divide value of 1.

NOTE

The reference divider select bits have built-in protection such that they cannot be written when the PLL is on (PLLON = 1).

The default divide value of 1 is recommended for all applications.

4.6 Interrupts

When the AUTO bit is set in the PLL bandwidth control register (PBWC), the PLL can generate a CPU interrupt request every time the LOCK bit changes state. The PLLIE bit in the PLL control register (PCTL) enables CPU interrupts from the PLL. PLLF, the interrupt flag in the PCTL, becomes set whether interrupts are enabled or not. When the AUTO bit is clear, CPU interrupts from the PLL are disabled and PLLF reads as logic 0.

Software should read the LOCK bit after a PLL interrupt request to see if the request was due to an entry into lock or an exit from lock. When the PLL enters lock, the VCO clock, CGMVCLK, divided by two can be selected as the CGMOUT source by setting BCS in the PCTL. When the PLL exits lock, the VCO clock frequency is corrupt, and appropriate precautions should be taken. If the application is not frequency sensitive, interrupts should be disabled to prevent PLL interrupt service routines from impeding software performance or from exceeding stack limitations.

NOTE

Software can select the CGMVCLK divided by two as the CGMOUT source even if the PLL is not locked (LOCK = 0). Therefore, software should make sure the PLL is locked before setting the BCS bit.

4.7 Special Modes

The WAIT instruction puts the MCU in low power-consumption standby modes.

4.7.1 Wait Mode

The WAIT instruction does not affect the CGM. Before entering wait mode, software can disengage and turn off the PLL by clearing the BCS and PLLON bits in the PLL control register (PCTL) to save power. Less power-sensitive applications can disengage the PLL without turning it off, so that the PLL clock is immediately available at WAIT exit. This would be the case also when the PLL is to wake the MCU from wait mode, such as when the PLL is first enabled and waiting for LOCK or LOCK is lost.

4.7.2 Stop Mode

If the OSCSTOPENB bit in the CONFIG register is cleared (default), then the STOP instruction disables the CGM (oscillator and phase locked loop) and holds low all CGM outputs (CGMXCLK, CGMOUT, and CGMINT).

If the STOP instruction is executed with the VCO clock, CGMVCLK, divided by two driving CGMOUT, the PLL automatically clears the BCS bit in the PLL control register (PCTL), thereby selecting the crystal clock, CGMXCLK, divided by two as the source of CGMOUT. When the MCU recovers from STOP, the crystal clock divided by two drives CGMOUT and BCS remains clear.

If the OSCSTOPENB bit in the CONFIG register is set, then the phase locked loop is shut off but the oscillator will continue to operate in stop mode.

4.7.3 CGM During Break Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. (See 14.7.1 SIM Break Status Register.)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the PLLF bit during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write the PLL control register during the break state without affecting the PLLF bit.

4.8 Acquisition/Lock Time Specifications

The acquisition and lock times of the PLL are, in many applications, the most critical PLL design parameters. Proper design and use of the PLL ensures the highest stability and lowest acquisition/lock times.

4.8.1 Acquisition/Lock Time Definitions

Typical control systems refer to the acquisition time or lock time as the reaction time, within specified tolerances, of the system to a step input. In a PLL, the step input occurs when the PLL is turned on or when it suffers a noise hit. The tolerance is usually specified as a percentage of the step input or when the output settles to the desired value plus or minus a percentage of the frequency change. Therefore, the reaction time is constant in this definition, regardless of the size of the step input. For example, consider a system with a 5 percent acquisition time tolerance. If a command instructs the system to change from 0 Hz to 1 MHz, the acquisition time is the time taken for the frequency to reach 1 MHz ±50 kHz. Fifty kHz = 5% of the 1-MHz step input. If the system is operating at 1 MHz and suffers a –100-kHz noise hit, the acquisition time is the time taken to return from 900 kHz to 1 MHz ±5 kHz. Five kHz = 5% of the 100-kHz step input.

Other systems refer to acquisition and lock times as the time the system takes to reduce the error between the actual output and the desired output to within specified tolerances. Therefore, the acquisition or lock time varies according to the original error in the output. Minor errors may not even be registered. Typical PLL applications prefer to use this definition because the system requires the output frequency to be within a certain tolerance of the desired frequency regardless of the size of the initial error.

4.8.2 Parametric Influences on Reaction Time

Acquisition and lock times are designed to be as short as possible while still providing the highest possible stability. These reaction times are not constant, however. Many factors directly and indirectly affect the acquisition time.

The most critical parameter which affects the reaction times of the PLL is the reference frequency, f_{RDV}. This frequency is the input to the phase detector and controls how often the PLL makes corrections. For stability, the corrections must be small compared to the desired frequency, so several corrections are required to reduce the frequency error. Therefore, the slower the reference the longer it takes to make these corrections. This parameter is under user control via the choice of crystal frequency f_{XCLK} and the R value programmed in the reference divider. (See 4.3.3 PLL Circuits, 4.3.6 Programming the PLL, and 4.5.6 PLL Reference Divider Select Register.)

Another critical parameter is the external filter network. The PLL modifies the voltage on the VCO by adding or subtracting charge from capacitors in this network. Therefore, the rate at which the voltage changes for a given frequency error (thus change in charge) is proportional to the capacitance. The size of the capacitor also is related to the stability of the PLL. If the capacitor is too small, the PLL cannot make small enough adjustments to the voltage and the system cannot lock. If the capacitor is too large, the PLL may not be able to adjust the voltage in a reasonable time. (See 4.8.3 Choosing a Filter.)

Also important is the operating voltage potential applied to V_{DDA}. The power supply potential alters the characteristics of the PLL. A fixed value is best. Variable supplies, such as batteries, are acceptable if they vary within a known range at very slow speeds. Noise on the power supply is not acceptable, because it causes small frequency errors which continually change the acquisition time of the PLL.

Temperature and processing also can affect acquisition time because the electrical characteristics of the PLL change. The part operates as specified as long as these influences stay within the specified limits. External factors, however, can cause drastic changes in the operation of the PLL. These factors include noise injected into the PLL through the filter capacitor, filter capacitor leakage, stray impedances on the circuit board, and even humidity or circuit board contamination.

4.8.3 Choosing a Filter

As described in 4.8.2 Parametric Influences on Reaction Time, the external filter network is critical to the stability and reaction time of the PLL. The PLL is also dependent on reference frequency and supply voltage.

Figure 4-10 shows two types of filter circuits. In low-cost applications, where stability and reaction time of the PLL are not critical, the three component filter network shown in Figure 4-10(B) can be replaced by a single capacitor, C_F, as shown in Figure 4-10(A). Refer to Table 4-4 for recommended filter components at various reference frequencies. For reference frequencies between the values listed in the table, extrapolate to the nearest common capacitor value. In general, a slightly larger capacitor provides more stability at the expense of increased lock time.

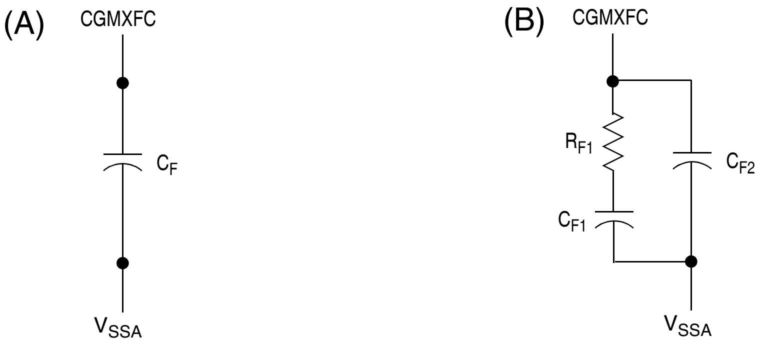


Figure 4-10. PLL Filter

Table 4-4. Example Filter Component Values

f_{RCLK}	C_{F1}	C_{F2}	R_{F1}	C_F
1 MHz	8.2 nF	820 pF	2k	18 nF
2 MHz	4.7 nF	470 pF	2k	6.8 nF
3 MHz	3.3 nF	330 pF	2k	5.6 nF
4 MHz	2.2 nF	220 pF	2k	4.7 nF
5 MHz	1.8 nF	180 pF	2k	3.9 nF
6 MHz	1.5 nF	150 pF	2k	3.3 nF
7 MHz	1.2 nF	120 pF	2k	2.7 nF
8 MHz	1 nF	100 pF	2k	2.2 nF

Chapter 5
Configuration Register (CONFIG)

5.1 Introduction

This section describes the configuration registers, CONFIG1 and CONFIG2. The configuration registers enable or disable these options:

- Stop mode recovery time (32 CGMXCLK cycles or 4096 CGMXCLK cycles)
- COP timeout period (262,128 or 8176 CGMXCLK cycles)
- STOP instruction
- Computer operating properly module (COP)
- Low-voltage inhibit (LVI) module control and voltage trip point selection
- Enable/disable the oscillator (OSC) during stop mode

5.2 Functional Description

The configuration registers are used in the initialization of various options. The configuration registers can be written once after each reset. All of the configuration register bits are cleared during reset. Since the various options affect the operation of the microcontroller unit (MCU), it is recommended that these registers be written immediately after reset. The configuration registers are located at $001E and $001F and may be read at anytime.

NOTE

On a FLASH device, the options except LVI5OR3 are one-time writable by the user after each reset. The LVI5OR3 bit is one-time writable by the user only after each POR (power-on reset). The CONFIG registers are not in the FLASH memory but are special registers containing one-time writable latches after each reset. Upon a reset, the CONFIG registers default to predetermined settings as shown in Figure 5-1 and Figure 5-2.

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	OSCSTOPENB	SCIBDSRC
Write:	▨	▨	▨	▨	▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented   R = Reserved

Figure 5-1. Configuration Register 2 (CONFIG2)

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVI5OR3	SSREC	STOP	COPD
Write:								
Reset:	0	0	0	0	See note	0	0	0

Note: LVI5OR3 bit is only reset via POR (power-on reset).

Figure 5-2. Configuration Register 1 (CONFIG1)

OSCSTOPENB — Oscillator Stop Mode Enable Bar Bit

OSCSTOPENB, when set, will enable the oscillator to continue to generate clocks in stop mode. See Chapter 4 Clock Generator Module (CGM). This function is used to keep the timebase running while the rest of the MCU stops. See Chapter 16 Timebase Module (TBM). When clear, the oscillator will cease to generate clocks while in stop mode. The default state for this option is clear, disabling the oscillator in stop mode.

1 = Oscillator enabled to operate during stop mode
0 = Oscillator disabled during stop mode (default)

SCIBDSRC — SCI Baud Rate Clock Source Bit

SCIBDSRC controls the clock source used for the serial communications interface (SCI). The setting of this bit affects the frequency at which the SCI operates.See Chapter 13 Serial Communications Interface (SCI) Module.

1 = Internal bus clock used as clock source for SCI
0 = External oscillator used as clock source for SCI

COPRS — COP Rate Select Bit

COPRS selects the COP timeout period. Reset clears COPRS. See Chapter 6 Computer Operating Properly (COP) Module

1 = COP timeout period = 8176 CGMXCLK cycles
0 = COP timeout period = 262,128 CGMXCLK cycles

LVISTOP — LVI Enable in Stop Mode Bit

Setting the LVISTOP bit enables the LVI to operate during stop mode. Reset clears LVISTOP.

1 = LVI enabled during stop mode
0 = LVI disabled during stop mode

LVIRSTD — LVI Reset Disable Bit

LVIRSTD disables the reset signal from the LVI module. See Chapter 11 Low-Voltage Inhibit (LVI).

0 = LVI module resets enabled

NOTE

On the MC68HC908GR8B, LVIRSTD is always clear.

LVIPWRD — LVI Power Disable Bit

LVIPWRD disables the LVI module. See Chapter 11 Low-Voltage Inhibit (LVI).

0 = LVI module power enabled

NOTE

On the MC68HC908GR8B, LVIPWRD is always clear.

LVI5OR3 — LVI 5-V or 3-V Operating Mode Bit

LVI5OR3 selects the voltage operating mode of the LVI module (see Chapter 11 Low-Voltage Inhibit (LVI)). The voltage mode selected for the LVI should match the operating V_{DD} (see Chapter 19 Electrical Specifications) for the LVI's voltage trip points for each of the modes.

0 = LVI operates in 3-V mode

NOTE

On the MC68HC908GR8B, LVI5OR3 is always clear.

SSREC — Short Stop Recovery Bit

SSREC enables the CPU to exit stop mode with a delay of 32 CGMXCLK cycles instead of a 4096-CGMXCLK cycle delay.

1 = Stop mode recovery after 32 CGMXCLK cycles
0 = Stop mode recovery after 4096 CGMXCLCK cycles

NOTE

Exiting stop mode by an LVI reset will result in the long stop recovery.

If the system clock source is an external crystal and the OSCSTOPENB configuration bit is not set, the oscillator will be disabled during stop mode. The short stop recovery does not provide enough time for oscillator stabilization and for this reason the SSREC bit should not be set.

The system stabilization time for power-on reset and long stop recovery (both 4096 CGMXCLK cycles) gives a delay longer than the LVI enable time for these startup scenarios. There is no period where the MCU is not protected from a low-power condition. However, when using the short stop recovery configuration option, the 32-CGMXCLK delay must be greater than the LVI's turn on time to avoid a period in startup where the LVI is not protecting the MCU.

STOP — STOP Instruction Enable Bit

STOP enables the STOP instruction.

1 = STOP instruction enabled
0 = STOP instruction treated as illegal opcode

COPD — COP Disable Bit

COPD disables the COP module. See Chapter 6 Computer Operating Properly (COP) Module.

1 = COP module disabled
0 = COP module enabled

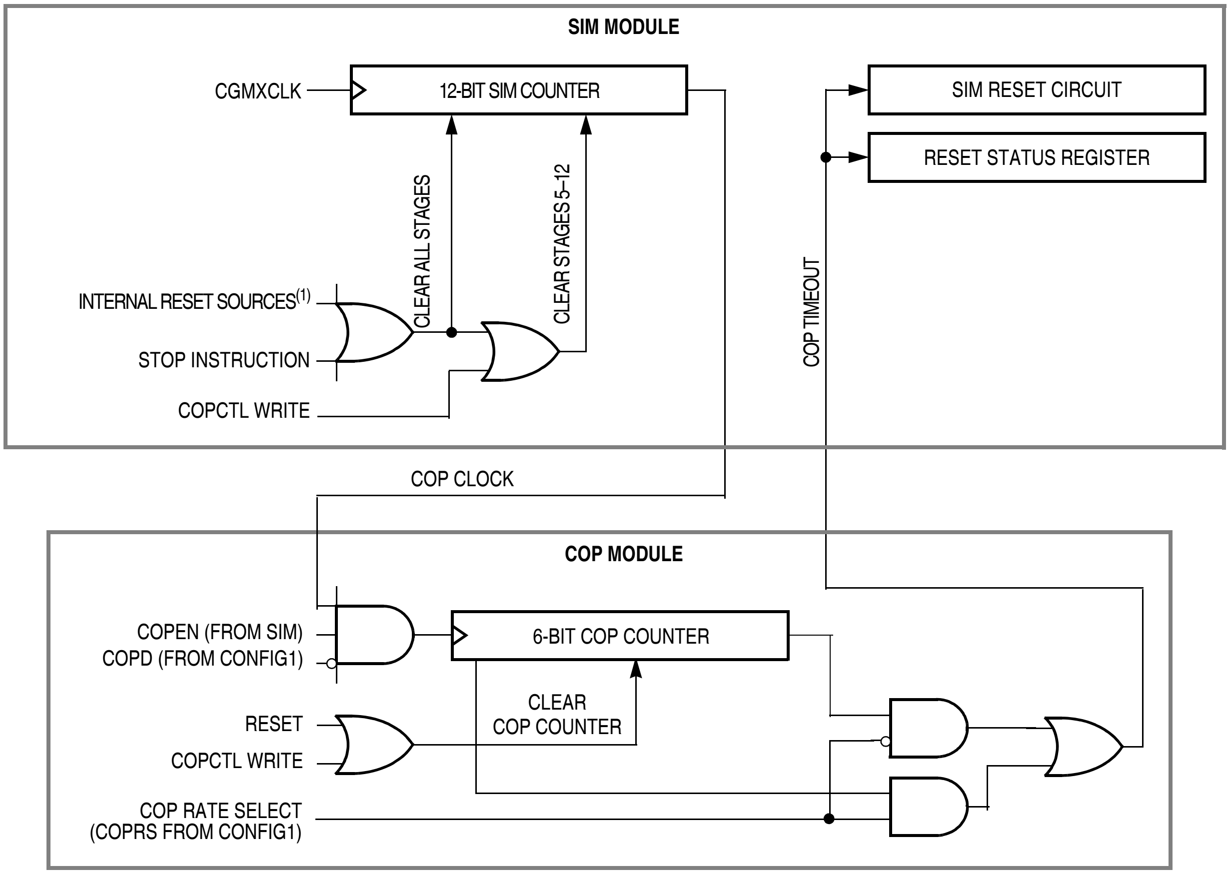
Chapter 6
Computer Operating Properly (COP) Module

6.1 Introduction

The computer operating properly (COP) module contains a free-running counter that generates a reset if allowed to overflow. The COP module helps software recover from runaway code. Prevent a COP reset by clearing the COP counter periodically. The COP module can be disabled through the COPD bit in the CONFIG register.

6.2 Functional Description

Figure 6-1 shows the structure of the COP module.



1. See Chapter 14 System Integration Module (SIM) for more details.

Figure 6-1. COP Block Diagram

The COP counter is a free-running 6-bit counter preceded by a 12-bit prescaler counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after 262,128 or 8176 CGMXCLK cycles, depending on the state of the COP rate select bit, COPRS, in the configuration register. With a 262,128 CGMXCLK cycle overflow option, a 4.9152-MHz crystal gives a COP timeout period of 53.3 ms. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12–5 of the prescaler.

NOTE

Service the COP immediately after reset and before entering or after exiting stop mode to guarantee the maximum time before the first COP counter overflow.

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 CGMXCLK cycles and sets the COP bit in the reset status register (RSR).

In monitor mode, the COP is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ}}$ is held at V_{TST}. During the break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP.

NOTE

Place COP clearing instructions in the main program and not in an interrupt subroutine. Such an interrupt subroutine could keep the COP from generating a reset even while the main program is not working properly.

6.3 I/O Signals

The following paragraphs describe the signals shown in Figure 6-1.

6.3.1 CGMXCLK

CGMXCLK is the crystal oscillator output signal. CGMXCLK frequency is equal to the crystal frequency.

6.3.2 STOP Instruction

The STOP instruction clears the COP prescaler.

6.3.3 COPCTL Write

Writing any value to the COP control register (COPCTL) clears the COP counter and clears bits 12–5 of the prescaler. Reading the COP control register returns the low byte of the reset vector. See 6.4 COP Control Register.

6.3.4 Power-On Reset

The power-on reset (POR) circuit clears the COP prescaler 4096 CGMXCLK cycles after power-up.

6.3.5 Internal Reset

An internal reset clears the COP prescaler and the COP counter.

6.3.6 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the configuration register. See Chapter 5 Configuration Register (CONFIG).

6.3.7 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the configuration register. See Chapter 5 Configuration Register (CONFIG).

6.4 COP Control Register

The COP control register (COPCTL) is located at address $FFFF and overlaps the reset vector. Writing any value to $FFFF clears the COP counter and starts a new timeout period. Reading location $FFFF returns the low byte of the reset vector.

Address: $FFFF

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Low byte of reset vector							
Write:	Clear COP counter							
Reset:	Unaffected by reset							

Figure 6-2. COP Control Register (COPCTL)

6.5 Interrupts

The COP does not generate central processor unit (CPU) interrupt requests.

6.6 Monitor Mode

When monitor mode is entered with V_{TST} on the $\overline{\text{IRQ}}$ pin, the COP is disabled as long as V_{TST} remains on the $\overline{\text{IRQ}}$ pin or the $\overline{\text{RST}}$ pin. When monitor mode is entered by having blank reset vectors and not having V_{TST} on the $\overline{\text{IRQ}}$ pin, the COP is automatically disabled until a POR occurs.

6.7 Low-Power Modes

The WAIT and STOP instructions put the microcontroller unit (MCU) in low power-consumption standby modes.

6.7.1 Wait Mode

The COP remains active during wait mode. If COP is enabled, a reset will occur at COP timeout.

6.7.2 Stop Mode

Stop mode turns off the CGMXCLK input to the COP and clears the COP prescaler. Service the COP immediately before entering or after exiting stop mode to ensure a full COP timeout period after entering or exiting stop mode.

To prevent inadvertently turning off the COP with a STOP instruction, a configuration option is available that disables the STOP instruction. When the STOP bit in the configuration register has the STOP instruction disabled, execution of a STOP instruction results in an illegal opcode reset.

6.8 COP Module During Break Mode

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{\text{RST}}$ pin.

Chapter 7
Central Processor Unit (CPU)

7.1 Introduction

The M68HC08 CPU (central processor unit) is an enhanced and fully object-code-compatible version of the M68HC05 CPU. The *CPU08 Reference Manual* (document order number CPU08RM/AD) contains a description of the CPU instruction set, addressing modes, and architecture.

7.2 Features

Features of the CPU include:
- Object code fully upward-compatible with M68HC05 Family
- 16-bit stack pointer with stack manipulation instructions
- 16-bit index register with x-register manipulation instructions
- 8-MHz CPU internal bus frequency
- 64-Kbyte program/data memory space
- 16 addressing modes
- Memory-to-memory data moves without using accumulator
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- Enhanced binary-coded decimal (BCD) data handling
- Modular architecture with expandable internal bus definition for extension of addressing range beyond 64 Kbytes
- Low-power stop and wait modes

7.3 CPU Registers

Figure 7-1 shows the five CPU registers. CPU registers are not part of the memory map.

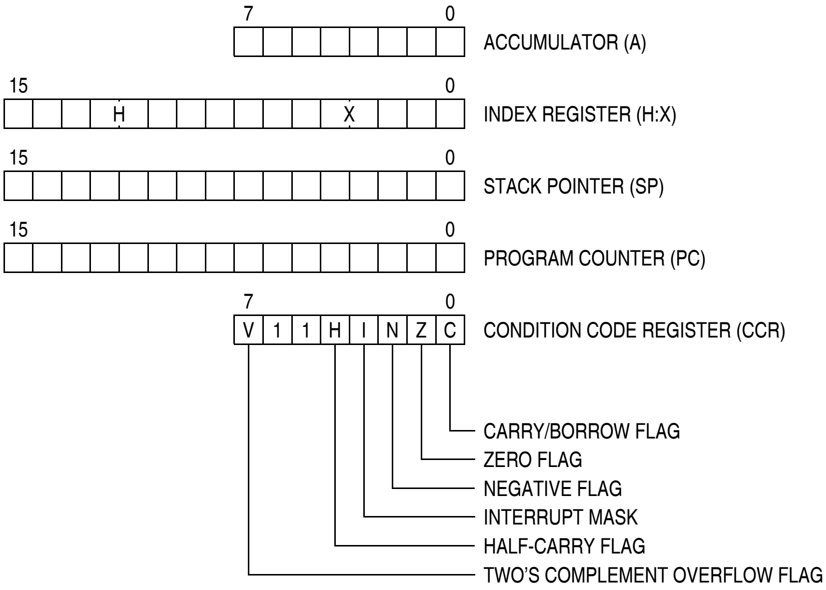


Figure 7-1. CPU Registers

7.3.1 Accumulator

The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and the results of arithmetic/logic operations.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:								
Write:								
Reset:	Unaffected by reset							

Figure 7-2. Accumulator (A)

7.3.2 Index Register

The 16-bit index register allows indexed addressing of a 64-Kbyte memory space. H is the upper byte of the index register, and X is the lower byte. H:X is the concatenated 16-bit index register.

In the indexed addressing modes, the CPU uses the contents of the index register to determine the conditional address of the operand.

The index register can serve also as a temporary data storage location.

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

X = Indeterminate

Figure 7-3. Index Register (H:X)

7.3.3 Stack Pointer

The stack pointer is a 16-bit register that contains the address of the next location on the stack. During a reset, the stack pointer is preset to $00FF. The reset stack pointer (RSP) instruction sets the least significant byte to $FF and does not affect the most significant byte. The stack pointer decrements as data is pushed onto the stack and increments as data is pulled from the stack.

In the stack pointer 8-bit offset and 16-bit offset addressing modes, the stack pointer can function as an index register to access data on the stack. The CPU uses the contents of the stack pointer to determine the conditional address of the operand.

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

Figure 7-4. Stack Pointer (SP)

NOTE

The location of the stack is arbitrary and may be relocated anywhere in random-access memory (RAM). Moving the SP out of page 0 ($0000 to $00FF) frees direct address (page 0) space. For correct operation, the stack pointer must point only to RAM locations.

7.3.4 Program Counter

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched.

Normally, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

During reset, the program counter is loaded with the reset vector address located at $FFFE and $FFFF. The vector address is the address of the first instruction to be executed after exiting the reset state.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
Read: Write:																
Reset:	Loaded with vector from $FFFE and $FFFF															

Figure 7-5. Program Counter (PC)

7.3.5 Condition Code Register

The 8-bit condition code register contains the interrupt mask and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to 1. The following paragraphs describe the functions of the condition code register.

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	V	1	1	H	I	N	Z	C
Reset:	X	1	1	X	1	X	X	X

X = Indeterminate

Figure 7-6. Condition Code Register (CCR)

V — Overflow Flag

The CPU sets the overflow flag when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow flag.

1 = Overflow
0 = No overflow

H — Half-Carry Flag

The CPU sets the half-carry flag when a carry occurs between accumulator bits 3 and 4 during an add-without-carry (ADD) or add-with-carry (ADC) operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations. The DAA instruction uses the states of the H and C flags to determine the appropriate correction factor.

1 = Carry between bits 3 and 4
0 = No carry between bits 3 and 4

I — Interrupt Mask

When the interrupt mask is set, all maskable CPU interrupts are disabled. CPU interrupts are enabled when the interrupt mask is cleared. When a CPU interrupt occurs, the interrupt mask is set automatically after the CPU registers are saved on the stack, but before the interrupt vector is fetched.

1 = Interrupts disabled
0 = Interrupts enabled

NOTE

To maintain M6805 Family compatibility, the upper byte of the index register (H) is not stacked automatically. If the interrupt service routine modifies H, then the user must stack and unstack H using the PSHH and PULH instructions.

After the I bit is cleared, the highest-priority interrupt request is serviced first.

A return-from-interrupt (RTI) instruction pulls the CPU registers from the stack and restores the interrupt mask from the stack. After any reset, the interrupt mask is set and can be cleared only by the clear interrupt mask software instruction (CLI).

N — Negative Flag

The CPU sets the negative flag when an arithmetic operation, logic operation, or data manipulation produces a negative result, setting bit 7 of the result.

1 = Negative result
0 = Non-negative result

Z — Zero Flag

The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of $00.

1 = Zero result
0 = Non-zero result

C — Carry/Borrow Flag

The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some instructions — such as bit test and branch, shift, and rotate — also clear or set the carry/borrow flag.

1 = Carry out of bit 7
0 = No carry out of bit 7

7.4 Arithmetic/Logic Unit (ALU)

The ALU performs the arithmetic and logic operations defined by the instruction set.

Refer to the *CPU08 Reference Manual* (document order number CPU08RM/AD) for a description of the instructions and addressing modes and more detail about the architecture of the CPU.

7.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

7.5.1 Wait Mode

The WAIT instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling interrupts. After exit from wait mode by interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

7.5.2 Stop Mode

The STOP instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling external interrupts. After exit from stop mode by external interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

After exiting stop mode, the CPU clock begins running after the oscillator stabilization delay.

7.6 CPU During Break Interrupts

If a break module is present on the MCU, the CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC:$FFFD or with $FEFC:$FEFD in monitor mode

The break interrupt begins after completion of the CPU instruction in progress. If the break address register match occurs on the last cycle of a CPU instruction, the break interrupt begins immediately.

A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the MCU to normal operation if the break interrupt has been deasserted.

7.7 Instruction Set Summary

Table 7-1 provides a summary of the M68HC08 instruction set.

Table 7-1. Instruction Set Summary (Sheet 1 of 6)

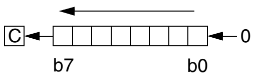
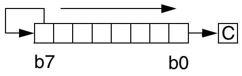
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
ADC #*opr*	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM	A9	ii	2
ADC *opr*									DIR	B9	dd	3
ADC *opr*									EXT	C9	hh ll	4
ADC *opr*,X									IX2	D9	ee ff	4
ADC *opr*,X									IX1	E9	ff	3
ADC ,X									IX	F9		2
ADC *opr*,SP									SP1	9EE9	ff	4
ADC *opr*,SP									SP2	9ED9	ee ff	5
ADD #*opr*	Add without Carry	A ← (A) + (M)	↕	↕	–	↕	↕	↕	IMM	AB	ii	2
ADD *opr*									DIR	BB	dd	3
ADD *opr*									EXT	CB	hh ll	4
ADD *opr*,X									IX2	DB	ee ff	4
ADD *opr*,X									IX1	EB	ff	3
ADD ,X									IX	FB		2
ADD *opr*,SP									SP1	9EEB	ff	4
ADD *opr*,SP									SP2	9EDB	ee ff	5
AIS #*opr*	Add Immediate Value (Signed) to SP	SP ← (SP) + (16 « M)	–	–	–	–	–	–	IMM	A7	ii	2
AIX #*opr*	Add Immediate Value (Signed) to H:X	H:X ← (H:X) + (16 « M)	–	–	–	–	–	–	IMM	AF	ii	2
AND #*opr*	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM	A4	ii	2
AND *opr*									DIR	B4	dd	3
AND *opr*									EXT	C4	hh ll	4
AND *opr*,X									IX2	D4	ee ff	4
AND *opr*,X									IX1	E4	ff	3
AND ,X									IX	F4		2
AND *opr*,SP									SP1	9EE4	ff	4
AND *opr*,SP									SP2	9ED4	ee ff	5
ASL *opr*	Arithmetic Shift Left (Same as LSL)		↕	–	–	↕	↕	↕	DIR	38	dd	4
ASLA									INH	48		1
ASLX									INH	58		1
ASL *opr*,X									IX1	68	ff	4
ASL ,X									IX	78		3
ASL *opr*,SP									SP1	9E68	ff	5
ASR *opr*	Arithmetic Shift Right		↕	–	–	↕	↕	↕	DIR	37	dd	4
ASRA									INH	47		1
ASRX									INH	57		1
ASR *opr*,X									IX1	67	ff	4
ASR *opr*,X									IX	77		3
ASR *opr*,SP									SP1	9E67	ff	5
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + 2 + rel ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BCLR *n*, *opr*	Clear Bit n in M	Mn ← 0	–	–	–	–	–	–	DIR (b0)	11	dd	4
									DIR (b1)	13	dd	4
									DIR (b2)	15	dd	4
									DIR (b3)	17	dd	4
									DIR (b4)	19	dd	4
									DIR (b5)	1B	dd	4
									DIR (b6)	1D	dd	4
									DIR (b7)	1F	dd	4
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 0	–	–	–	–	–	–	REL	22	rr	3

Table 7-1. Instruction Set Summary (Sheet 2 of 6)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{IRQ}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{IRQ}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{IRQ}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{IRQ}$ = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #*opr*	Bit Test	(A) & (M)	0	–	–	↕	↕	–	IMM	A5	ii	2
BIT *opr*									DIR	B5	dd	3
BIT *opr*									EXT	C5	hh ll	4
BIT *opr*,X									IX2	D5	ee ff	4
BIT *opr*,X									IX1	E5	ff	3
BIT ,X									IX	F5		2
BIT *opr*,SP									SP1	9EE5	ff	4
BIT *opr*,SP									SP2	9ED5	ee ff	5
BLE *opr*	Branch if Less Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) =1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0)	01	dd rr	5
									DIR (b1)	03	dd rr	5
									DIR (b2)	05	dd rr	5
									DIR (b3)	07	dd rr	5
									DIR (b4)	09	dd rr	5
									DIR (b5)	0B	dd rr	5
									DIR (b6)	0D	dd rr	5
									DIR (b7)	0F	dd rr	5
BRN *rel*	Branch Never	PC ← (PC) + 2	–	–	–	–	–	–	REL	21	rr	3
BRSET *n*,*opr*,*rel*	Branch if Bit *n* in M Set	PC ← (PC) + 3 + *rel* ? (Mn) = 1	–	–	–	–	–	↕	DIR (b0)	00	dd rr	5
									DIR (b1)	02	dd rr	5
									DIR (b2)	04	dd rr	5
									DIR (b3)	06	dd rr	5
									DIR (b4)	08	dd rr	5
									DIR (b5)	0A	dd rr	5
									DIR (b6)	0C	dd rr	5
									DIR (b7)	0E	dd rr	5
BSET *n*,*opr*	Set Bit *n* in M	Mn ← 1	–	–	–	–	–	–	DIR (b0)	10	dd	4
									DIR (b1)	12	dd	4
									DIR (b2)	14	dd	4
									DIR (b3)	16	dd	4
									DIR (b4)	18	dd	4
									DIR (b5)	1A	dd	4
									DIR (b6)	1C	dd	4
									DIR (b7)	1E	dd	4
BSR *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) – 1; push (PCH) SP ← (SP) – 1 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	4
CBEQ *opr,rel*	Compare and Branch if Equal	PC ← (PC) + 3 + rel ? (A) – (M) = $00	–	–	–	–	–	–	DIR	31	dd rr	5
CBEQA #*opr,rel*		PC ← (PC) + 3 + rel ? (A) – (M) = $00							IMM	41	ii rr	4
CBEQX #*opr,rel*		PC ← (PC) + 3 + rel ? (X) – (M) = $00							IMM	51	ii rr	4
CBEQ *opr*,X+,*rel*		PC ← (PC) + 3 + rel ? (A) – (M) = $00							IX1+	61	ff rr	5
CBEQ X+,*rel*		PC ← (PC) + 2 + rel ? (A) – (M) = $00							IX+	71	rr	4
CBEQ *opr*,SP,*rel*		PC ← (PC) + 4 + rel ? (A) – (M) = $00							SP1	9E61	ff rr	6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask	I ← 0	–	–	0	–	–	–	INH	9A		2

Table 7-1. Instruction Set Summary (Sheet 3 of 6)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
CLR *opr*	Clear	M ← \$00	0	–	–	0	1	–	DIR	3F	dd	3
CLRA		A ← \$00							INH	4F		1
CLRX		X ← \$00							INH	5F		1
CLRH		H ← \$00							INH	8C		1
CLR *opr*,X		M ← \$00							IX1	6F	ff	3
CLR ,X		M ← \$00							IX	7F		2
CLR *opr*,SP		M ← \$00							SP1	9E6F	ff	4
CMP #*opr*	Compare A with M	(A) – (M)	↕	–	–	↕	↕	↕	IMM	A1	ii	2
CMP *opr*									DIR	B1	dd	3
CMP *opr*									EXT	C1	hh ll	4
CMP *opr*,X									IX2	D1	ee ff	4
CMP *opr*,X									IX1	E1	ff	3
CMP ,X									IX	F1		2
CMP *opr*,SP									SP1	9EE1	ff	4
CMP *opr*,SP									SP2	9ED1	ee ff	5
COM *opr*	Complement (One’s Complement)	M ← $(\overline{M})$ = \$FF – (M)	0	–	–	↕	↕	1	DIR	33	dd	4
COMA		A ← $(\overline{A})$ = \$FF – (M)							INH	43		1
COMX		X ← $(\overline{X})$ = \$FF – (M)							INH	53		1
COM *opr*,X		M ← $(\overline{M})$ = \$FF – (M)							IX1	63	ff	4
COM ,X		M ← $(\overline{M})$ = \$FF – (M)							IX	73		3
COM *opr*,SP		M ← $(\overline{M})$ = \$FF – (M)							SP1	9E63	ff	5
CPHX #*opr*	Compare H:X with M	(H:X) – (M:M + 1)	↕	–	–	↕	↕	↕	IMM	65	ii ii+1	3
CPHX *opr*									DIR	75	dd	4
CPX #*opr*	Compare X with M	(X) – (M)	↕	–	–	↕	↕	↕	IMM	A3	ii	2
CPX *opr*									DIR	B3	dd	3
CPX *opr*									EXT	C3	hh ll	4
CPX ,X									IX2	D3	ee ff	4
CPX *opr*,X									IX1	E3	ff	3
CPX *opr*,X									IX	F3		2
CPX *opr*,SP									SP1	9EE3	ff	4
CPX *opr*,SP									SP2	9ED3	ee ff	5
DAA	Decimal Adjust A	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		2
	Decrement and Branch if Not Zero	A ← (A) – 1 or M ← (M) – 1 or X ← (X) – 1	–	–	–	–	–	–				
DBNZ *opr*,*rel*		PC ← (PC) + 3 + *rel* ? (result) ≠ 0							DIR	3B	dd rr	5
DBNZA *rel*		PC ← (PC) + 2 + *rel* ? (result) ≠ 0							INH	4B	rr	3
DBNZX *rel*		PC ← (PC) + 2 + *rel* ? (result) ≠ 0							INH	5B	rr	3
DBNZ *opr*,X,*rel*		PC ← (PC) + 3 + *rel* ? (result) ≠ 0							IX1	6B	ff rr	5
DBNZ X,*rel*		PC ← (PC) + 2 + *rel* ? (result) ≠ 0							IX	7B	rr	4
DBNZ *opr*,SP,*rel*		PC ← (PC) + 4 + *rel* ? (result) ≠ 0							SP1	9E6B	ff rr	6
DEC *opr*	Decrement	M ← (M) – 1	↕	–	–	↕	↕	–	DIR	3A	dd	4
DECA		A ← (A) – 1							INH	4A		1
DECX		X ← (X) – 1							INH	5A		1
DEC *opr*,X		M ← (M) – 1							IX1	6A	ff	4
DEC ,X		M ← (M) – 1							IX	7A		3
DEC *opr*,SP		M ← (M) – 1							SP1	9E6A	ff	5
DIV	Divide	A ← (H:A)/(X) H ← Remainder	–	–	–	–	↕	↕	INH	52		7
EOR #*opr*	Exclusive OR M with A	A ← (A ⊕ M)	0	–	–	↕	↕	–	IMM	A8	ii	2
EOR *opr*									DIR	B8	dd	3
EOR *opr*									EXT	C8	hh ll	4
EOR *opr*,X									IX2	D8	ee ff	4
EOR *opr*,X									IX1	E8	ff	3
EOR ,X									IX	F8		2
EOR *opr*,SP									SP1	9EE8	ff	4
EOR *opr*,SP									SP2	9ED8	ee ff	5
INC *opr*	Increment	M ← (M) + 1	↕	–	–	↕	↕	–	DIR	3C	dd	4
INCA		A ← (A) + 1							INH	4C		1
INCX		X ← (X) + 1							INH	5C		1
INC *opr*,X		M ← (M) + 1							IX1	6C	ff	4
INC ,X		M ← (M) + 1							IX	7C		3
INC *opr*,SP		M ← (M) + 1							SP1	9E6C	ff	5

Table 7-1. Instruction Set Summary (Sheet 4 of 6)

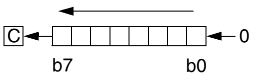
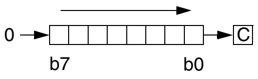
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
JMP *opr* JMP *opr* JMP *opr*,X JMP *opr*,X JMP ,X	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff 	2 3 4 3 2
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Unconditional Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff 	4 5 6 5 4
LDA #*opr* LDA *opr* LDA *opr* LDA *opr*,X LDA *opr*,X LDA ,X LDA *opr*,SP LDA *opr*,SP	Load A from M	A ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A6 B6 C6 D6 E6 F6 9EE6 9ED6	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LDHX #*opr* LDHX *opr*	Load H:X from M	H:X ← (M:M + 1)	0	–	–	↕	↕	–	IMM DIR	45 55	ii jj dd	3 4
LDX #*opr* LDX *opr* LDX *opr* LDX *opr*,X LDX *opr*,X LDX ,X LDX *opr*,SP LDX *opr*,SP	Load X from M	X ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AE BE CE DE EE FE 9EEE 9EDE	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
LSL *opr* LSLA LSLX LSL *opr*,X LSL ,X LSL *opr*,SP	Logical Shift Left (Same as ASL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
LSR *opr* LSRA LSR*X* LSR *opr*,X LSR ,X LSR *opr*,SP	Logical Shift Right		↕	–	–	0	↕	↕	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	4 1 1 4 3 5
MOV *opr,opr* MOV *opr*,X+ MOV #*opr,opr* MOV X+,*opr*	Move	$(M)_{\text{Destination}} \leftarrow (M)_{\text{Source}}$ H:X ← (H:X) + 1 (IX+D, DIX+)	0	–	–	↕	↕	–	DD DIX+ IMD IX+D	4E 5E 6E 7E	dd dd dd ii dd dd	5 4 4 4
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG *opr* NEGA NEGX NEG *opr*,X NEG ,X NEG *opr*,SP	Negate (Two's Complement)	M ← –(M) = $00 – (M) A ← –(A) = $00 – (A) X ← –(X) = $00 – (X) M ← –(M) = $00 – (M) M ← –(M) = $00 – (M)	↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	4 1 1 4 3 5
NOP	No Operation	None	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap A	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		3
ORA #*opr* ORA *opr* ORA *opr* ORA *opr*,X ORA *opr*,X ORA ,X ORA *opr*,SP ORA *opr*,SP	Inclusive OR A and M	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AA BA CA DA EA FA 9EEA 9EDA	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
PSHA	Push A onto Stack	Push (A); SP ← (SP) – 1	–	–	–	–	–	–	INH	87		2
PSHH	Push H onto Stack	Push (H); SP ← (SP) – 1	–	–	–	–	–	–	INH	8B		2
PSHX	Push X onto Stack	Push (X); SP ← (SP) – 1	–	–	–	–	–	–	INH	89		2

Table 7-1. Instruction Set Summary (Sheet 5 of 6)

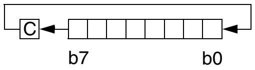
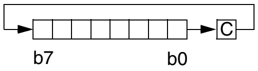
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
PULA	Pull A from Stack	SP ← (SP + 1); Pull (A)	–	–	–	–	–	–	INH	86		2
PULH	Pull H from Stack	SP ← (SP + 1); Pull (H)	–	–	–	–	–	–	INH	8A		2
PULX	Pull X from Stack	SP ← (SP + 1); Pull (X)	–	–	–	–	–	–	INH	88		2
ROL *opr* ROLA ROLX ROL *opr*,X ROL ,X ROL *opr*,SP	Rotate Left through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	4 1 1 4 3 5
ROR *opr* RORA RORX ROR *opr*,X ROR ,X ROR *opr*,SP	Rotate Right through Carry		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	4 1 1 4 3 5
RSP	Reset Stack Pointer	SP ← $FF	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		7
RTS	Return from Subroutine	SP ← SP + 1; Pull (PCH) SP ← SP + 1; Pull (PCL)	–	–	–	–	–	–	INH	81		4
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X SBC *opr*,SP SBC *opr*,SP	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A2 B2 C2 D2 E2 F2 9EE2 9ED2	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask	I ← 1	–	–	1	–	–	–	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X STA *opr*,SP STA *opr*,SP	Store A in M	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	B7 C7 D7 E7 F7 9EE7 9ED7	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
STHX *opr*	Store H:X in M	(M:M + 1) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STOP	Enable Interrupts, Stop Processing, Refer to MCU Documentation	I ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		1
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X STX *opr*,SP STX *opr*,SP	Store X in M	M ← (X)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	BF CF DF EF FF 9EEF 9EDF	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X SUB *opr*,SP SUB *opr*,SP	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A0 B0 C0 D0 E0 F0 9EE0 9ED0	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5

Table 7-1. Instruction Set Summary (Sheet 6 of 6)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
SWI	Software Interrupt	PC ← (PC) + 1; Push (PCL) SP ← (SP) – 1; Push (PCH) SP ← (SP) – 1; Push (X) SP ← (SP) – 1; Push (A) SP ← (SP) – 1; Push (CCR) SP ← (SP) – 1; I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		9
TAP	Transfer A to CCR	CCR ← (A)	↕	↕	↕	↕	↕	↕	INH	84		2
TAX	Transfer A to X	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to A	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST *opr* TSTA TSTX TST *opr*,X TST ,X TST *opr*,SP	Test for Negative or Zero	(A) – $00 or (X) – $00 or (M) – $00	0	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	3 1 1 3 2 4
TSX	Transfer SP to H:X	H:X ← (SP) + 1	–	–	–	–	–	–	INH	95		2
TXA	Transfer X to A	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer H:X to SP	(SP) ← (H:X) – 1	–	–	–	–	–	–	INH	94		2
WAIT	Enable Interrupts; Wait for Interrupt	I bit ← 0; Inhibit CPU clocking until interrupted	–	–	0	–	–	–	INH	8F		1

A	Accumulator
C	Carry/borrow bit
CCR	Condition code register
dd	Direct address of operand
dd rr	Direct address of operand and relative offset of branch instruction
DD	Direct to direct addressing mode
DIR	Direct addressing mode
DIX+	Direct to indexed with post increment addressing mode
ee ff	High and low bytes of offset in indexed, 16-bit offset addressing
EXT	Extended addressing mode
ff	Offset byte in indexed, 8-bit offset addressing
H	Half-carry bit
H	Index register high byte
hh ll	High and low bytes of operand address in extended addressing
I	Interrupt mask
ii	Immediate operand byte
IMD	Immediate source to direct destination addressing mode
IMM	Immediate addressing mode
INH	Inherent addressing mode
IX	Indexed, no offset addressing mode
IX+	Indexed, no offset, post increment addressing mode
IX+D	Indexed with post increment to direct addressing mode
IX1	Indexed, 8-bit offset addressing mode
IX1+	Indexed, 8-bit offset, post increment addressing mode
IX2	Indexed, 16-bit offset addressing mode
M	Memory location
N	Negative bit
n	Any bit
opr	Operand (one or two bytes)
PC	Program counter
PCH	Program counter high byte
PCL	Program counter low byte
REL	Relative addressing mode
rel	Relative program counter offset byte
rr	Relative program counter offset byte
SP1	Stack pointer, 8-bit offset addressing mode
SP2	Stack pointer 16-bit offset addressing mode
SP	Stack pointer
U	Undefined
V	Overflow bit
X	Index register low byte
Z	Zero bit
&	Logical AND
|	Logical OR
⊕	Logical EXCLUSIVE OR
( )	Contents of
–( )	Negation (two's complement)
#	Immediate value
«	Sign extend
←	Loaded with
?	If
:	Concatenated with
↕	Set or cleared
—	Not affected

7.8 Opcode Map

See Table 7-2.

Table 7-2. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write						Control		Register/Memory							
	DIR	DIR	REL	DIR	INH	INH	IX1	SP1	IX	INH	INH	IMM	DIR	EXT	IX2	SP2	IX1	SP1	IX
MSB LSB	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**9E6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**9ED**	**E**	**9EE**	**F**
0	5 BRSET0 3 DIR	4 BSET0 2 DIR	3 BRA 2 REL	4 NEG 2 DIR	1 NEGA 1 INH	1 NEGX 1 INH	4 NEG 2 IX1	5 NEG 3 SP1	3 NEG 1 IX	7 RTI 1 INH	3 BGE 2 REL	2 SUB 2 IMM	3 SUB 2 DIR	4 SUB 3 EXT	4 SUB 3 IX2	5 SUB 4 SP2	3 SUB 2 IX1	4 SUB 3 SP1	2 SUB 1 IX
1	5 BRCLR0 3 DIR	4 BCLR0 2 DIR	3 BRN 2 REL	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 CBEQX 3 IMM	5 CBEQ 3 IX1+	6 CBEQ 4 SP1	4 CBEQ 2 IX+	4 RTS 1 INH	3 BLT 2 REL	2 CMP 2 IMM	3 CMP 2 DIR	4 CMP 3 EXT	4 CMP 3 IX2	5 CMP 4 SP2	3 CMP 2 IX1	4 CMP 3 SP1	2 CMP 1 IX
2	5 BRSET1 3 DIR	4 BSET1 2 DIR	3 BHI 2 REL		5 MUL 1 INH	7 DIV 1 INH	3 NSA 1 INH		2 DAA 1 INH		3 BGT 2 REL	2 SBC 2 IMM	3 SBC 2 DIR	4 SBC 3 EXT	4 SBC 3 IX2	5 SBC 4 SP2	3 SBC 2 IX1	4 SBC 3 SP1	2 SBC 1 IX
3	5 BRCLR1 3 DIR	4 BCLR1 2 DIR	3 BLS 2 REL	4 COM 2 DIR	1 COMA 1 INH	1 COMX 1 INH	4 COM 2 IX1	5 COM 3 SP1	3 COM 1 IX	9 SWI 1 INH	3 BLE 2 REL	2 CPX 2 IMM	3 CPX 2 DIR	4 CPX 3 EXT	4 CPX 3 IX2	5 CPX 4 SP2	3 CPX 2 IX1	4 CPX 3 SP1	2 CPX 1 IX
4	5 BRSET2 3 DIR	4 BSET2 2 DIR	3 BCC 2 REL	4 LSR 2 DIR	1 LSRA 1 INH	1 LSRX 1 INH	4 LSR 2 IX1	5 LSR 3 SP1	3 LSR 1 IX	2 TAP 1 INH	2 TXS 1 INH	2 AND 2 IMM	3 AND 2 DIR	4 AND 3 EXT	4 AND 3 IX2	5 AND 4 SP2	3 AND 2 IX1	4 AND 3 SP1	2 AND 1 IX
5	5 BRCLR2 3 DIR	4 BCLR2 2 DIR	3 BCS 2 REL	4 STHX 2 DIR	3 LDHX 3 IMM	4 LDHX 2 DIR	3 CPHX 3 IMM		4 CPHX 2 DIR	1 TPA 1 INH	2 TSX 1 INH	2 BIT 2 IMM	3 BIT 2 DIR	4 BIT 3 EXT	4 BIT 3 IX2	5 BIT 4 SP2	3 BIT 2 IX1	4 BIT 3 SP1	2 BIT 1 IX
6	5 BRSET3 3 DIR	4 BSET3 2 DIR	3 BNE 2 REL	4 ROR 2 DIR	1 RORA 1 INH	1 RORX 1 INH	4 ROR 2 IX1	5 ROR 3 SP1	3 ROR 1 IX	2 PULA 1 INH		2 LDA 2 IMM	3 LDA 2 DIR	4 LDA 3 EXT	4 LDA 3 IX2	5 LDA 4 SP2	3 LDA 2 IX1	4 LDA 3 SP1	2 LDA 1 IX
7	5 BRCLR3 3 DIR	4 BCLR3 2 DIR	3 BEQ 2 REL	4 ASR 2 DIR	1 ASRA 1 INH	1 ASRX 1 INH	4 ASR 2 IX1	5 ASR 3 SP1	3 ASR 1 IX	2 PSHA 1 INH	1 TAX 1 INH	2 AIS 2 IMM	3 STA 2 DIR	4 STA 3 EXT	4 STA 3 IX2	5 STA 4 SP2	3 STA 2 IX1	4 STA 3 SP1	2 STA 1 IX
8	5 BRSET4 3 DIR	4 BSET4 2 DIR	3 BHCC 2 REL	4 LSL 2 DIR	1 LSLA 1 INH	1 LSLX 1 INH	4 LSL 2 IX1	5 LSL 3 SP1	3 LSL 1 IX	2 PULX 1 INH	1 CLC 1 INH	2 EOR 2 IMM	3 EOR 2 DIR	4 EOR 3 EXT	4 EOR 3 IX2	5 EOR 4 SP2	3 EOR 2 IX1	4 EOR 3 SP1	2 EOR 1 IX
9	5 BRCLR4 3 DIR	4 BCLR4 2 DIR	3 BHCS 2 REL	4 ROL 2 DIR	1 ROLA 1 INH	1 ROLX 1 INH	4 ROL 2 IX1	5 ROL 3 SP1	3 ROL 1 IX	2 PSHX 1 INH	1 SEC 1 INH	2 ADC 2 IMM	3 ADC 2 DIR	4 ADC 3 EXT	4 ADC 3 IX2	5 ADC 4 SP2	3 ADC 2 IX1	4 ADC 3 SP1	2 ADC 1 IX
A	5 BRSET5 3 DIR	4 BSET5 2 DIR	3 BPL 2 REL	4 DEC 2 DIR	1 DECA 1 INH	1 DECX 1 INH	4 DEC 2 IX1	5 DEC 3 SP1	3 DEC 1 IX	2 PULH 1 INH	2 CLI 1 INH	2 ORA 2 IMM	3 ORA 2 DIR	4 ORA 3 EXT	4 ORA 3 IX2	5 ORA 4 SP2	3 ORA 2 IX1	4 ORA 3 SP1	2 ORA 1 IX
B	5 BRCLR5 3 DIR	4 BCLR5 2 DIR	3 BMI 2 REL	5 DBNZ 3 DIR	3 DBNZA 2 INH	3 DBNZX 2 INH	5 DBNZ 3 IX1	6 DBNZ 4 SP1	4 DBNZ 2 IX	2 PSHH 1 INH	2 SEI 1 INH	2 ADD 2 IMM	3 ADD 2 DIR	4 ADD 3 EXT	4 ADD 3 IX2	5 ADD 4 SP2	3 ADD 2 IX1	4 ADD 3 SP1	2 ADD 1 IX
C	5 BRSET6 3 DIR	4 BSET6 2 DIR	3 BMC 2 REL	4 INC 2 DIR	1 INCA 1 INH	1 INCX 1 INH	4 INC 2 IX1	5 INC 3 SP1	3 INC 1 IX	1 CLRH 1 INH	1 RSP 1 INH		2 JMP 2 DIR	3 JMP 3 EXT	4 JMP 3 IX2		3 JMP 2 IX1		2 JMP 1 IX
D	5 BRCLR6 3 DIR	4 BCLR6 2 DIR	3 BMS 2 REL	3 TST 2 DIR	1 TSTA 1 INH	1 TSTX 1 INH	3 TST 2 IX1	4 TST 3 SP1	2 TST 1 IX		1 NOP 1 INH	4 BSR 2 REL	4 JSR 2 DIR	5 JSR 3 EXT	6 JSR 3 IX2		5 JSR 2 IX1		4 JSR 1 IX
E	5 BRSET7 3 DIR	4 BSET7 2 DIR	3 BIL 2 REL		5 MOV 3 DD	4 MOV 2 DIX+	4 MOV 3 IMD		4 MOV 2 IX+D	1 STOP 1 INH	*	2 LDX 2 IMM	3 LDX 2 DIR	4 LDX 3 EXT	4 LDX 3 IX2	5 LDX 4 SP2	3 LDX 2 IX1	4 LDX 3 SP1	2 LDX 1 IX
F	5 BRCLR7 3 DIR	4 BCLR7 2 DIR	3 BIH 2 REL	3 CLR 2 DIR	1 CLRA 1 INH	1 CLRX 1 INH	3 CLR 2 IX1	4 CLR 3 SP1	2 CLR 1 IX	1 WAIT 1 INH	1 TXA 1 INH	2 AIX 2 IMM	3 STX 2 DIR	4 STX 3 EXT	4 STX 3 IX2	5 STX 4 SP2	3 STX 2 IX1	4 STX 3 SP1	2 STX 1 IX

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
DD	Direct-Direct
IX+D	Indexed-Direct
REL	Relative
IX	Indexed, No Offset
IX1	Indexed, 8-Bit Offset
IX2	Indexed, 16-Bit Offset
IMD	Immediate-Direct
DIX+	Direct-Indexed
SP1	Stack Pointer, 8-Bit Offset
SP2	Stack Pointer, 16-Bit Offset
IX+	Indexed, No Offset with Post Increment
IX1+	Indexed, 1-Byte Offset with Post Increment

*Pre-byte for stack pointer indexed instructions

	MSB LSB	0	High Byte of Opcode in Hexadecimal
Low Byte of Opcode in Hexadecimal	0	5 BRSET0 3 DIR	Cycles Opcode Mnemonic Number of Bytes / Addressing Mode

Chapter 8
External Interrupt (IRQ)

8.1 Introduction

The IRQ (external interrupt) module provides a maskable interrupt input.

8.2 Features

Features of the IRQ module include:

- A dedicated external interrupt pin ($\overline{\text{IRQ}}$)
- IRQ interrupt control bits
- Hysteresis buffer
- Programmable edge-only or edge and level interrupt sensitivity
- Automatic interrupt acknowledge
- Internal pullup resistor

8.3 Functional Description

A falling edge applied to the external interrupt pin can latch a central processor unit (CPU) interrupt request. Figure 8-1 shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ}}$ pin are latched into the IRQ latch. An interrupt latch remains set until one of the following actions occurs:

- Vector fetch — A vector fetch automatically generates an interrupt acknowledge signal that clears the latch that caused the vector fetch.
- Software clear — Software can clear an interrupt latch by writing to the appropriate acknowledge bit in the interrupt status and control register (INTSCR). Writing a 1 to the ACK bit clears the IRQ latch.
- Reset — A reset automatically clears the interrupt latch.

The external interrupt pin is falling-edge triggered out of reset and is software-configurable to be either falling-edge or falling-edge and low-level triggered. The MODE bit in the INTSCR controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin.

When an interrupt pin is edge-triggered only (MODE = 0), the interrupt remains set until a vector fetch, software clear, or reset occurs.

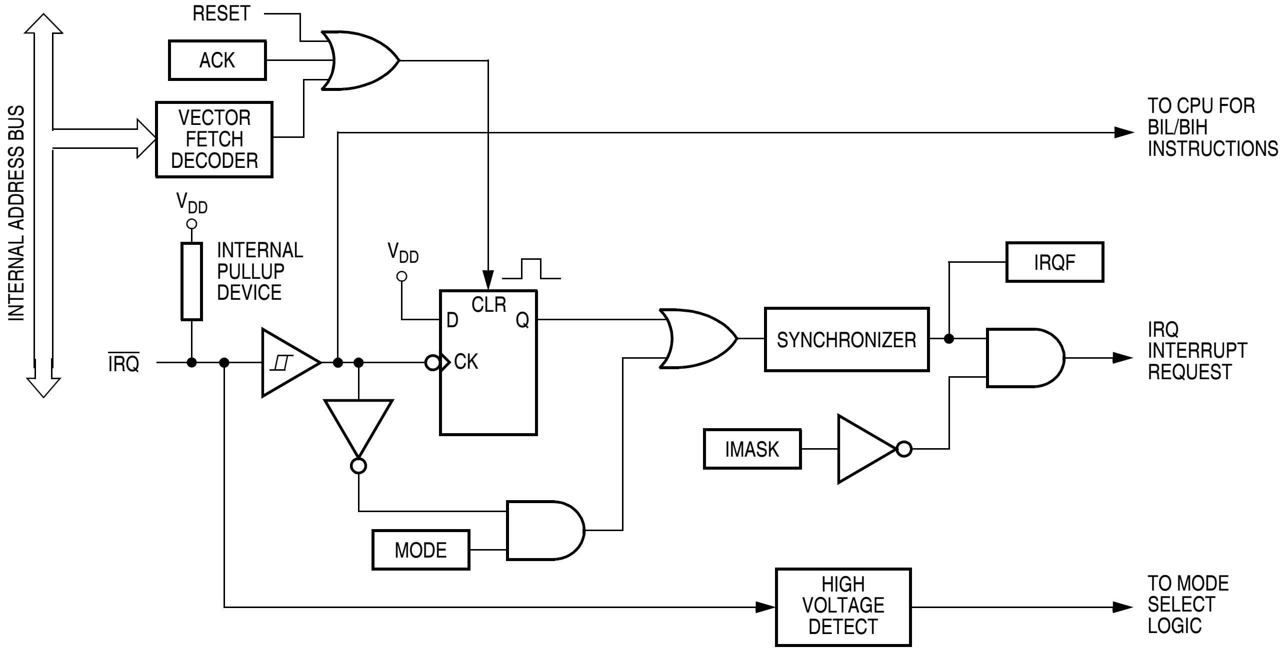


Figure 8-1. IRQ Module Block Diagram

When an interrupt pin is both falling-edge and low-level triggered (MODE = 1), the interrupt remains set until both of these events occur:

- Vector fetch or software clear
- Return of the interrupt pin to a high level

The vector fetch or software clear may occur before or after the interrupt pin returns to a high level. As long as the pin is low, the interrupt request remains pending. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

When set, the IMASK bit in the INTSCR masks all external interrupt requests. A latched interrupt request is not presented to the interrupt priority logic unless the IMASK bit is clear.

NOTE

The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including external interrupt requests.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001D	IRQ Status and Control Register (INTSCR) See page 96.	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:	▨	▨	▨	▨	▨	ACK		
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 8-2. IRQ I/O Register Summary

8.4 $\overline{\text{IRQ}}$ Pin

A falling edge on the $\overline{\text{IRQ}}$ pin can latch an interrupt request into the IRQ latch. A vector fetch, software clear, or reset clears the IRQ latch.

If the MODE bit is set, the $\overline{\text{IRQ}}$ pin is both falling-edge-sensitive and low-level sensitive. With MODE set, both of the following actions must occur to clear IRQ:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the latch. Software may generate the interrupt acknowledge signal by writing a 1 to the ACK bit in the interrupt status and control register (INTSCR). The ACK bit is useful in applications that poll the $\overline{\text{IRQ}}$ pin and require software to clear the IRQ latch. Writing to the ACK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACK does not affect subsequent transitions on the $\overline{\text{IRQ}}$ pin. A falling edge that occurs after writing to the ACK bit latches another interrupt request. If the IRQ mask bit, IMASK, is clear, the CPU loads the program counter with the vector address at locations $FFFA and $FFFB.
- Return of the $\overline{\text{IRQ}}$ pin to a high level — As long as the $\overline{\text{IRQ}}$ pin is low, IRQ remains active.

The vector fetch or software clear and the return of the $\overline{\text{IRQ}}$ pin to a high level may occur in any order. The interrupt request remains pending as long as the $\overline{\text{IRQ}}$ pin is low. A reset will clear the latch and the MODE control bit, thereby clearing the interrupt even if the pin stays low.

If the MODE bit is clear, the $\overline{\text{IRQ}}$ pin is falling-edge-sensitive only. With MODE clear, a vector fetch or software clear immediately clears the IRQ latch.

The IRQF bit in the INTSCR register can be used to check for pending interrupts. The IRQF bit is not affected by the IMASK bit, which makes it useful in applications where polling is preferred.

Use the BIH or BIL instruction to read the logic level on the $\overline{\text{IRQ}}$ pin.

NOTE

When using the level-sensitive interrupt trigger, avoid false interrupts by masking interrupt requests in the interrupt routine.

8.5 IRQ Module During Break Interrupts

The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear the latch during the break state. See Chapter 18 Development Support.

To allow software to clear the IRQ latch during a break interrupt, write a 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect CPU interrupt flags during the break state, write a 0 to the BCFE bit. With BCFE at 0 (its default state), writing to the ACK bit in the IRQ status and control register during the break state has no effect on the IRQ interrupt flags.

8.6 IRQ Status and Control Register

The IRQ status and control register (INTSCR) controls and monitors operation of the IRQ module. The INTSCR:

- Shows the state of the IRQ flag
- Clears the IRQ latch
- Masks IRQ interrupt request
- Controls triggering sensitivity of the $\overline{\text{IRQ}}$ interrupt pin

Address: $001D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	IRQF	0	IMASK	MODE
Write:	▨	▨	▨	▨	▨	ACK		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 8-3. IRQ Status and Control Register (INTSCR)

IRQF — IRQ Flag Bit

This read-only status bit is high when the IRQ interrupt is pending.

1 = $\overline{\text{IRQ}}$ interrupt pending
0 = $\overline{\text{IRQ}}$ interrupt not pending

ACK — IRQ Interrupt Request Acknowledge Bit

Writing a 1 to this write-only bit clears the IRQ latch. ACK always reads as 0. Reset clears ACK.

IMASK — IRQ Interrupt Mask Bit

Writing a 1 to this read/write bit disables IRQ interrupt requests. Reset clears IMASK.

1 = IRQ interrupt requests disabled
0 = IRQ interrupt requests enabled

MODE — IRQ Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin. Reset clears MODE.

1 = $\overline{\text{IRQ}}$ interrupt requests on falling edges and low levels
0 = $\overline{\text{IRQ}}$ interrupt requests on falling edges only

Chapter 9
Keyboard Interrupt Module (KBI)

9.1 Introduction

The keyboard interrupt module (KBI) provides four independently maskable external interrupts which are accessible via PTA0–PTA3. When a port pin is enabled for keyboard interrupt function, an internal pullup device is also enabled on the pin.

9.2 Features

Features include:

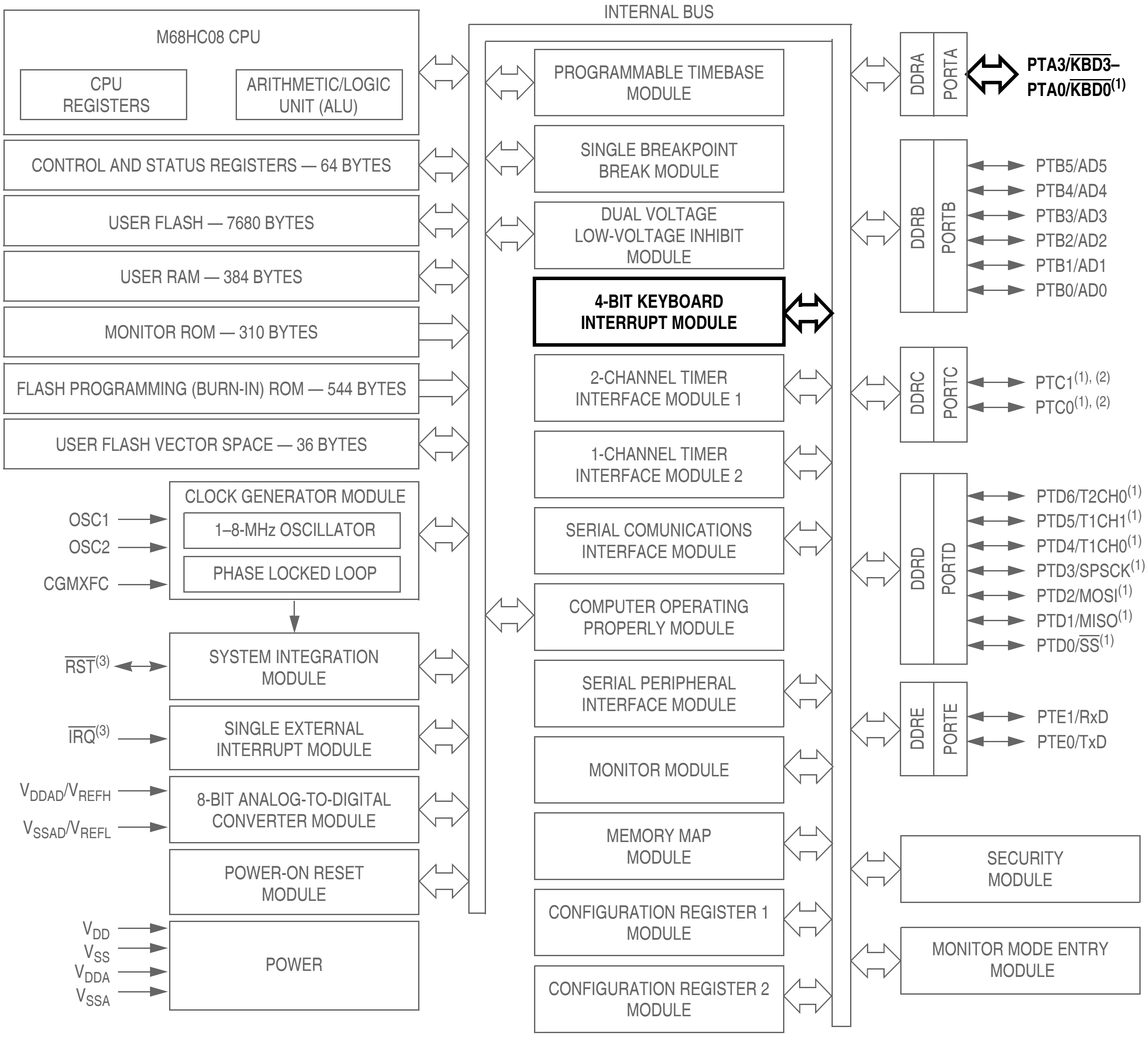
- Four keyboard interrupt pins with separate keyboard interrupt enable bits and one keyboard interrupt mask
- Hysteresis buffers
- Programmable edge-only or edge- and level- interrupt sensitivity
- Exit from low-power modes
- I/O (input/output) port bit(s) software configurable with pullup device(s) if configured as input port bit(s)

9.3 Functional Description

Writing to the KBIE3–KBIE0 bits in the keyboard interrupt enable register independently enables or disables each port A pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin also enables its internal pullup device. A low level applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

A keyboard interrupt is latched when one or more keyboard pins goes low after all were high. The MODEK bit in the keyboard status and control register controls the triggering mode of the keyboard interrupt.

- If the keyboard interrupt is edge-sensitive only, a falling edge on a keyboard pin does not latch an interrupt request if another keyboard pin is already low. To prevent losing an interrupt request on one pin because another pin is still low, software can disable the latter pin while it is low.
- If the keyboard interrupt is falling edge- and low-level sensitive, an interrupt request is present as long as any keyboard interrupt pin is low and the pin is keyboard interrupt enabled.



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 9-1. Block Diagram Highlighting KBI Block and Pins

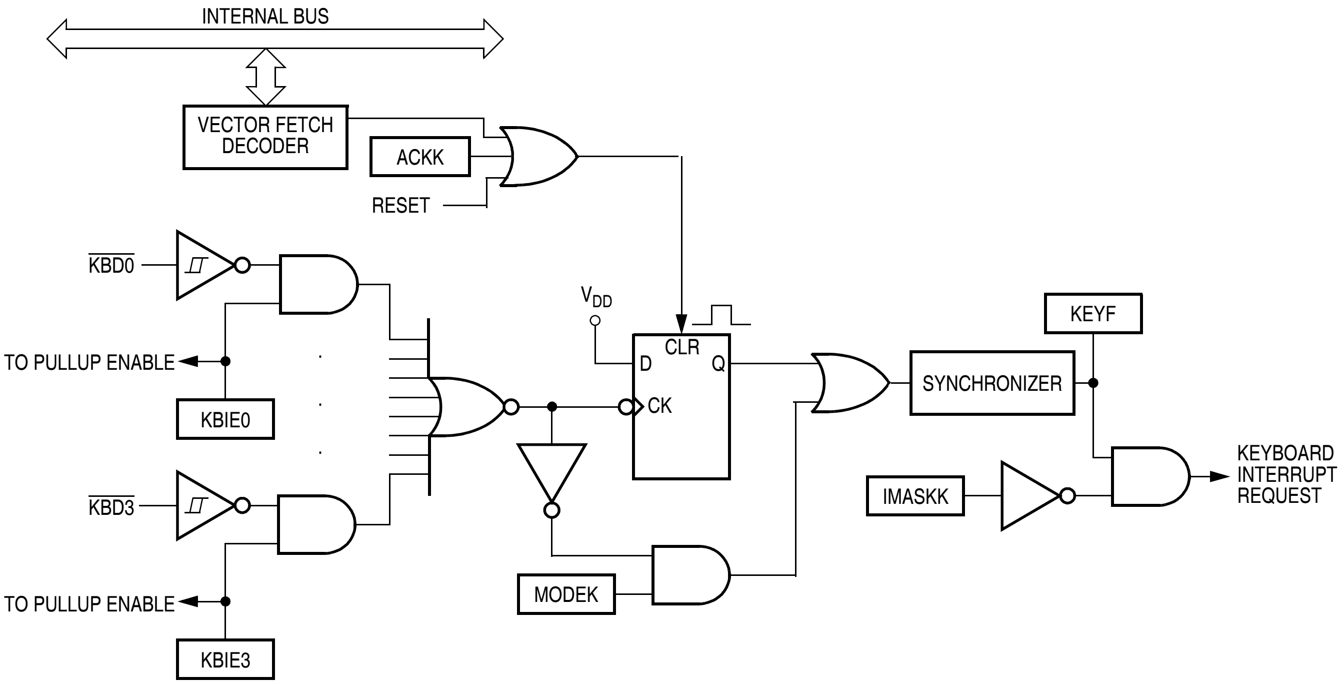


Figure 9-2. Keyboard Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001A	Keyboard Status and Control Register (INTKBSCR) See page 101.	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
		Write:	▨	▨	▨	▨	▨	ACKK		
		Reset:	0	0	0	0	0	0	0	0
$001B	Keyboard Interrupt Enable Register (INTKBIER) See page 102.	Read:	▨	▨	▨	▨	KBIE3	KBIE2	KBIE1	KBIE0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-3. I/O Register Summary

If the MODEK bit is set, the keyboard interrupt pins are both falling edge- and low-level sensitive, and both of the following actions must occur to clear a keyboard interrupt request:

- Vector fetch or software clear — A vector fetch generates an interrupt acknowledge signal to clear the interrupt request. Software may generate the interrupt acknowledge signal by writing a 1 to the ACKK bit in the keyboard status and control register (INTKBSCR). The ACKK bit is useful in applications that poll the keyboard interrupt pins and require software to clear the keyboard interrupt request. Writing to the ACKK bit prior to leaving an interrupt service routine can also prevent spurious interrupts due to noise. Setting ACKK does not affect subsequent transitions on the keyboard interrupt pins. A falling edge that occurs after writing to the ACKK bit latches another interrupt request. If the keyboard interrupt mask bit, IMASKK, is clear, the CPU loads the program counter with the vector address at locations $FFE0 and $FFE1.
- Return of all enabled keyboard interrupt pins to a high level — As long as any enabled keyboard interrupt pin is low, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to a high level may occur in any order.

If the MODEK bit is clear, the keyboard interrupt pin is falling-edge-sensitive only. With MODEK clear, a vector fetch or software clear immediately clears the keyboard interrupt request.

Reset clears the keyboard interrupt request and the MODEK bit, clearing the interrupt request even if a keyboard interrupt pin stays at low.

The keyboard flag bit (KEYF) in the keyboard status and control register can be used to see if a pending interrupt exists. The KEYF bit is not affected by the keyboard interrupt mask bit (IMASKK) which makes it useful in applications where polling is preferred.

To determine the logic level on a keyboard interrupt pin, use the data direction register to configure the pin as an input and read the data register.

NOTE

Setting a keyboard interrupt enable bit (KBIEx) forces the corresponding keyboard interrupt pin to be an input, overriding the data direction register. However, the data direction register bit must be a 0 for software to read the pin.

9.4 Keyboard Initialization

When a keyboard interrupt pin is enabled, it takes time for the internal pullup to reach a 1. Therefore, a false interrupt can occur as soon as the pin is enabled.

To prevent a false interrupt on keyboard initialization:

1. Mask keyboard interrupts by setting the IMASKK bit in the keyboard status and control register.
2. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.
3. Write to the ACKK bit in the keyboard status and control register to clear any false interrupts.
4. Clear the IMASKK bit.

An interrupt signal on an edge-triggered pin can be acknowledged immediately after enabling the pin. An interrupt signal on an edge- and level-triggered interrupt pin must be acknowledged after a delay that depends on the external load.

Another way to avoid a false interrupt:

1. Configure the keyboard pins as outputs by setting the appropriate DDRA bits in data direction register A.
2. Write 1s to the appropriate port A data register bits.
3. Enable the KBI pins by setting the appropriate KBIEx bits in the keyboard interrupt enable register.

9.5 Low-Power Modes

The WAIT and STOP instructions put the microcontroller unit (MCU) in low power-consumption standby modes.

9.5.1 Wait Mode

The keyboard module remains active in wait mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of wait mode.

9.5.2 Stop Mode

The keyboard module remains active in stop mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of stop mode.

9.6 Keyboard Module During Break Interrupts

The system integration module (SIM) controls whether the keyboard interrupt latch can be cleared during the break state. The BCFE bit in the break flag control register (SBFCR) enables software to clear status bits during the break state.

To allow software to clear the keyboard interrupt latch during a break interrupt, write a 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latch during the break state, write a 0 to the BCFE bit. With BCFE at 0 (its default state), writing to the keyboard acknowledge bit (ACKK) in the keyboard status and control register during the break state has no effect. See 9.7.1 Keyboard Status and Control Register.

9.7 I/O Registers

These registers control and monitor operation of the keyboard module:

- Keyboard status and control register (INTKBSCR)
- Keyboard interrupt enable register (INTKBIER)

9.7.1 Keyboard Status and Control Register

The keyboard status and control register:

- Flags keyboard interrupt requests
- Acknowledges keyboard interrupt requests
- Masks keyboard interrupt requests
- Controls keyboard interrupt triggering sensitivity

Address: $001A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
Write:	▨	▨	▨	▨	▨	ACKK		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-4. Keyboard Status and Control Register (INTKBSCR)

Bits 7–4 — Not used

These read-only bits always read as 0s.

KEYF — Keyboard Flag Bit

This read-only bit is set when a keyboard interrupt is pending. Reset clears the KEYF bit.

1 = Keyboard interrupt pending
0 = No keyboard interrupt pending

ACKK — Keyboard Acknowledge Bit

Writing a 1 to this write-only bit clears the keyboard interrupt request. ACKK always reads as 0. Reset clears ACKK.

IMASKK — Keyboard Interrupt Mask Bit

Writing a 1 to this read/write bit prevents the output of the keyboard interrupt mask from generating interrupt requests. Reset clears the IMASKK bit.

1 = Keyboard interrupt requests masked
0 = Keyboard interrupt requests not masked

MODEK — Keyboard Triggering Sensitivity Bit

This read/write bit controls the triggering sensitivity of the keyboard interrupt pins. Reset clears MODEK.

1 = Keyboard interrupt requests on falling edges and low levels
0 = Keyboard interrupt requests on falling edges only

9.7.2 Keyboard Interrupt Enable Register

The keyboard interrupt enable register enables or disables each port A pin to operate as a keyboard interrupt pin.

Address: $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	▨	▨	▨	▨	KBIE3	KBIE2	KBIE1	KBIE0
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-5. Keyboard Interrupt Enable Register (INTKBIER)

KBIE3–KBIE0 — Keyboard Interrupt Enable Bits

Each of these read/write bits enables the corresponding keyboard interrupt pin to latch interrupt requests. Reset clears the keyboard interrupt enable register.

1 = PTAx pin enabled as keyboard interrupt pin
0 = PTAx pin not enabled as keyboard interrupt pin

Chapter 10
Low-Power Modes

10.1 Introduction

The microcontroller (MCU) may enter two low-power modes: wait mode and stop mode. They are common to all HC08 MCUs and are entered through instruction execution. This section describes how each module acts in the low-power modes.

10.1.1 Wait Mode

The WAIT instruction puts the MCU in a low-power standby mode in which the central processor unit (CPU) clock is disabled but the bus clock continues to run. Power consumption can be further reduced by disabling the low-voltage inhibit (LVI) module through bits in the CONFIG1 register. See Chapter 5 Configuration Register (CONFIG).

10.1.2 Stop Mode

Stop mode is entered when a STOP instruction is executed. The CPU clock is disabled and the bus clock is disabled if the OSCSTOPENB bit in the CONFIG2 register is a 0. See Chapter 5 Configuration Register (CONFIG).

10.2 Analog-to-Digital Converter (ADC)

10.2.1 Wait Mode

The analog-to-digital converter (ADC) continues normal operation during wait mode. Any enabled CPU interrupt request from the ADC can bring the MCU out of wait mode. If the ADC is not required to bring the MCU out of wait mode, power down the ADC by setting ADCH4–ADCH0 bits in the ADC status and control register before executing the WAIT instruction.

10.2.2 Stop Mode

The ADC module is inactive after the execution of a STOP instruction. Any pending conversion is aborted. ADC conversions resume when the MCU exits stop mode after an external interrupt. Allow one conversion cycle to stabilize the analog circuitry.

10.3 Break Module (BRK)

10.3.1 Wait Mode

The break (BRK) module is active in wait mode. In the break routine, the user can subtract one from the return address on the stack if the SBSW bit in the break status register is set.

10.3.2 Stop Mode

The break module is inactive in stop mode. The STOP instruction does not affect break module register states.

10.4 Central Processor Unit (CPU)

10.4.1 Wait Mode

The WAIT instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling interrupts. After exit from wait mode by interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

10.4.2 Stop Mode

The STOP instruction:

- Clears the interrupt mask (I bit) in the condition code register, enabling external interrupts. After exit from stop mode by external interrupt, the I bit remains clear. After exit by reset, the I bit is set.
- Disables the CPU clock

After exiting stop mode, the CPU clock begins running after the oscillator stabilization delay.

10.5 Clock Generator Module (CGM)

10.5.1 Wait Mode

The clock generator module (CGM) remains active in wait mode. Before entering wait mode, software can disengage and turn off the PLL by clearing the BCS and PLLON bits in the PLL control register (PCTL). Less power-sensitive applications can disengage the PLL without turning it off. Applications that require the PLL to wake the MCU from wait mode also can deselect the PLL output without turning off the PLL.

10.5.2 Stop Mode

If the OSCSTOPENB bit in the CONFIG2 register is cleared (default), then the STOP instruction disables the CGM (oscillator and phase-locked loop) and holds low all CGM outputs (CGMXCLK, CGMOUT, and CGMINT).

If the OSCSTOPENB bit in the CONFIG2 register is set, then the phase locked loop is shut off, but the oscillator will continue to operate in stop mode.

10.6 Computer Operating Properly Module (COP)

10.6.1 Wait Mode

The COP remains active during wait mode. If COP is enabled, a reset will occur at COP timeout.

10.6.2 Stop Mode

Stop mode turns off the CGMXCLK input to the COP and clears the SIM counter. Service the COP immediately before entering or after exiting stop mode to ensure a full COP timeout period after entering or exiting stop mode.

The STOP bit in the CONFIG1 register enables the STOP instruction. To prevent inadvertently turning off the COP with a STOP instruction, disable the STOP instruction by clearing the STOP bit.

10.7 External Interrupt Module (IRQ)

10.7.1 Wait Mode

The external interrupt (IRQ) module remains active in wait mode. Clearing the IMASK bit in the IRQ status and control register enables $\overline{\text{IRQ}}$ CPU interrupt requests to bring the MCU out of wait mode.

10.7.2 Stop Mode

The IRQ module remains active in stop mode. Clearing the IMASK bit in the IRQ status and control register enables $\overline{\text{IRQ}}$ CPU interrupt requests to bring the MCU out of stop mode.

10.8 Keyboard Interrupt Module (KBI)

10.8.1 Wait Mode

The keyboard interrupt (KBI) module remains active in wait mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of wait mode.

10.8.2 Stop Mode

The keyboard module remains active in stop mode. Clearing the IMASKK bit in the keyboard status and control register enables keyboard interrupt requests to bring the MCU out of stop mode.

10.9 Low-Voltage Inhibit Module (LVI)

10.9.1 Wait Mode

If enabled, the low-voltage inhibit (LVI) module remains active in wait mode. If enabled to generate resets, the LVI module can generate a reset and bring the MCU out of wait mode.

10.9.2 Stop Mode

If enabled, the LVI module remains active in stop mode. If enabled to generate resets, the LVI module can generate a reset and bring the MCU out of stop mode.

10.10 Serial Communications Interface Module (SCI)

10.10.1 Wait Mode

The serial communications interface (SCI) module remains active in wait mode. Any enabled CPU interrupt request from the SCI module can bring the MCU out of wait mode.

If SCI module functions are not required during wait mode, reduce power consumption by disabling the module before executing the WAIT instruction.

10.10.2 Stop Mode

The SCI module is inactive in stop mode. The STOP instruction does not affect SCI register states. SCI module operation resumes after the MCU exits stop mode.

Because the internal clock is inactive during stop mode, entering stop mode during an SCI transmission or reception results in invalid data.

10.11 Serial Peripheral Interface Module (SPI)

10.11.1 Wait Mode

The serial peripheral interface (SPI) module remains active in wait mode. Any enabled CPU interrupt request from the SPI module can bring the MCU out of wait mode.

If SPI module functions are not required during wait mode, reduce power consumption by disabling the SPI module before executing the WAIT instruction.

10.11.2 Stop Mode

The SPI module is inactive in stop mode. The STOP instruction does not affect SPI register states. SPI operation resumes after an external interrupt. If stop mode is exited by reset, any transfer in progress is aborted, and the SPI is reset.

10.12 Timer Interface Module (TIM1 and TIM2)

10.12.1 Wait Mode

The timer interface modules (TIM) remain active in wait mode. Any enabled CPU interrupt request from the TIM can bring the MCU out of wait mode.

If TIM functions are not required during wait mode, reduce power consumption by stopping the TIM before executing the WAIT instruction.

10.12.2 Stop Mode

The TIM is inactive in stop mode. The STOP instruction does not affect register states or the state of the TIM counter. TIM operation resumes when the MCU exits stop mode after an external interrupt.

10.13 Timebase Module (TBM)

10.13.1 Wait Mode

The timebase module (TBM) remains active after execution of the WAIT instruction. In wait mode, the timebase register is not accessible by the CPU.

If the timebase functions are not required during wait mode, reduce the power consumption by stopping the timebase before enabling the WAIT instruction.

10.13.2 Stop Mode

The timebase module may remain active after execution of the STOP instruction if the oscillator has been enabled to operate during stop mode through the OSCSTOPENB bit in the CONFIG2 register. The timebase module can be used in this mode to generate a periodic wakeup from stop mode.

If the oscillator has not been enabled to operate in stop mode, the timebase module will not be active during stop mode. In stop mode, the timebase register is not accessible by the CPU.

If the timebase functions are not required during stop mode, reduce the power consumption by stopping the timebase before enabling the STOP instruction.

10.14 Exiting Stop Mode

These events restart the system clocks and load the program counter with the reset vector or with an interrupt vector:

- External reset — A 0 on the $\overline{\text{RST}}$ pin resets the MCU and loads the program counter with the contents of locations $FFFE and $FFFF.
- External interrupt — A high-to-low transition on an external interrupt pin loads the program counter with the contents of locations:
  - $FFFA and $FFFB; $\overline{\text{IRQ}}$ pin
  - $FFE0 and $FFE1; keyboard interrupt pins
- Low-voltage inhibit (LVI) reset — A power supply voltage below the V_{TRIPF} voltage resets the MCU and loads the program counter with the contents of locations $FFFE and $FFFF.

- Timebase module (TBM) interrupt — A TBM interrupt loads the program counter with the contents of locations $FFDC and $FFDD when the timebase counter has rolled over. This allows the TBM to generate a periodic wakeup from stop mode.

Upon exit from stop mode, the system clocks begin running after an oscillator stabilization delay. A 12-bit stop recovery counter inhibits the system clocks for 4096 CGMXCLK cycles after the reset or external interrupt.

The short stop recovery bit, SSREC, in the CONFIG1 register controls the oscillator stabilization delay during stop recovery. Setting SSREC reduces stop recovery time from 4096 CGMXCLK cycles to 32 CGMXCLK cycles.

NOTE

Use the full stop recovery time (SSREC = 0) in applications that use an external crystal unless the OSCSTOPENB bit is set.

Chapter 11
Low-Voltage Inhibit (LVI)

11.1 Introduction

This section describes the low-voltage inhibit (LVI) module, which monitors the voltage on the V_{DD} pin and will force a reset when the V_{DD} voltage falls below the LVI trip falling voltage, V_{TRIPF}.

11.2 Features

Features of the LVI module include:

- LVI reset
- Stop mode operation

11.3 Functional Description

Figure 11-1 shows the structure of the LVI module. The LVI is enabled out of reset. The LVI module contains a bandgap reference circuit and comparator. The LVI power disable bit, LVIPWRD, is always clear to enable the LVI to monitor V_{DD} voltage. The LVI reset disable bit, LVIRSTD, is always clear to enable the LVI module to generate a reset when V_{DD} falls below a voltage, V_{TRIPF}. The LVI enable in stop mode bit, LVISTOP, is always clear to enable the LVI to operate in stop mode. The LVI 5-V or 3-V trip point bit, LVI5OR3, is always clear to enable the trip point voltage, V_{TRIPF}, to be configured for 3-V operation. The actual trip points are shown in Chapter 19 Electrical Specifications.

NOTE

After a power-on reset (POR) the LVI's default mode of operation is 3 V.

LVISTOP, LVIPWRD, LVI5OR3, and LVIRSTD are in the configuration register (CONFIG1). See Figure 5-2. Configuration Register 1 (CONFIG1) for details of the LVI's configuration bits. Once an LVI reset occurs, the MCU remains in reset until V_{DD} rises above a voltage, V_{TRIPR}, which causes the MCU to exit reset. See 14.3.2.5 Low-Voltage Inhibit (LVI) Reset for details of the interaction between the SIM and the LVI. The output of the comparator controls the state of the LVIOUT flag in the LVI status register (LVISR).

An LVI reset also drives the $\overline{\text{RST}}$ pin low to provide low-voltage protection to external peripheral devices.

11.3.1 Polled LVI Operation

Polled LVI operation is not available on the MC68HC908GR8B.

11.3.2 Forced Reset Operation

In applications that require V_{DD} to remain above the V_{TRIPF} level, LVI resets allow the LVI module to reset the MCU when V_{DD} falls below the V_{TRIPF} level. The LVIPWRD and LVIRSTD bits are always clear to enable the LVI module and to enable LVI resets.

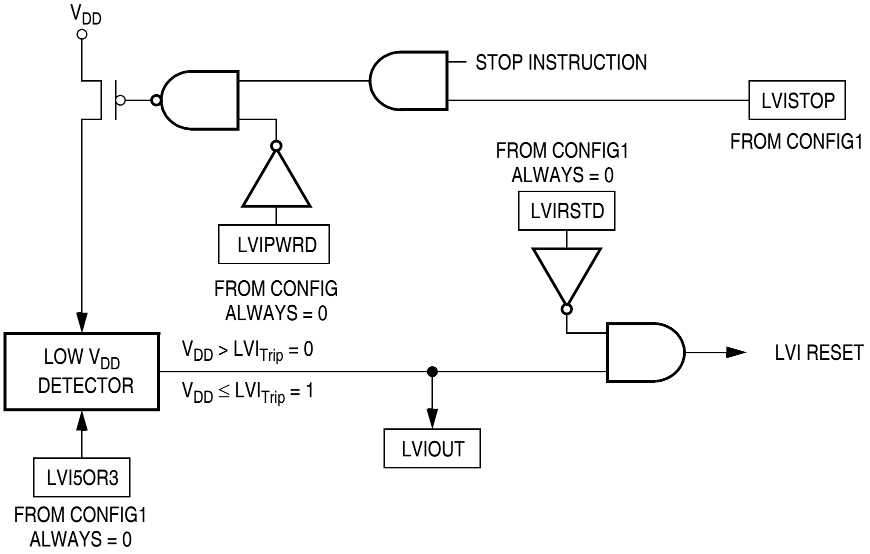


Figure 11-1. LVI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0C	LVI Status Register (LVISR) See page 111.	Read:	LVIOUT	0	0	0	0	0	0	0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 11-2. LVI I/O Register Summary

11.3.3 Voltage Hysteresis Protection

Once the LVI has triggered (by having V_{DD} fall below V_{TRIPF}), the LVI will maintain a reset condition until V_{DD} rises above the rising trip point voltage, V_{TRIPR}. This prevents a condition in which the MCU is continually entering and exiting reset if V_{DD} is approximately equal to V_{TRIPF}. V_{TRIPR} is greater than V_{TRIPF} by the hysteresis voltage, V_{HYS}.

11.3.4 LVI Trip Selection

The LVI5OR3 bit in the configuration register selects the LVI to be configured for 3-V protection.

NOTE

The microcontroller is guaranteed to operate at a minimum supply voltage. The trip point (V_{TRIPF} [3 V]) may be lower than this. See Chapter 19 Electrical Specifications for the actual trip point voltages.

11.4 LVI Status Register

The LVI status register (LVISR) indicates if the V_{DD} voltage was detected below the V_{TRIPF} level.

Address: $FE0C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	LVIOUT	0	0	0	0	0	0	0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 11-3. LVI Status Register (LVISR)

LVIOUT — LVI Output Bit

This read-only flag becomes set when the V_{DD} voltage falls below the V_{TRIPF} trip voltage (see Table 11-1). Reset clears the LVIOUT bit.

Table 11-1. LVIOUT Bit Indication

V_{DD}	LVIOUT
$V_{DD} > V_{TRIPR}$	0
$V_{DD} < V_{TRIPF}$	1
$V_{TRIPF} < V_{DD} < V_{TRIPR}$	Previous value

11.5 LVI Interrupts

The LVI module does not generate interrupt requests.

11.6 Low-Power Modes

The STOP and WAIT instructions put the MCU in low power-consumption standby modes.

11.6.1 Wait Mode

The LVI module remains active in wait mode. The LVI module will generate a reset and bring the MCU out of wait mode.

11.6.2 Stop Mode

If enabled in stop mode (LVISTOP bit in the configuration register is set), the LVI module remains active in stop mode. If enabled to generate resets, the LVI module can generate a reset and bring the MCU out of stop mode.

Chapter 12
Input/Output (I/O) Ports

12.1 Introduction

Bidirectional input-output (I/O) pins form five parallel ports. All I/O pins are programmable as inputs or outputs. All individual bits within port A, port C, and port D are software configurable with pullup devices if configured as input port bits. The pullup devices are automatically and dynamically disabled when a port bit is switched to output mode.

NOTE

Connect any unused I/O pins to an appropriate logic level, either V_{DD} *or* V_{SS}*. Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.*

Not all port pins are bonded out in all packages. Care sure be taken to make any unbonded port pins an output to prevent them from being floating inputs.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA) See page 115.	Read:	0	0	0	0	PTA3	PTA2	PTA1	PTA0
		Write:	▨	▨	▨	▨				
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB) See page 118.	Read:	0	0	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Write:	▨	▨						
		Reset:	Unaffected by reset							
$0002	Port C Data Register (PTC) See page 120.	Read:	0	0	0	0	0	0	PTC1	PTC0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	Unaffected by reset							
$0003	Port D Data Register (PTD) See page 122.	Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Write:	▨							
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA) See page 116.	Read:	0	0	0	0	DDRA3	DDRA2	DDRA1	DDRA0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-1. I/O Port Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0005	Data Direction Register B (DDRB) See page 118.	Read:	0	0	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Write:	▨	▨						
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC) See page 120.	Read:	0	0	0	0	0	0	DDRC1	DDRC0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD) See page 123.	Read:	0	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Write:	▨							
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE) See page 125.	Read:	0	0	0	0	0	0	PTE1	PTE0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	Unaffected by reset							
$000C	Data Direction Register E (DDRE) See page 126.	Read:	0	0	0	0	0	0	DDRE1	DDRE0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	0	0	0	0	0	0	0	0
$000D	Port A Input Pullup Enable Register (PTAPUE) See page 117.	Read:	0	0	0	0	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$000E	Port C Input Pullup Enable Register (PTCPUE) See page 122.	Read:	0	0	0	0	0	0	PTCPUE1	PTCPUE0
		Write:	▨	▨	▨	▨	▨	▨		
		Reset:	0	0	0	0	0	0	0	0
$000F	Port D Input Pullup Enable Register (PTDPUE) See page 125.	Read:	0	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
		Write:	▨							
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-1. I/O Port Register Summary (Continued)

Table 12-1. Port Control Register Bits Summary

Port	Bit	DDR	Module Control		Pin
A	0	DDRA0	KBD	KBIE0	PTA0/KBD0
	1	DDRA1		KBIE1	PTA1/KBD1
	2	DDRA2		KBIE2	PTA2/KBD2
	3	DDRA3		KBIE3	PTA3/KBD3
B	0	DDRB0	ADC	ADCH4:0	PTB0/AD0
	1	DDRB1		ADCH4:0	PTB1/AD1
	2	DDRB2		ADCH4:0	PTB2/AD2
	3	DDRB3		ADCH4:0	PTB3/AD3
	4	DDRB4		ADCH4:0	PTB4/AD4
	5	DDRB5		ADCH4:0	PTB5/AD5
C	0	DDRC0			PTC0
	1	DDRC1			PTC1
D	0	DDRD0	SPI	SPE	PTD0/$\overline{SS}$
	1	DDRD1			PTD1/MISO
	2	DDRD2			PTD2/MOSI
	3	DDRD3			PTD3/SPSCK
	4	DDRD4	TIM1	ELS0B:ELS0A	PTD4/T1CH0
	5	DDRD5		ELS1B:ELS1A	PTD5/T1CH1
	6	DDRD6	TIM2	ELS0B:ELS0A	PTD6/T2CH0
E	0	DDRE0	SCI	ENSCI	PTE0/TxD
	1	DDRE1			PTE1/RxD

12.2 Port A

Port A is an 4-bit special-function port that shares all four of its pins with the keyboard interrupt (KBI) module. Port A also has software configurable pullup devices if configured as an input port.

12.2.1 Port A Data Register

The port A data register (PTA) contains a data latch for each of the four port A pins.

Address: $0000

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	PTA3	PTA2	PTA1	PTA0
Write:	▨	▨	▨	▨				
Reset:	Unaffected by reset							
Alternative Function:	▨	▨	▨	▨	KBD3	KBD2	KBD1	KBD0

▨ = Unimplemented

Figure 12-2. Port A Data Register (PTA)

PTA3–PTA0 — Port A Data Bits

These read/write bits are software programmable. Data direction of each port A pin is under the control of the corresponding bit in data direction register A. Reset has no effect on port A data.

KBD3–KBD0 — Keyboard Inputs

The keyboard interrupt enable bits, KBIE3–KBIE0, in the keyboard interrupt control register (KBICR) enable the port A pins as external interrupt pins. See Chapter 9 Keyboard Interrupt Module (KBI).

12.2.2 Data Direction Register A

Data direction register A (DDRA) determines whether each port A pin is an input or an output. Writing a 1 to a DDRA bit enables the output buffer for the corresponding port A pin; a 0 disables the output buffer.

Address: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	DDRA3	DDRA2	DDRA1	DDRA0
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-3. Data Direction Register A (DDRA)

DDRA3–DDRA0 — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA3–DDRA0, configuring all port A pins as inputs.

1 = Corresponding port A pin configured as output
0 = Corresponding port A pin configured as input

NOTE

Avoid glitches on port A pins by writing to the port A data register before changing data direction register A bits from 0 to 1.

Figure 12-4 shows the port A I/O logic.

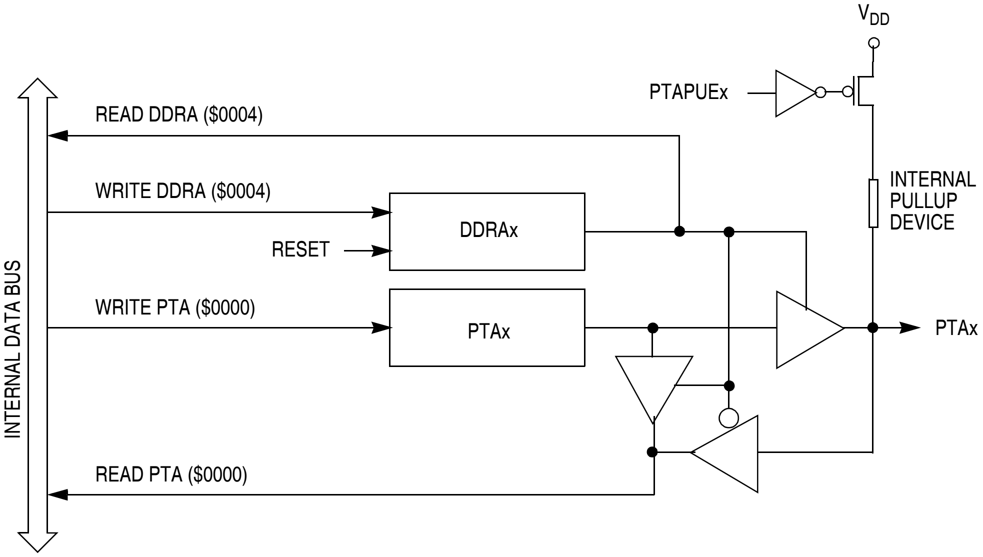


Figure 12-4. Port A I/O Circuit

When bit DDRAx is a 1, reading address $0000 reads the PTAx data latch. When bit DDRAx is a 0, reading address $0000 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-2 summarizes the operation of the port A pins.

Table 12-2. Port A Pin Functions

PTAPUE Bit	DDRA Bit	PTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PTA	
				Read/Write	Read	Write
1	0	X(1)	Input, V_{DD}(2)	DDRA3–DDRA0	Pin	PTA3–PTA0(3)
0	0	X	Input, Hi-Z(4)	DDRA3–DDRA0	Pin	PTA3–PTA0(3)
X	1	X	Output	DDRA3–DDRA0	PTA3–PTA0	PTA3–PTA0

1. X = Don’t care
2. I/O pin pulled up to V_{DD} by internal pullup device
3. Writing affects data register, but does not affect input.
4. Hi-Z = High impedance

12.2.3 Port A Input Pullup Enable Register

The port A input pullup enable register (PTAPUE) contains a software configurable pullup device for each of the four port A pins. Each bit is individually configurable and requires that the data direction register, DDRA, bit be configured as an input. Each pullup is automatically and dynamically disabled when a port bit’s DDRA is configured for output mode.

Address: $000D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-5. Port A Input Pullup Enable Register (PTAPUE)

PTAPUE3–PTAPUE0 — Port A Input Pullup Enable Bits

These writeable bits are software programmable to enable pullup devices on an input port bit.

1 = Corresponding port A pin configured to have internal pullup

0 = Corresponding port A pin has internal pullup disconnected

12.3 Port B

Port B is a 6-bit special-function port that shares all six of its pins with the analog-to-digital converter (ADC) module.

12.3.1 Port B Data Register

The port B data register (PTB) contains a data latch for each of the six port pins.

Address: $0001

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
Write:	▨	▨						
Reset:	Unaffected by reset							
Alternative Function:	▨	▨	AD5	AD4	AD3	AD2	AD1	AD0

▨ = Unimplemented

Figure 12-6. Port B Data Register (PTB)

PTB5–PTB0 — Port B Data Bits

These read/write bits are software-programmable. Data direction of each port B pin is under the control of the corresponding bit in data direction register B. Reset has no effect on port B data.

AD5–AD0 — Analog-to-Digital Input Bits

AD5–AD0 are pins used for the input channels to the analog-to-digital converter module. The channel select bits in the ADC status and control register define which port B pin will be used as an ADC input and overrides any control from the port I/O logic by forcing that pin as the input to the analog circuitry.

NOTE

Care must be taken when reading port B while applying analog voltages to AD5–AD0 pins. If the appropriate ADC channel is not enabled, excessive current drain may occur if analog voltages are applied to the PTBx/ADx pin, while PTB is read as a digital input. Those ports not selected as analog input channels are considered digital I/O ports.

PTB4 and PTB5 are not available in 28-pin DIP and SOIC packages.

12.3.2 Data Direction Register B

Data direction register B (DDRB) determines whether each port B pin is an input or an output. Writing a 1 to a DDRB bit enables the output buffer for the corresponding port B pin; a 0 disables the output buffer.

Address: $0005

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Write:	▨	▨						
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-7. Data Direction Register B (DDRB)

DDRB5–DDRB0 — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB5–DDRB0, configuring all port B pins as inputs.

1 = Corresponding port B pin configured as output
0 = Corresponding port B pin configured as input

NOTE

Avoid glitches on port B pins by writing to the port B data register before changing data direction register B bits from 0 to 1.

For those devices packaged in a 28-pin DIP and SOIC package, PTB5,4 are not connected. Set DDRB5,4 to a 1 to configure PTB5,4 as outputs.

Figure 12-8 shows the port B I/O logic.

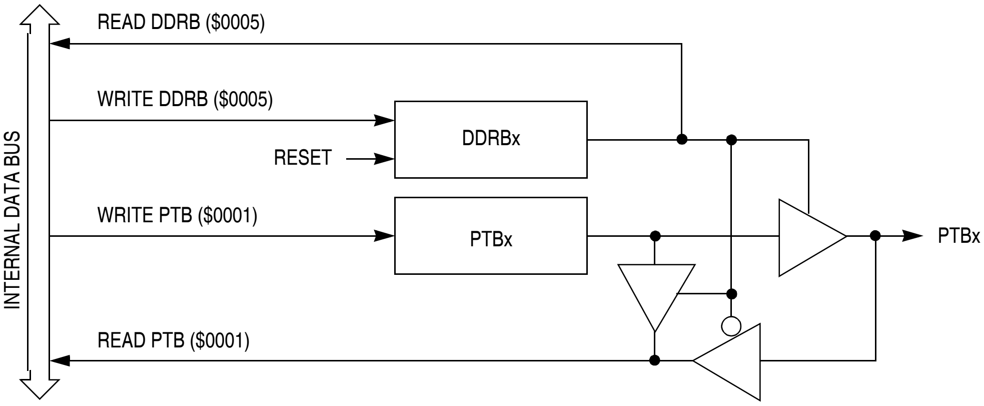


Figure 12-8. Port B I/O Circuit

When bit DDRBx is a 1, reading address $0001 reads the PTBx data latch. When bit DDRBx is a 0, reading address $0001 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-3 summarizes the operation of the port B pins.

Table 12-3. Port B Pin Functions

DDRB Bit	PTB Bit	I/O Pin Mode	Accesses to DDRB	Accesses to PTB	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRB5–DDRB0	Pin	PTB5–PTB0(3)
1	X	Output	DDRB5–DDRB0	PTB5–PTB0	PTB5–PTB0

1. X = Don’t care
2. Hi-Z = High impedance
3. Writing affects data register, but does not affect input.

12.4 Port C

Port C is a 2-bit, general-purpose bidirectional I/O port. Port C also has software configurable pullup devices if configured as an input port.

12.4.1 Port C Data Register

The port C data register (PTC) contains a data latch for each of the two port C pins.

Address: $0002

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	PTC1	PTC0
Write:	▨	▨	▨	▨	▨	▨		
Reset:	Unaffected by reset							

▨ = Unimplemented

Figure 12-9. Port C Data Register (PTC)

PTC1–PTC0 — Port C Data Bits

These read/write bits are software-programmable. Data direction of each port C pin is under the control of the corresponding bit in data direction register C. Reset has no effect on port C data.

NOTE

PTC is not available in a 28-pin DIP or SOIC package.

12.4.2 Data Direction Register C

Data direction register C (DDRC) determines whether each port C pin is an input or an output. Writing a 1 to a DDRC bit enables the output buffer for the corresponding port C pin; a 0 disables the output buffer.

Address: $0006

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	DDRC1	DDRC0
Write:	▨	▨	▨	▨	▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-10. Data Direction Register C (DDRC)

DDRC1–DDRC0 — Data Direction Register C Bits

These read/write bits control port C data direction. Reset clears DDRC1–DDRC0, configuring all port C pins as inputs.

1 = Corresponding port C pin configured as output
0 = Corresponding port C pin configured as input

NOTE

Avoid glitches on port C pins by writing to the port C data register before changing data direction register C bits from 0 to 1.

For those devices packaged in a 28-pin DIP or SOIC package, PTC1–PTC0 are not connected. Set DDRC1 and DDRC0 to a 1 to configure PTC1–PTC0 as outputs.

Figure 12-11 shows the port C I/O logic.

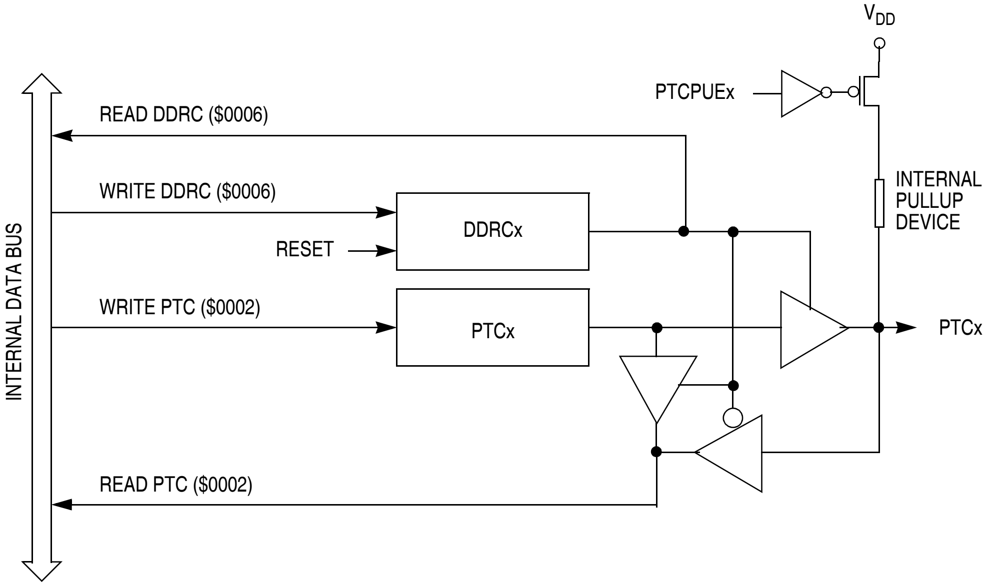


Figure 12-11. Port C I/O Circuit

When bit DDRCx is a 1, reading address $0002 reads the PTCx data latch. When bit DDRCx is a 0, reading address $0002 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-4 summarizes the operation of the port C pins.

Table 12-4. Port C Pin Functions

PTCPUE Bit	DDRC Bit	PTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PTC	
				Read/Write	Read	Write
1	0	X(1)	Input, V_{DD}(2)	DDRC1–DDRC0	Pin	PTC1–PTC0(3)
0	0	X	Input, Hi-Z(4)	DDRC1–DDRC0	Pin	PTC1–PTC0(3)
X	1	X	Output	DDRC1–DDRC0	PTC1–PTC0	PTC1–PTC0

1. X = Don’t care
2. I/O pin pulled up to V_{DD} by internal pullup device.
3. Writing affects data register, but does not affect input.
4. Hi-Z = High impedance

12.4.3 Port C Input Pullup Enable Register

The port C input pullup enable register (PTCPUE) contains a software configurable pullup device for each of the two port C pins. Each bit is individually configurable and requires that the data direction register, DDRC, bit be configured as an input. Each pullup is automatically and dynamically disabled when a port bit's DDRC is configured for output mode.

Address: $000E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	PTCPUE1	PTCPUE0
Write:	▨	▨	▨	▨	▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-12. Port C Input Pullup Enable Register (PTCPUE)

PTCPUE1–PTCPUE0 — Port C Input Pullup Enable Bits

These writeable bits are software programmable to enable pullup devices on an input port bit.

1 = Corresponding port C pin configured to have internal pullup
0 = Corresponding port C pin internal pullup disconnected

12.5 Port D

Port D is a 7-bit special-function port that shares four of its pins with the serial peripheral interface (SPI) module and three of its pins with two timer interface (TIM1 and TIM2) modules. Port D also has software configurable pullup devices if configured as an input port.

12.5.1 Port D Data Register

The port D data register (PTD) contains a data latch for each of the seven port D pins.

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Write:	▨							
Reset:	Unaffected by reset							
Alternative Function:	▨	T2CH0	T1CH1	T1CH0	SPSCK	MOSI	MISO	$\overline{SS}$

▨ = Unimplemented

Figure 12-13. Port D Data Register (PTD)

PTD6–PTD0 — Port D Data Bits

These read/write bits are software-programmable. Data direction of each port D pin is under the control of the corresponding bit in data direction register D. Reset has no effect on port D data.

T2CH0 — Timer 2 Channel I/O Bit

The PTD6/T2CH0 pin is a TIM2 input capture/output compare pin. The edge/level select bits, ELSxB:ELSxA, determine whether the PTD6/T2CH0 pin is a timer channel I/O pin or a general-purpose I/O pin. See Chapter 17 Timer Interface Module (TIM1 and TIM2).

T1CH1 and T1CH0 — Timer 1 Channel I/O Bits

The PTD5/T1CH1–PTD4/T1CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB and ELSxA, determine whether the PTD5/T1CH1–PTD4/T1CH0 pins are timer channel I/O pins or general-purpose I/O pins. See Chapter 17 Timer Interface Module (TIM1 and TIM2).

SPSCK — SPI Serial Clock

The PTD3/SPSCK pin is the serial clock input of the SPI module. When the SPE bit is clear, the PTD3/SPSCK pin is available for general-purpose I/O.

MOSI — Master Out/Slave In

The PTD2/MOSI pin is the master out/slave in terminal of the SPI module. When the SPE bit is clear, the PTD2/MOSI pin is available for general-purpose I/O.

MISO — Master In/Slave Out

The PTD1/MISO pin is the master in/slave out terminal of the SPI module. When the SPI enable bit, SPE, is clear, the SPI module is disabled, and the PTD0/$\overline{\text{SS}}$ pin is available for general-purpose I/O.

Data direction register D (DDRD) does not affect the data direction of port D pins that are being used by the SPI module. However, the DDRD bits always determine whether reading port D returns the states of the latches or the states of the pins. See Table 12-5.

$\overline{\text{SS}}$ — Slave Select

The PTD0/$\overline{\text{SS}}$ pin is the slave select input of the SPI module. When the SPE bit is clear, or when the SPI master bit, SPMSTR, is set, the PTD0/$\overline{\text{SS}}$ pin is available for general-purpose I/O. When the SPI is enabled, the DDRB0 bit in data direction register B (DDRB) has no effect on the PTD0/$\overline{\text{SS}}$ pin.

12.5.2 Data Direction Register D

Data direction register D (DDRD) determines whether each port D pin is an input or an output. Writing a 1 to a DDRD bit enables the output buffer for the corresponding port D pin; a 0 disables the output buffer.

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Write:	▨							
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-14. Data Direction Register D (DDRD)

DDRD6–DDRD0 — Data Direction Register D Bits

These read/write bits control port D data direction. Reset clears DDRD6–DDRD0, configuring all port D pins as inputs.

1 = Corresponding port D pin configured as output
0 = Corresponding port D pin configured as input

NOTE

Avoid glitches on port D pins by writing to the port D data register before changing data direction register D bits from 0 to 1.

Figure 12-15 shows the port D I/O logic.

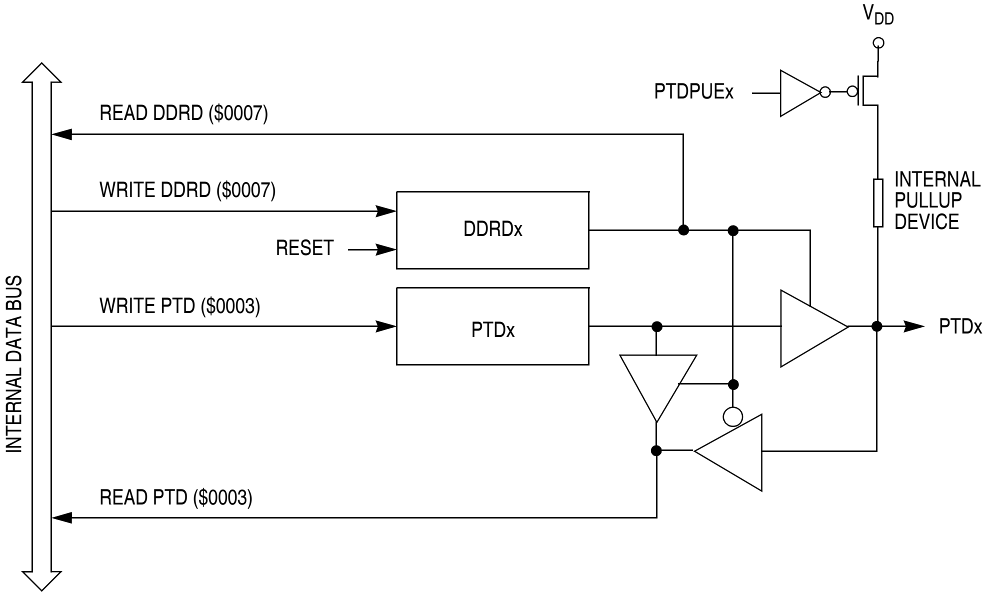


Figure 12-15. Port D I/O Circuit

When bit DDRDx is a 1, reading address $0003 reads the PTDx data latch. When bit DDRDx is a 0, reading address $0003 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-5 summarizes the operation of the port D pins.

Table 12-5. Port D Pin Functions

PTDPUE Bit	DDRD Bit	PTD Bit	I/O Pin Mode	Accesses to DDRD	Accesses to PTD	
				Read/Write	Read	Write
1	0	X(1)	Input, V_{DD}(2)	DDRD6–DDRD0	Pin	PTD6–PTD0(3)
0	0	X	Input, Hi-Z(4)	DDRD6–DDRD0	Pin	PTD6–PTD0(3)
X	1	X	Output	DDRD6–DDRD0	PTD6–PTD0	PTD6–PTD0

1. X = Don’t care
2. I/O pin pulled up to V_{DD} by internal pullup device.
3. Writing affects data register, but does not affect input.
4. Hi-Z = High impedance

12.5.3 Port D Input Pullup Enable Register

The port D input pullup enable register (PTDPUE) contains a software configurable pullup device for each of the seven port D pins. Each bit is individually configurable and requires that the data direction register, DDRD, bit be configured as an input. Each pullup is automatically and dynamically disabled when a port bit's DDRD is configured for output mode.

Address: $000F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
Write:	▨							
Reset:	0	0	0	0	0	0	0	0

Figure 12-16. Port D Input Pullup Enable Register (PTDPUE)

PTDPUE6–PTDPUE0 — Port D Input Pullup Enable Bits

These writable bits are software programmable to enable pullup devices on an input port bit.

1 = Corresponding port D pin configured to have internal pullup
0 = Corresponding port D pin has internal pullup disconnected

12.6 Port E

Port E is a 2-bit special-function port that shares two of its pins with the serial communications interface (SCI) module.

12.6.1 Port E Data Register

The port E data register contains a data latch for each of the two port E pins.

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	PTE1	PTE0
Write:	▨	▨	▨	▨	▨	▨		
Reset:	Unaffected by reset							
Alternative Function:	▨	▨	▨	▨	▨	▨	RxD	TxD

▨ = Unimplemented

Figure 12-17. Port E Data Register (PTE)

PTE1–PTE0 — Port E Data Bits

These read/write bits are software-programmable. Data direction of each port E pin is under the control of the corresponding bit in data direction register E. Reset has no effect on port E data.

NOTE

Data direction register E (DDRE) does not affect the data direction of port E pins that are being used by the ESCI module. However, the DDRE bits always determine whether reading port E returns the states of the latches or the states of the pins. See Table 12-6.

RxD — SCI Receive Data Input

The PTE1/RxD pin is the receive data input for the ESCI module. When the enable SCI bit, ENSCI, is clear, the ESCI module is disabled, and the PTE1/RxD pin is available for general-purpose I/O. See Chapter 13 Serial Communications Interface (SCI) Module.

TxD — SCI Transmit Data Output

The PTE0/TxD pin is the transmit data output for the ESCI module. When the enable SCI bit, ENSCI, is clear, the ESCI module is disabled, and the PTE0/TxD pin is available for general-purpose I/O. See Chapter 13 Serial Communications Interface (SCI) Module.

12.6.2 Data Direction Register E

Data direction register E (DDRE) determines whether each port E pin is an input or an output. Writing a 1 to a DDRE bit enables the output buffer for the corresponding port E pin; a 0 disables the output buffer.

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	DDRE1	DDRE0
Write:	▨	▨	▨	▨	▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-18. Data Direction Register E (DDRE)

DDRE1–DDRE0 — Data Direction Register E Bits

These read/write bits control port E data direction. Reset clears DDRE1–DDRE0, configuring all port E pins as inputs.

1 = Corresponding port E pin configured as output
0 = Corresponding port E pin configured as input

NOTE

Avoid glitches on port E pins by writing to the port E data register before changing data direction register E bits from 0 to 1.

Figure 12-19 shows the port E I/O logic.

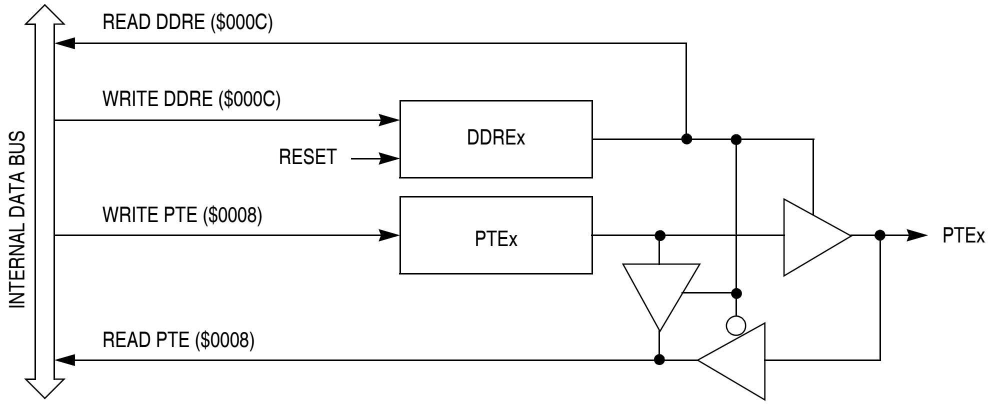


Figure 12-19. Port E I/O Circuit

When bit DDREx is a 1, reading address $0008 reads the PTEx data latch. When bit DDREx is a 0, reading address $0008 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. Table 12-6 summarizes the operation of the port E pins.

Table 12-6. Port E Pin Functions

DDRE Bit	PTE Bit	I/O Pin Mode	Accesses to DDRE	Accesses to PTE	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRE1–DDRE0	Pin	PTE1–PTE0(3)
1	X	Output	DDRE1–DDRE0	PTE1–PTE0	PTE1–PTE0

1. X = Don’t care
2. Hi-Z = High impedance
3. Writing affects data register, but does not affect input.

Chapter 13
Serial Communications Interface (SCI) Module

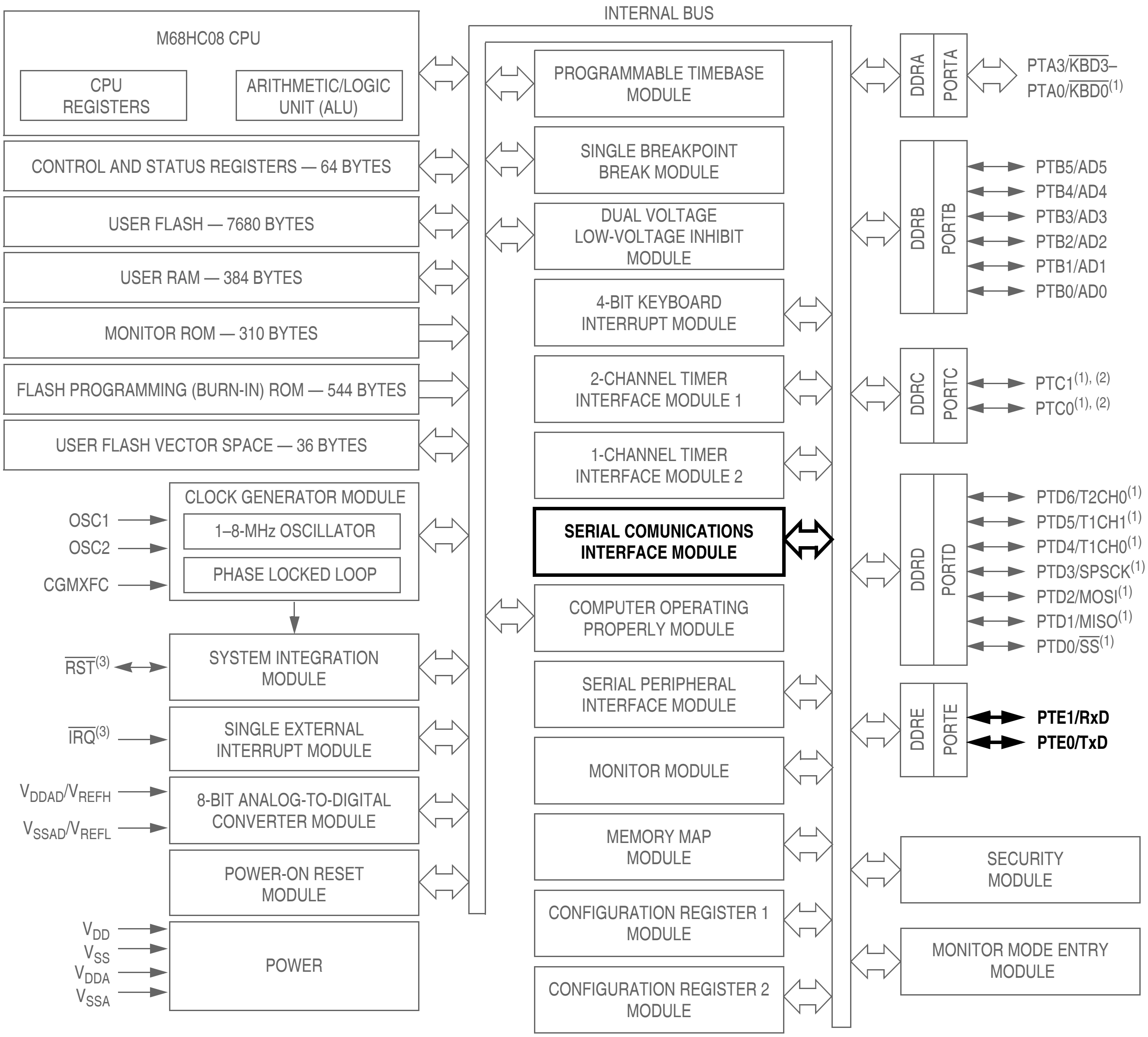
13.1 Introduction

This section describes the serial communications interface (SCI) module, which allows high-speed asynchronous communications with peripheral devices and other MCUs.

13.2 Features

Features of the SCI module include:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 32 programmable baud rates
- Programmable 8-bit or 9-bit character length
- Separately enabled transmitter and receiver
- Separate receiver and transmitter CPU interrupt requests
- Programmable transmitter output polarity
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Interrupt-driven operation with eight interrupt flags:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Idle receiver input
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection
- Mask option register bit, SCIBDSRC, to allow selection of baud rate clock source



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 13-1. Block Diagram Highlighting SCI Block and Pins

13.3 Pin Name Conventions

The generic names of the SCI I/O pins are:

- RxD (receive data)
- TxD (transmit data)

SCI I/O (input/output) lines are implemented by sharing parallel I/O port pins. The full name of an SCI input or output reflects the name of the shared port pin. Table 13-1 shows the full names and the generic names of the SCI I/O pins.

The generic pin names appear in the text of this section.

Table 13-1. Pin Name Conventions

Generic Pin Names:	RxD	TxD
Full Pin Names:	PTE1/RxD	PTE0/TxD

13.4 Functional Description

Figure 13-3 shows the structure of the SCI module. The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

The baud rate clock source for the SCI can be selected via the configuration bit, SCIBDSRC, of the CONFIG2 register ($001E). Source selection values are shown in Figure 13-3.

13.4.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in Figure 13-2.

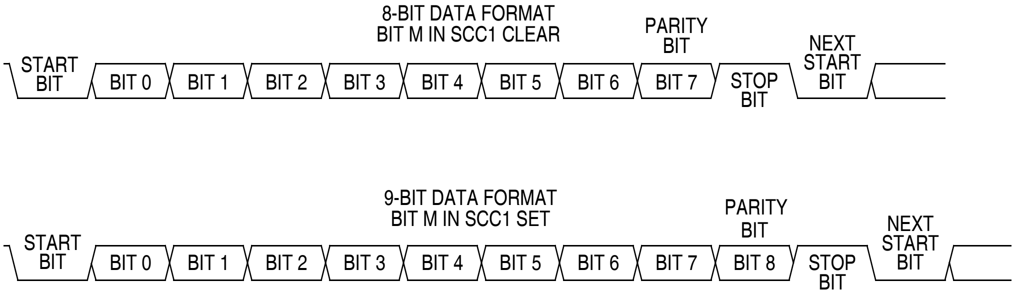


Figure 13-2. SCI Data Formats

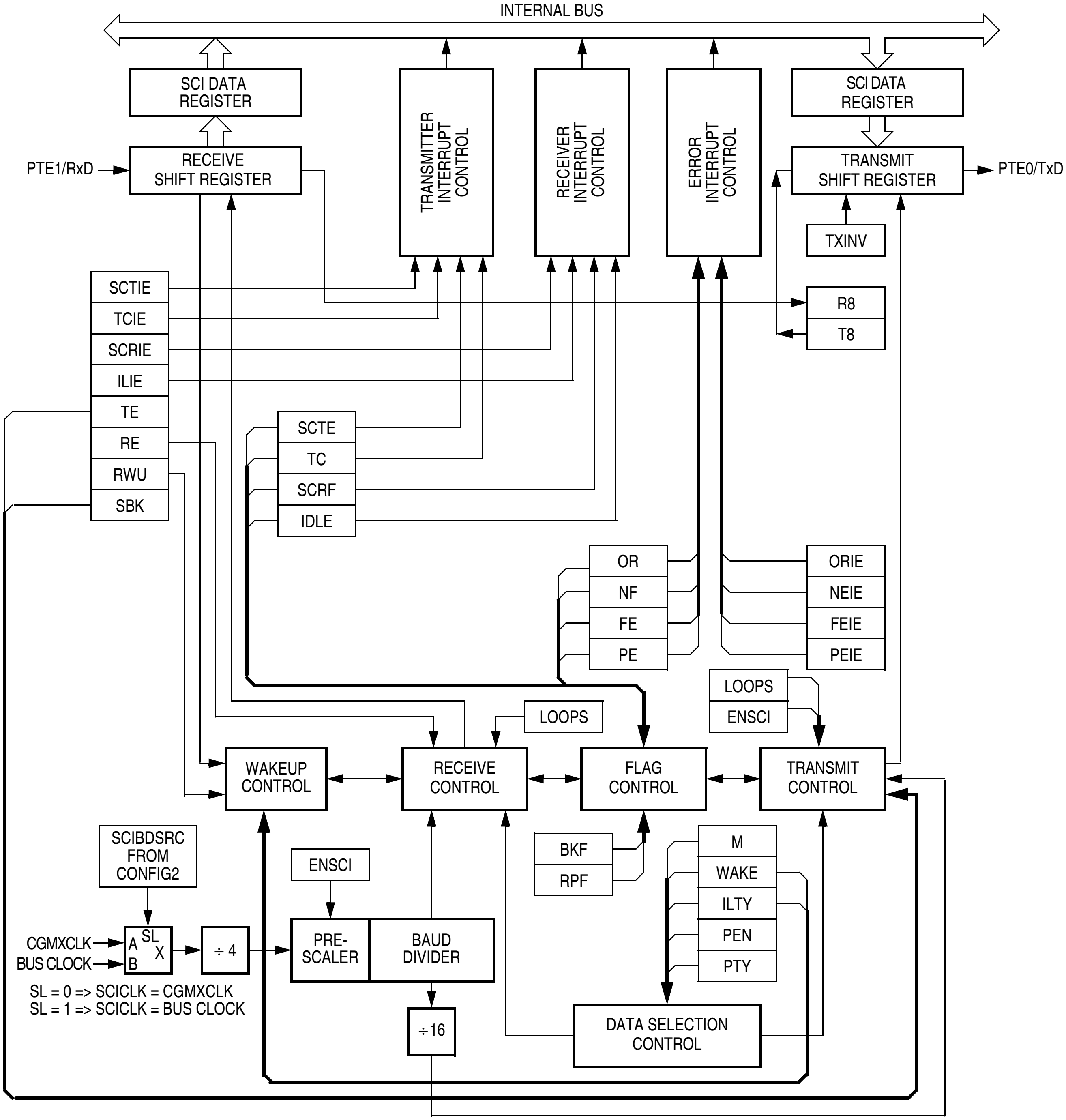


Figure 13-3. SCI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0013	SCI Control Register 1 (SCC1) See page 144.	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0014	SCI Control Register 2 (SCC2) See page 146.	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0015	SCI Control Register 3 (SCC3) See page 147.	Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
		Write:	▨							
		Reset:	U	U	0	0	0	0	0	0
$0016	SCI Status Register 1 (SCS1) See page 149.	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	1	1	0	0	0	0	0	0
$0017	SCI Status Register 2 (SCS2) See page 151.	Read:	▨	▨	▨	▨	▨	▨	BKF	RPF
		Write:							▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0018	SCI Data Register (SCDR) See page 152.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	SCI Baud Rate Register (SCBR) See page 152.	Read:	▨	▨	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented　R = Reserved　U = Unaffected

Figure 13-4. SCI I/O Register Summary

13.4.2 Transmitter

Figure 13-5 shows the structure of the SCI transmitter.

The baud rate clock source for the SCI can be selected via the configuration bit, SCIBDSRC. Source selection values are shown in Figure 13-5.

13.4.2.1 Character Length

The transmitter can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When transmitting 9-bit data, bit T8 in SCI control register 3 (SCC3) is the ninth bit (bit 8).

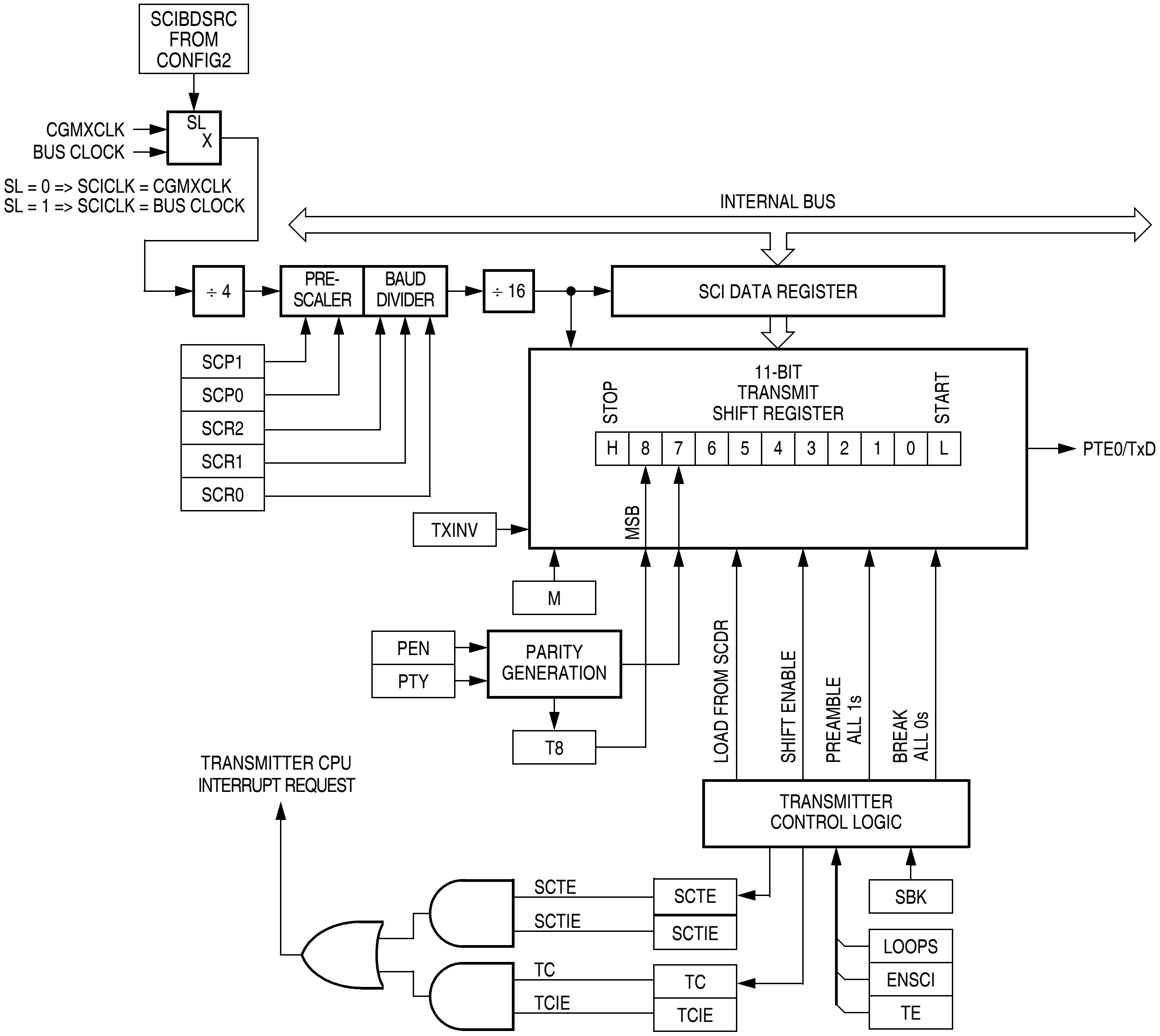


Figure 13-5. SCI Transmitter

13.4.2.2 Character Transmission

During an SCI transmission, the transmit shift register shifts a character out to the PTE0/TxD pin. The SCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register. To initiate an SCI transmission:

1. Enable the SCI by writing a 1 to the enable SCI bit (ENSCI) in SCI control register 1 (SCC1).
2. Enable the transmitter by writing a 1 to the transmitter enable bit (TE) in SCI control register 2 (SCC2).
3. Clear the SCI transmitter empty bit by first reading SCI status register 1 (SCS1) and then writing to the SCDR.
4. Repeat step 3 for each subsequent transmission.

At the start of a transmission, transmitter control logic automatically loads the transmit shift register with a preamble of 1s. After the preamble shifts out, control logic transfers the SCDR data into the transmit shift register. A 0 start bit automatically goes into the least significant bit position of the transmit shift register. A 1 stop bit goes into the most significant bit position.

The SCI transmitter empty bit, SCTE, in SCS1 becomes set when the SCDR transfers a byte to the transmit shift register. The SCTE bit indicates that the SCDR can accept new data from the internal data bus. If the SCI transmit interrupt enable bit, SCTIE, in SCC2 is also set, the SCTE bit generates a transmitter CPU interrupt request.

When the transmit shift register is not transmitting a character, the PTE0/TxD pin goes to the idle condition, 1. If at any time software clears the ENSCI bit in SCI control register 1 (SCC1), the transmitter and receiver relinquish control of the port E pins.

13.4.2.3 Break Characters

Writing a 1 to the send break bit, SBK, in SCC2 loads the transmit shift register with a break character. A break character contains all 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCC1. As long as SBK is 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one 1. The automatic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

The SCI recognizes a break character when a start bit is followed by eight or nine 0 data bits and a 0 where the stop bit should be.

Receiving a break character has these effects on SCI registers:

- Sets the framing error bit (FE) in SCS1
- Sets the SCI receiver full bit (SCRF) in SCS1
- Clears the SCI data register (SCDR)
- Clears the R8 bit in SCC3
- Sets the break flag bit (BKF) in SCS2
- May set the overrun (OR), noise flag (NF), parity error (PE), or reception in progress flag (RPF) bits

13.4.2.4 Idle Characters

An idle character contains all 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCC1. The preamble is a synchronizing idle character that begins every transmission.

If the TE bit is cleared during a transmission, the PTE0/TxD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the character currently being transmitted.

> ***NOTE***
>
> *When queueing an idle character, return the TE bit to 1 before the stop bit of the current character shifts out to the TxD pin. Setting TE after the stop bit appears on TxD causes data previously written to the SCDR to be lost. Toggle the TE bit for a queued idle character when the SCTE bit becomes set and just before writing the next byte to the SCDR.*

13.4.2.5 Inversion of Transmitted Output

The transmit inversion bit (TXINV) in SCI control register 1 (SCC1) reverses the polarity of transmitted data. All transmitted values, including idle, break, start, and stop bits, are inverted when TXINV is 1. See 13.8.1 SCI Control Register 1.

13.4.2.6 Transmitter Interrupts

These conditions can generate CPU interrupt requests from the SCI transmitter:

- SCI transmitter empty (SCTE) — The SCTE bit in SCS1 indicates that the SCDR has transferred a character to the transmit shift register. SCTE can generate a transmitter CPU interrupt request. Setting the SCI transmit interrupt enable bit, SCTIE, in SCC2 enables the SCTE bit to generate transmitter CPU interrupt requests.
- Transmission complete (TC) — The TC bit in SCS1 indicates that the transmit shift register and the SCDR are empty and that no break or idle character has been generated. The transmission complete interrupt enable bit, TCIE, in SCC2 enables the TC bit to generate transmitter CPU interrupt requests.

13.4.3 Receiver

Figure 13-6 shows the structure of the SCI receiver.

13.4.3.1 Character Length

The receiver can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCC1) determines character length. When receiving 9-bit data, bit R8 in SCI control register 2 (SCC2) is the ninth bit (bit 8). When receiving 8-bit data, bit R8 is a copy of the eighth bit (bit 7).

13.4.3.2 Character Reception

During an SCI reception, the receive shift register shifts characters in from the PTE1/RxD pin. The SCI data register (SCDR) is the read-only buffer between the internal data bus and the receive shift register.

After a complete character shifts into the receive shift register, the data portion of the character transfers to the SCDR. The SCI receiver full bit, SCRF, in SCI status register 1 (SCS1) becomes set, indicating that the received byte can be read. If the SCI receive interrupt enable bit, SCRIE, in SCC2 is also set, the SCRF bit generates a receiver CPU interrupt request.

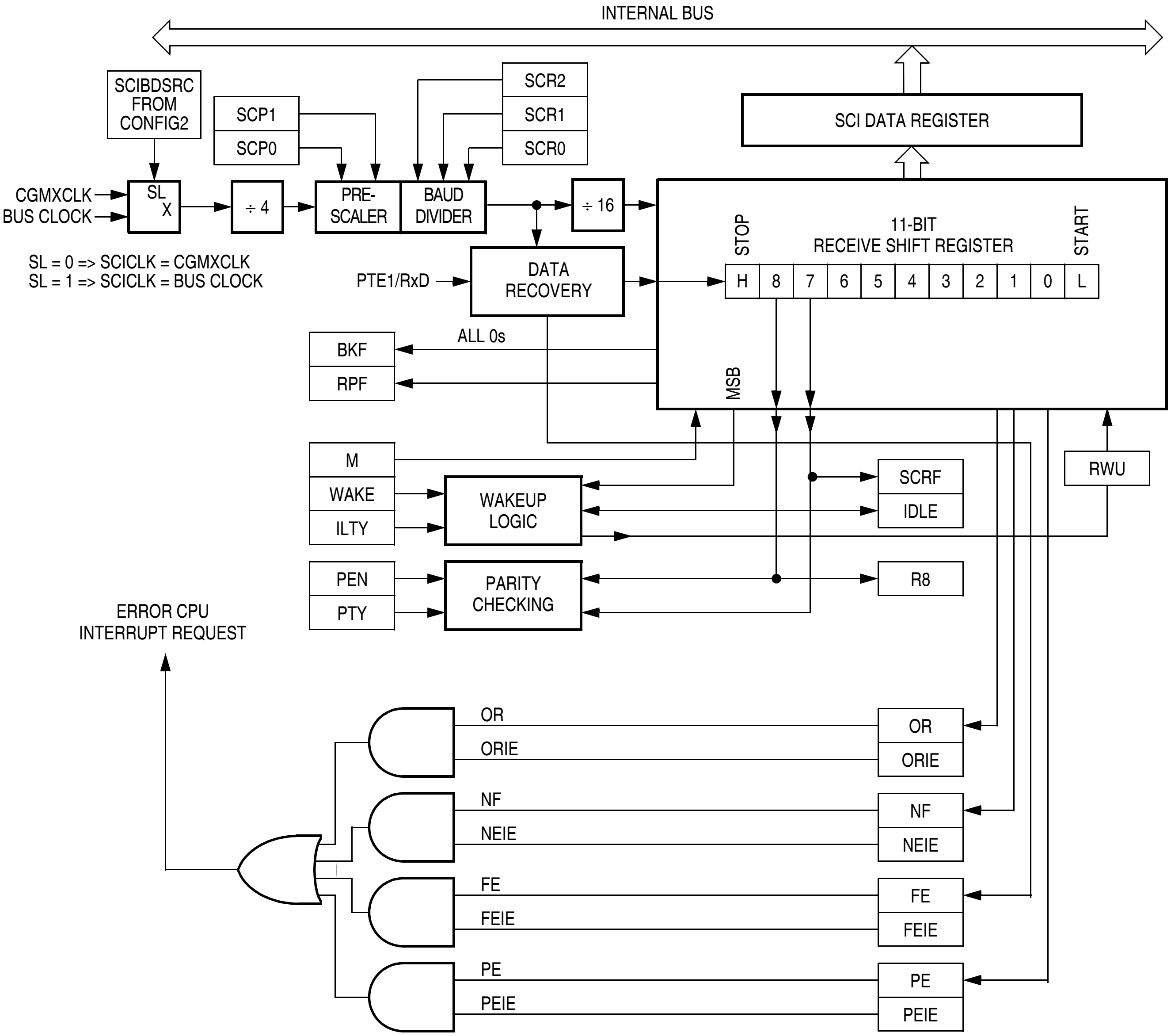


Figure 13-6. SCI Receiver Block Diagram

13.4.3.3 Data Sampling

The receiver samples the PTE1/RxD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock is resynchronized at the following times (see Figure 13-7):

- After every start bit
- After the receiver detects a data bit change from 1 to 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid 0)

To locate the start bit, data recovery logic does an asynchronous search for a 0 preceded by three 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

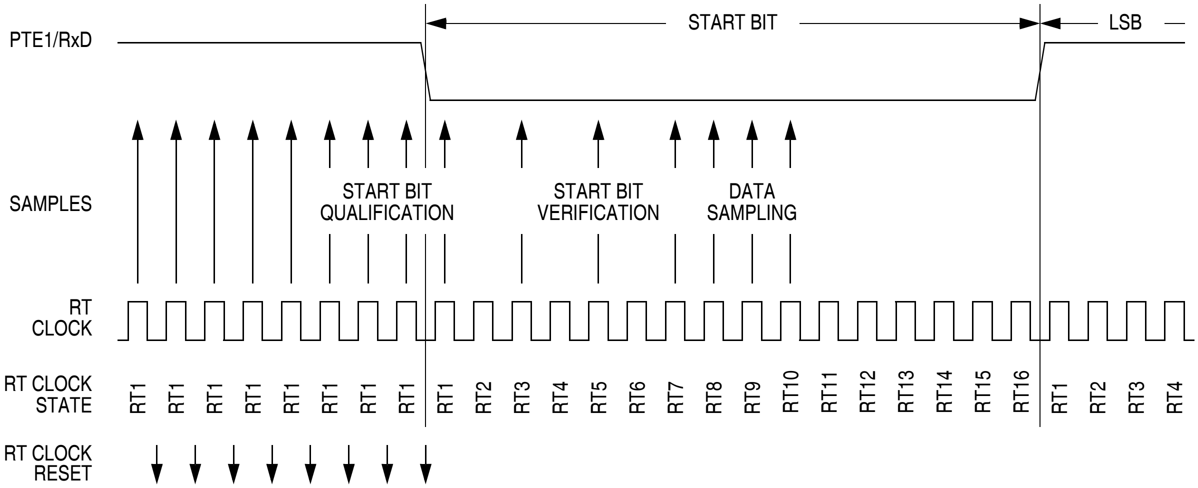


Figure 13-7. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Table 13-2 summarizes the results of the start bit verification samples.

Table 13-2. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

Start bit verification is not successful if any two of the three verification samples are 1s. If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 13-3 summarizes the results of the data bit samples.

Table 13-3. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit.

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 13-4 summarizes the results of the stop bit samples.

Table 13-4. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

13.4.3.4 Framing Errors

If the data recovery logic does not detect a 1 where the stop bit should be in an incoming character, it sets the framing error bit, FE, in SCS1. A break character also sets the FE bit because a break character has no stop bit. The FE bit is set at the same time that the SCRF bit is set.

13.4.3.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. Then a noise error occurs. If more than one of the samples is outside the stop bit, a framing error occurs. In most applications, the baud rate tolerance is much more than the degree of misalignment that is likely to occur.

As the receiver samples an incoming character, it resynchronizes the RT clock on any valid falling edge within the character. Resynchronization within characters corrects misalignments between transmitter bit times and receiver bit times.

Slow Data Tolerance

Figure 13-8 shows how much a slow received character can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

For an 8-bit character, data sampling of the stop bit takes the receiver 9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 13-8, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 9 bit times × 16 RT cycles + 3 RT cycles = 147 RT cycles.

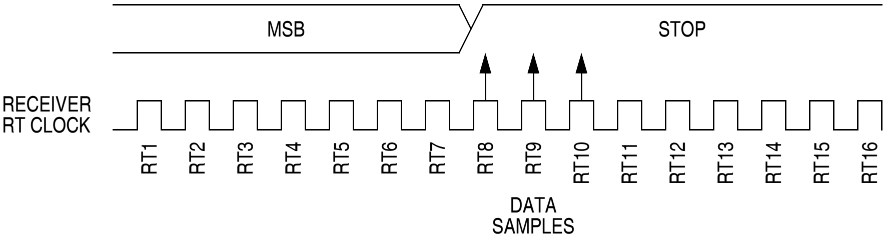


Figure 13-8. Slow Data

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit character with no errors is

$$\left|\frac{154-147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver 10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 13-8, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles + 3 RT cycles = 163 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is

$$\left|\frac{170-163}{170}\right| \times 100 = 4.12\%$$

Fast Data Tolerance

Figure 13-9 shows how much a fast received character can be misaligned without causing a noise error or a framing error. The fast stop bit ends at RT10 instead of RT16 but is still there for the stop bit data samples at RT8, RT9, and RT10.

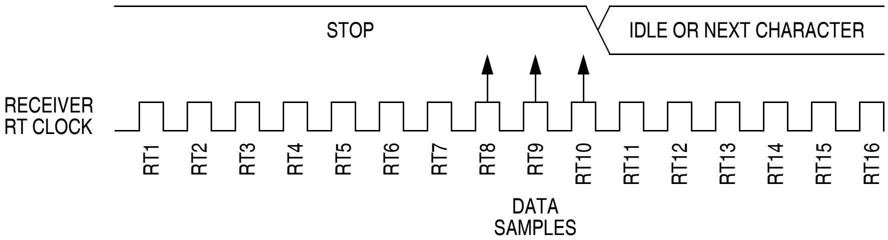


Figure 13-9. Fast Data

For an 8-bit character, data sampling of the stop bit takes the receiver 9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 13-9, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles = 160 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is

$$\left|\frac{154 - 160}{154}\right| \times 100 = 3.9\dot{0}\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver 10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 13-9, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 11 bit times × 16 RT cycles = 176 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

13.4.3.6 Receiver Wakeup

So that the MCU can ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCC2 puts the receiver into a standby state during which receiver interrupts are disabled.

Depending on the state of the WAKE bit in SCC1, either of two conditions on the PTE1/RxD pin can bring the receiver out of the standby state:

- Address mark — An address mark is a 1 in the most significant bit position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the SCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.

- Idle input line condition — When the WAKE bit is clear, an idle character on the PTE1/RxD pin wakes the receiver from the standby state by clearing the RWU bit. The idle character that wakes the receiver does not set the receiver idle bit, IDLE, or the SCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting 1s as idle character bits after the start bit or after the stop bit.

NOTE

With the WAKE bit clear, setting the RWU bit after the RxD pin has been idle may cause the receiver to wake up immediately.

13.4.3.7 Receiver Interrupts

The following sources can generate CPU interrupt requests from the SCI receiver:

- SCI receiver full (SCRF) — The SCRF bit in SCS1 indicates that the receive shift register has transferred a character to the SCDR. SCRF can generate a receiver CPU interrupt request. Setting the SCI receive interrupt enable bit, SCRIE, in SCC2 enables the SCRF bit to generate receiver CPU interrupts.
- Idle input (IDLE) — The IDLE bit in SCS1 indicates that 10 or 11 consecutive 1s shifted in from the PTE1/RxD pin. The idle line interrupt enable bit, ILIE, in SCC2 enables the IDLE bit to generate CPU interrupt requests.

13.4.3.8 Error Interrupts

The following receiver error flags in SCS1 can generate CPU interrupt requests:

- Receiver overrun (OR) — The OR bit indicates that the receive shift register shifted in a new character before the previous character was read from the SCDR. The previous character remains in the SCDR, and the new character is lost. The overrun interrupt enable bit, ORIE, in SCC3 enables OR to generate SCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the SCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in SCC3 enables NF to generate SCI error CPU interrupt requests.
- Framing error (FE) — The FE bit in SCS1 is set when a 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate SCI error CPU interrupt requests.
- Parity error (PE) — The PE bit in SCS1 is set when the SCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate SCI error CPU interrupt requests.

13.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power- consumption standby modes.

13.5.1 Wait Mode

The SCI module remains active after the execution of a WAIT instruction. In wait mode, the SCI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SCI module can bring the MCU out of wait mode.

If SCI module functions are not required during wait mode, reduce power consumption by disabling the module before executing the WAIT instruction.

Refer to Chapter 10 Low-Power Modes for information on exiting wait mode.

13.5.2 Stop Mode

The SCI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect SCI register states. SCI module operation resumes after an external interrupt.

Because the internal clock is inactive during stop mode, entering stop mode during an SCI transmission or reception results in invalid data.

Refer to Chapter 10 Low-Power Modes for information on exiting stop mode.

13.6 SCI During Break Module Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state.

To allow software to clear status bits during a break interrupt, write a 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a 0 to the BCFE bit. With BCFE at 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at 0. After the break, doing the second step clears the status bit.

13.7 I/O Signals

Port E shares two of its pins with the SCI module. The two SCI I/O pins are:

- PTE0/TxD — Transmit data
- PTE1/RxD — Receive data

13.7.1 PTE0/TxD (Transmit Data)

The PTE0/TxD pin is the serial data output from the SCI transmitter. The SCI shares the PTE0/TxD pin with port E. When the SCI is enabled, the PTE0/TxD pin is an output regardless of the state of the DDRE0 bit in data direction register E (DDRE).

13.7.2 PTE1/RxD (Receive Data)

The PTE1/RxD pin is the serial data input to the SCI receiver. The SCI shares the PTE1/RxD pin with port E. When the SCI is enabled, the PTE1/RxD pin is an input regardless of the state of the DDRE1 bit in data direction register E (DDRE).

13.8 I/O Registers

These I/O registers control and monitor SCI operation:

- SCI control register 1 (SCC1)
- SCI control register 2 (SCC2)
- SCI control register 3 (SCC3)
- SCI status register 1 (SCS1)
- SCI status register 2 (SCS2)
- SCI data register (SCDR)
- SCI baud rate register (SCBR)

13.8.1 SCI Control Register 1

SCI control register 1:

- Enables loop mode operation
- Enables the SCI
- Controls output polarity
- Controls character length
- Controls SCI wakeup method
- Controls idle character detection
- Enables parity function
- Controls parity type

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
Reset:	0	0	0	0	0	0	0	0

Figure 13-10. SCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation. In loop mode the PTE1/RxD pin is disconnected from the SCI, and the transmitter output goes into the receiver input. Both the transmitter and the receiver must be enabled to use loop mode. Reset clears the LOOPS bit.

1 = Loop mode enabled
0 = Normal operation enabled

ENSCI — Enable SCI Bit

This read/write bit enables the SCI and the SCI baud rate generator. Clearing ENSCI sets the SCTE and TC bits in SCI status register 1 and disables transmitter interrupts. Reset clears the ENSCI bit.

1 = SCI enabled
0 = SCI disabled

TXINV — Transmit Inversion Bit

This read/write bit reverses the polarity of transmitted data. Reset clears the TXINV bit.

1 = Transmitter output inverted
0 = Transmitter output not inverted

NOTE

Setting the TXINV bit inverts all transmitted values, including idle, break, start, and stop bits.

M — Mode (Character Length) Bit

This read/write bit determines whether SCI characters are eight or nine bits long. See Table 13-5. The ninth bit can serve as an extra stop bit, as a receiver wakeup signal, or as a parity bit. Reset clears the M bit.

1 = 9-bit SCI characters
0 = 8-bit SCI characters

WAKE — Wakeup Condition Bit

This read/write bit determines which condition wakes up the SCI: a 1 (address mark) in the most significant bit position of a received character or an idle condition on the PTE1/RxD pin. Reset clears the WAKE bit.

1 = Address mark wakeup
0 = Idle line wakeup

ILTY — Idle Line Type Bit

This read/write bit determines when the SCI starts counting 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. Reset clears the ILTY bit.

1 = Idle character bit count begins after stop bit
0 = Idle character bit count begins after start bit

PEN — Parity Enable Bit

This read/write bit enables the SCI parity function. See Table 13-5. When enabled, the parity function inserts a parity bit in the most significant bit position. See Figure 13-2. Reset clears the PEN bit.

1 = Parity function enabled
0 = Parity function disabled

PTY — Parity Bit

This read/write bit determines whether the SCI generates and checks for odd parity or even parity. See Table 13-5. Reset clears the PTY bit.

1 = Odd parity
0 = Even parity

NOTE

Changing the PTY bit in the middle of a transmission or reception can generate a parity error.

Table 13-5. Character Format Selection

Control Bits		Character Format				
M	**PEN and PTY**	**Start Bits**	**Data Bits**	**Parity**	**Stop Bits**	**Character Length**
0	0X	1	8	None	1	10 bits
1	0X	1	9	None	1	11 bits
0	10	1	7	Even	1	10 bits
0	11	1	7	Odd	1	10 bits
1	10	1	8	Even	1	11 bits
1	11	1	8	Odd	1	11 bits

13.8.2 SCI Control Register 2

SCI control register 2:

- Enables the following CPU interrupt requests:
  - Enables the SCTE bit to generate transmitter CPU interrupt requests
  - Enables the TC bit to generate transmitter CPU interrupt requests
  - Enables the SCRF bit to generate receiver CPU interrupt requests
  - Enables the IDLE bit to generate receiver CPU interrupt requests
- Enables the transmitter
- Enables the receiver
- Enables SCI wakeup
- Transmits SCI break characters

Address: $0014

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

Figure 13-11. SCI Control Register 2 (SCC2)

SCTIE — SCI Transmit Interrupt Enable Bit

This read/write bit enables the SCTE bit to generate SCI transmitter CPU interrupt requests. Reset clears the SCTIE bit.

1 = SCTE enabled to generate CPU interrupt
0 = SCTE not enabled to generate CPU interrupt

TCIE — Transmission Complete Interrupt Enable Bit

This read/write bit enables the TC bit to generate SCI transmitter CPU interrupt requests. Reset clears the TCIE bit.

1 = TC enabled to generate CPU interrupt requests
0 = TC not enabled to generate CPU interrupt requests

SCRIE — SCI Receive Interrupt Enable Bit

This read/write bit enables the SCRF bit to generate SCI receiver CPU interrupt requests. Reset clears the SCRIE bit.

1 = SCRF enabled to generate CPU interrupt
0 = SCRF not enabled to generate CPU interrupt

ILIE — Idle Line Interrupt Enable Bit

This read/write bit enables the IDLE bit to generate SCI receiver CPU interrupt requests. Reset clears the ILIE bit.

1 = IDLE enabled to generate CPU interrupt requests
0 = IDLE not enabled to generate CPU interrupt requests

TE — Transmitter Enable Bit

Setting this read/write bit begins the transmission by sending a preamble of 10 or 11 1s from the transmit shift register to the PTE0/TxD pin. If software clears the TE bit, the transmitter completes any transmission in progress before the PTE0/TxD returns to the idle condition (1). Clearing and then setting TE during a transmission queues an idle character to be sent after the character currently being transmitted. Reset clears the TE bit.

1 = Transmitter enabled
0 = Transmitter disabled

NOTE

Writing to the TE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.

RE — Receiver Enable Bit

Setting this read/write bit enables the receiver. Clearing the RE bit disables the receiver but does not affect receiver interrupt flag bits. Reset clears the RE bit.

1 = Receiver enabled
0 = Receiver disabled

NOTE

Writing to the RE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.

RWU — Receiver Wakeup Bit

This read/write bit puts the receiver in a standby state during which receiver interrupts are disabled. The WAKE bit in SCC1 determines whether an idle input or an address mark brings the receiver out of the standby state and clears the RWU bit. Reset clears the RWU bit.

1 = Standby state
0 = Normal operation

SBK — Send Break Bit

Setting and then clearing this read/write bit transmits a break character followed by a 1. The 1 after the break character guarantees recognition of a valid start bit. If SBK remains set, the transmitter continuously transmits break characters with no 1s between them. Reset clears the SBK bit.

1 = Transmit break characters
0 = No break characters being transmitted

NOTE

Do not toggle the SBK bit immediately after setting the SCTE bit. Toggling SBK before the preamble begins causes the SCI to send a break character instead of a preamble.

13.8.3 SCI Control Register 3

SCI control register 3:

- Stores the ninth SCI data bit received and the ninth SCI data bit to be transmitted
- Enables these interrupts:
  - Receiver overrun interrupts
  - Noise error interrupts
  - Framing error interrupts
- Parity error interrupts

Address: $0015

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
Write:	▨							
Reset:	U	U	0	0	0	0	0	0

▨ = Unimplemented R = Reserved U = Unaffected

Figure 13-12. SCI Control Register 3 (SCC3)

R8 — Received Bit 8

When the SCI is receiving 9-bit characters, R8 is the read-only ninth bit (bit 8) of the received character. R8 is received at the same time that the SCDR receives the other 8 bits.

When the SCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7). Reset has no effect on the R8 bit.

T8 — Transmitted Bit 8

When the SCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register. Reset has no effect on the T8 bit.

ORIE — Receiver Overrun Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the receiver overrun bit, OR.

1 = SCI error CPU interrupt requests from OR bit enabled
0 = SCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.

1 = SCI error CPU interrupt requests from NE bit enabled
0 = SCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.

1 = SCI error CPU interrupt requests from FE bit enabled
0 = SCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the parity error bit, PE. See 13.8.4 SCI Status Register 1. Reset clears PEIE.

1 = SCI error CPU interrupt requests from PE bit enabled
0 = SCI error CPU interrupt requests from PE bit disabled

13.8.4 SCI Status Register 1

SCI status register 1 (SCS1) contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	1	1	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-13. SCI Status Register 1 (SCS1)

SCTE — SCI Transmitter Empty Bit

This clearable, read-only bit is set when the SCDR transfers a character to the transmit shift register. SCTE can generate an SCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an SCI transmitter CPU interrupt request. In normal operation, clear the SCTE bit by reading SCS1 with SCTE set and then writing to SCDR. Reset sets the SCTE bit.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

TC — Transmission Complete Bit

This read-only bit is set when the SCTE bit is set, and no data, preamble, or break character is being transmitted. TC generates an SCI transmitter CPU interrupt request if the TCIE bit in SCC2 is also set. TC is automatically cleared when data, preamble or break is queued and ready to be sent. There may be up to 1.5 transmitter clocks of latency between queueing data, preamble, and break and the transmission actually starting. Reset sets the TC bit.

1 = No transmission in progress
0 = Transmission in progress

SCRF — SCI Receiver Full Bit

This clearable, read-only bit is set when the data in the receive shift register transfers to the SCI data register. SCRF can generate an SCI receiver CPU interrupt request. When the SCRIE bit in SCC2 is set, SCRF generates a CPU interrupt request. In normal operation, clear the SCRF bit by reading SCS1 with SCRF set and then reading the SCDR. Reset clears SCRF.

1 = Received data available in SCDR
0 = Data not available in SCDR

IDLE — Receiver Idle Bit

This clearable, read-only bit is set when 10 or 11 consecutive 1s appear on the receiver input. IDLE generates an SCI receiver CPU interrupt request if the ILIE bit in SCC2 is also set. Clear the IDLE bit by reading SCS1 with IDLE set and then reading the SCDR. After the receiver is enabled, it must receive a valid character that sets the SCRF bit before an idle condition can set the IDLE bit. Also, after the IDLE bit has been cleared, a valid character must again set the SCRF bit before an idle condition can set the IDLE bit. Reset clears the IDLE bit.

1 = Receiver input idle
0 = Receiver input active (or idle since the IDLE bit was cleared)

OR — Receiver Overrun Bit

This clearable, read-only bit is set when software fails to read the SCDR before the receive shift register receives the next character. The OR bit generates an SCI error CPU interrupt request if the ORIE bit in SCC3 is also set. The data in the shift register is lost, but the data already in the SCDR is not affected. Clear the OR bit by reading SCS1 with OR set and then reading the SCDR. Reset clears the OR bit.

1 = Receive shift register full and SCRF = 1
0 = No receiver overrun

Software latency may allow an overrun to occur between reads of SCS1 and SCDR in the flag-clearing sequence. Figure 13-14 shows the normal flag-clearing sequence and an example of an overrun caused by a delayed flag-clearing sequence. The delayed read of SCDR does not clear the OR bit because OR was not set when SCS1 was read. Byte 2 caused the overrun and is lost. The next flag-clearing sequence reads byte 3 in the SCDR instead of byte 2.

In applications that are subject to software latency or in which it is important to know which byte is lost due to an overrun, the flag-clearing routine can check the OR bit in a second read of SCS1 after reading the data register.

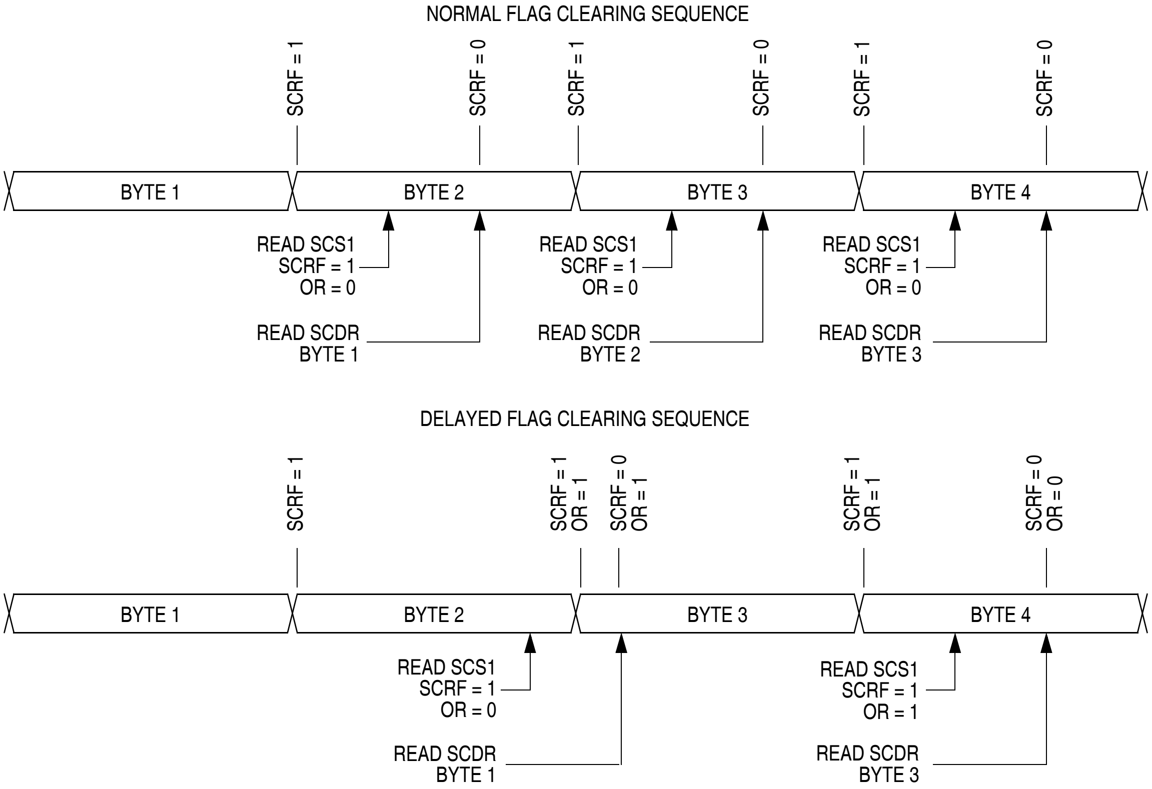


Figure 13-14. Flag Clearing Sequence

NF — Receiver Noise Flag Bit

This clearable, read-only bit is set when the SCI detects noise on the PTE1/RxD pin. NF generates an SCI error CPU interrupt request if the NEIE bit in SCC3 is also set. Clear the NF bit by reading SCS1 and then reading the SCDR. Reset clears the NF bit.

1 = Noise detected
0 = No noise detected

FE — Receiver Framing Error Bit

This clearable, read-only bit is set when a 0 is accepted as the stop bit. FE generates an SCI error CPU interrupt request if the FEIE bit in SCC3 also is set. Clear the FE bit by reading SCS1 with FE set and then reading the SCDR. Reset clears the FE bit.

1 = Framing error detected
0 = No framing error detected

PE — Receiver Parity Error Bit

This clearable, read-only bit is set when the SCI detects a parity error in incoming data. PE generates an SCI error CPU interrupt request if the PEIE bit in SCC3 is also set. Clear the PE bit by reading SCS1 with PE set and then reading the SCDR. Reset clears the PE bit.

1 = Parity error detected
0 = No parity error detected

13.8.5 SCI Status Register 2

SCI status register 2 contains flags to signal the following conditions:

- Break character detected
- Incoming data

Address: $0017

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	▨	▨	▨	▨	▨	▨	BKF	RPF
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 13-15. SCI Status Register 2 (SCS2)

BKF — Break Flag Bit

This clearable, read-only bit is set when the SCI detects a break character on the PTE1/RxD pin. In SCS1, the FE and SCRF bits are also set. In 9-bit character transmissions, the R8 bit in SCC3 is cleared. BKF does not generate a CPU interrupt request. Clear BKF by reading SCS2 with BKF set and then reading the SCDR. Once cleared, BKF can become set again only after 1s again appear on the PTE1/RxD pin followed by another break character. Reset clears the BKF bit.

1 = Break character detected
0 = No break character detected

RPF — Reception in Progress Flag Bit

This read-only bit is set when the receiver detects a 0 during the RT1 time period of the start bit search. RPF does not generate an interrupt request. RPF is reset after the receiver detects false start bits (usually from noise or a baud rate mismatch) or when the receiver detects an idle character. Polling RPF before disabling the SCI module or entering stop mode can show whether a reception is in progress.

1 = Reception in progress
0 = No reception in progress

13.8.6 SCI Data Register

The SCI data register (SCDR) is the buffer between the internal data bus and the receive and transmit shift registers. Reset has no effect on data in the SCI data register.

Address: $0018

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	Unaffected by reset							

Figure 13-16. SCI Data Register (SCDR)

R7/T7–R0/T0 — Receive/Transmit Data Bits

Reading the SCDR accesses the read-only received data bits, R7:R0. Writing to the SCDR writes the data to be transmitted, T7:T0.

Reset has no effect on the SCDR.

NOTE

Do not use read/modify/write instructions on the SCI data register.

13.8.7 SCI Baud Rate Register

The baud rate register (SCBR) selects the baud rate for both the receiver and the transmitter.

Address: $0019

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	▨	▨	SCP1	SCP0	R	SCR2	SCR1	SCR0
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented R = Reserved

Figure 13-17. SCI Baud Rate Register (SCBR)

SCP1 and SCP0 — SCI Baud Rate Prescaler Bits

These read/write bits select the baud rate prescaler divisor as shown in Table 13-6. Reset clears SCP1 and SCP0.

Table 13-6. SCI Baud Rate Prescaling

SCP1 and SCP0	Prescaler Divisor (PD)
00	1
01	3
10	4
11	13

SCR2–SCR0 — SCI Baud Rate Select Bits

These read/write bits select the SCI baud rate divisor as shown in Table 13-7. Reset clears SCR2–SCR0.

Table 13-7. SCI Baud Rate Selection

SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Use this formula to calculate the SCI baud rate:

$$\text{Baud rate} = \frac{\text{SCI clock source}}{64 \times \text{PD} \times \text{BD}}$$

where:

SCI clock source = f_{BUS} or CGMXCLK (selected by SCIBDSRC bit in CONFIG2)

PD = prescaler divisor

BD = baud rate divisor

Table 13-8 shows the SCI baud rates that can be generated with a 4.9152-MHz bus clock when f_{BUS} is selected as SCI clock source.

Table 13-8. SCI Baud Rate Selection Examples

SCP1 and SCP0	Prescaler Divisor (PD)	SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)	Baud Rate (f_{BUS} = 4.9152 MHz)
00	1	000	1	76,800
00	1	001	2	38,400
00	1	010	4	19,200
00	1	011	8	9600
00	1	100	16	4800
00	1	101	32	2400
00	1	110	64	1200
00	1	111	128	600
01	3	000	1	25,600
01	3	001	2	12,800
01	3	010	4	6400
01	3	011	8	3200
01	3	100	16	1600
01	3	101	32	800
01	3	110	64	400
01	3	111	128	200
10	4	000	1	19,200
10	4	001	2	9600
10	4	010	4	4800
10	4	011	8	2400
10	4	100	16	1200
10	4	101	32	600
10	4	110	64	300
10	4	111	128	150
11	13	000	1	5908
11	13	001	2	2954
11	13	010	4	1477
11	13	011	8	739
11	13	100	16	369
11	13	101	32	185
11	13	110	64	92
11	13	111	128	46

Chapter 14
System Integration Module (SIM)

14.1 Introduction

This section describes the system integration module (SIM). Together with the central processor unit (CPU), the SIM controls all microcontroller unit (MCU) activities. A block diagram of the SIM is shown in Figure 14-1. Table 14-1 is a summary of the SIM input/output (I/O) registers. The SIM is a system state controller that coordinates CPU and exception timing.

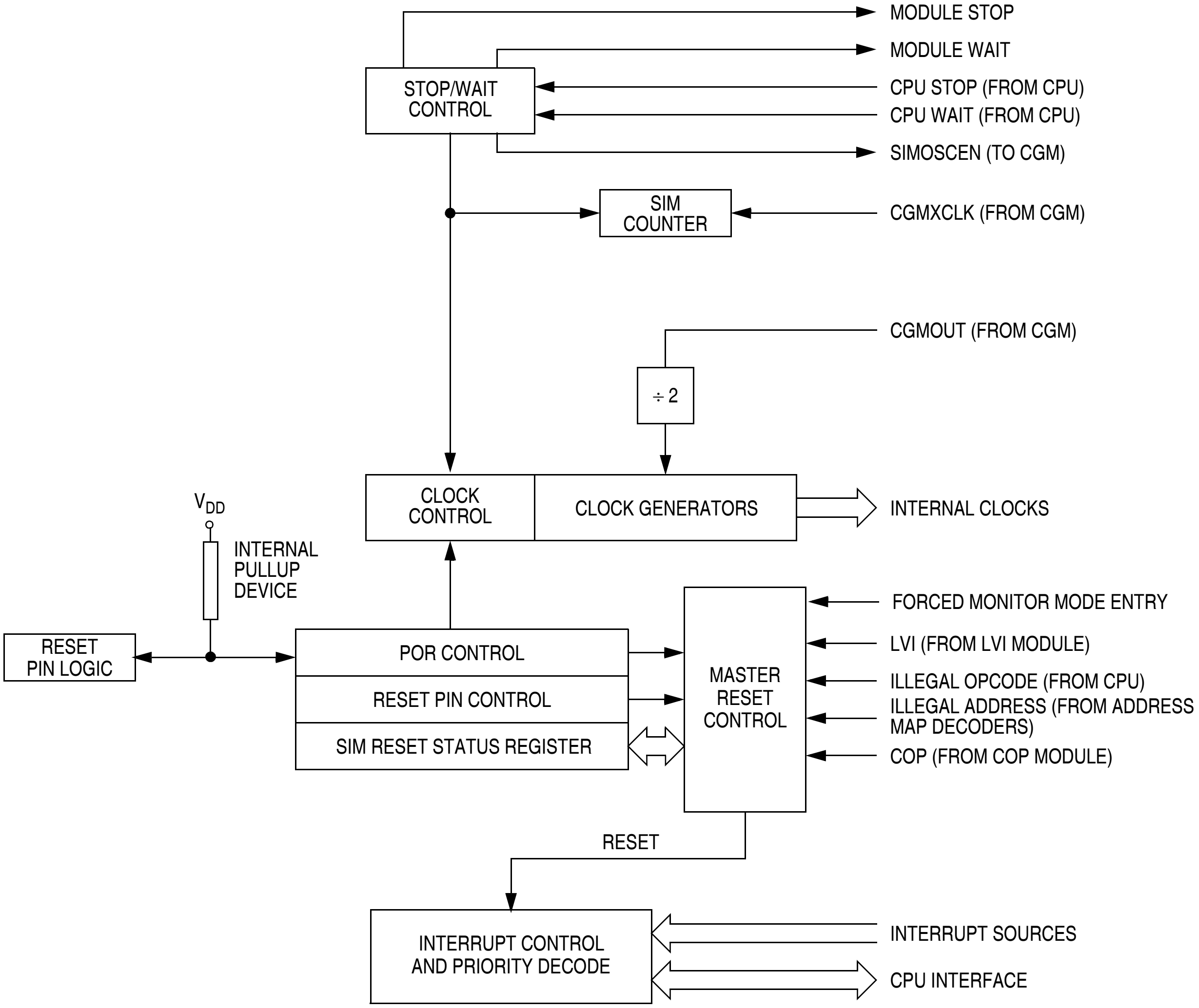


Figure 14-1. SIM Block Diagram

The SIM is responsible for:

- Bus clock generation and control for CPU and peripherals:
  - Stop/wait/reset/break entry and recovery
  - Internal clock control
- Master reset control, including power-on reset (POR) and computer operating properly (COP) timeout
- Interrupt arbitration

Table 14-1 shows the internal signal names used in this section.

Table 14-1. Signal Name Conventions

Signal Name	Description
CGMXCLK	Buffered version of OSC1 from clock generator module (CGM)
CGMVCLK	PLL output
CGMOUT	PLL-based or OSC1-based clock output from CGM module (Bus clock = CGMOUT divided by two)
IAB	Internal address bus
IDB	Internal data bus
PORRST	Signal from the power-on reset module to the SIM
IRST	Internal reset signal
R/$\overline{W}$	Read/write signal

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR) See page 169.	Read:	R	R	R	R	R	R	SBSW	R
		Write:							Note[1]	
		Reset:	0	0	0	0	0	0	0	0
$FE01	SIM Reset Status Register (SRSR) See page 170.	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		POR:	1	0	0	0	0	0	0	0
$FE03	SIM Break Flag Control Register (SBFCR) See page 171.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE04	Interrupt Status Register 1 (INT1) See page 165.	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE05	Interrupt Status Register 2 (INT2) See page 166.	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FE06	Interrupt Status Register 3 (INT3) See page 166.	Read:	0	0	0	0	0	0	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

1. Writing a 0 clears SBSW.

▨ = Unimplemented R = Reserved

Figure 14-2. SIM I/O Register Summary

14.2 SIM Bus Clock Control and Generation

The bus clock generator provides system clock signals for the CPU and peripherals on the MCU. The system clocks are generated from an incoming clock, CGMOUT, as shown in Figure 14-3. This clock originates from either an external oscillator or from the on-chip PLL.

14.2.1 Bus Timing

In user mode, the internal bus frequency is either the crystal oscillator output (CGMXCLK) divided by four or the PLL output (CGMVCLK) divided by four.

14.2.2 Clock Startup from POR or LVI Reset

When the power-on reset module or the low-voltage inhibit module generates a reset, the clocks to the CPU and peripherals are inactive and held in an inactive phase until after the 4096 CGMXCLK cycle POR timeout has completed. The $\overline{\text{RST}}$ pin is driven low by the SIM during this entire period. The bus clocks start upon completion of the timeout.

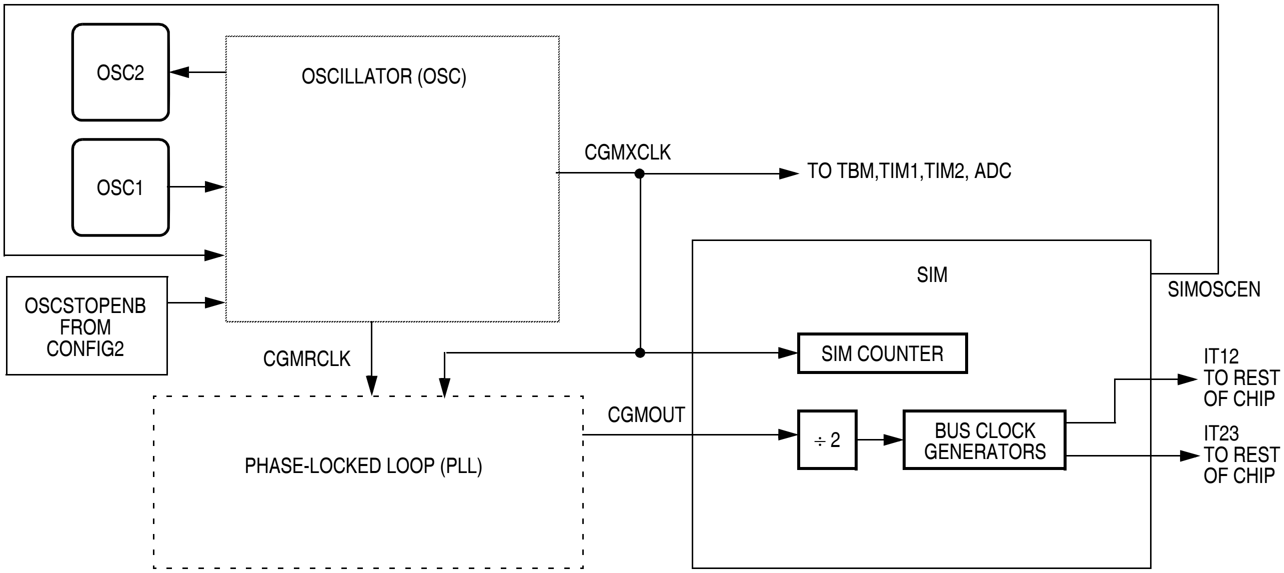


Figure 14-3. System Clock Signals

14.2.3 Clocks in Stop Mode and Wait Mode

Upon exit from stop mode by an interrupt or reset, the SIM allows CGMXCLK to clock the SIM counter. The CPU and peripheral clocks do not become active until after the stop delay timeout. This timeout is selectable as 4096 or 32 CGMXCLK cycles. See 14.6.2 Stop Mode.

In wait mode, the CPU clocks are inactive. The SIM also produces two sets of clocks for other modules. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

14.3 Reset and System Initialization

The MCU has these reset sources:

- Power-on reset module (POR)
- External reset pin ($\overline{\text{RST}}$)
- Computer operating properly module (COP)
- Low-voltage inhibit module (LVI)
- Illegal opcode
- Illegal address
- Forced monitor mode entry reset (MODRST)

All of these resets produce the vector \$FFFE:\$FFFF (\$FEFE:\$FEFF in monitor mode) and assert the internal reset signal (IRST). IRST causes all registers to be returned to their default values and all modules to be returned to their reset states.

An internal reset clears the SIM counter (see 14.4 SIM Counter), but an external reset does not. Each of the resets sets a corresponding bit in the SIM reset status register (SRSR). See 14.7 SIM Registers.

14.3.1 External Pin Reset

The $\overline{\text{RST}}$ pin circuit includes an internal pullup device. Pulling the asynchronous $\overline{\text{RST}}$ pin low halts all processing. The PIN bit of the SIM reset status register (SRSR) is set as long as $\overline{\text{RST}}$ is held low for a minimum of 67 CGMXCLK cycles, assuming that neither the POR nor the LVI was the source of the reset. See Table 14-2 for details. Figure 14-4 shows the relative timing.

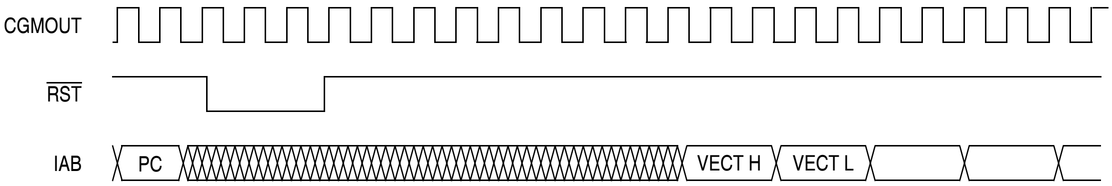


Figure 14-4. External Reset Timing

14.3.2 Active Resets from Internal Sources

All internal reset sources actively pull the $\overline{\text{RST}}$ pin low for 32 CGMXCLK cycles to allow resetting of external peripherals. The internal reset continues to be asserted for an additional 32 cycles at which point the reset vector will be fetched. See Figure 14-5. An internal reset can be caused by an illegal address, illegal opcode, COP timeout, LVI, or POR. See Figure 14-6.

NOTE

For LVI or POR resets, the SIM cycles through 4096 + 32 CGMXCLK cycles during which the SIM forces the $\overline{\text{RST}}$ *pin low. The internal reset signal then follows the sequence from the falling edge of* $\overline{\text{RST}}$ *shown in Figure 14-5.*

The COP reset is asynchronous to the bus clock.

The active reset feature allows the part to issue a reset to peripherals and other chips within a system built around the MCU.

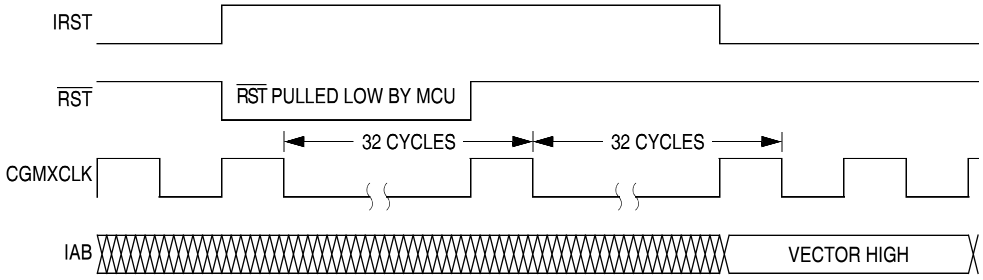


Figure 14-5. Internal Reset Timing

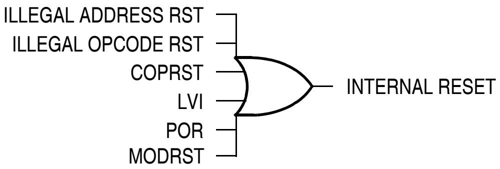


Figure 14-6. Sources of Internal Reset

Table 14-2. Reset Recovery Type

Reset Recovery Type	Actual Number of Cycles
POR/LVI	4163 (4096 + 64 + 3)
All others	67 (64 + 3)

14.3.2.1 Power-On Reset

When power is first applied to the MCU, the power-on reset module (POR) generates a pulse to indicate that power-on has occurred. The external reset pin ($\overline{\text{RST}}$) is held low while the SIM counter counts out 4096 + 32 CGMXCLK cycles. Thirty-two CGMXCLK cycles later, the CPU and memories are released from reset to allow the reset vector sequence to occur.

At power-on, these events occur:

- A POR pulse is generated.
- The internal reset signal is asserted.
- The SIM enables CGMOUT.
- Internal clocks to the CPU and modules are held inactive for 4096 CGMXCLK cycles to allow stabilization of the oscillator.
- The $\overline{\text{RST}}$ pin is driven low during the oscillator stabilization time.
- The POR bit of the SIM reset status register (SRSR) is set and all other bits in the register are cleared.

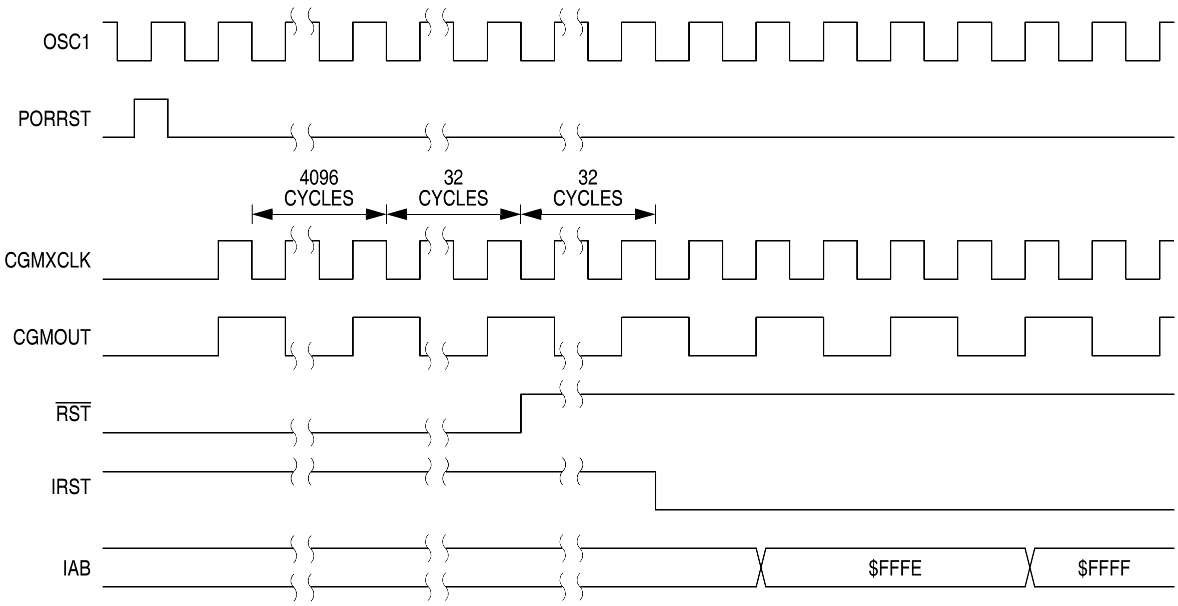


Figure 14-7. POR Recovery

14.3.2.2 Computer Operating Properly (COP) Reset

An input to the SIM is reserved for the COP reset signal. The overflow of the COP counter causes an internal reset and sets the COP bit in the SIM reset status register (SRSR). The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

The COP module is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ}}$ pin is held at V_{TST} while the MCU is in monitor mode. The COP module can be disabled only through combinational logic conditioned with the high voltage signal on the $\overline{\text{RST}}$ or the $\overline{\text{IRQ}}$ pin. This prevents the COP from becoming disabled as a result of external noise. During a break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP module.

14.3.2.3 Illegal Opcode Reset

The SIM decodes signals from the CPU to detect illegal instructions. An illegal instruction sets the ILOP bit in the SIM reset status register (SRSR) and causes a reset.

If the stop enable bit, STOP, in the CONFIG1 register is 0, the SIM treats the STOP instruction as an illegal opcode and causes an illegal opcode reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

14.3.2.4 Illegal Address Reset

An opcode fetch from an unmapped address generates an illegal address reset. The SIM verifies that the CPU is fetching an opcode prior to asserting the ILAD bit in the SIM reset status register (SRSR) and resetting the MCU. A data fetch from an unmapped address does not generate a reset. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

14.3.2.5 Low-Voltage Inhibit (LVI) Reset

The low-voltage inhibit module (LVI) asserts its output to the SIM when the V_{DD} voltage falls to the V_{TRIPF} voltage. The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{\text{RST}}$) is held low while the SIM counter counts out 4096 + 32 CGMXCLK cycles. Thirty-two CGMXCLK cycles later, the CPU is released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

14.3.2.6 Monitor Mode Entry Module Reset (MODRST)

The monitor mode entry module reset (MODRST) asserts its output to the SIM when monitor mode is entered in the condition where the reset vectors are erased ($FF) (see 18.3.1.1 Normal Monitor Mode). When MODRST gets asserted, an internal reset occurs. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

14.4 SIM Counter

The SIM counter is used by the power-on reset module (POR) and in stop mode recovery to allow the oscillator time to stabilize before enabling the internal bus (IBUS) clocks. The SIM counter also serves as a prescaler for the computer operating properly module (COP). The SIM counter is 12 bits long.

14.4.1 SIM Counter During Power-On Reset

The power-on reset module (POR) detects power applied to the MCU. At power-on, the POR circuit asserts the signal PORRST. Once the SIM is initialized, it enables the clock generation module (CGM) to drive the bus clock state machine.

14.4.2 SIM Counter During Stop Mode Recovery

The SIM counter also is used for stop mode recovery. The STOP instruction clears the SIM counter. After an interrupt, break, or reset, the SIM senses the state of the short stop recovery bit, SSREC, in the CONFIG1 register. If the SSREC bit is a 1, then the stop recovery is reduced from the normal delay of 4096 CGMXCLK cycles down to 32 CGMXCLK cycles. This is ideal for applications using crystals with the OSCSTOPENB bit set. External crystal applications should use the full stop recovery time, SSREC cleared, if the OSCSTOPENB bit is cleared.

14.4.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. See 14.6.2 Stop Mode for details. The SIM counter is free-running after all reset states. See 14.3.2 Active Resets from Internal Sources for counter control and internal reset recovery sequences.

14.5 Exception Control

Normal, sequential program execution can be changed in three different ways:

- Interrupts:
  - Maskable hardware CPU interrupts
  - Non-maskable software interrupt instruction (SWI)
- Reset
- Break interrupts

14.5.1 Interrupts

At the beginning of an interrupt, the CPU saves the CPU register contents on the stack and sets the interrupt mask (I bit) to prevent additional interrupts. At the end of an interrupt, the RTI instruction recovers the CPU register contents from the stack so that normal processing can resume. Figure 14-8 shows interrupt entry timing. Figure 14-9 shows interrupt recovery timing.

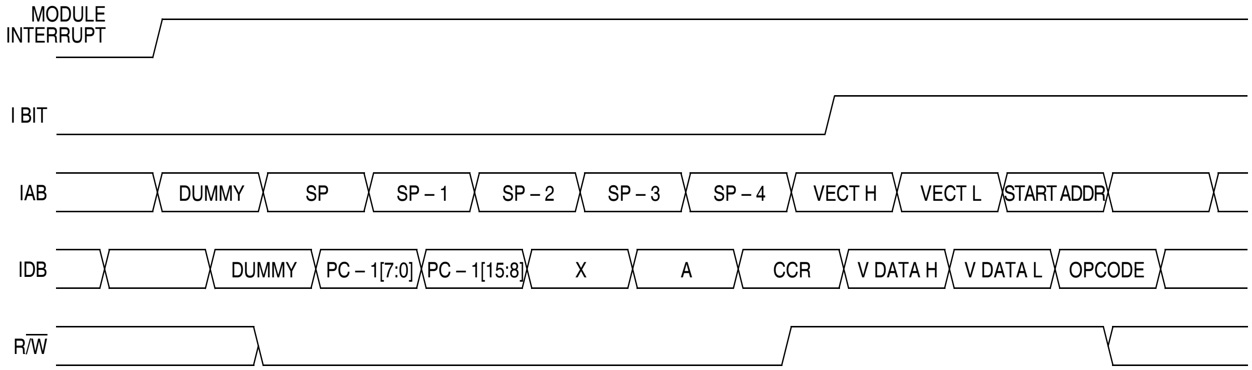


Figure 14-8. Interrupt Entry Timing

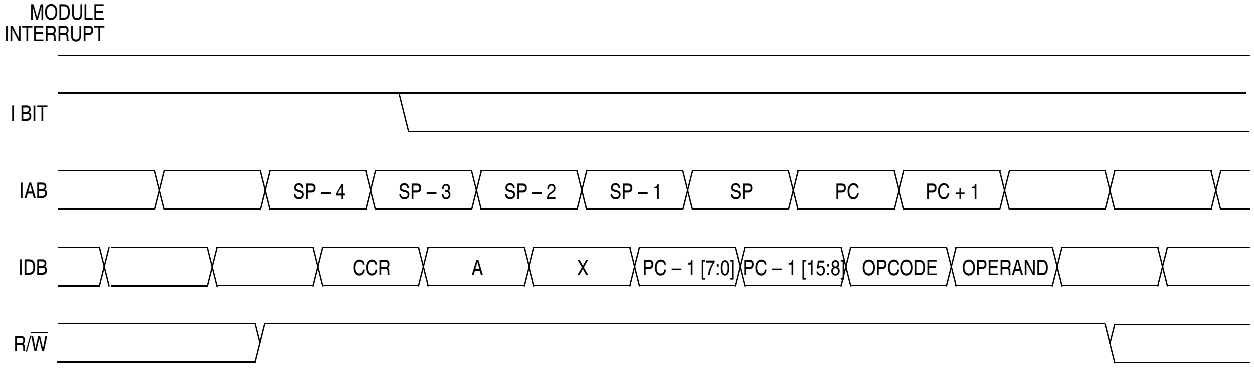


Figure 14-9. Interrupt Recovery Timing

Interrupts are latched, and arbitration is performed in the SIM at the start of interrupt processing. The arbitration result is a constant that the CPU uses to determine which vector to fetch. Once an interrupt is latched by the SIM, no other interrupt can take precedence, regardless of priority, until the latched interrupt is serviced (or the I bit is cleared). See Figure 14-10.

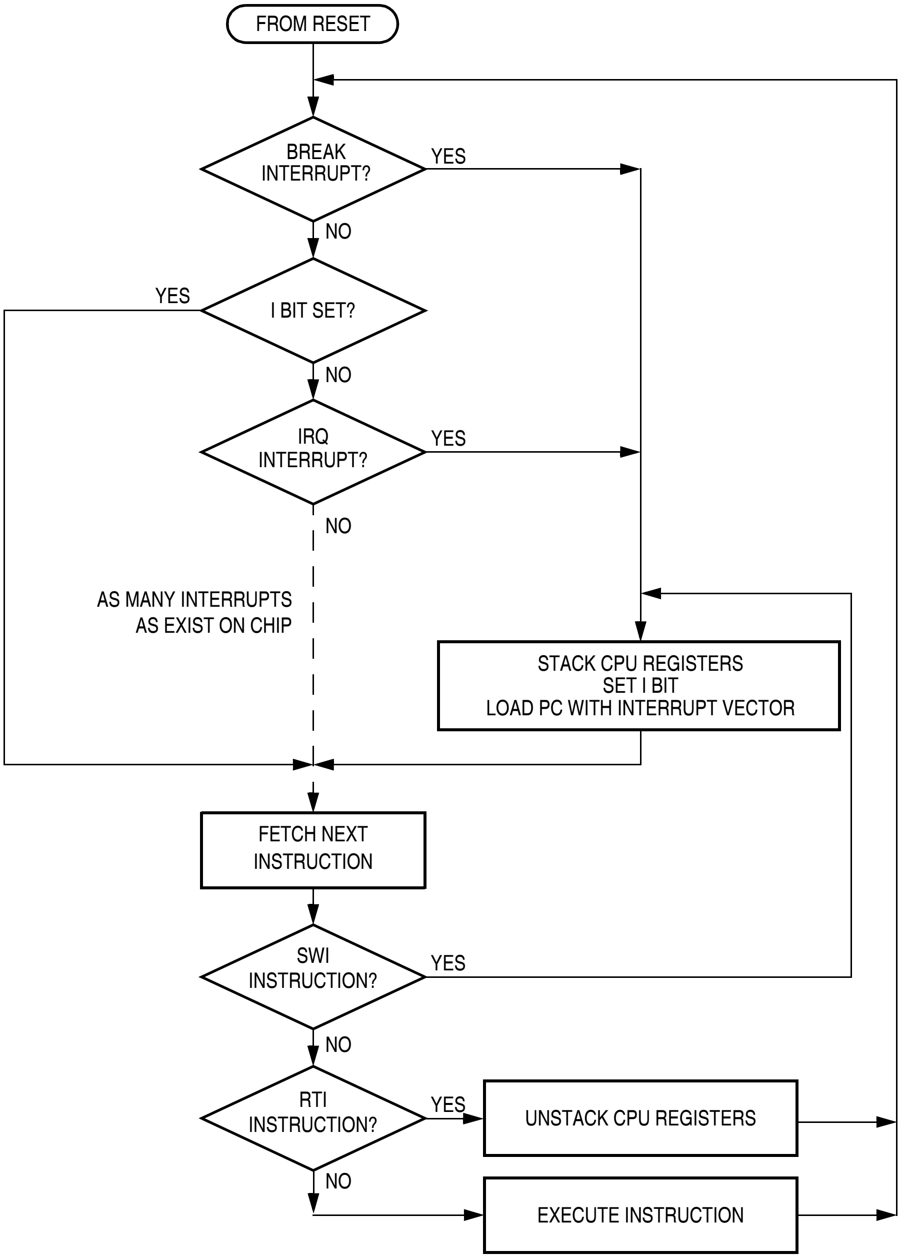


Figure 14-10. Interrupt Processing

14.5.1.1 Hardware Interrupts

A hardware interrupt does not stop the current instruction. Processing of a hardware interrupt begins after completion of the current instruction. When the current instruction is complete, the SIM checks all pending hardware interrupts. If interrupts are not masked (I bit clear in the condition code register) and if the corresponding interrupt enable bit is set, the SIM proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If more than one interrupt is pending at the end of an instruction execution, the highest priority interrupt is serviced first. Figure 14-11 demonstrates what happens when two interrupts are pending. If an interrupt is pending upon exit from the original interrupt service routine, the pending interrupt is serviced before the LDA instruction is executed.

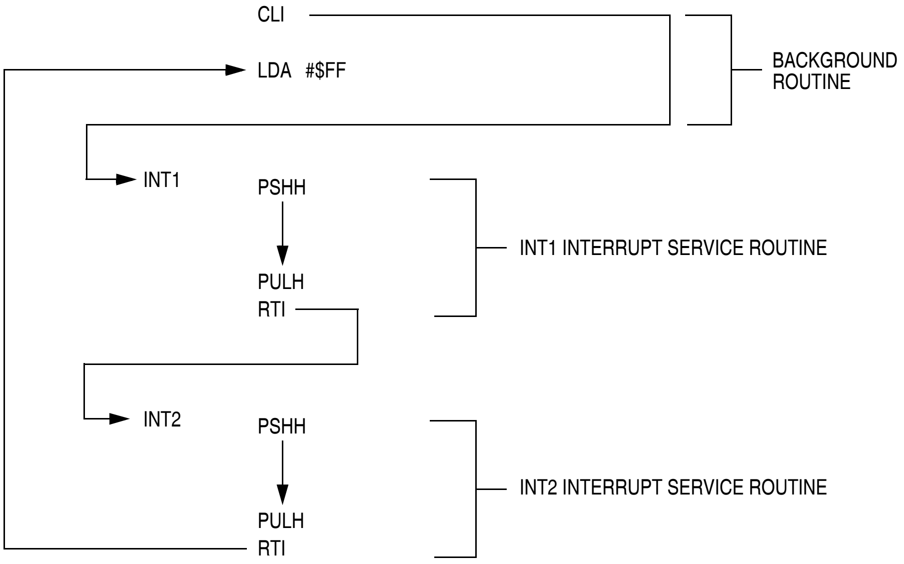


Figure 14-11. Interrupt Recognition Example

The LDA opcode is prefetched by both the INT1 and INT2 RTI instructions. However, in the case of the INT1 RTI prefetch, this is a redundant operation.

NOTE

To maintain compatibility with the M6805 Family, the H register is not pushed on the stack during interrupt entry. If the interrupt service routine modifies the H register or uses the indexed addressing mode, software should save the H register and then restore it prior to exiting the routine.

14.5.1.2 SWI Instruction

The SWI instruction is a non-maskable instruction that causes an interrupt regardless of the state of the interrupt mask (I bit) in the condition code register.

NOTE

A software interrupt pushes PC onto the stack. A software interrupt does not push PC – 1, as a hardware interrupt does.

14.5.1.3 *Interrupt Status Registers*

The flags in the interrupt status registers identify maskable interrupt sources. Table 14-3 summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

Table 14-3. Interrupt Sources

Priority	Interrupt Source	Interrupt Status Register Flag
Highest	Reset	—
	SWI instruction	—
	$\overline{\text{IRQ}}$ pin	I1
	CGM clock monitor	I2
	TIM1 channel 0	I3
	TIM1 channel 1	I4
	TIM1 overflow	I5
	TIM2 channel 0	I6
	TIM2 channel 1	I7
	TIM2 overflow	I8
	SPI receiver full	I9
	SPI transmitter empty	I10
	SCI receive error	I11
	SCI receive	I12
	SCI transmit	I13
	Keyboard	I14
	ADC conversion complete	I15
Lowest	Timebase module	I16

Interrupt Status Register 1

Address: $FE04

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	I6	I5	I4	I3	I2	I1	0	0
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 14-12. Interrupt Status Register 1 (INT1)

I6–I1 — Interrupt Flags 1–6

These flags indicate the presence of interrupt requests from the sources shown in Table 14-3.

1 = Interrupt request present
0 = No interrupt request present

Bit 0 and Bit 1 — Always read 0

Interrupt Status Register 2

Address: $FE05

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	I14	I13	I12	I11	I10	I9	I8	I7
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 14-13. Interrupt Status Register 2 (INT2)

I14–I7 — Interrupt Flags 14–7

These flags indicate the presence of interrupt requests from the sources shown in Table 14-3.

1 = Interrupt request present
0 = No interrupt request present

Interrupt Status Register 3

Address: $FE06

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	I16	I15
Write:	R	R	R	R	R	R	R	R
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 14-14. Interrupt Status Register 3 (INT3)

Bits 7–6 — Always read 0

I16 and I15 — Interrupt Flags 16 and 15

These flags indicate the presence of an interrupt request from the source shown in Table 14-3.

1 = Interrupt request present
0 = No interrupt request present

14.5.2 Reset

All reset sources always have equal and highest priority and cannot be arbitrated.

14.5.3 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output (see Chapter 17 Timer Interface Module (TIM1 and TIM2)). The SIM puts the CPU into the break state by forcing it to the SWI vector location. Refer to the break interrupt subsection of each module to see how each module is affected by the break state.

14.5.4 Status Flag Protection in Break Mode

The SIM controls whether status flags contained in other modules can be cleared during break mode. The user can select whether flags are protected from being cleared by properly initializing the break clear flag enable bit (BCFE) in the SIM break flag control register (SBFCR).

Protecting flags in break mode ensures that set flags will not be cleared while in break mode. This protection allows registers to be freely read and written during break mode without losing status flag information.

Setting the BCFE bit enables the clearing mechanisms. Once cleared in break mode, a flag remains cleared even when break mode is exited. Status flags with a 2-step clearing mechanism — for example, a read of one register followed by the read or write of another — are protected, even when the first step is accomplished prior to entering break mode. Upon leaving break mode, execution of the second step will clear the flag as normal.

14.6 Low-Power Modes

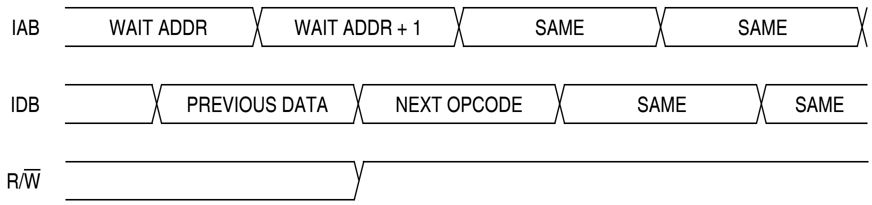
Executing the WAIT or STOP instruction puts the MCU in a low power- consumption mode for standby situations. The SIM holds the CPU in a non-clocked state. The operation of each of these modes is described in the following subsections. Both STOP and WAIT clear the interrupt mask (I) in the condition code register, allowing interrupts to occur.

14.6.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. Figure 14-15 shows the timing for wait mode entry.

A module that is active during wait mode can wakeup the CPU with an interrupt if the interrupt is enabled. Stacking for the interrupt begins one cycle after the WAIT instruction during which the interrupt occurred. In wait mode, the CPU clocks are inactive. Refer to the wait mode subsection of each module to see if the module is active or inactive in wait mode. Some modules can be programmed to be active in wait mode.

Wait mode also can be exited by a reset (or break in emulation mode). A break interrupt during wait mode sets the SIM break stop/wait bit, SBSW, in the SIM break status register (SBSR). If the COP disable bit, COPD, in the CONFIG1 register is 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.



Note: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 14-15. Wait Mode Entry Timing

Figure 14-16 and Figure 14-17 show the timing for WAIT recovery.

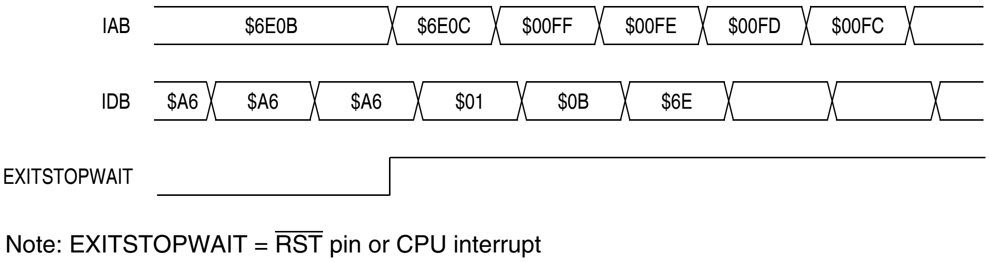


Figure 14-16. Wait Recovery from Interrupt

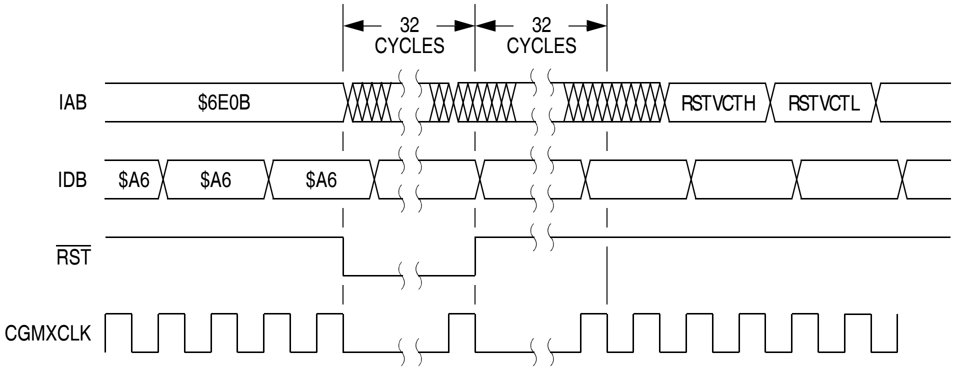


Figure 14-17. Wait Recovery from Internal Reset

14.6.2 Stop Mode

In stop mode, the SIM counter is reset and the system clocks are disabled. An interrupt request from a module can cause an exit from stop mode. Stacking for interrupts begins after the selected stop recovery time has elapsed. Reset causes an exit from stop mode.

The SIM disables the clock generator module outputs (CGMOUT and CGMXCLK) in stop mode, stopping the CPU and peripherals. Stop recovery time is selectable using the SSREC bit in the CONFIG1 register. If SSREC is set, stop recovery is reduced from the normal delay of 4096 CGMXCLK cycles down to 32. This is ideal for applications using canned oscillators that do not require long startup times from stop mode.

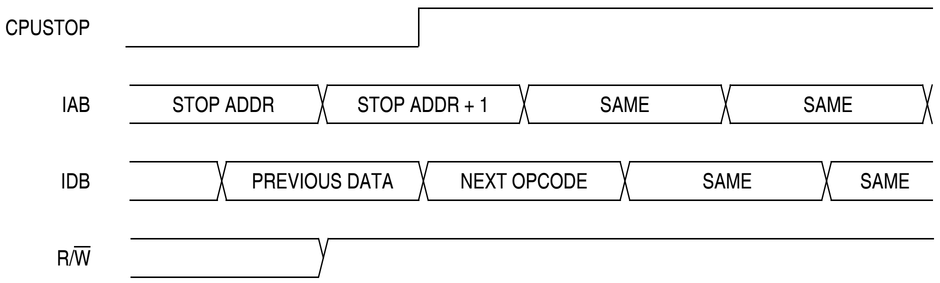
NOTE

External crystal applications should use the full stop recovery time by clearing the SSREC bit unless the OSCSTOPENB bit is set in CONFIG2.

The SIM counter is held in reset from the execution of the STOP instruction until the beginning of stop recovery. It is then used to time the recovery period. Figure 14-18 shows stop mode entry timing. Figure 14-19 shows stop mode recovery time from interrupt.

NOTE

To minimize stop current, all pins configured as inputs should be driven to a 1 or 0.



Note: Previous data can be operand data or the STOP opcode, depending on the last instruction.

Figure 14-18. Stop Mode Entry Timing

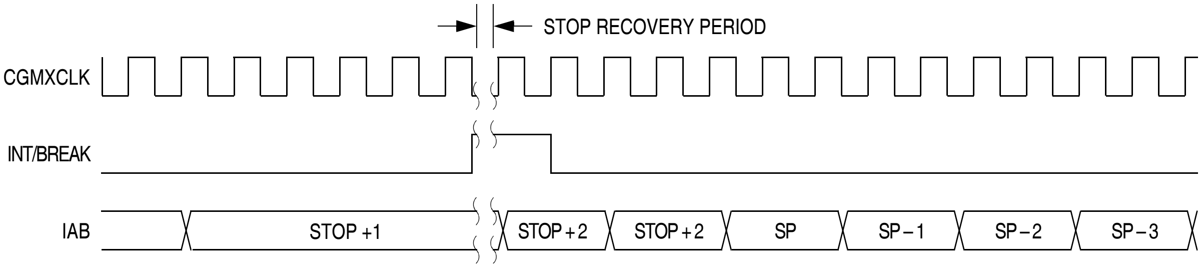


Figure 14-19. Stop Mode Recovery from Interrupt

14.7 SIM Registers

The SIM has three memory-mapped registers. Table 14-4 shows the mapping of these registers.

Table 14-4. SIM Registers

Address	Register	Access Mode
$FE00	SBSR	User
$FE01	SRSR	User
$FE03	SBFCR	User

14.7.1 SIM Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from wait mode. This register is only used in emulation mode.

Address: $FE00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R	R	R	R	R	R	SBSW	R
Write:							Note[1]	
Reset:	0	0	0	0	0	0	0	0

R = Reserved

1. Writing a 0 clears SBSW.

Figure 14-20. Break Status Register (SBSR)

SBSW — SIM Break Stop/Wait

SBSW can be read within the break state SWI routine. The user can modify the return address on the stack by subtracting one from it.

1 = Wait mode was exited by break interrupt.
0 = Wait mode was not exited by break interrupt.

14.7.2 SIM Reset Status Register

The SRSR register contains flags that show the source of the last reset. The status register will automatically clear after reading SRSR. A power-on reset sets the POR bit and clears all other bits in the register. All other reset sources set the individual flag bits but do not clear the register. More than one reset source can be flagged at any time depending on the conditions at the time of the internal or external reset. For example, the POR and LVI bit can both be set if the power supply has a slow rise time.

Address: $FE01

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	1	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 14-21. SIM Reset Status Register (SRSR)

POR — Power-On Reset Bit

1 = Last reset caused by POR circuit
0 = Read of SRSR

PIN — External Reset Bit

1 = Last reset caused by external reset pin ($\overline{\text{RST}}$)
0 = POR or read of SRSR

COP — Computer Operating Properly Reset Bit

1 = Last reset caused by COP counter
0 = POR or read of SRSR

ILOP — Illegal Opcode Reset Bit

1 = Last reset caused by an illegal opcode
0 = POR or read of SRSR

ILAD — Illegal Address Reset Bit (opcode fetches only)

1 = Last reset caused by an opcode fetch from an illegal address
0 = POR or read of SRSR

MODRST — Monitor Mode Entry Module Reset Bit

1 = Last reset caused by monitor mode entry when vector locations $FFFE and $FFFF are $FF after POR while $\overline{\text{IRQ}}$ = V_{DD}
0 = POR or read of SRSR

LVI — Low-Voltage Inhibit Reset Bit

1 = Last reset caused by the LVI circuit
0 = POR or read of SRSR

14.7.3 SIM Break Flag Control Register

The SIM break flag control register (SBFCR) contains a bit that enables software to clear status bits while the MCU is in a break state.

Address: $FE03

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BCFE	R	R	R	R	R	R	R
Reset:	0							

R = Reserved

Figure 14-22. SIM Break Flag Control Register (BFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break
0 = Status bits not clearable during break

Chapter 15
Serial Peripheral Interface (SPI) Module

15.1 Introduction

This section describes the serial peripheral interface (SPI) module, which allows full-duplex, synchronous, serial communications with peripheral devices.

The text that follows describes the SPI. The SPI I/O pin names are $\overline{\text{SS}}$ (slave select), SPSCK (SPI serial clock), MOSI (master out slave in), and MISO (master in/slave out). The SPI shares four I/O pins with four parallel I/O ports.

15.2 Features

Features of the SPI module include:

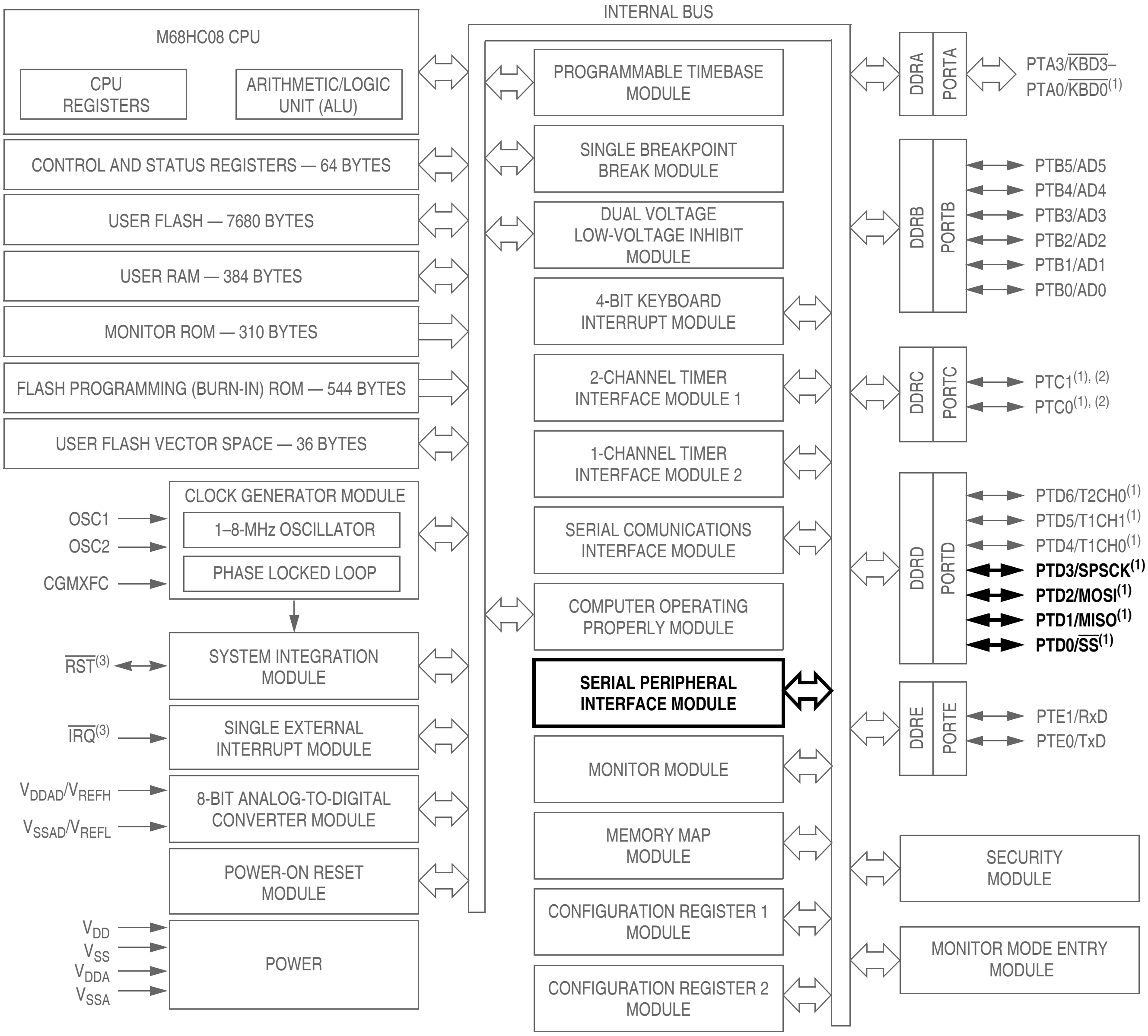
- Full-duplex operation
- Master and slave modes
- Double-buffered operation with separate transmit and receive registers
- Four master mode frequencies (maximum = bus frequency ÷ 2)
- Maximum slave mode frequency = bus frequency
- Serial clock with programmable polarity and phase
- Two separately enabled interrupts:
  - SPRF (SPI receiver full)
  - SPTE (SPI transmitter empty)
- Mode fault error flag with CPU interrupt capability
- Overflow error flag with CPU interrupt capability
- Programmable wired-OR mode
- I/O (input/output) port bit(s) software configurable with pullup device(s) if configured as input port bit(s)

15.3 Functional Description

The SPI module allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs. Software can poll the SPI status flags or SPI operation can be interrupt driven.

If a port bit is configured for input, then an internal pullup device may be enabled for that port bit.

The following paragraphs describe the operation of the SPI module. Refer to Figure 15-3 for a summary of the SPI I/O registers.



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 15-1. Block Diagram Highlighting SPI Block and Pins

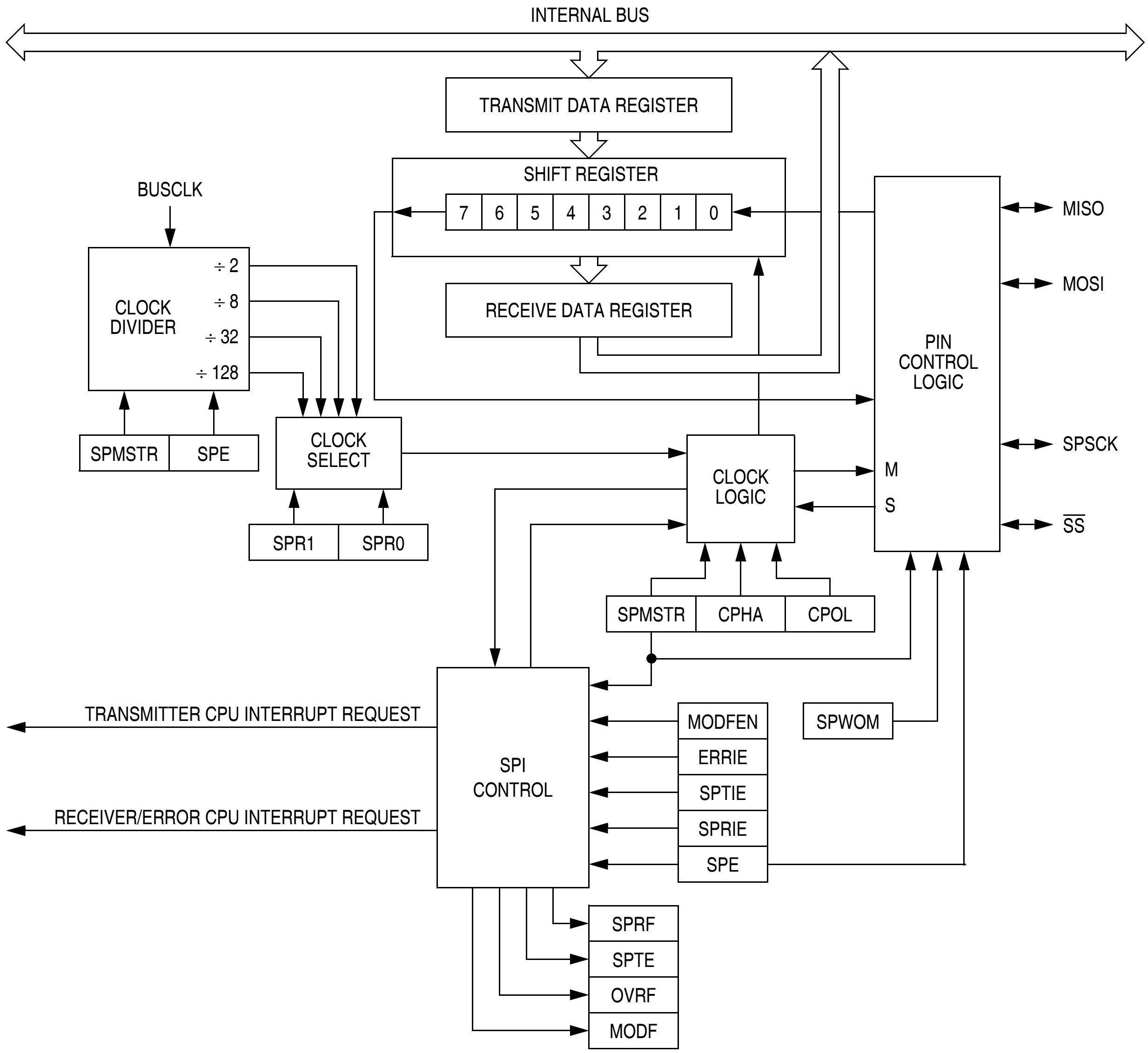


Figure 15-2. SPI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0010	SPI Control Register (SPCR) See page 189.	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0011	SPI Status and Control Register (SPSCR) See page 190.	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:	▨		▨	▨	▨			
		Reset:	0	0	0	0	1	0	0	0
$0012	SPI Data Register (SPDR) See page 192.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							

R = Reserved

▨ = Unimplemented

Figure 15-3. SPI I/O Register Summary

15.3.1 Master Mode

The SPI operates in master mode when the SPI master bit, SPMSTR, is set.

NOTE

In a multi-SPI system, configure the SPI modules as master or slave before enabling them. Enable the master SPI before enabling the slave SPI. Disable the slave SPI before disabling the master SPI. See 15.12.1 SPI Control Register.

Only a master SPI module can initiate transmissions. Software begins the transmission from a master SPI module by writing to the transmit data register. If the shift register is empty, the byte immediately transfers to the shift register, setting the SPI transmitter empty bit, SPTE. The byte begins shifting out on the MOSI pin under the control of the serial clock. See Figure 15-4.

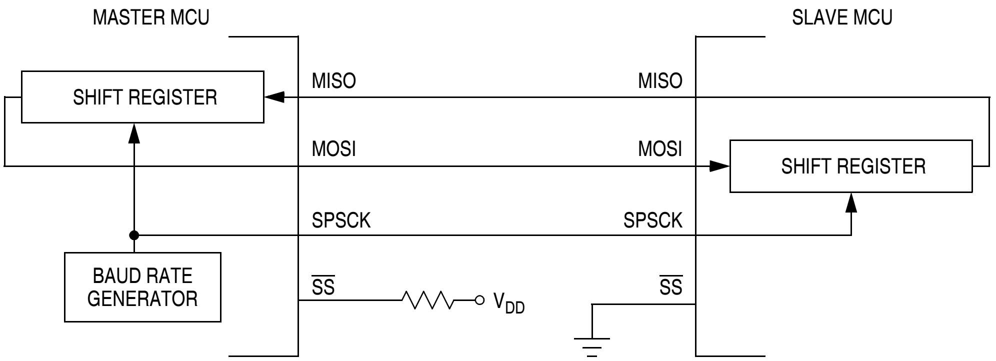


Figure 15-4. Full-Duplex Master-Slave Connections

The SPR1 and SPR0 bits control the baud rate generator and determine the speed of the shift register. (See 15.12.2 SPI Status and Control Register.) Through the SPSCK pin, the baud rate generator of the master also controls the shift register of the slave peripheral.

As the byte shifts out on the MOSI pin of the master, another byte shifts in from the slave on the master's MISO pin. The transmission ends when the receiver full bit, SPRF, becomes set. At the same time that SPRF becomes set, the byte from the slave transfers to the receive data register. In normal operation, SPRF signals the end of a transmission. Software clears SPRF by reading the SPI status and control register with SPRF set and then reading the SPI data register. Writing to the SPI data register (SPDR) clears SPTE.

15.3.2 Slave Mode

The SPI operates in slave mode when SPMSTR is clear. In slave mode, the SPSCK pin is the input for the serial clock from the master MCU. Before a data transmission occurs, the $\overline{SS}$ pin of the slave SPI must be low. $\overline{SS}$ must remain low until the transmission is complete. See 15.6.2 Mode Fault Error.

In a slave SPI module, data enters the shift register under the control of the serial clock from the master SPI module. After a byte enters the shift register of a slave SPI, it transfers to the receive data register, and the SPRF bit is set. To prevent an overflow condition, slave software then must read the receive data register before another full byte enters the shift register.

The maximum frequency of the SPSCK for an SPI configured as a slave is the bus clock speed (which is twice as fast as the fastest master SPSCK clock that can be generated). The frequency of the SPSCK for an SPI configured as a slave does not have to correspond to any SPI baud rate. The baud rate only controls the speed of the SPSCK generated by an SPI configured as a master. Therefore, the frequency of the SPSCK for an SPI configured as a slave can be any frequency less than or equal to the bus speed.

When the master SPI starts a transmission, the data in the slave shift register begins shifting out on the MISO pin. The slave can load its shift register with a new byte for the next transmission by writing to its transmit data register. The slave must write to its transmit data register at least one bus cycle before the master starts the next transmission. Otherwise, the byte already in the slave shift register shifts out on the MISO pin. Data written to the slave shift register during a transmission remains in a buffer until the end of the transmission.

When the clock phase bit (CPHA) is set, the first edge of SPSCK starts a transmission. When CPHA is clear, the falling edge of $\overline{\text{SS}}$ starts a transmission. See 15.4 Transmission Formats.

NOTE

SPSCK must be in the proper idle state before the slave is enabled to prevent SPSCK from appearing as a clock edge.

15.4 Transmission Formats

During an SPI transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock synchronizes shifting and sampling on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. On a master SPI device, the slave select line can optionally be used to indicate multiple-master bus contention.

15.4.1 Clock Phase and Polarity Controls

Software can select any of four combinations of serial clock (SPSCK) phase and polarity using two bits in the SPI control register (SPCR). The clock polarity is specified by the CPOL control bit, which selects an active high or low clock and has no significant effect on the transmission format.

The clock phase (CPHA) control bit selects one of two fundamentally different transmission formats. The clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

NOTE

Before writing to the CPOL bit or the CPHA bit, disable the SPI by clearing the SPI enable bit (SPE).

15.4.2 Transmission Format When CPHA = 0

Figure 15-5 shows an SPI transmission in which CPHA = 0. The figure should not be used as a replacement for data sheet parametric information.

Two waveforms are shown for SPSCK: one for CPOL = 0 and another for CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the serial clock (SPSCK), master in/slave out (MISO), and master out/slave in (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{\text{SS}}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{\text{SS}}$) is low, so that only the selected slave drives to the master. The $\overline{\text{SS}}$ pin of the master is not shown but is assumed to be inactive. The $\overline{\text{SS}}$ pin of the master must be high or must be reconfigured as general-purpose I/O not affecting the SPI. (See 15.6.2 Mode Fault Error.) When CPHA = 0, the first SPSCK edge is the MSB capture strobe. Therefore, the slave must begin driving its data before the first SPSCK edge, and a falling edge on the $\overline{\text{SS}}$ pin is used to start the slave data transmission. The slave's $\overline{\text{SS}}$ pin must be toggled back to high and then low again between each byte transmitted as shown in Figure 15-6.

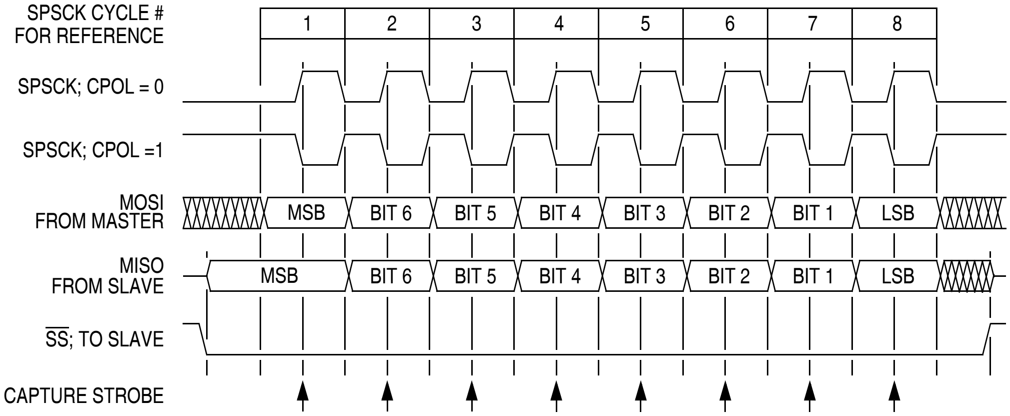


Figure 15-5. Transmission Format (CPHA = 0)

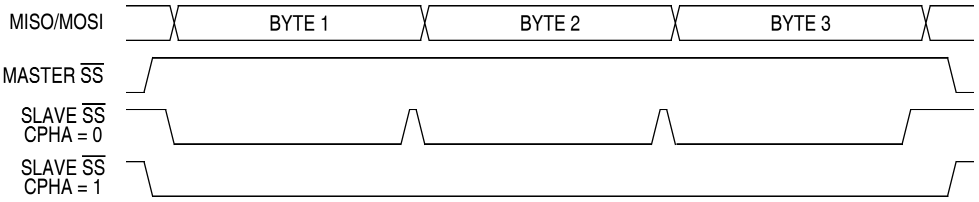


Figure 15-6. CPHA/$\overline{\text{SS}}$ Timing

When CPHA = 0 for a slave, the falling edge of $\overline{\text{SS}}$ indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the MSB of its data. Once the transmission begins, no new data is allowed into the shift register from the transmit data register. Therefore, the SPI data register of the slave must be loaded with transmit data before the falling edge of $\overline{\text{SS}}$. Any data written after the falling edge is stored in the transmit data register and transferred to the shift register after the current transmission.

15.4.3 Transmission Format When CPHA = 1

Figure 15-7 shows an SPI transmission in which CPHA = 1. The figure should not be used as a replacement for data sheet parametric information. Two waveforms are shown for SPSCK: one for CPOL = 0 and another for CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the serial clock (SPSCK), master in/slave out (MISO), and master out/slave in (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{\text{SS}}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{\text{SS}}$) is low, so that only the selected slave drives to the master. The $\overline{\text{SS}}$ pin of the master is not shown but is assumed to be inactive. The $\overline{\text{SS}}$

pin of the master must be high or must be reconfigured as general-purpose I/O not affecting the SPI. (See 15.6.2 Mode Fault Error.) When CPHA = 1, the master begins driving its MOSI pin on the first SPSCK edge. Therefore, the slave uses the first SPSCK edge as a start transmission signal. The $\overline{SS}$ pin can remain low between transmissions. This format may be preferable in systems having only one master and only one slave driving the MISO data line.

When CPHA = 1 for a slave, the first edge of the SPSCK indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the MSB of its data. Once the transmission begins, no new data is allowed into the shift register from the transmit data register. Therefore, the SPI data register of the slave must be loaded with transmit data before the first edge of SPSCK. Any data written after the first edge is stored in the transmit data register and transferred to the shift register after the current transmission.

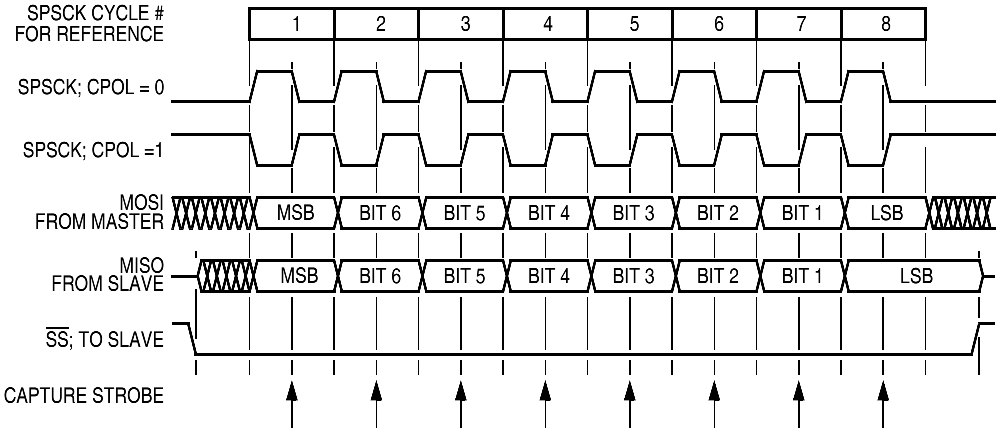


Figure 15-7. Transmission Format (CPHA = 1)

15.4.4 Transmission Initiation Latency

When the SPI is configured as a master (SPMSTR = 1), writing to the SPDR starts a transmission. CPHA has no effect on the delay to the start of the transmission, but it does affect the initial state of the SPSCK signal. When CPHA = 0, the SPSCK signal remains inactive for the first half of the first SPSCK cycle. When CPHA = 1, the first SPSCK cycle begins with an edge on the SPSCK line from its inactive to its active level. The SPI clock rate (selected by SPR1:SPR0) affects the delay from the write to SPDR and the start of the SPI transmission. (See Figure 15-8.) The internal SPI clock in the master is a free-running derivative of the internal MCU clock. To conserve power, it is enabled only when both the SPE and SPMSTR bits are set. Since the SPI clock is free-running, it is uncertain where the write to the SPDR occurs relative to the slower SPSCK. This uncertainty causes the variation in the initiation delay shown in Figure 15-8. This delay is no longer than a single SPI bit time. That is, the maximum delay is two MCU bus cycles for DIV2, eight MCU bus cycles for DIV8, 32 MCU bus cycles for DIV32, and 128 MCU bus cycles for DIV128.

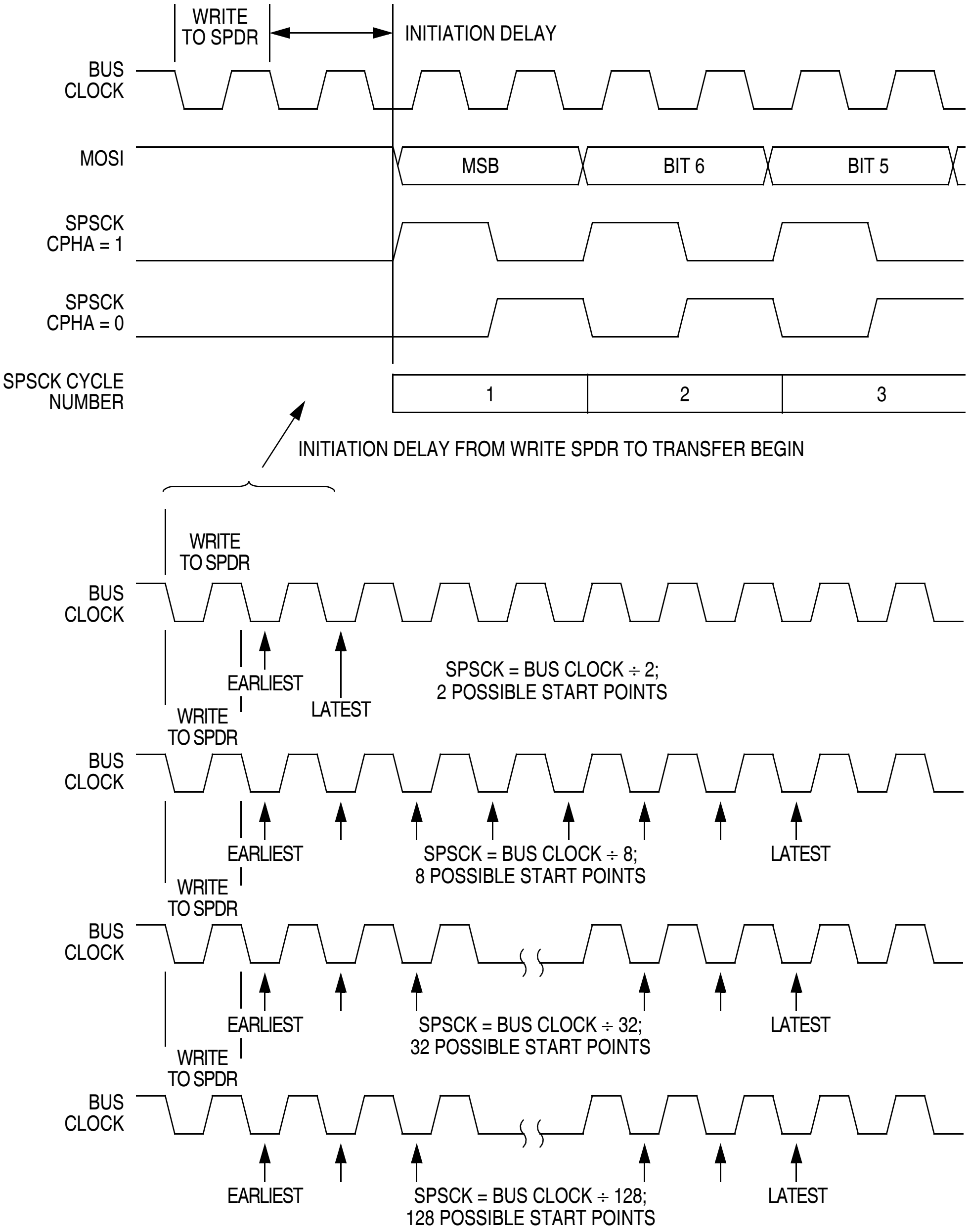
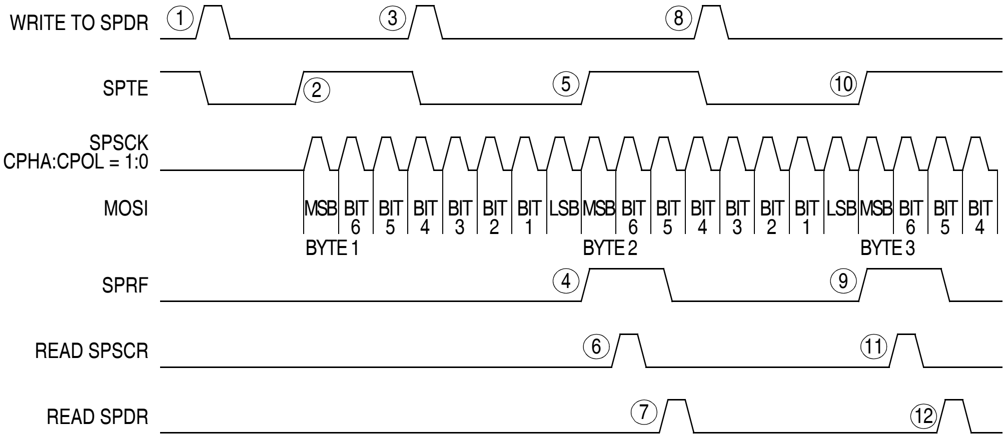


Figure 15-8. Transmission Start Delay (Master)

15.5 Queuing Transmission Data

The double-buffered transmit data register allows a data byte to be queued and transmitted. For an SPI configured as a master, a queued data byte is transmitted immediately after the previous transmission has completed. The SPI transmitter empty flag (SPTE) indicates when the transmit data buffer is ready to accept new data. Write to the transmit data register only when SPTE is high. Figure 15-9 shows the timing associated with doing back-to-back transmissions with the SPI (SPSCK has CPHA: CPOL = 1:0).



① CPU WRITES BYTE 1 TO SPDR, CLEARING SPTE BIT.

② BYTE 1 TRANSFERS FROM TRANSMIT DATA REGISTER TO SHIFT REGISTER, SETTING SPTE BIT.

③ CPU WRITES BYTE 2 TO SPDR, QUEUEING BYTE 2 AND CLEARING SPTE BIT.

④ FIRST INCOMING BYTE TRANSFERS FROM SHIFT REGISTER TO RECEIVE DATA REGISTER, SETTING SPRF BIT.

⑤ BYTE 2 TRANSFERS FROM TRANSMIT DATA REGISTER TO SHIFT REGISTER, SETTING SPTE BIT.

⑥ CPU READS SPSCR WITH SPRF BIT SET.

⑦ CPU READS SPDR, CLEARING SPRF BIT.

⑧ CPU WRITES BYTE 3 TO SPDR, QUEUEING BYTE 3 AND CLEARING SPTE BIT.

⑨ SECOND INCOMING BYTE TRANSFERS FROM SHIFT REGISTER TO RECEIVE DATA REGISTER, SETTING SPRF BIT.

⑩ BYTE 3 TRANSFERS FROM TRANSMIT DATA REGISTER TO SHIFT REGISTER, SETTING SPTE BIT.

⑪ CPU READS SPSCR WITH SPRF BIT SET.

⑫ CPU READS SPDR, CLEARING SPRF BIT.

Figure 15-9. SPRF/SPTE CPU Interrupt Timing

The transmit data buffer allows back-to-back transmissions without the slave precisely timing its writes between transmissions as in a system with a single data buffer. Also, if no new data is written to the data buffer, the last value contained in the shift register is the next data word to be transmitted.

For an idle master or idle slave that has no data loaded into its transmit buffer, the SPTE is set again no more than two bus cycles after the transmit buffer empties into the shift register. This allows the user to queue up a 16-bit value to send. For an already active slave, the load of the shift register cannot occur until the transmission is completed. This implies that a back-to-back write to the transmit data register is not possible. SPTE indicates when the next write can occur.

15.6 Error Conditions

The following flags signal SPI error conditions:

- Overflow (OVRF) — Failing to read the SPI data register before the next full byte enters the shift register sets the OVRF bit. The new byte does not transfer to the receive data register, and the unread byte still can be read. OVRF is in the SPI status and control register.
- Mode fault error (MODF) — The MODF bit indicates that the voltage on the slave select pin ($\overline{SS}$) is inconsistent with the mode of the SPI. MODF is in the SPI status and control register.

15.6.1 Overflow Error

The overflow flag (OVRF) becomes set if the receive data register still has unread data from a previous transmission when the capture strobe of bit 1 of the next transmission occurs. The bit 1 capture strobe occurs in the middle of SPSCK cycle 7 (see Figure 15-5 and Figure 15-7.) If an overflow occurs, all data received after the overflow and before the OVRF bit is cleared does not transfer to the receive data register and does not set the SPI receiver full bit (SPRF). The unread data that transferred to the receive data register before the overflow occurred can still be read. Therefore, an overflow error always indicates the loss of data. Clear the overflow flag by reading the SPI status and control register and then reading the SPI data register.

OVRF generates a receiver/error CPU interrupt request if the error interrupt enable bit (ERRIE) is also set. The SPRF, MODF, and OVRF interrupts share the same CPU interrupt vector (see Figure 15-12.) It is not possible to enable MODF or OVRF individually to generate a receiver/error CPU interrupt request. However, leaving MODFEN low prevents MODF from being set.

If the CPU SPRF interrupt is enabled and the OVRF interrupt is not, watch for an overflow condition. Figure 15-10 shows how it is possible to miss an overflow. The first part of Figure 15-10 shows how it is possible to read the SPSCR and SPDR to clear the SPRF without problems. However, as illustrated by the second transmission example, the OVRF bit can be set in between the time that SPSCR and SPDR are read.

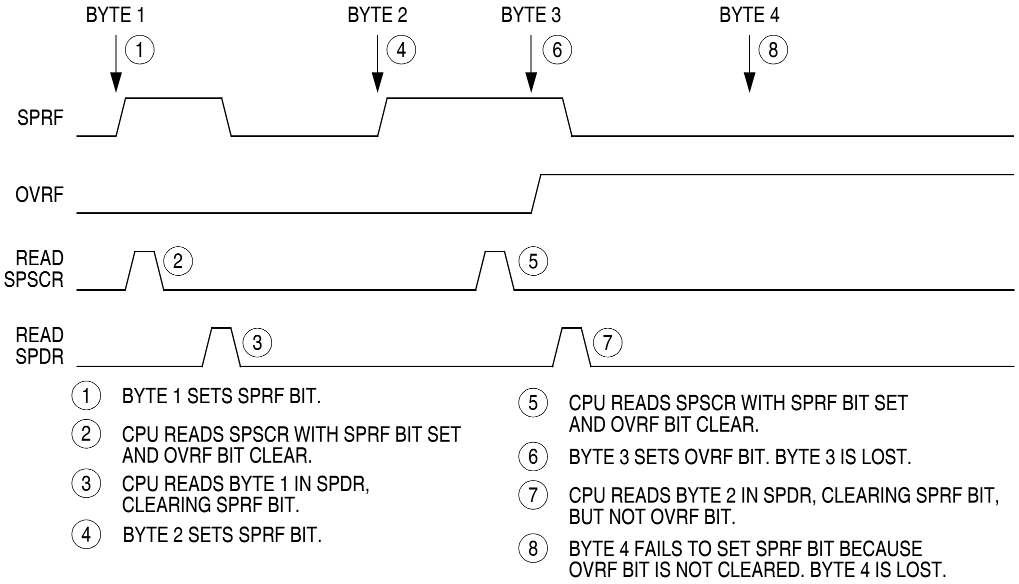
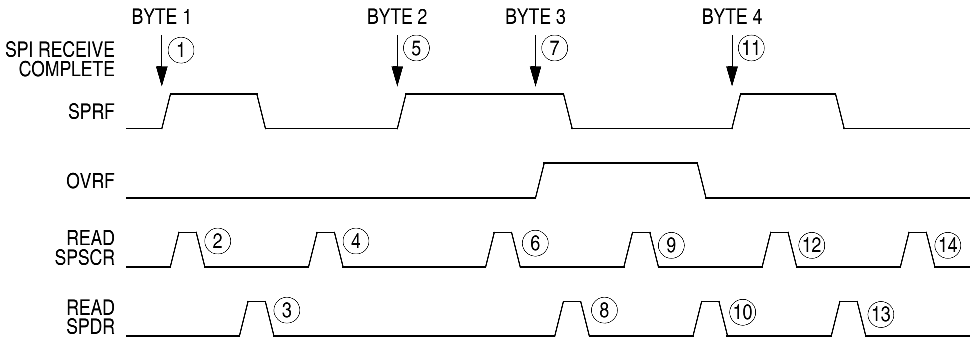


Figure 15-10. Missed Read of Overflow Condition

In this case, an overflow can be missed easily. Since no more SPRF interrupts can be generated until this OVRF is serviced, it is not obvious that bytes are being lost as more transmissions are completed. To prevent this, either enable the OVRF interrupt or do another read of the SPSCR following the read of the SPDR. This ensures that the OVRF was not set before the SPRF was cleared and that future transmissions can set the SPRF bit. Figure 15-11 illustrates this process. Generally, to avoid this second SPSCR read, enable the OVRF to the CPU by setting the ERRIE bit.



① BYTE 1 SETS SPRF BIT.

② CPU READS SPSCR WITH SPRF BIT SET AND OVRF BIT CLEAR.

③ CPU READS BYTE 1 IN SPDR, CLEARING SPRF BIT.

④ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

⑤ BYTE 2 SETS SPRF BIT.

⑥ CPU READS SPSCR WITH SPRF BIT SET AND OVRF BIT CLEAR.

⑦ BYTE 3 SETS OVRF BIT. BYTE 3 IS LOST.

⑧ CPU READS BYTE 2 IN SPDR, CLEARING SPRF BIT.

⑨ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

⑩ CPU READS BYTE 2 SPDR, CLEARING OVRF BIT.

⑪ BYTE 4 SETS SPRF BIT.

⑫ CPU READS SPSCR.

⑬ CPU READS BYTE 4 IN SPDR, CLEARING SPRF BIT.

⑭ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

Figure 15-11. Clearing SPRF When OVRF Interrupt Is Not Enabled

15.6.2 Mode Fault Error

Setting SPMSTR selects master mode and configures the SPSCK and MOSI pins as outputs and the MISO pin as an input. Clearing SPMSTR selects slave mode and configures the SPSCK and MOSI pins as inputs and the MISO pin as an output. The mode fault bit, MODF, becomes set any time the state of the slave select pin, $\overline{SS}$, is inconsistent with the mode selected by SPMSTR.

To prevent SPI pin contention and damage to the MCU, a mode fault error occurs if:

- The $\overline{SS}$ pin of a slave SPI goes high during a transmission
- The $\overline{SS}$ pin of a master SPI goes low at any time

For the MODF flag to be set, the mode fault error enable bit (MODFEN) must be set. Clearing the MODFEN bit does not clear the MODF flag but does prevent MODF from being set again after MODF is cleared.

MODF generates a receiver/error CPU interrupt request if the error interrupt enable bit (ERRIE) is also set. The SPRF, MODF, and OVRF interrupts share the same CPU interrupt vector. (See Figure 15-12.) It is not possible to enable MODF or OVRF individually to generate a receiver/error CPU interrupt request. However, leaving MODFEN low prevents MODF from being set.

In a master SPI with the mode fault enable bit (MODFEN) set, the mode fault flag (MODF) is set if $\overline{SS}$ goes low. A mode fault in a master SPI causes the following events to occur:

- If ERRIE = 1, the SPI generates an SPI receiver/error CPU interrupt request.
- The SPE bit is cleared.
- The SPTE bit is set.
- The SPI state counter is cleared.
- The data direction register of the shared I/O port regains control of port drivers.

NOTE

To prevent bus contention with another master SPI after a mode fault error, clear all SPI bits of the data direction register of the shared I/O port before enabling the SPI.

When configured as a slave (SPMSTR = 0), the MODF flag is set if $\overline{SS}$ goes high during a transmission. When CPHA = 0, a transmission begins when $\overline{SS}$ goes low and ends once the incoming SPSCK goes back to its idle level following the shift of the eighth data bit. When CPHA = 1, the transmission begins when the SPSCK leaves its idle level and $\overline{SS}$ is already low. The transmission continues until the SPSCK returns to its idle level following the shift of the last data bit. See 15.4 Transmission Formats.

NOTE

Setting the MODF flag does not clear the SPMSTR bit. SPMSTR has no function when SPE = 0. Reading SPMSTR when MODF = 1 shows the difference between a MODF occurring when the SPI is a master and when it is a slave.

NOTE

When CPHA = 0, a MODF occurs if a slave is selected ($\overline{SS}$ is low) and later unselected ($\overline{SS}$ is high) even if no SPSCK is sent to that slave. This happens because $\overline{SS}$ low indicates the start of the transmission (MISO driven out with the value of MSB) for CPHA = 0. When CPHA = 1, a slave can be selected and then later unselected with no transmission occurring. Therefore, MODF does not occur since a transmission was never begun.

In a slave SPI (MSTR = 0), MODF generates an SPI receiver/error CPU interrupt request if the ERRIE bit is set. The MODF bit does not clear the SPE bit or reset the SPI in any way. Software can abort the SPI transmission by clearing the SPE bit of the slave.

NOTE

A high on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. Also, the slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.

To clear the MODF flag, read the SPSCR with the MODF bit set and then write to the SPCR register. This entire clearing mechanism must occur with no MODF condition existing or else the flag is not cleared.

15.7 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests. See Table 15-1.

Table 15-1. SPI Interrupts

Flag	Request
SPTE Transmitter empty	SPI transmitter CPU interrupt request (SPTIE = 1, SPE = 1)
SPRF Receiver full	SPI receiver CPU interrupt request (SPRIE = 1)
OVRF Overflow	SPI receiver/error interrupt request (ERRIE = 1)
MODF Mode fault	SPI receiver/error interrupt request (ERRIE = 1)

Reading the SPI status and control register with SPRF set and then reading the receive data register clears SPRF. The clearing mechanism for the SPTE flag is always just a write to the transmit data register.

The SPI transmitter interrupt enable bit (SPTIE) enables the SPTE flag to generate transmitter CPU interrupt requests, provided that the SPI is enabled (SPE = 1).

The SPI receiver interrupt enable bit (SPRIE) enables SPRF to generate receiver CPU interrupt requests, regardless of the state of SPE. See Figure 15-12.

The error interrupt enable bit (ERRIE) enables both the MODF and OVRF bits to generate a receiver/error CPU interrupt request.

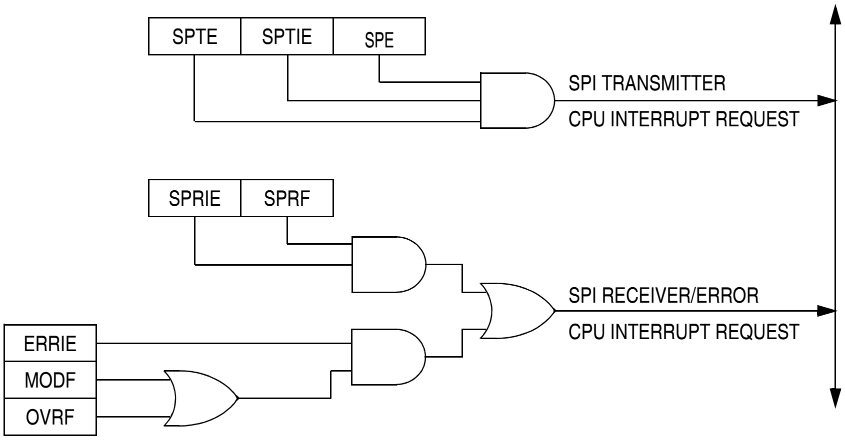


Figure 15-12. SPI Interrupt Request Generation

The mode fault enable bit (MODFEN) can prevent the MODF flag from being set so that only the OVRF bit is enabled by the ERRIE bit to generate receiver/error CPU interrupt requests.

- The following sources in the SPI status and control register can generate CPU interrupt requests:
- SPI receiver full bit (SPRF) — SPRF becomes set every time a byte transfers from the shift register to the receive data register. If the SPI receiver interrupt enable bit, SPRIE, is also set, SPRF generates an SPI receiver/error CPU interrupt request.
- SPI transmitter empty (SPTE) — SPTE becomes set every time a byte transfers from the transmit data register to the shift register. If the SPI transmit interrupt enable bit, SPTIE, is also set, SPTE generates an SPTE CPU interrupt request.

15.8 Resetting the SPI

Any system reset completely resets the SPI. Partial resets occur whenever the SPI enable bit (SPE) is 0. Whenever SPE is 0, the following occurs:

- The SPTE flag is set.
- Any transmission currently in progress is aborted.
- The shift register is cleared.
- The SPI state counter is cleared, making it ready for a new complete transmission.
- All the SPI port logic is defaulted back to being general-purpose I/O.

These items are reset only by a system reset:

- All control bits in the SPCR register
- All control bits in the SPSCR register (MODFEN, ERRIE, SPR1, and SPR0)
- The status flags SPRF, OVRF, and MODF

By not resetting the control bits when SPE is low, the user can clear SPE between transmissions without having to set all control bits again when SPE is set back high for the next transmission.

By not resetting the SPRF, OVRF, and MODF flags, the user can still service these interrupts after the SPI has been disabled. The user can disable the SPI by writing 0 to the SPE bit. The SPI can also be disabled by a mode fault occurring in an SPI that was configured as a master with the MODFEN bit set.

15.9 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

15.9.1 Wait Mode

The SPI module remains active after the execution of a WAIT instruction. In wait mode the SPI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SPI module can bring the MCU out of wait mode.

If SPI module functions are not required during wait mode, reduce power consumption by disabling the SPI module before executing the WAIT instruction.

To exit wait mode when an overflow condition occurs, enable the OVRF bit to generate CPU interrupt requests by setting the error interrupt enable bit (ERRIE). See 15.7 Interrupts.

15.9.2 Stop Mode

The SPI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions. SPI operation resumes after an external interrupt. If stop mode is exited by reset, any transfer in progress is aborted, and the SPI is reset.

15.10 SPI During Break Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. BCFE in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. See Chapter 14 System Integration Module (SIM).

To allow software to clear status bits during a break interrupt, write a 1 to BCFE. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a 0 to BCFE. With BCFE at 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is 0. After the break, doing the second step clears the status bit.

Since the SPTE bit cannot be cleared during a break with BCFE cleared, a write to the transmit data register in break mode does not initiate a transmission nor is this data transferred into the shift register. Therefore, a write to the SPDR in break mode with BCFE cleared has no effect.

15.11 I/O Signals

The SPI module has four I/O pins:

- MISO — Master input/slave output
- MOSI — Master output/slave input
- SPSCK — Serial clock
- $\overline{\text{SS}}$ — Slave select

The SPI has limited inter-integrated circuit (I^2C) capability (requiring software support) as a master in a single-master environment. To communicate with I^2C peripherals, MOSI becomes an open-drain output when the SPWOM bit in the SPI control register is set. In I^2C communication, the MOSI and MISO pins are connected to a bidirectional pin from the I^2C peripheral and through a pullup resistor to V_{DD}.

15.11.1 MISO (Master In/Slave Out)

MISO is one of the two SPI module pins that transmits serial data. In full duplex operation, the MISO pin of the master SPI module is connected to the MISO pin of the slave SPI module. The master SPI simultaneously receives data on its MISO pin and transmits data from its MOSI pin.

Slave output data on the MISO pin is enabled only when the SPI is configured as a slave. The SPI is configured as a slave when its SPMSTR bit is 0 and its $\overline{\text{SS}}$ pin is low. To support a multiple-slave system, a high on the $\overline{\text{SS}}$ pin puts the MISO pin in a high-impedance state.

When enabled, the SPI controls data direction of the MISO pin regardless of the state of the data direction register of the shared I/O port.

15.11.2 MOSI (Master Out/Slave In)

MOSI is one of the two SPI module pins that transmits serial data. In full-duplex operation, the MOSI pin of the master SPI module is connected to the MOSI pin of the slave SPI module. The master SPI simultaneously transmits data from its MOSI pin and receives data on its MISO pin.

When enabled, the SPI controls data direction of the MOSI pin regardless of the state of the data direction register of the shared I/O port.

15.11.3 SPSCK (Serial Clock)

The serial clock synchronizes data transmission between master and slave devices. In a master MCU, the SPSCK pin is the clock output. In a slave MCU, the SPSCK pin is the clock input. In full-duplex operation, the master and slave MCUs exchange a byte of data in eight serial clock cycles.

When enabled, the SPI controls data direction of the SPSCK pin regardless of the state of the data direction register of the shared I/O port.

15.11.4 $\overline{\text{SS}}$ (Slave Select)

The $\overline{\text{SS}}$ pin has various functions depending on the current state of the SPI. For an SPI configured as a slave, $\overline{\text{SS}}$ is used to select a slave. For CPHA = 0, $\overline{\text{SS}}$ is used to define the start of a transmission. (See 15.4 Transmission Formats.) Since it is used to indicate the start of a transmission, $\overline{\text{SS}}$ must be toggled high and low between each byte transmitted for the CPHA = 0 format. However, it can remain low between transmissions for the CPHA = 1 format. See Figure 15-13.

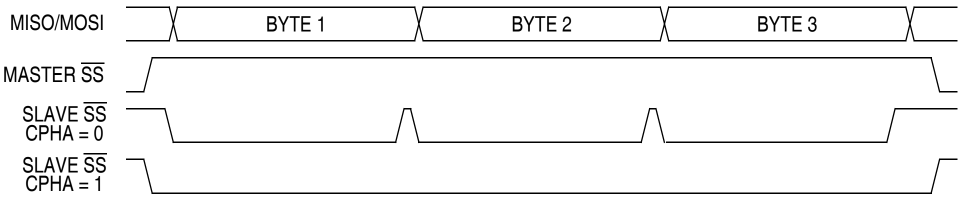


Figure 15-13. CPHA/$\overline{\text{SS}}$ Timing

When an SPI is configured as a slave, the $\overline{\text{SS}}$ pin is always configured as an input. It cannot be used as a general-purpose I/O regardless of the state of the MODFEN control bit. However, the MODFEN bit can still prevent the state of $\overline{\text{SS}}$ from creating a MODF error. See 15.12.2 SPI Status and Control Register.

NOTE

A high on the $\overline{\text{SS}}$ pin of a slave SPI puts the MISO pin in a high-impedance state. The slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.

When an SPI is configured as a master, the $\overline{\text{SS}}$ input can be used in conjunction with the MODF flag to prevent multiple masters from driving MOSI and SPSCK. (See 15.6.2 Mode Fault Error.) For the state of the $\overline{\text{SS}}$ pin to set the MODF flag, the MODFEN bit in the SPSCK register must be set. If MODFEN is 0 for an SPI master, the $\overline{\text{SS}}$ pin can be used as a general-purpose I/O under the control of the data direction register of the shared I/O port. When MODFEN is 1, $\overline{\text{SS}}$ is an input-only pin to the SPI regardless of the state of the data direction register of the shared I/O port.

The CPU can always read the state of the $\overline{\text{SS}}$ pin by configuring the appropriate pin as an input and reading the port data register. See Table 15-2.

Table 15-2. SPI Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	Function of $\overline{\text{SS}}$ Pin
0	X[1]	X	Not enabled	General-purpose I/O; $\overline{\text{SS}}$ ignored by SPI
1	0	X	Slave	Input-only to SPI
1	1	0	Master without MODF	General-purpose I/O; $\overline{\text{SS}}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

1. X = Don't care

15.12 I/O Registers

Three registers control and monitor SPI operation:

- SPI control register (SPCR)
- SPI status and control register (SPSCR)
- SPI data register (SPDR)

15.12.1 SPI Control Register

The SPI control register:

- Enables SPI module interrupt requests
- Configures the SPI module as master or slave
- Selects serial clock polarity and phase
- Configures the SPSCK, MOSI, and MISO pins as open-drain outputs
- Enables the SPI module

Address: $0010

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
Reset:	0	0	1	0	1	0	0	0

R = Reserved

Figure 15-14. SPI Control Register (SPCR)

SPRIE — SPI Receiver Interrupt Enable Bit

This read/write bit enables CPU interrupt requests generated by the SPRF bit. The SPRF bit is set when a byte transfers from the shift register to the receive data register. Reset clears the SPRIE bit.

1 = SPRF CPU interrupt requests enabled
0 = SPRF CPU interrupt requests disabled

SPMSTR — SPI Master Bit

This read/write bit selects master mode operation or slave mode operation. Reset sets the SPMSTR bit.

1 = Master mode
0 = Slave mode

CPOL — Clock Polarity Bit

This read/write bit determines the logic state of the SPSCK pin between transmissions. (See Figure 15-5 and Figure 15-7.) To transmit data between SPI modules, the SPI modules must have identical CPOL values. Reset clears the CPOL bit.

CPHA — Clock Phase Bit

This read/write bit controls the timing relationship between the serial clock and SPI data. (See Figure 15-5 and Figure 15-7.) To transmit data between SPI modules, the SPI modules must have identical CPHA values. When CPHA = 0, the $\overline{SS}$ pin of the slave SPI module must be high between bytes. (See Figure 15-13.) Reset sets the CPHA bit.

SPWOM — SPI Wired-OR Mode Bit

This read/write bit disables the pullup devices on pins SPSCK, MOSI, and MISO so that those pins become open-drain outputs.

1 = Wired-OR SPSCK, MOSI, and MISO pins
0 = Normal push-pull SPSCK, MOSI, and MISO pins

SPE — SPI Enable

This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. (See 15.8 Resetting the SPI.) Reset clears the SPE bit.

1 = SPI module enabled
0 = SPI module disabled

SPTIE— SPI Transmit Interrupt Enable

This read/write bit enables CPU interrupt requests generated by the SPTE bit. SPTE is set when a byte transfers from the transmit data register to the shift register. Reset clears the SPTIE bit.

1 = SPTE CPU interrupt requests enabled
0 = SPTE CPU interrupt requests disabled

15.12.2 SPI Status and Control Register

The SPI status and control register contains flags to signal these conditions:

- Receive data register full
- Failure to clear SPRF bit before next byte is received (overflow error)
- Inconsistent logic level on $\overline{SS}$ pin (mode fault error)
- Transmit data register empty

The SPI status and control register also contains bits that perform these functions:

- Enable error interrupts
- Enable mode fault error detection
- Select master SPI baud rate

Address: $0011

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
Write:	▨		▨	▨	▨			
Reset:	0	0	0	0	1	0	0	0

▨ = Unimplemented

Figure 15-15. SPI Status and Control Register (SPSCR)

SPRF — SPI Receiver Full Bit

This clearable, read-only flag is set each time a byte transfers from the shift register to the receive data register. SPRF generates a CPU interrupt request if the SPRIE bit in the SPI control register is set also.

During an SPRF CPU interrupt, the CPU clears SPRF by reading the SPI status and control register with SPRF set and then reading the SPI data register.
Reset clears the SPRF bit.

1 = Receive data register full
0 = Receive data register not full

ERRIE — Error Interrupt Enable Bit

This read/write bit enables the MODF and OVRF bits to generate CPU interrupt requests. Reset clears the ERRIE bit.

1 = MODF and OVRF can generate CPU interrupt requests
0 = MODF and OVRF cannot generate CPU interrupt requests

OVRF — Overflow Bit

This clearable, read-only flag is set if software does not read the byte in the receive data register before the next full byte enters the shift register. In an overflow condition, the byte already in the receive data register is unaffected, and the byte that shifted in last is lost. Clear the OVRF bit by reading the SPI status and control register with OVRF set and then reading the receive data register. Reset clears the OVRF bit.

1 = Overflow
0 = No overflow

MODF — Mode Fault Bit

This clearable, read-only flag is set in a slave SPI if the $\overline{SS}$ pin goes high during a transmission with MODFEN set. In a master SPI, the MODF flag is set if the $\overline{SS}$ pin goes low at any time with the MODFEN bit set. Clear MODF by reading the SPI status and control register (SPSCR) with MODF set and then writing to the SPI control register (SPCR). Reset clears the MODF bit.

1 = $\overline{SS}$ pin at inappropriate logic level
0 = $\overline{SS}$ pin at appropriate logic level

SPTE — SPI Transmitter Empty Bit

This clearable, read-only flag is set each time the transmit data register transfers a byte into the shift register. SPTE generates an SPTE CPU interrupt request if SPTIE in the SPI control register is set also.

NOTE

Do not write to the SPI data register unless SPTE is high.

During an SPTE CPU interrupt, the CPU clears SPTE bit writing to the transmit data register. Reset sets the SPTE bit.

1 = Transmit data register empty
0 = Transmit data register not empty

MODFEN — Mode Fault Enable Bit

This read/write bit, when set, allows the MODF flag to be set. If the MODF flag is set, clearing MODFEN does not clear the MODF flag. If the SPI is enabled as a master and the MODFEN bit is 0, then the $\overline{SS}$ pin is available as a general-purpose I/O.

If the MODFEN bit is 1, then the $\overline{SS}$ is not available as a general-purpose I/O. When the SPI is enabled as a slave, the $\overline{SS}$ pin is not available as a general-purpose I/O regardless of the value of MODFEN. See 15.11.4 SS (Slave Select).

If the MODFEN bit is 0, the level of the $\overline{SS}$ pin does not affect the operation of an enabled SPI configured as a master. For an enabled SPI configured as a slave, having MODFEN low only prevents the MODF flag from being set. It does not affect any other part of SPI operation. See 15.6.2 Mode Fault Error.

SPR1 and SPR0 — SPI Baud Rate Select Bits

In master mode, these read/write bits select one of four baud rates as shown in Table 15-3. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Table 15-3. SPI Master Baud Rate Selection

SPR1 and SPR0	Baud Rate Divisor (BD)
00	2
01	8
10	32
11	128

Use this formula to calculate the SPI baud rate:

$$\text{Baud rate} = \frac{\text{BUSCLK}}{\text{BD}}$$

15.12.3 SPI Data Register

The SPI data register consists of the read-only receive data register and the write-only transmit data register. Writing to the SPI data register writes data into the transmit data register. Reading the SPI data register reads data from the receive data register. The transmit data and receive data registers are separate registers that can contain different values. See Figure 15-2.

Address: $0012

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	Unaffected by reset							

Figure 15-16. SPI Data Register (SPDR)

R7–R0/T7–T0 — Receive/Transmit Data Bits

NOTE

Do not use read-modify-write instructions on the SPI data register since the register read is not the same as the register written.

Chapter 16
Timebase Module (TBM)

16.1 Introduction

This section describes the timebase module (TBM). The TBM will generate periodic interrupts at user selectable rates using a counter clocked by the external clock source. This TBM version uses 15 divider stages, eight of which are user selectable.

16.2 Features

Features of the TBM module include:

- Software configurable periodic interrupts with divide-by: 8, 16, 32, 64, 128, 2048, 8192, and 32768 taps of the selected clock source
- Configurable for operation during stop mode to allow periodic wakeup from stop

16.3 Functional Description

This module can generate a periodic interrupt by dividing the clock source supplied from the clock generator module, CGMXCLK.

The counter is initialized to all 0s when TBON bit is cleared. The counter, shown in Figure 16-1, starts counting when the TBON bit is set. When the counter overflows at the tap selected by TBR2–TBR0, the TBIF bit gets set. If the TBIE bit is set, an interrupt request is sent to the CPU. The TBIF flag is cleared by writing a 1 to the TACK bit. The first time the TBIF flag is set after enabling the timebase module, the interrupt is generated at approximately half of the overflow period. Subsequent events occur at the exact period.

The timebase module may remain active after execution of the STOP instruction if the crystal oscillator has been enabled to operate during stop mode through the OSCSTOPENB bit in the configuration register. The timebase module can be used in this mode to generate a periodic wakeup from stop mode.

16.4 Interrupts

The timebase module can periodically interrupt the CPU with a rate defined by the selected TBMCLK and the select bits TBR2–TBR0. When the timebase counter chain rolls over, the TBIF flag is set. If the TBIE bit is set, enabling the timebase interrupt, the counter chain overflow will generate a CPU interrupt request.

NOTE

Interrupts must be acknowledged by writing a 1 to the TACK bit.

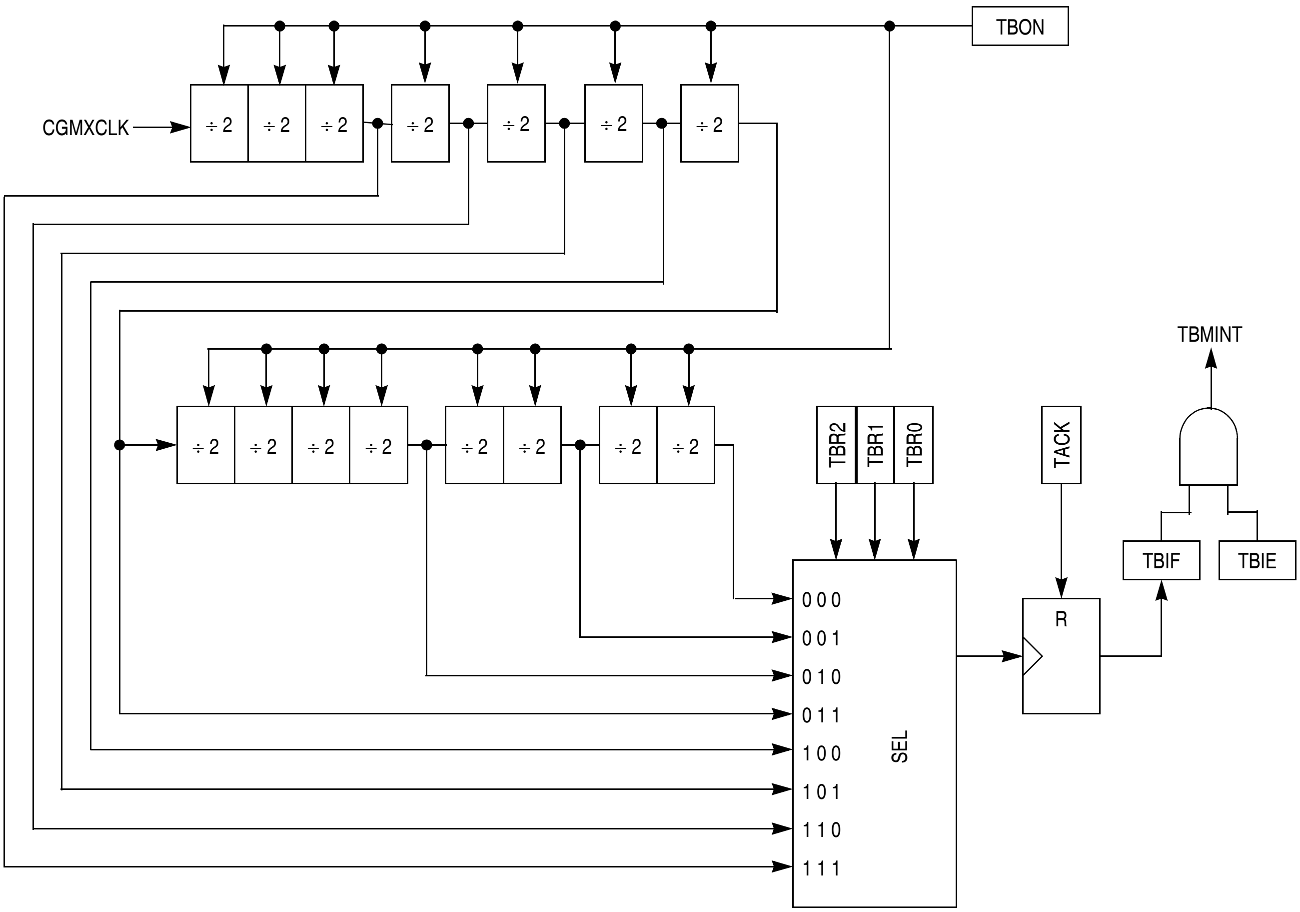


Figure 16-1. Timebase Block Diagram

16.5 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

16.5.1 Wait Mode

The timebase module remains active after execution of the WAIT instruction. In wait mode the timebase register is not accessible by the CPU.

If the timebase functions are not required during wait mode, reduce the power consumption by stopping the timebase before executing the WAIT instruction.

16.5.2 Stop Mode

The timebase module may remain active after execution of the STOP instruction if the internal clock generator has been enabled to operate during stop mode through the OSCSTOPENB bit in the configuration register. The timebase module can be used in this mode to generate a periodic wakeup from stop mode.

If the internal clock generator has not been enabled to operate in stop mode, the timebase module will not be active during stop mode. In stop mode, the timebase register is not accessible by the CPU.

If the timebase functions are not required during stop mode, reduce power consumption by disabling the timebase module before executing the STOP instruction.

16.6 Timebase Control Register

The timebase has one register, the timebase control register (TBCR), which is used to enable the timebase interrupts and set the rate.

Address: $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TBIF	TBR2	TBR1	TBR0	0	TBIE	TBON	R
Write:	▨				TACK			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

R = Reserved

Figure 16-2. Timebase Control Register (TBCR)

TBIF — Timebase Interrupt Flag

This read-only flag bit is set when the timebase counter has rolled over.

1 = Timebase interrupt pending
0 = Timebase interrupt not pending

TBR2–TBR0 — Timebase Divider Selection Bits

These read/write bits select the tap in the counter to be used for timebase interrupts as shown in Table 16-1.

Table 16-1. Timebase Divider Selection

TBR2	TBR1	TBR0	Divider
0	0	0	32,768
0	0	1	8192
0	1	0	2048
0	1	1	128
1	0	0	64
1	0	1	32
1	1	0	16
1	1	1	8

NOTE

Do not change TBR2–TBR0 bits while the timebase is enabled (TBON = 1).

TACK— Timebase Acknowledge Bit

The TACK bit is a write-only bit and always reads as 0. Writing a 1 to this bit clears TBIF, the timebase interrupt flag bit. Writing a 0 to this bit has no effect.

1 = Clear timebase interrupt flag
0 = No effect

TBIE — Timebase Interrupt Enabled Bit

This read/write bit enables the timebase interrupt when the TBIF bit becomes set. Reset clears the TBIE bit.

1 = Timebase interrupt is enabled.
0 = Timebase interrupt is disabled.

TBON — Timebase Enabled Bit

This read/write bit enables the timebase. Timebase may be turned off to reduce power consumption when its function is not necessary. The counter can be initialized by clearing and then setting this bit. Reset clears the TBON bit.

1 = Timebase is enabled.
0 = Timebase is disabled and the counter initialized to 0s.

Chapter 17
Timer Interface Module (TIM1 and TIM2)

17.1 Introduction

This section describes the timer interface (TIM) module. The timer on this part consists of a 2-channel and a 1-channel timer that provides a timing reference with input capture, output compare, and pulse-width-modulation functions. Figure 17-1 is a block diagram of the TIM.

This particular MCU has two timer interface modules which are denoted as TIM1 and TIM2.

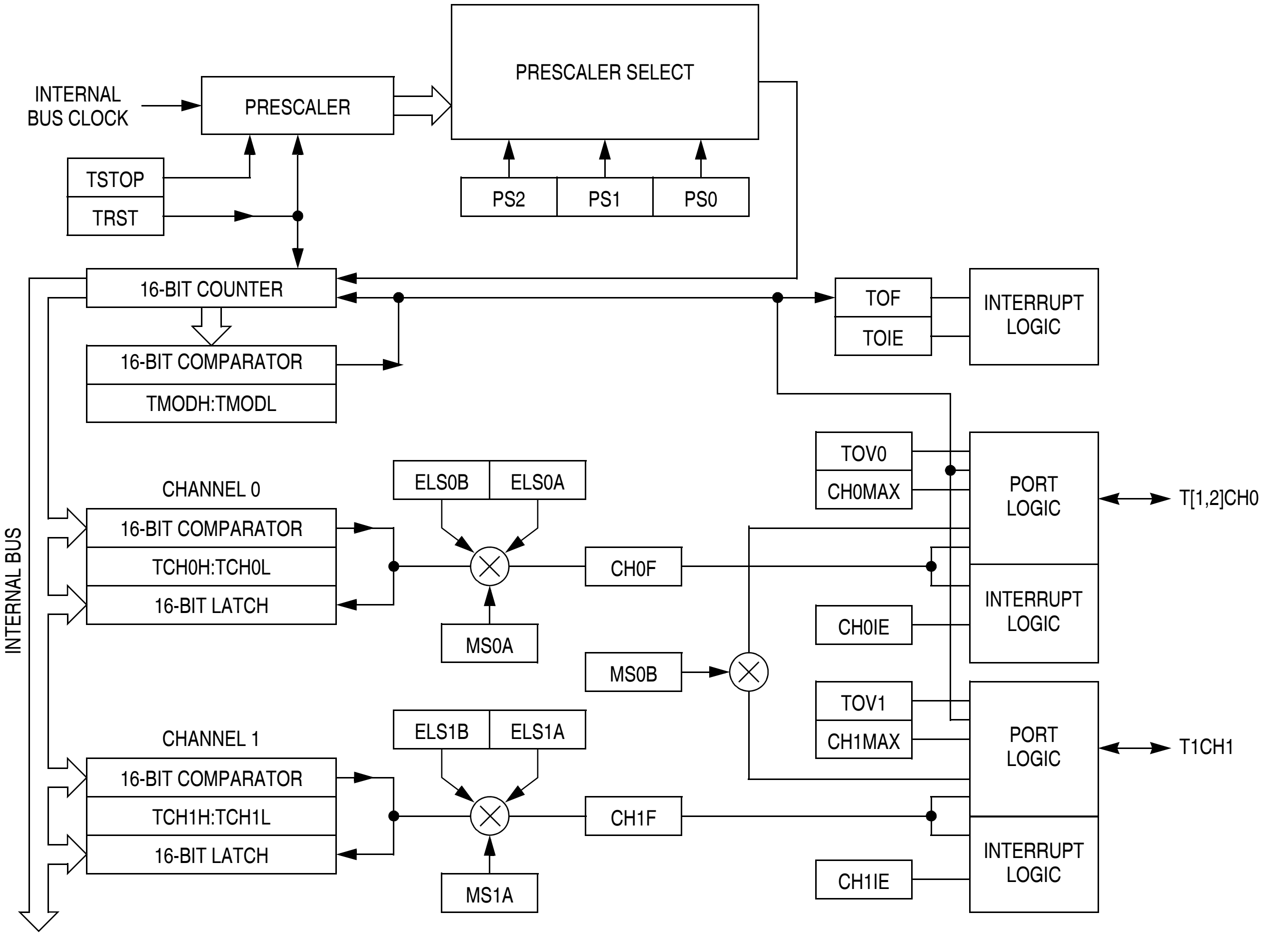
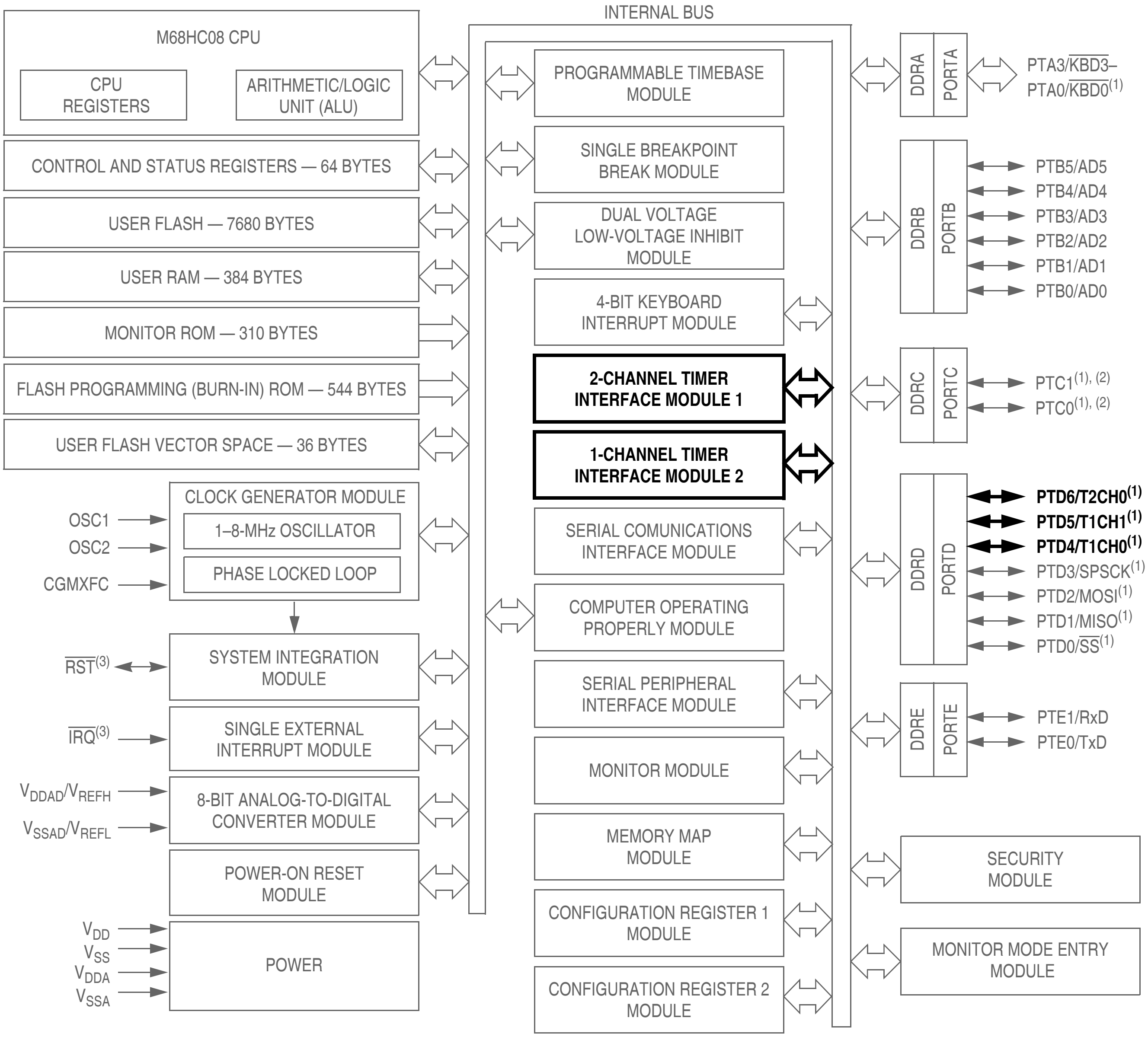


Figure 17-1. TIM Block Diagram



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 17-2. Block Diagram Highlighting TIM Block and Pins

17.2 Features

Features of the TIM include:

- Three input capture/output compare channels:
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
- Buffered and unbuffered pulse-width-modulation (PWM) signal generation
- Programmable TIM clock input with 7-frequency internal bus clock prescaler selection
- Free-running or modulo up-count operation
- Toggle any channel pin on overflow
- TIM counter stop and reset bits

17.3 Pin Name Conventions

The text that follows describes both timers, TIM1 and TIM2. The TIM input/output (I/O) pin names are T[1,2]CH0 (timer channel 0) and T[1,2]CH1 (timer channel 1), where “1” is used to indicate TIM1 and “2” is used to indicate TIM2. The two TIMs share three I/O pins with three port D I/O port pins. The full names of the TIM I/O pins are listed in Table 17-1. The generic pin names appear in the text that follows.

Table 17-1. Pin Name Conventions

TIM Generic Pin Names:		**T[1,2]CH0**	**T[1,2]CH1**
Full TIM Pin Names:	TIM1	PTD4/T1CH0	PTD5/T1CH1
	TIM2	PTD6/T2CH0	—

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TCH0 may refer generically to T1CH0 and T2CH0.

17.4 Functional Description

Figure 17-1 shows the structure of the TIM. The central component of the TIM is the 16-bit TIM counter that can operate as a free-running counter or a modulo up-counter. The TIM counter provides the timing reference for the input capture and output compare functions. The TIM counter modulo registers, TMODH:TMODL, control the modulo value of the TIM counter. Software can read the TIM counter value at any time without affecting the counting sequence.

The two TIM channels (per timer) are programmable independently as input capture or output compare channels. If a channel is configured as input capture, then an internal pullup device may be enabled for that channel. See 12.5.3 Port D Input Pullup Enable Register.

Figure 17-3 summarizes the timer registers.

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC and T2SC.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	Timer 1 Status and Control Register (T1SC) See page 207.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	▨			
		Reset:	0	0	1	0	0	0	0	0
$0021	Timer 1 Counter Register High (T1CNTH) See page 208.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0022	Timer 1 Counter Register Low (T1CNTL) See page 208.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0023	Timer 1 Counter Modulo Register High (T1MODH) See page 209.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	Timer 1 Counter Modulo Register Low (T1MODL) See page 209.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	Timer 1 Channel 0 Status and Control Register (T1SC0) See page 209.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	Timer 1 Channel 0 Register High (T1CH0H) See page 212.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0027	Timer 1 Channel 0 Register Low (T1CH0L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0028	Timer 1 Channel 1 Status and Control Register (T1SC1) See page 209.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		▨					
		Reset:	0	0	0	0	0	0	0	0
$0029	Timer 1 Channel 1 Register High (T1CH1H) See page 212.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$002A	Timer 1 Channel 1 Register Low (T1CH1L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$002B	Timer 2 Status and Control Register (T2SC) See page 207.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	▨			
		Reset:	0	0	1	0	0	0	0	0

▨ = Unimplemented

Figure 17-3. TIM I/O Register Summary (Sheet 1 of 2)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$002C	Timer 2 Counter Register High (T2CNTH) See page 208.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$002D	Timer 2 Counter Register Low (T2CNTL) See page 208.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$002E	Timer 2 Counter Modulo Register High (T2MODH) See page 209.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$002F	Timer 2 Counter Modulo Register Low (T2MODL) See page 209.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0030	Timer 2 Channel 0 Status and Control Register (T2SC0) See page 209.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0031	Timer 2 Channel 0 Register High (T2CH0H) See page 212.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0032	Timer 2 Channel 0 Register Low (T2CH0L) See page 212.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0033	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0034 ↓ $0035	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	Indeterminate after reset							

▨ = Unimplemented

Figure 17-3. TIM I/O Register Summary (Sheet 2 of 2)

17.4.1 TIM Counter Prescaler

The TIM clock source can be one of the seven prescaler outputs. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM status and control register select the TIM clock source.

17.4.2 Input Capture

With the input capture function, the TIM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TIM latches the contents of the TIM counter into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

17.4.3 Output Compare

With the output compare function, the TIM can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

17.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in 17.4.3 Output Compare. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change an output compare value could cause incorrect operation for up to two counter overflow periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that counter overflow period. Also, using a TIM overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the output compare value on channel x:

- When changing to a smaller value, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current output compare pulse. The interrupt routine has until the end of the counter overflow period to write the new value.
- When changing to a larger output compare value, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current counter overflow period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same counter overflow period.

17.4.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The output compare value in the TIM channel 0 registers initially controls the output on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the output after the TIM overflows. At each subsequent overflow, the TIM channel registers (0 or 1) that control the output are the ones written to last. TSC0 controls and monitors the buffered output compare function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE

In buffered output compare operation, do not write new output compare values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered output compares.

17.4.4 Pulse Width Modulation (PWM)

By using the toggle-on-overflow feature with an output compare channel, the TIM can generate a PWM signal. The value in the TIM counter modulo registers determines the period of the PWM signal. The channel pin toggles when the counter reaches the value in the TIM counter modulo registers. The time between overflows is the period of the PWM signal.

As Figure 17-4 shows, the output compare value in the TIM channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIM to clear the channel pin on output compare if the polarity of the PWM pulse is 1 (ELSxA = 0). Program the TIM to set the pin if the polarity of the PWM pulse is 0 (ELSxA = 0).

The value in the TIM counter modulo registers and the selected prescaler output determines the frequency of the PWM output. The frequency of an 8-bit PWM signal is variable in 256 increments. Writing $00FF (255) to the TIM counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000. See 17.9.1 TIM Status and Control Register.

The value in the TIM channel registers determines the pulse width of the PWM output. The pulse width of an 8-bit PWM signal is variable in 256 increments. Writing $0080 (128) to the TIM channel registers produces a duty cycle of 128/256 or 50%.

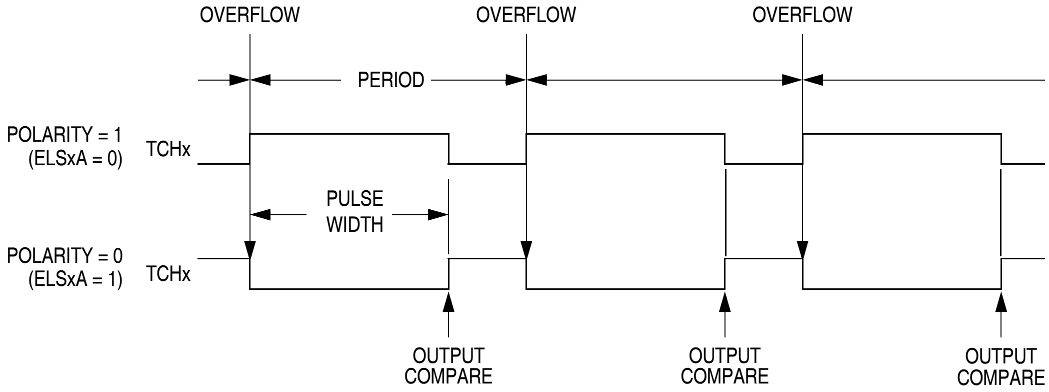


Figure 17-4. PWM Period and Pulse Width

17.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in 17.4.4 Pulse Width Modulation (PWM). The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change a pulse width value could cause incorrect operation for up to two PWM periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that PWM period. Also, using a TIM overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the PWM pulse width on channel x:

- When changing to a shorter pulse width, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current pulse. The interrupt routine has until the end of the PWM period to write the new value.

- When changing to a longer pulse width, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current PWM period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same PWM period.

NOTE

In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare also can cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.

17.4.4.2 Buffered PWM Signal Generation

Channels 0 and 1 can be linked to form a buffered PWM channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The TIM channel 0 registers initially control the pulse width on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIM channel registers (0 or 1) that control the pulse width are the ones written to last. TSC0 controls and monitors the buffered PWM function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE

In buffered PWM signal generation, do not write new pulse width values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered PWM signals.

17.4.4.3 PWM Initialization

To ensure correct operation when generating unbuffered or buffered PWM signals, use the following initialization procedure:

1. In the TIM status and control register (TSC):
    a. Stop the TIM counter by setting the TIM stop bit, TSTOP.
    b. Reset the TIM counter and prescaler by setting the TIM reset bit, TRST.
2. In the TIM counter modulo registers (TMODH:TMODL), write the value for the required PWM period.
3. In the TIM channel x registers (TCHxH:TCHxL), write the value for the required pulse width.
4. In TIM channel x status and control register (TSCx):
    a. Write 0:1 (for unbuffered output compare or PWM signals) or 1:0 (for buffered output compare or PWM signals) to the mode select bits, MSxB:MSxA. See Table 17-3.
    b. Write 1 to the toggle-on-overflow bit, TOVx.

c. Write 1:0 (polarity 1 — to clear output on compare) or 1:1 (polarity 0 — to set output on compare) to the edge/level select bits, ELSxB:ELSxA. The output action on compare must force the output to the complement of the pulse width level. See Table 17-3.

NOTE

In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0% duty cycle generation and removes the ability of the channel to self-correct in the event of software error or noise. Toggling on output compare can also cause incorrect PWM signal generation when changing the PWM pulse width to a new, much larger value.

5. In the TIM status control register (TSC), clear the TIM stop bit, TSTOP.

Setting MS0B links channels 0 and 1 and configures them for buffered PWM operation. The TIM channel 0 registers (TCH0H:TCH0L) initially control the buffered PWM output. TIM status control register 0 (TSCR0) controls and monitors the PWM signal from the linked channels.

Clearing the toggle-on-overflow bit, TOVx, inhibits output toggles on TIM overflows. Subsequent output compares try to force the output to a state it is already in and have no effect. The result is a 0% duty cycle output.

Setting the channel x maximum duty cycle bit (CHxMAX) and setting the TOVx bit generates a 100% duty cycle output. See 17.9.4 TIM Channel Status and Control Registers.

17.5 Interrupts

The following TIM sources can generate interrupt requests:

- TIM overflow flag (TOF) — The TOF bit is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. The TIM overflow interrupt enable bit, TOIE, enables TIM overflow CPU interrupt requests. TOF and TOIE are in the TIM status and control register.
- TIM channel flags (CH1F:CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. Channel x TIM CPU interrupt requests are controlled by the channel x interrupt enable bit, CHxIE. Channel x TIM CPU interrupt requests are enabled when CHxIE = 1. CHxF and CHxIE are in the TIM channel x status and control register.

17.6 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

17.6.1 Wait Mode

The TIM remains active after the execution of a WAIT instruction. In wait mode, the TIM registers are not accessible by the CPU. Any enabled CPU interrupt request from the TIM can bring the MCU out of wait mode.

If TIM functions are not required during wait mode, reduce power consumption by stopping the TIM before executing the WAIT instruction.

17.6.2 Stop Mode

The TIM is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions or the state of the TIM counter. TIM operation resumes when the MCU exits stop mode after an external interrupt.

17.7 TIM During Break Interrupts

A break interrupt stops the TIM counter and inhibits captures.

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. See 14.7.3 SIM Break Flag Control Register.

To allow software to clear status bits during a break interrupt, write a 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a 0 to the BCFE bit. With BCFE at 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at 0. After the break, doing the second step clears the status bit.

17.8 I/O Signals

Port D shares four of its pins with the TIM. The four TIM channel I/O pins are T1CH0, T1CH1, T2CH0, and T2CH1 as described in 17.3 Pin Name Conventions.

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. T1CH0 and T2CH0 can be configured as buffered output compare or buffered PWM pins.

17.9 I/O Registers

NOTE

References to either timer 1 or timer 2 may be made in the following text by omitting the timer number. For example, TSC may generically refer to both T1SC AND T2SC.

These I/O registers control and monitor operation of the TIM:
- TIM status and control register (TSC)
- TIM counter registers (TCNTH:TCNTL)
- TIM counter modulo registers (TMODH:TMODL)
- TIM channel status and control registers (TSC0 and TSC1)
- TIM channel registers (TCH0H:TCH0L, TCH1H:TCH1L)

17.9.1 TIM Status and Control Register

The TIM status and control register (TSC):
- Enables TIM overflow interrupts
- Flags TIM overflows
- Stops the TIM counter
- Resets the TIM counter
- Prescales the TIM counter clock

Address: T1SC, $002 and T2SC, $002B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
Write:	0			TRST	▨			
Reset:	0	0	1	0	0	0	0	0

▨ = Unimplemented

Figure 17-5. TIM Status and Control Register (TSC)

TOF — TIM Overflow Flag Bit

This read/write flag is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. Clear TOF by reading the TIM status and control register when TOF is set and then writing a 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing 0 to TOF has no effect. Therefore, a TOF interrupt request cannot be lost due to inadvertent clearing of TOF. Reset clears the TOF bit. Writing a 1 to TOF has no effect.

1 = TIM counter has reached modulo value
0 = TIM counter has not reached modulo value

TOIE — TIM Overflow Interrupt Enable Bit

This read/write bit enables TIM overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIM overflow interrupts enabled
0 = TIM overflow interrupts disabled

TSTOP — TIM Stop Bit

This read/write bit stops the TIM counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIM counter until software clears the TSTOP bit.

1 = TIM counter stopped
0 = TIM counter active

NOTE

Do not set the TSTOP bit before entering wait mode if the TIM is required to exit wait mode. Also, when the TSTOP bit is set and the timer is configured for input capture operation, input captures are inhibited until the TSTOP bit is cleared.

TRST — TIM Reset Bit

Setting this write-only bit resets the TIM counter and the TIM prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIM counter is reset and always reads as 0. Reset clears the TRST bit.

1 = Prescaler and TIM counter cleared
0 = No effect

NOTE

Setting the TSTOP and TRST bits simultaneously stops the TIM counter at a value of $0000.

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIM counter as Table 17-2 shows. Reset clears the PS[2:0] bits.

Table 17-2. Prescaler Selection

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal bus clock ÷ 1
0	0	1	Internal bus clock ÷ 2
0	1	0	Internal bus clock ÷ 4
0	1	1	Internal bus clock ÷ 8
1	0	0	Internal bus clock ÷ 16
1	0	1	Internal bus clock ÷ 32
1	1	0	Internal bus clock ÷ 64
1	1	1	Not available

17.9.2 TIM Counter Registers

The two read-only TIM counter registers contain the high and low bytes of the value in the TIM counter. Reading the high byte (TCNTH) latches the contents of the low byte (TCNTL) into a buffer. Subsequent reads of TCNTH do not affect the latched TCNTL value until TCNTL is read. Reset clears the TIM counter registers. Setting the TIM reset bit (TRST) also clears the TIM counter registers.

NOTE

If you read TCNTH during a break interrupt, be sure to unlatch TCNTL by reading TCNTL before exiting the break interrupt. Otherwise, TCNTL retains the value latched during the break.

Address: T1CNTH, $0021 and T2CNTH, $002C

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

▨ = Unimplemented

Figure 17-6. TIM Counter Registers High (TCNTH)

Address: T1CNTL, $0022 and T2CNTL, $002D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	6	5	4	3	2	1	Bit 0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-7. TIM Counter Registers Low (TCNTL)

17.9.3 TIM Counter Modulo Registers

The read/write TIM modulo registers contain the modulo value for the TIM counter. When the TIM counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIM counter resumes counting from $0000 at the next timer clock. Writing to the high byte (TMODH) inhibits the TOF bit and overflow interrupts until the low byte (TMODL) is written. Reset sets the TIM counter modulo registers.

Address: T1MODH, $0023 and T2MODH, $002E

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

Figure 17-8. TIM Counter Modulo Register High (TMODH)

Address: T1MODL, $0024 and T2MODL, $002F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	6	5	4	3	2	1	Bit 0
Write:								
Reset:	1	1	1	1	1	1	1	1

Figure 17-9. TIM Counter Modulo Register Low (TMODL)

NOTE

Reset the TIM counter before writing to the TIM counter modulo registers.

17.9.4 TIM Channel Status and Control Registers

Each of the TIM channel status and control registers:

- Flags input captures and output compares
- Enables input capture and output compare interrupts
- Selects input capture, output compare, or PWM operation
- Selects high, low, or toggling output on output compare
- Selects rising edge, falling edge, or any edge as the active input capture trigger
- Selects output toggling on TIM overflow
- Selects 0% and 100% PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

Address: T1SC0, $0025 and T2SC0, $0030

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

Figure 17-10. TIM Channel 0 Status and Control Register (TSC0)

Address: T1SC1, $0028

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
Write:	0		▨					
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 17-11. TIM Channel 1 Status and Control Register (TSC1)

CHxF — Channel x Flag Bit

When channel x is an input capture channel, this read/write bit is set when an active edge occurs on the channel x pin. When channel x is an output compare channel, CHxF is set when the value in the TIM counter registers matches the value in the TIM channel x registers.

When TIM CPU interrupt requests are enabled (CHxIE = 1), clear CHxF by reading TIM channel x status and control register with CHxF set and then writing a 0 to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing 0 to CHxF has no effect. Therefore, an interrupt request cannot be lost due to inadvertent clearing of CHxF.

Reset clears the CHxF bit. Writing a 1 to CHxF has no effect.
1 = Input capture or output compare on channel x
0 = No input capture or output compare on channel x

CHxIE — Channel x Interrupt Enable Bit

This read/write bit enables TIM CPU interrupt service requests on channel x.
Reset clears the CHxIE bit.
1 = Channel x CPU interrupt requests enabled
0 = Channel x CPU interrupt requests disabled

MSxB — Mode Select Bit B

This read/write bit selects buffered output compare/PWM operation. MSxB exists only in the TIM1 channel 0 and TIM2 channel 0 status and control registers.

Setting MS0B disables the channel 1 status and control register and reverts TCH1 to general-purpose I/O.

Reset clears the MSxB bit.
1 = Buffered output compare/PWM operation enabled
0 = Buffered output compare/PWM operation disabled

MSxA — Mode Select Bit A

When ELSxB:A ≠ 00, this read/write bit selects either input capture operation or unbuffered output compare/PWM operation. See Table 17-3.
1 = Unbuffered output compare/PWM operation
0 = Input capture operation

When ELSxB:A = 00, this read/write bit selects the initial output level of the TCHx pin. See Table 17-3. Reset clears the MSxA bit.
1 = Initial output level low
0 = Initial output level high

NOTE

Before changing a channel function by writing to the MSxB or MSxA bit, set the TSTOP and TRST bits in the TIM status and control register (TSC).

ELSxB and ELSxA — Edge/Level Select Bits

When channel x is an input capture channel, these read/write bits control the active edge-sensing logic on channel x.

When channel x is an output compare channel, ELSxB and ELSxA control the channel x output behavior when an output compare occurs.

When ELSxB and ELSxA are both clear, channel x is not connected to port D, and pin PTDx/TCHx is available as a general-purpose I/O pin. Table 17-3 shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

Table 17-3. Mode, Edge, and Level Selection

MSxB	MSxA	ELSxB	ELSxA	Mode	Configuration
X	0	0	0	Output preset	Pin under port control; initial output level high
X	1	0	0		Pin under port control; initial output level low
0	0	0	1	Input capture	Capture on rising edge only
0	0	1	0		Capture on falling edge only
0	0	1	1		Capture on rising or falling edge
0	1	0	0	Output compare or PWM	Software compare only
0	1	0	1		Toggle output on compare
0	1	1	0		Clear output on compare
0	1	1	1		Set output on compare
1	X	0	1	Buffered output compare or buffered PWM	Toggle output on compare
1	X	1	0		Clear output on compare
1	X	1	1		Set output on compare

NOTE

Before enabling a TIM channel register for input capture operation, make sure that the PTD/TCHx pin is stable for at least two bus clocks.

TOVx — Toggle On Overflow Bit

When channel x is an output compare channel, this read/write bit controls the behavior of the channel x output when the TIM counter overflows. When channel x is an input capture channel, TOVx has no effect. Reset clears the TOVx bit.

1 = Channel x pin toggles on TIM counter overflow.
0 = Channel x pin does not toggle on TIM counter overflow.

NOTE

When TOVx is set, a TIM counter overflow takes precedence over a channel x output compare if both occur at the same time.

CHxMAX — Channel x Maximum Duty Cycle Bit

When the TOVx bit is at 1, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100%. As Figure 17-12 shows, the CHxMAX bit takes effect in the cycle after it is set or cleared. The output stays at the 100% duty cycle level until the cycle after CHxMAX is cleared.

NOTE

The 100% PWM duty cycle is defined as a continuous high level if the PWM polarity is 1 and a continuous low level if the PWM polarity is 0. Conversely, a 0% PWM duty cycle is defined as a continuous low level if the PWM polarity is 1 and a continuous high level if the PWM polarity is 0.

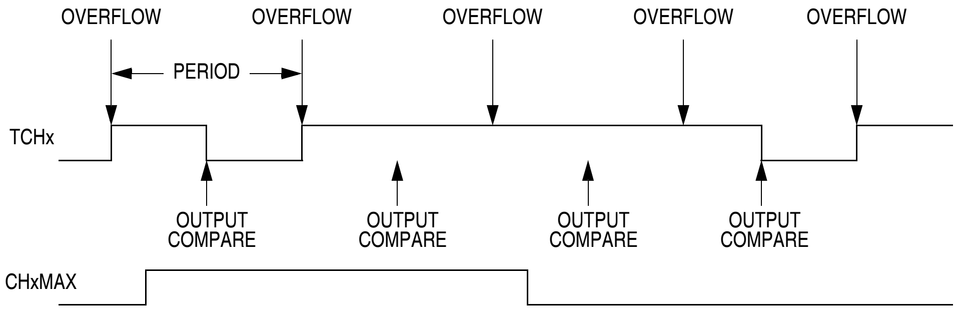


Figure 17-12. CHxMAX Latency

17.9.5 TIM Channel Registers

These read/write registers contain the captured TIM counter value of the input capture function or the output compare value of the output compare function. The state of the TIM channel registers after reset is unknown.

In input capture mode (MSxB:MSxA = 0:0), reading the high byte of the TIM channel x registers (TCHxH) inhibits input captures until the low byte (TCHxL) is read.

In output compare mode (MSxB:MSxA ≠ 0:0), writing to the high byte of the TIM channel x registers (TCHxH) inhibits output compares until the low byte (TCHxL) is written.

See Figure 17-13 through Figure 17-16.

Address: T1CH0H, \$0026 and T2CH0H, \$0031

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
Reset:	Indeterminate after reset							

Figure 17-13. TIM Channel 0 Register High (TCH0H)

Address: T1CH0L, \$0027 and T2CH0L \$0032

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	Indeterminate after reset							

Figure 17-14. TIM Channel 0 Register Low (TCH0L)

Address: T1CH1H, \$0029

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
Reset:	Indeterminate after reset							

Figure 17-15. TIM Channel 1 Register High (TCH1H)

Address: T1CH1L, \$002A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	Indeterminate after reset							

Figure 17-16. TIM Channel 1 Register Low (TCH1L)

Chapter 18
Development Support

18.1 Introduction

This section describes the break module, the monitor module (MON), and the monitor mode entry methods.

18.2 Break Module (BRK)

The break module can generate a break interrupt that stops normal program flow at a defined address to enter a background program.

Features of the break module include:

- Accessible input/output (I/O) registers during the break Interrupt
- Central processor unit (CPU) generated break interrupts
- Software-generated break interrupts
- Computer operating properly (COP) disabling during break interrupts

18.2.1 Functional Description

When the internal address bus matches the value written in the break address registers, the break module issues a breakpoint signal ($\overline{\text{BKPT}}$) to the system integration module (SIM). The SIM then causes the CPU to load the instruction register with a software interrupt instruction (SWI). The program counter vectors to \$FFFC and \$FFFD (\$FEFC and \$FEFD in monitor mode).

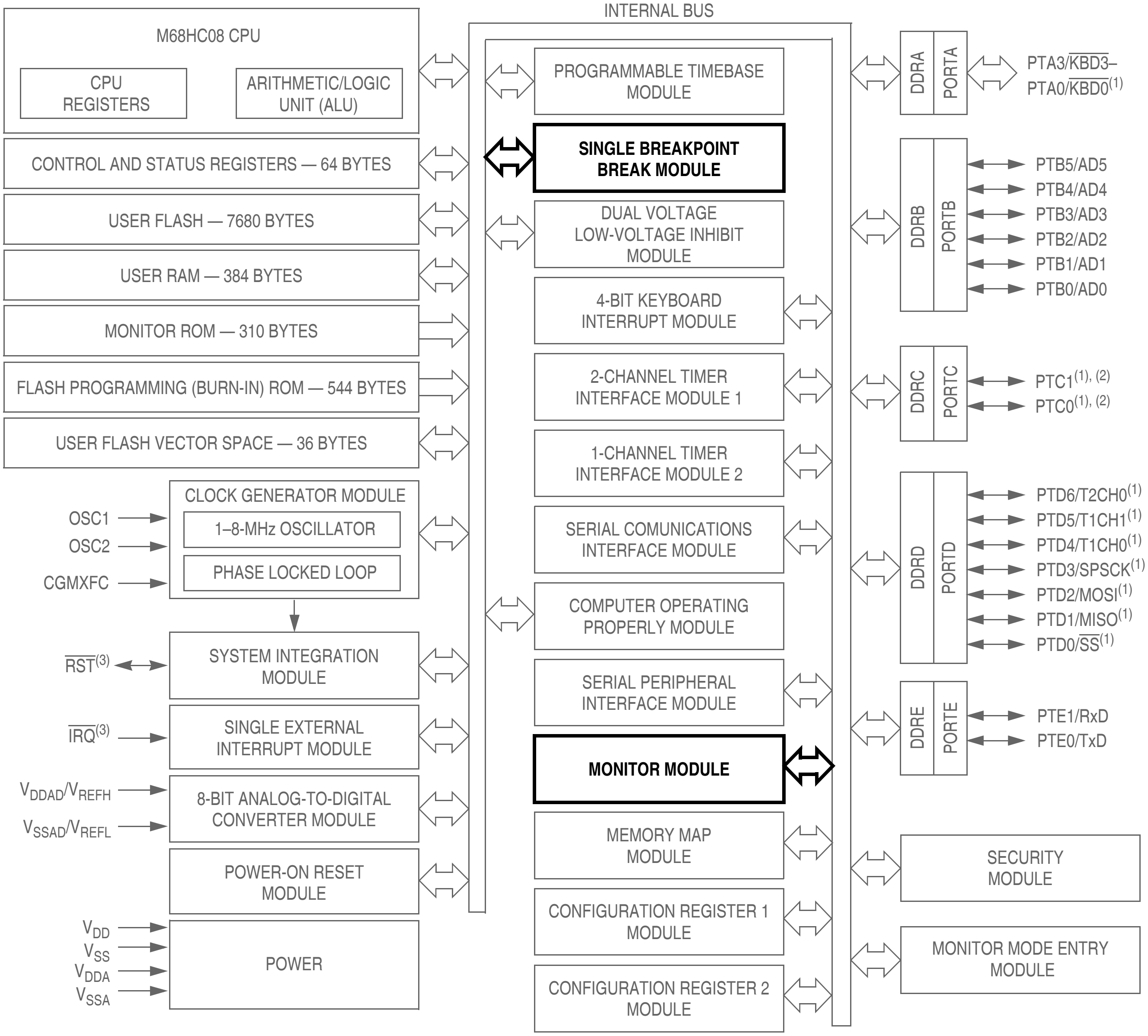
The following events can cause a break interrupt to occur:

- A CPU generated address (the address in the program counter) matches the contents of the break address registers.
- Software writes a 1 to the BRKA bit in the break status and control register.

When a CPU generated address matches the contents of the break address registers, the break interrupt is generated. A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the microcontroller unit (MCU) to normal operation.

Figure 18-2 shows the structure of the break module.

Figure 18-3 provides a summary of the I/O registers.



1. Ports are software configurable with pullup device if input port.
2. Higher current drive port pins
3. Pin contains integrated pullup device

Figure 18-1. Block Diagram Highlighting BRK and MON Blocks

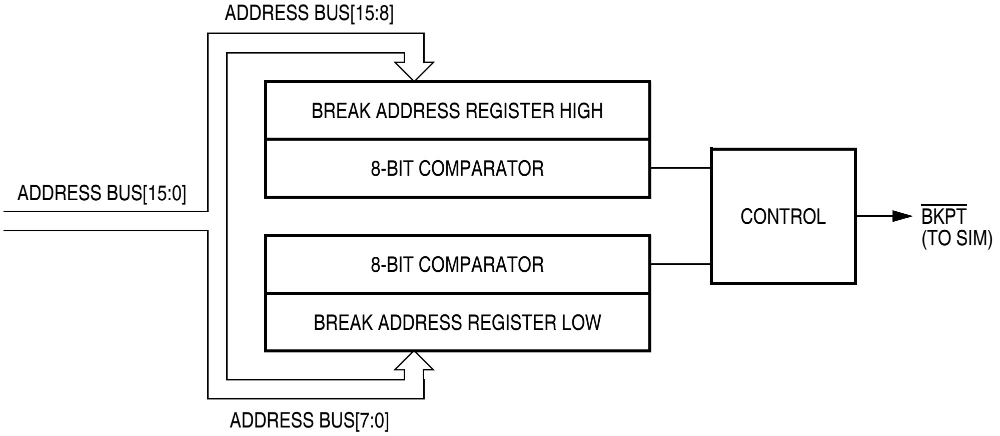


Figure 18-2. Break Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR) See page 218.	Read:	R	R	R	R	R	R	SBSW	R
		Write:							Note(1)	
		Reset:							0	
$FE02	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE03	SIM Break Flag Control Register (SBFCR) See page 218.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE09	Break Address High Register (BRKH) See page 217.	Read:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Break Address Low Register (BRKL) See page 217.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$FE0B	Break Status and Control Register (BRKSCR) See page 217.	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:			▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0

1. Writing a 0 clears SBSW.

▨ = Unimplemented

R = Reserved

Figure 18-3. Break I/O Register Summary

When the internal address bus matches the value written in the break address registers or when software writes a 1 to the BRKA bit in the break status and control register, the CPU starts a break interrupt by:

- Loading the instruction register with the SWI instruction
- Loading the program counter with $FFFC and $FFFD ($FEFC and $FEFD in monitor mode)

The break interrupt timing is:

- When a break address is placed at the address of the instruction opcode, the instruction is not executed until after completion of the break interrupt routine.
- When a break address is placed at an address of an instruction operand, the instruction is executed before the break interrupt.
- When software writes a 1 to the BRKA bit, the break interrupt occurs just before the next instruction is executed.

By updating a break address and clearing the BRKA bit in a break interrupt routine, a break interrupt can be generated continuously.

CAUTION

A break address should be placed at the address of the instruction opcode. When software does not change the break address and clears the BRKA bit in the first break interrupt routine, the next break interrupt will not be generated after exiting the interrupt routine even when the internal address bus matches the value written in the break address registers.

18.2.1.1 Flag Protection During Break Interrupts

The system integration module (SIM) controls whether or not module status bits can be cleared during the break state. The BCFE bit in the break flag control register (SBFCR) enables software to clear status bits during the break state. See 14.7.3 SIM Break Flag Control Register and the **Break Interrupts** subsection for each module.

18.2.1.2 TIM During Break Interrupts

A break interrupt stops the timer counter and inhibits captures.

18.2.1.3 COP During Break Interrupts

The COP is disabled during a break interrupt when V_{TST} is present on the $\overline{\text{RST}}$ pin.

18.2.2 Break Module Registers

These registers control and monitor operation of the break module:

- Break status and control register (BRKSCR)
- Break address register high (BRKH)
- Break address register low (BRKL)
- Break status register (SBSR)
- Break flag control register (SBFCR)

18.2.2.1 *Break Status and Control Register*

The break status and control register (BRKSCR) contains break module enable and status bits.

Address: $FE0B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	BRKE	BRKA	0	0	0	0	0	0
Write:			▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 18-4. Break Status and Control Register (BRKSCR)

BRKE — Break Enable Bit

This read/write bit enables breaks on break address register matches. Clear BRKE by writing a 0 to bit 7. Reset clears the BRKE bit.

1 = Breaks enabled on 16-bit address match
0 = Breaks disabled

BRKA — Break Active Bit

This read/write status and control bit is set when a break address match occurs. Writing a 1 to BRKA generates a break interrupt. Clear BRKA by writing a 0 to it before exiting the break routine. Reset clears the BRKA bit.

1 = Break address match
0 = No break address match

18.2.2.2 *Break Address Registers*

The break address registers (BRKH and BRKL) contain the high and low bytes of the desired breakpoint address. Reset clears the break address registers.

Address: $FE09

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

Figure 18-5. Break Address Register High (BRKH)

Address: $FE0A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 18-6. Break Address Register Low (BRKL)

18.2.2.3 SIM Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from wait mode. This register is only used in emulation mode.

Address: $FE00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R	R	R	R	R	R	SBSW	R
Write:							Note(1)	
Reset:							0	

R = Reserved

1. Writing a 0 clears SBSW.

Figure 18-7. SIM Break Status Register (SBSR)

SBSW — SIM Break Stop/Wait

SBSW can be read within the break state SWI routine. The user can modify the return address on the stack by subtracting one from it.

1 = Wait mode was exited by break interrupt
0 = Wait mode was not exited by break interrupt

18.2.2.4 SIM Break Flag Control Register

The SIM break control register (SBFCR) contains a bit that enables software to clear status bits while the MCU is in a break state.

Address: $FE03

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	BCFE	R	R	R	R	R	R	R
Write:								
Reset:	0							

R = Reserved

Figure 18-8. SIM Break Flag Control Register (SBFCR)

BCFE — Break Clear Flag Enable Bit

This read/write bit enables software to clear status bits by accessing status registers while the MCU is in a break state. To clear status bits during the break state, the BCFE bit must be set.

1 = Status bits clearable during break
0 = Status bits not clearable during break

18.2.3 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power- consumption standby modes. If enabled, the break module will remain enabled in wait and stop modes. However, since the internal address bus does not increment in these modes, a break interrupt will never be triggered.

18.3 Monitor Module (MON)

The monitor module allows debugging and programming of the microcontroller unit (MCU) through a single-wire interface with a host computer. Monitor mode entry can be achieved without use of the higher test voltage, V_{TST}, as long as vector addresses $FFFE and $FFFF are blank, thus reducing the hardware requirements for in-circuit programming.

Features of the monitor module include:

- Normal user-mode pin functionality
- One pin dedicated to serial communication between MCU and host computer
- Standard non-return-to-zero (NRZ) communication with host computer
- Standard communication baud rate
- Execution of code in random-access memory (RAM) or FLASH
- FLASH memory security feature(1)
- FLASH memory programming interface
- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)
- Normal monitor mode entry if V_{TST} is applied to $\overline{\text{IRQ}}$

18.3.1 Functional Description

Figure 18-9 shows a simplified diagram of the monitor mode.

The monitor module receives and executes commands from a host computer. Figure 18-10 and Figure 18-11 show example circuits used to enter monitor mode and communicate with a host computer via a standard RS-232 interface.

Simple monitor commands can access any memory address. In monitor mode, the MCU can execute code downloaded into RAM by a host computer while most MCU pins retain normal operating mode functions. All communication between the host computer and the MCU is through the PTA0 pin. A level-shifting and multiplexing interface is required between PTA0 and the host computer. PTA0 is used in a wired-OR configuration and requires a pullup resistor.

Table 18-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode must be entered after a power-on reset (POR) and will allow communication at 9600 baud provided one of the following sets of conditions is met:

- If $FFFE and $FFFF do not contain $FF (programmed state):
  - The external clock is 9.8304 MHz
  - IRQ = V_{TST}
- If $FFFE and $FFFF contain $FF (erased state):
  - The external clock is 9.8304 MHz
  - $\overline{\text{IRQ}}$ = V_{DD} (this can be implemented through the internal $\overline{\text{IRQ}}$ pullup)

---

1. No security feature is absolutely secure. However, Freescale's strategy is to make reading or copying the FLASH difficult for unauthorized users.

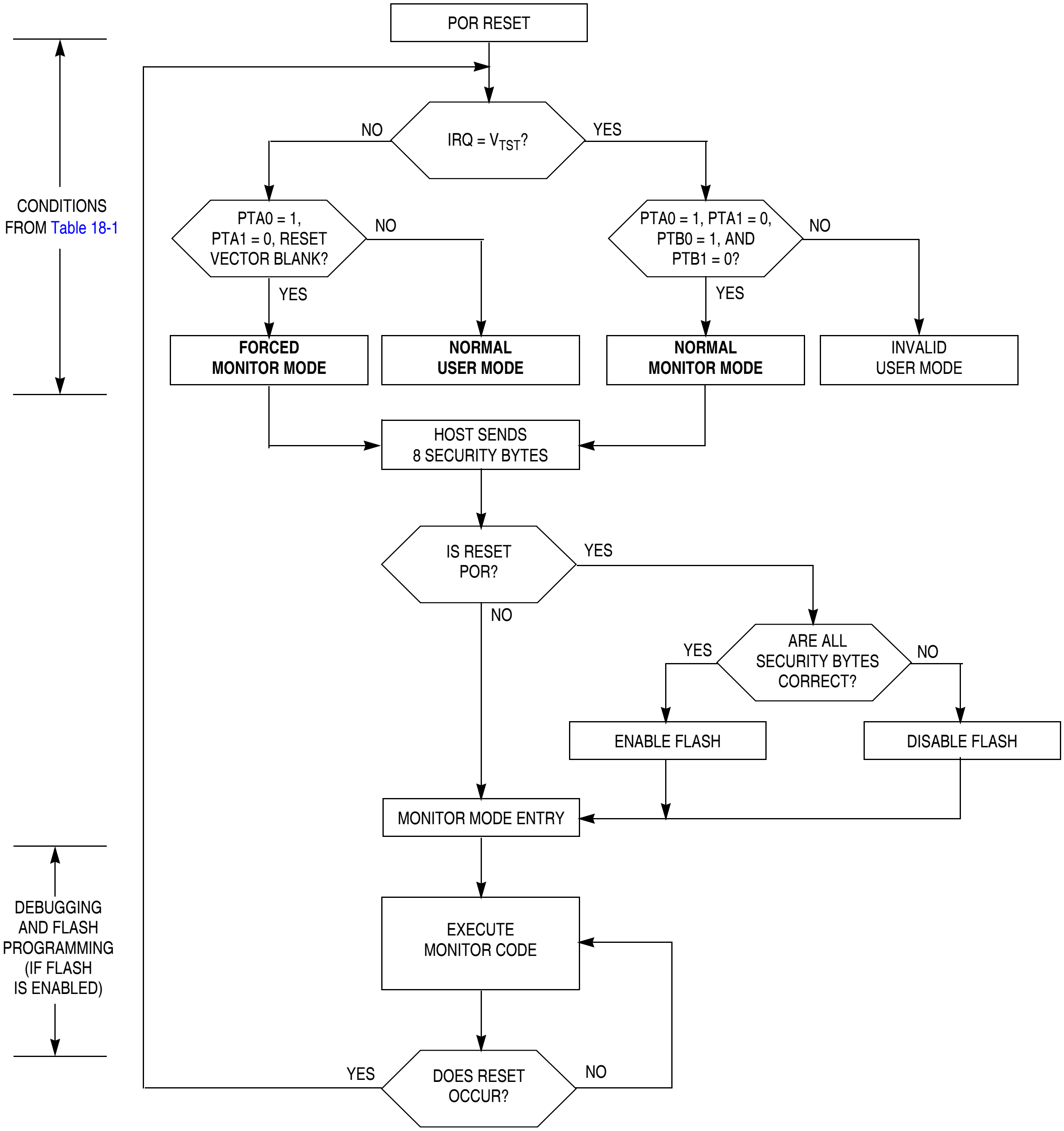


Figure 18-9. Simplified Monitor Mode Entry Flowchart

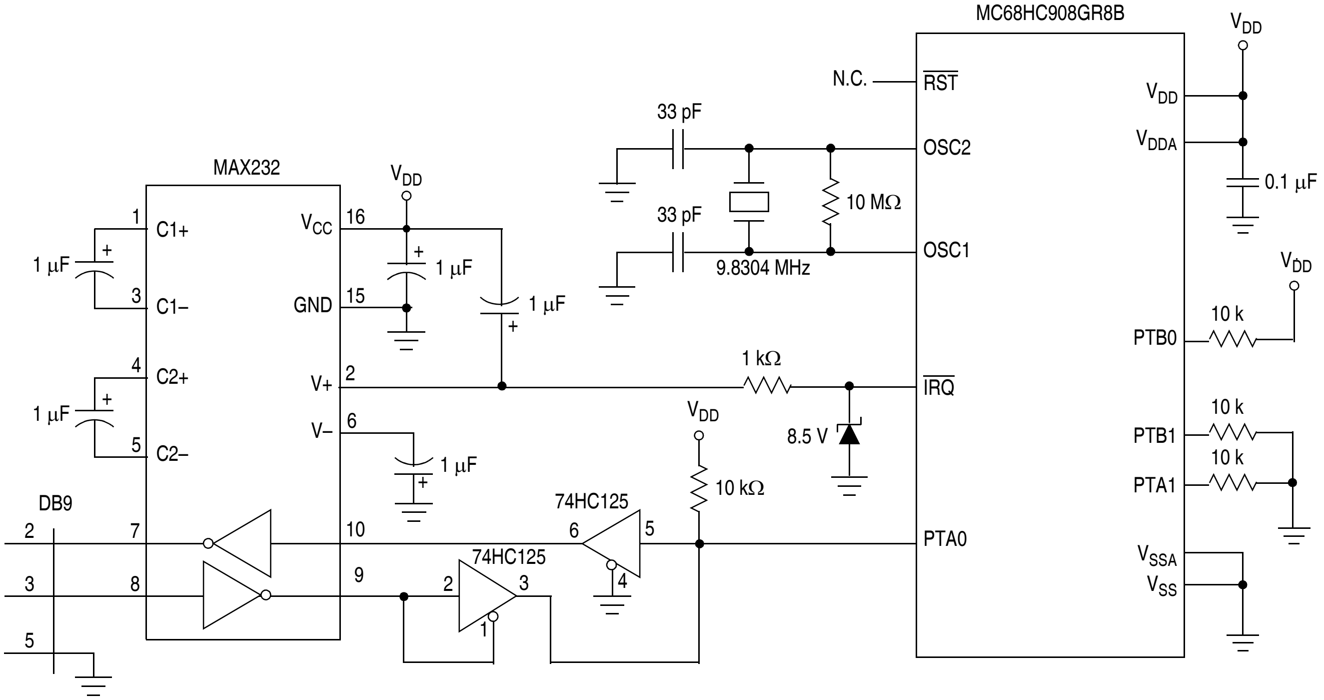


Figure 18-10. Normal Monitor Mode Circuit

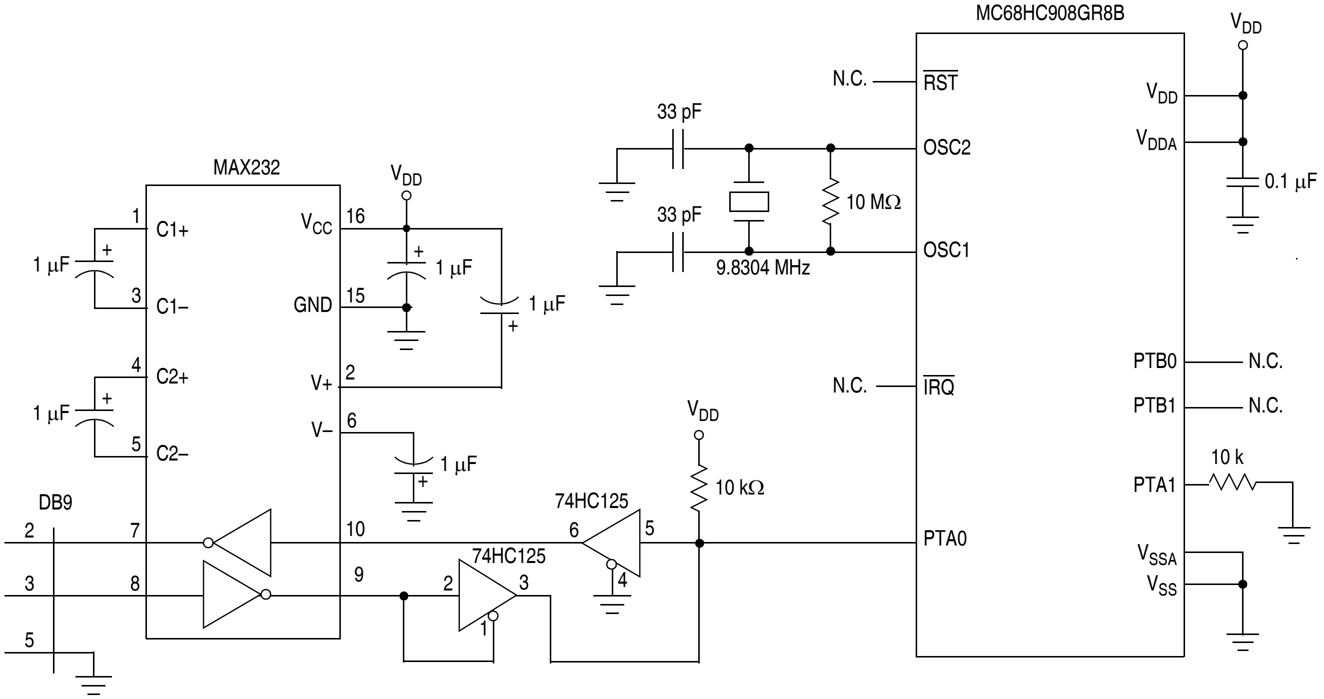


Figure 18-11. Forced Monitor Mode

Table 18-1. Monitor Mode Signal Requirements and Options

Mode	$\overline{\text{IRQ}}$	$\overline{\text{RST}}$	Reset Vector	Serial Communication		Mode Selection		PLL	COP	Communication Speed		
				PTA0	PTA1	PTB0	PTB1			External Clock	Bus Frequency	Baud Rate
Normal Monitor	V_{TST}	V_{DD} or V_{TST}	X	1	0	1	0	OFF	Disabled	9.8304 MHz	2.4576 MHz	9600
Forced Monitor	V_{DD}	V_{DD}	\$FFFF (blank)	1	0	X	X	OFF	Disabled	9.8304 MHz	2.4576 MHz	9600
User	V_{DD} or V_{SS}	V_{DD} or V_{TST}	Not \$FFFF	X	X	X	X	X	Enabled	X	X	X
MON08 Function [Pin No.]	V_{TST} [6]	$\overline{\text{RST}}$ [4]	—	COM [8]	SSEL [10]	MOD0 [12]	MOD1 [14]	—	—	OSC1 [13]	—	—

1. PTA0 must have a pullup resistor to V_{DD} in monitor mode.
2. Communication speed in the table is an example to obtain a baud rate of 9600.
   Baud rate using external oscillator is bus frequency / 256.
3. External clock is an 9.8304 MHz crystal on OSC1 and OSC2 or a 9.8304 MHz canned oscillator on OSC1.
4. X = don't care
5. MON08 pin refers to P&E Microcomputer Systems' MON08-Cyclone 2 by 8-pin connector.

NC	1	2	GND
NC	3	4	$\overline{\text{RST}}$
NC	5	6	$\overline{\text{IRQ}}$
NC	7	8	PTA0
NC	9	10	PTA1
NC	11	12	PTB0
OSC1	13	14	PTB1
V_{DD}	15	16	NC

Enter monitor mode with pin configuration shown in Table 18-1 with a power-on reset. The rising edge of $\overline{\text{RST}}$ latches monitor mode. Once monitor mode is latched, the levels on the port pins except PTA0 can change.

Once out of reset, the MCU waits for the host to send eight security bytes (see 18.3.2 Security). After the security bytes, the MCU sends a break signal (10 consecutive 0s) to the host, indicating that it is ready to receive a command.

18.3.1.1 Normal Monitor Mode

When monitor mode is entered with V_{TST} on $\overline{\text{IRQ}}$, the computer operating properly (COP) is disabled as long as V_{TST} is applied to either $\overline{\text{IRQ}}$ or $\overline{\text{RST}}$.

This condition states that as long as V_{TST} is maintained on the $\overline{\text{IRQ}}$ pin after entering monitor mode, or if V_{TST} is applied to $\overline{\text{RST}}$ after the initial reset to get into monitor mode (when V_{TST} was applied to $\overline{\text{IRQ}}$), then the COP will be disabled. In the latter situation, after V_{TST} is applied to the $\overline{\text{RST}}$ pin, V_{TST} can be removed from the $\overline{\text{IRQ}}$ pin in the interest of freeing the $\overline{\text{IRQ}}$ for normal functionality in monitor mode.

18.3.1.2 Forced Monitor Mode

If entering monitor mode without high voltage on $\overline{\text{IRQ}}$, all port B pin requirements and conditions are not in effect. This is to reduce circuit requirements when performing in-circuit programming.

NOTE

If the reset vector is blank and monitor mode is entered, the chip will see an additional reset cycle after the initial power-on reset (POR). Once the reset vector has been programmed, the traditional method of applying a voltage, V_{TST}, to $\overline{\text{IRQ}}$ must be used to enter monitor mode.

An external oscillator of 9.8304 MHz is required for a baud rate of 9600, as the internal bus frequency is automatically set to the external frequency divided by four.

When the forced monitor mode is entered the COP is always disabled regardless of the state of $\overline{\text{IRQ}}$ or $\overline{\text{RST}}$.

18.3.1.3 Monitor Vectors

In monitor mode, the MCU uses different vectors for reset, SWI (software interrupt), and break interrupt than those for user mode. The alternate vectors are in the $FE page instead of the $FF page and allow code execution from the internal monitor firmware instead of user code.

Table 18-2 summarizes the differences between user mode and monitor mode.

Table 18-2. Mode Differences

Modes	Functions					
	Reset Vector High	Reset Vector Low	Break Vector High	Break Vector Low	SWI Vector High	SWI Vector Low
User	$FFFE	$FFFF	$FFFC	$FFFD	$FFFC	$FFFD
Monitor	$FEFE	$FEFF	$FEFC	$FEFD	$FEFC	$FEFD

18.3.1.4 Data Format

Communication with the monitor ROM is in standard non-return-to-zero (NRZ) mark/space data format. Transmit and receive baud rates must be identical.

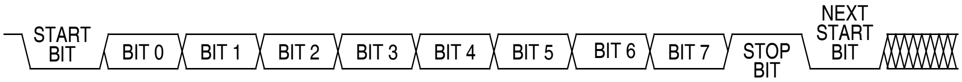


Figure 18-12. Monitor Data Format

18.3.1.5 Break Signal

A start bit (0) followed by nine 0 bits is a break signal. When the monitor receives a break signal, it drives the PTA0 pin high for the duration of approximately two bits and then echoes back the break signal.

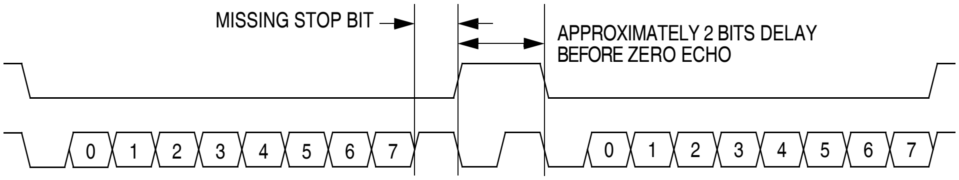


Figure 18-13. Break Transaction

18.3.1.6 Baud Rate

The communication baud rate is controlled by the crystal frequency or external clock and the state of the PTB4 pin (when $\overline{\text{IRQ}}$ is set to V_{TST}) upon entry into monitor mode. If monitor mode was entered with V_{DD} on $\overline{\text{IRQ}}$ and the reset vector blank, then the baud rate is independent of PTB4.

Table 18-1 also lists external frequencies required to achieve a standard baud rate of 9600 bps. The effective baud rate is the bus frequency divided by 256. If using a crystal as the clock source, be aware of the upper frequency limit that the internal clock module can handle. See 19.5 5.0 V DC Electrical Characteristics or 19.6 3.0 V DC Electrical Characteristics for this limit.

18.3.1.7 Commands

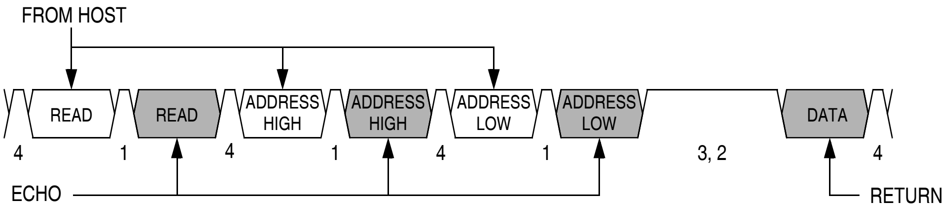
The monitor ROM firmware uses these commands:

- READ (read memory)
- WRITE (write memory)
- IREAD (indexed read)
- IWRITE (indexed write)
- READSP (read stack pointer)
- RUN (run user program)

The monitor ROM firmware echoes each received byte back to the PTA0 pin for error checking. An 11-bit delay at the end of each command allows the host to send a break character to cancel the command. A delay of two bit times occurs before each echo and before READ, IREAD, or READSP data is returned. The data returned by a read command appears after the echo of the last byte of the command.

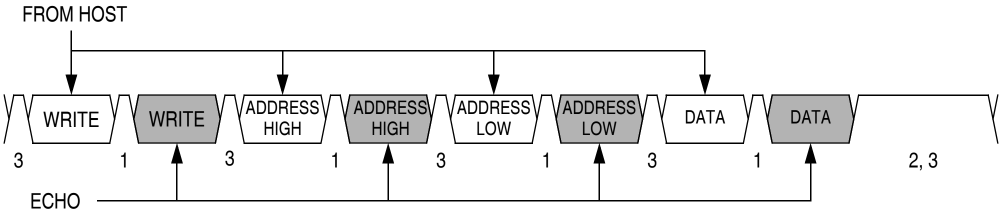
NOTE

Wait one bit time after each echo before sending the next byte.



Notes:
1 = Echo delay, approximately 2 bit times
2 = Data return delay, approximately 2 bit times
3 = Cancel command delay, 11 bit times
4 = Wait 1 bit time before sending next byte.

Figure 18-14. Read Transaction



Notes:
1 = Echo delay, approximately 2 bit times
2 = Cancel command delay, 11 bit times
3 = Wait 1 bit time before sending next byte.

Figure 18-15. Write Transaction

A brief description of each monitor mode command is given in Table 18-3 through Table 18-8.

Table 18-3. READ (Read Memory) Command

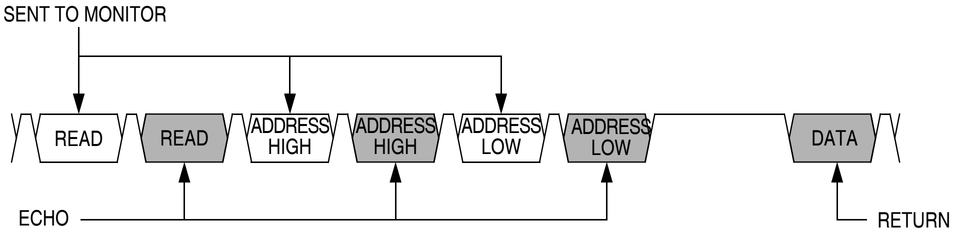
Description	Read byte from memory
Operand	2-byte address in high-byte:low-byte order
Data Returned	Returns contents of specified address
Opcode	$4A
Command Sequence 	

Table 18-4. WRITE (Write Memory) Command

Description	Write byte to memory
Operand	2-byte address in high-byte:low-byte order; low byte followed by data byte
Data Returned	None
Opcode	$49

Command Sequence

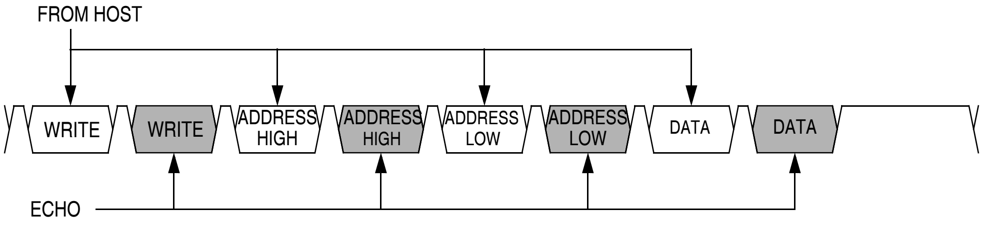


Table 18-5. IREAD (Indexed Read) Command

Description	Read next 2 bytes in memory from last address accessed
Operand	None
Data Returned	Returns contents of next two addresses
Opcode	$1A

Command Sequence

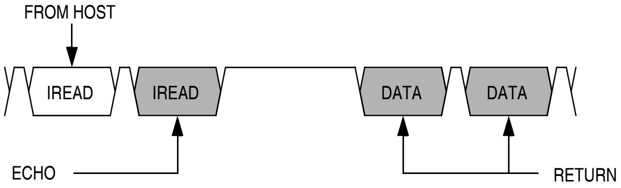
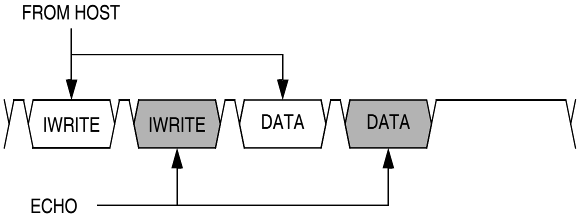


Table 18-6. IWRITE (Indexed Write) Command

Description	Write to last address accessed + 1
Operand	Single data byte
Data Returned	None
Opcode	$19

Command Sequence



A sequence of IREAD or IWRITE commands can access a block of memory sequentially over the full 64-Kbyte memory map.

Table 18-7. READSP (Read Stack Pointer) Command

Description	Reads stack pointer
Operand	None
Data Returned	Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order
Opcode	$0C

Command Sequence

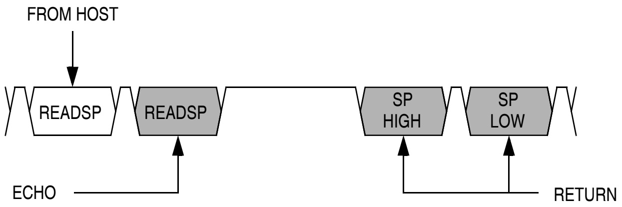
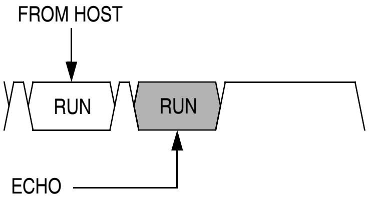


Table 18-8. RUN (Run User Program) Command

Description	Executes PULH and RTI instructions
Operand	None
Data Returned	None
Opcode	$28

Command Sequence



The MCU executes the SWI and PSHH instructions when it enters monitor mode. The RUN command tells the MCU to execute the PULH and RTI instructions. Before sending the RUN command, the host can modify the stacked CPU registers to prepare to run the host program. The READSP command returns the incremented stack pointer value, SP + 1. The high and low bytes of the program counter are at addresses SP + 5 and SP + 6.

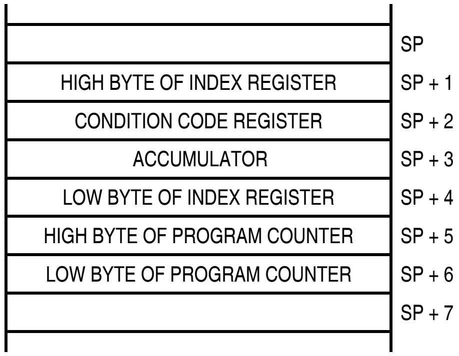


Figure 18-16. Stack Pointer at Monitor Mode Entry

18.3.2 Security

A security feature discourages unauthorized reading of FLASH locations while in monitor mode. The host can bypass the security feature at monitor mode entry by sending eight security bytes that match the bytes at locations \$FFF6–\$FFFD. Locations \$FFF6–\$FFFD contain user-defined data.

NOTE

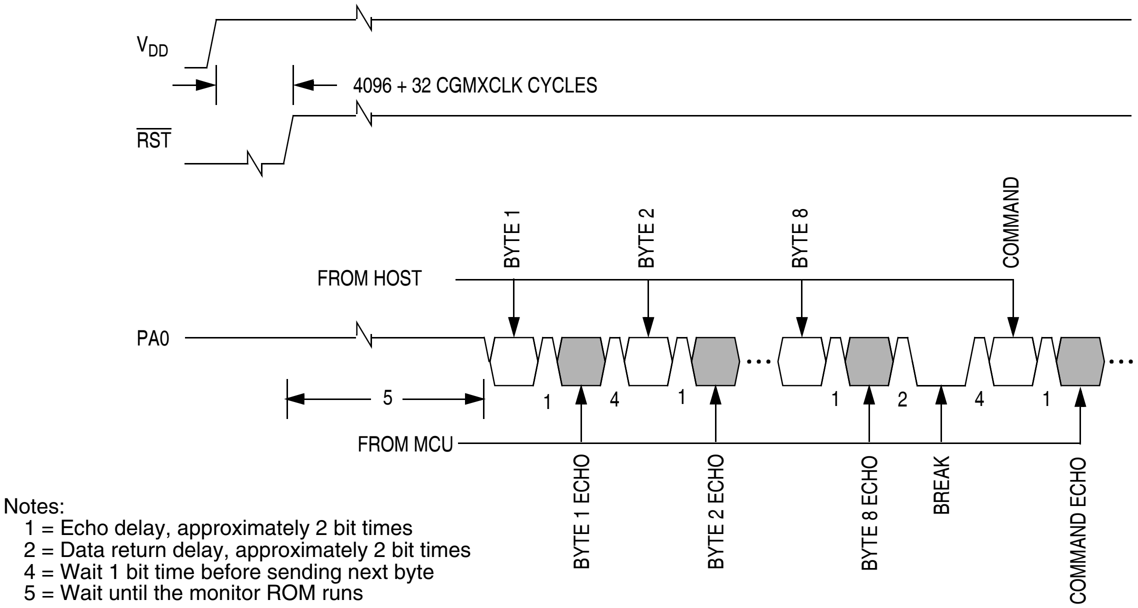
Do not leave locations \$FFF6–\$FFFD blank. For security reasons, program locations \$FFF6–\$FFFD even if they are not used for vectors.

During monitor mode entry, the MCU waits after the power-on reset for the host to send the eight security bytes on pin PTA0. If the received bytes match those at locations \$FFF6–\$FFFD, the host bypasses the security feature and can read all FLASH locations and execute code from FLASH. Security remains bypassed until a power-on reset occurs. If the reset was not a power-on reset, security remains bypassed and security code entry is not required. See Figure 18-17.

Upon power-on reset, if the received bytes of the security code do not match the data at locations \$FFF6–\$FFFD, the host fails to bypass the security feature. The MCU remains in monitor mode, but reading a FLASH location returns an invalid value and trying to execute code from FLASH causes an illegal address reset. After receiving the eight security bytes from the host, the MCU transmits a break character, signifying that it is ready to receive a command.

NOTE

The MCU does not transmit a break character until after the host sends the eight security bytes.



Notes:
1 = Echo delay, approximately 2 bit times
2 = Data return delay, approximately 2 bit times
4 = Wait 1 bit time before sending next byte
5 = Wait until the monitor ROM runs

Figure 18-17. Monitor Mode Entry Timing

To determine whether the security code entered is correct, check to see if bit 6 of RAM address \$40 is set. If it is, then the correct security code has been entered and FLASH can be accessed.

If the security sequence fails, the device should be reset by a power-on reset and brought up in monitor mode to attempt another entry. After failing the security sequence, the FLASH module can also be mass erased by executing an erase routine that was downloaded into internal RAM. The mass erase operation clears the security code locations so that all eight security bytes become \$FF (blank).

Chapter 19
Electrical Specifications

19.1 Introduction

This chapter contains electrical and timing specifications

19.2 Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the microcontroller unit (MCU) can be exposed without permanently damaging it.

NOTE

This device is not guaranteed to operate properly beyond the maximum ratings. Refer to 5.0 V DC Electrical Characteristics for guaranteed operating conditions.

Characteristic[1]	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to + 6.0	V
Input voltage	V_{In}	V_{SS} – 0.3 to V_{DD} + 0.3	V
Maximum current per pin excluding V_{DD}, V_{SS}, and PTC0–PTC	I	± 15	mA
Maximum current for pins PTC0–PTC1	$I_{PTC0–PTC1}$	± 25	mA
Maximum current into V_{DD}	I_{mvdd}	150	mA
Maximum current out of V_{SS}	I_{mvss}	150	mA
Storage temperature	T_{stg}	–55 to +150	°C

1. Voltages referenced to V_{SS}

NOTE

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{In} *and* V_{Out} *be constrained to the range* $V_{SS} \leq (V_{In}$ *or* $V_{Out}) \leq V_{DD}$*. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either* V_{SS} *or* V_{DD}*).*

19.3 Functional Operating Range

Characteristic	Symbol	Value	Unit
Operating temperature range	T_A	–40 to +125	°C
Operating voltage range	V_{DD}	3.0 ±10% 5.0 ±10%	V

19.4 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance PDIP (28-pin) SOIC (28-pin) QFP (32-pin)	θ_{JA}	 60 60 95	°C/W
I/O pin power dissipation	$P_{I/O}$	User-Determined	W
Power dissipation[1]	P_D	$P_D = (I_{DD} \times V_{DD}) + P_{I/O} = K/(T_J + 273\ °C)$	W
Constant[2]	K	$P_D \times (T_A + 273\ °C) + P_D^2 \times \theta_{JA}$	W/°C
Average junction temperature	T_J	$T_A + (P_D \times \theta_{JA})$	°C
Maximum junction temperature	T_{JM}	140	°C

1. Power dissipation is a function of temperature.
2. K is a constant unique to the device. K can be determined for a known T_A and measured P_D. With this value of K, P_D and T_J can be determined for any value of T_A.

19.5 5.0 V DC Electrical Characteristics

Characteristic(1)	Symbol	Min	Typ(2)	Max	Unit
Output high voltage					
(I_{Load} = –2.0 mA) all I/O pins	V_{OH}	V_{DD} – 0.8	—	—	V
(I_{Load} = –10.0 mA) all I/O pins	V_{OH}	V_{DD} – 1.5	—	—	V
(I_{Load} = –10.0 mA) pins PTC0–PTC1 only	V_{OH}	V_{DD} – 0.8	—	—	V
Maximum combined I_{OH} for port C, port E, port PTD0–PTD3	I_{OH1}	—	—	50	mA
Maximum combined I_{OH} for port PTD4–PTD6, port A, port B	I_{OH2}	—	—	50	mA
Maximum total I_{OH} for all port pins	I_{OHT}	—	—	100	mA
Output low voltage					
(I_{Load} = 1.6 mA) all I/O pins	V_{OL}	—	—	0.4	V
(I_{Load} = 10 mA) all I/O pins	V_{OL}	—	—	1.5	V
(I_{Load} = 15 mA) pins PTC0–PTC1 only	V_{OL}	—	—	1.0	V
Maximum combined I_{OL} for port C, port E, port PTD0–PTD3	I_{OL1}	—	—	50	mA
Maximum combined I_{OL} for port PTD4–PTD6, port A, port B	I_{OL2}	—	—	50	mA
Maximum total I_{OL} for all port pins	I_{OLT}	—	—	100	mA
Input high voltage					
All ports, IRQs, RESET	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
OSC1		0.8 x V_{DD}			
Input low voltage All ports, IRQs, RESET, OSC1	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V
V_{DD} supply current					
Run(3)	I_{DD}	—	15	20	mA
Wait(4)		—	4	8	mA
Stop with LVI		—	250	350	µA
Stop with LVI and TBM enabled(5)		—	1.25	1.6	mA
DC injection current(6), (7), (8), (9)					
Single pin limit					
$V_{IN} > V_{DD}$	I_{IC}	0	—	2	mA
$V_{IN} < V_{SS}$		0	—	–0.2	
Total MCU limit, includes sum of all stressed pins					
$V_{IN} > V_{DD}$		0	—	25	
$V_{IN} < V_{SS}$		0	—	–5	
I/O ports Hi-Z leakage current(10)	I_{IL}	—	10	±10	µA
Input current	I_{In}	—	—	1	µA
Pullup resistors (as input only) Ports PTA3/$\overline{\text{KBD3}}$–PTA0/$\overline{\text{KBD0}}$, PTC1–PTC0, PTD6/T2CH0–PTD0/$\overline{\text{SS}}$	R_{PU}	20	45	65	kΩ
Capacitance Ports (as input or output)	C_{Out} C_{In}	— —	— —	12 8	pF
Monitor mode entry voltage	V_{TST}	V_{DD} +2.5	—	8.5	V

Continued on next page

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Low-voltage inhibit, trip falling voltage – target	V_{TRIPF}	2.35	2.60	2.70	V
Low-voltage inhibit, trip rising voltage – target	V_{TRIPR}	2.45	2.66	2.80	V
Low-voltage inhibit reset/recover hysteresis – target ($V_{TRIPF} + V_{HYS} = V_{TRIPR}$)	V_{HYS}	—	60	—	mV
Input hysteresis (alternative input functions only)[6], [11]	V_{InHYS}	0.06 x V_{DD}	—	—	V
POR reset voltage[12]	V_{PORRST}	0	700	800	mV
POR rise time ramp rate[13]	R_{POR}	0.035	—	—	V/ms

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{osc} = 32.8 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}. Measured with all modules enabled.
4. Wait I_{DD} measured using external square wave clock source (f_{osc} = 32.8 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}. Measured with PLL and LVI enabled.
5. Stop I_{DD} with TBM enabled is measured using an external crystal clock source (f_{OSC} = 8 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All inputs configured as inputs.
6. This parameter is characterized and not tested on each device.
7. All functional non-supply pins are internally clamped to V_{SS} and V_{DD}.
8. Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.
9. Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{IN} > V_{DD}$) is greater than I_{DD}, the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if clock rate is very low (which would reduce overall power consumption).
10. Pullups and pulldowns are disabled. Port B leakage is specified in 19.12 ADC Characteristics.
11. Digital inputs have hysteresis whenever they are configured for any alternative peripheral input function other than as a general-purpose input.
12. Maximum is highest voltage that POR is possible.
13. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.

19.6 3.0 V DC Electrical Characteristics

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output high voltage					
(I_{Load} = –0.6 mA) all I/O pins	V_{OH}	V_{DD} – 0.3	—	—	V
(I_{Load} = –4.0 mA) all I/O pins	V_{OH}	V_{DD} – 1.0	—	—	V
(I_{Load} = –4.0 mA) pins PTC0–PTC1 only	V_{OH}	V_{DD} – 0.5	—	—	V
Maximum combined I_{OH} for port C, port E, port PTD0–PTD3	I_{OH1}	—	—	30	mA
Maximum combined I_{OH} for port PTD4–PTD6, port A, port B	I_{OH2}	—	—	30	mA
Maximum total I_{OH} for all port pins	I_{OHT}	—	—	60	mA
Output low voltage					
(I_{Load} = 0.5 mA) all I/O pins	V_{OL}	—	—	0.3	V
(I_{Load} = 6.0 mA) all I/O pins	V_{OL}	—	—	1.0	V
(I_{Load} = 10.0 mA) pins PTC0–PTC1 only	V_{OL}	—	—	0.8	V
Maximum combined I_{OL} for port C, port E, port PTD0–PTD3	I_{OL1}	—	—	30	mA
Maximum combined I_{OL} for port PTD4–PTD6, port A, port B	I_{OL2}	—	—	30	mA
Maximum total I_{OL} for all port pins	I_{OLT}	—	—	60	mA
Input high voltage					
All ports, IRQs, RESET	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
OSC1		0.8 x V_{DD}			
Input low voltage					
All ports, IRQs, RESET	V_{IL}	V_{SS}	—	0.3 x V_{DD}	V
OSC1				0.2 x V_{DD}	
V_{DD} supply current					
Run[3]	I_{DD}	—	4.5	8	mA
Wait[4]		—	1.65	4	mA
Stop with LVI		—	200	300	µA
Stop with LVI and TBM enabled[5]		—	500	700	µA
DC injection current[6], [7], [8], [9]					
Single pin limit					
$V_{IN} > V_{DD}$	I_{IC}	0	—	2	mA
$V_{IN} < V_{SS}$		0	—	–0.2	
Total MCU limit, includes sum of all stressed pins					
$V_{IN} > V_{DD}$		0	—	25	
$V_{IN} < V_{SS}$		0	—	–5	
I/O ports Hi-Z leakage current[10]	I_{IL}	—	10	±10	µA
Input current	I_{In}	—	—	1	µA
Pullup resistors (as input only) Ports PTA3/$\overline{\text{KBD37}}$–PTA0/$\overline{\text{KBD0}}$, PTC1–PTC0, PTD6/T2CH0–PTD0/$\overline{\text{SS}}$	R_{PU}	20	45	65	kΩ
Capacitance Ports (as input or output)	C_{Out} C_{In}	— —	— —	12 8	pF
Monitor mode entry voltage	V_{TST}	V_{DD} +2.5	—	8.5	V

Continued on next page

Characteristic(1)	Symbol	Min	Typ(2)	Max	Unit
Low-voltage inhibit, trip falling voltage – target	V_{TRIPF}	2.35	2.60	2.70	V
Low-voltage inhibit, trip rising voltage – target	V_{TRIPR}	2.45	2.66	2.80	V
Low-voltage inhibit reset/recover hysteresis – target ($V_{TRIPF} + V_{HYS} = V_{TRIPR}$)	V_{HYS}	—	60	—	mV
Input hysteresis (alternative input functions only)(6), (11)	V_{InHYS}	0.06 x V_{DD}	—	—	V
POR rearm voltage(12)	V_{POR}	0	—	100	mV
POR reset voltage(13)	V_{PORRST}	0	700	800	mV
POR rise time ramp rate(14)	R_{POR}	0.02	—	—	V/ms

1. V_{DD} = 3.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. Typical values reflect average measurements at midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{osc} = 16.4 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}. Measured with all modules enabled.
4. Wait I_{DD} measured using external square wave clock source (f_{osc} = 16.4 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}. Measured with PLL and LVI enabled.
5. Stop I_{DD} with TBM enabled is measured using an external crystal clock source (f_{OSC} = 4 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All inputs configured as inputs.
6. This parameter is characterized and not tested on each device.
7. All functional non-supply pins are internally clamped to V_{SS} and V_{DD}.
8. Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.
9. Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{IN} > V_{DD}$) is greater than I_{DD}, the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if clock rate is very low (which would reduce overall power consumption).
10. Pullups and pulldowns are disabled.
11. Digital inputs have hysteresis whenever they are configured for any alternative peripheral input function other than as a general-purpose input.
12. Maximum is highest voltage that POR is guaranteed.
13. Maximum is highest voltage that POR is possible.
14. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.

19.7 5.0 V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation[2] Crystal option External clock option[3]	f_{OSC}	 1 dc[4]	 8 32.8	 MHz MHz
Internal operating frequency	f_{OP}	—	8.2	MHz
Internal clock period ($1/f_{OP}$)	t_{cyc}	122	—	ns
$\overline{\text{RESET}}$ input pulse width low[5]	t_{IRL}	50	—	ns
$\overline{\text{IRQ}}$ interrupt pulse width low[6] (edge-triggered)	t_{ILIH}	50	—	ns
$\overline{\text{IRQ}}$ interrupt pulse period	t_{ILIL}	Note 8	—	t_{cyc}
16-bit timer[7] Input capture pulse width Input capture period	 t_{TH}, t_{TL} t_{TLTL}	Note 8	 — —	 ns t_{cyc}

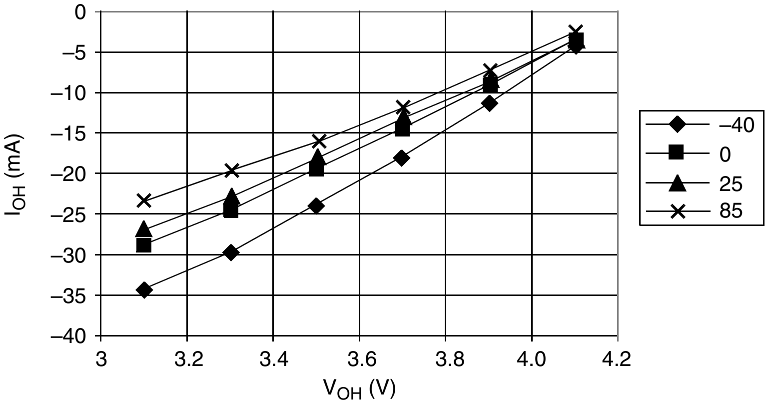
1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{SS} unless otherwise noted.
2. See 19.15 Clock Generation Module Characteristics for more information.
3. No more than 10% duty cycle deviation from 50%
4. Some modules may require a minimum frequency greater than dc for proper operation. See appropriate table for this information.
5. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
6. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
7. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
8. The minimum period, t_{ILIL} or t_{TLTL}, should not be less than the number of cycles it takes to execute the interrupt service routine plus t_{cyc}.

19.8 3.0 V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation[2] Crystal option External clock option[3]	f_{OSC}	 1 dc[4]	 8 16.4	 MHz MHz
Internal operating frequency	f_{OP}	—	4.1	MHz
Internal clock period ($1/f_{OP}$)	t_{cyc}	244	—	ns
$\overline{\text{RESET}}$ input pulse width low[5]	t_{IRL}	125	—	ns
$\overline{\text{IRQ}}$ interrupt pulse width low[6] (edge-triggered)	t_{ILIH}	125	—	ns
$\overline{\text{IRQ}}$ interrupt pulse period	t_{ILIL}	Note 8	—	t_{cyc}
16-bit timer[7] Input capture pulse width Input capture period	 t_{TH}, t_{TL} t_{TLTL}	Note 8	 — —	 ns t_{cyc}

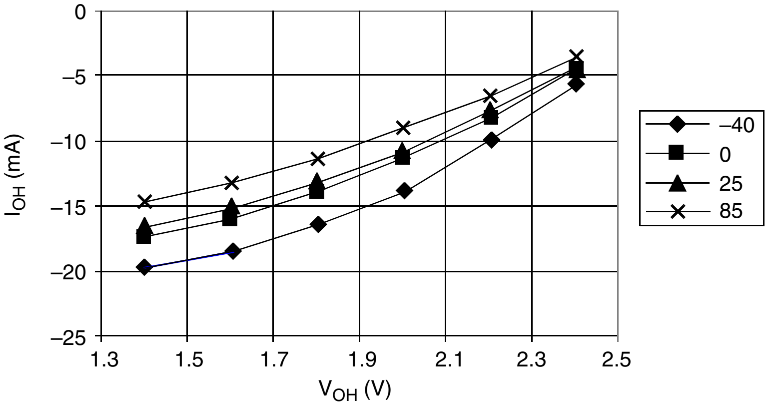
1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{SS} unless otherwise noted.
2. See 19.15 Clock Generation Module Characteristics for more information.
3. No more than 10% duty cycle deviation from 50%
4. Some modules may require a minimum frequency greater than dc for proper operation. See appropriate table for this information.
5. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
6. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
7. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
8. The minimum period, t_{ILIL} or t_{TLTL}, should not be less than the number of cycles it takes to execute the interrupt service routine plus t_{cyc}.

19.9 Output High-Voltage Characteristics



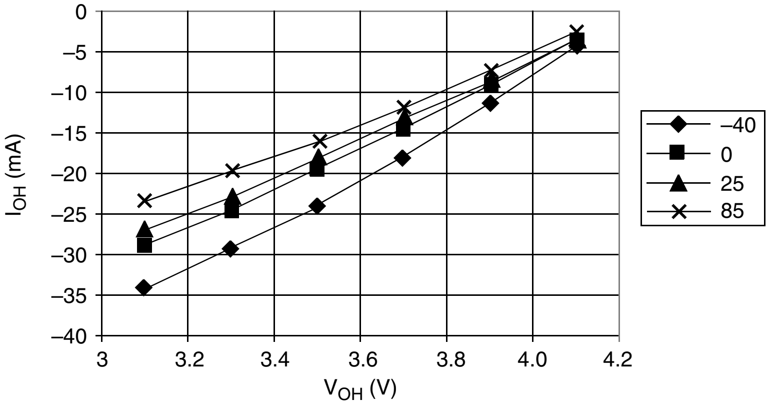
$V_{OH} > V_{DD}$ –0.8 V @ I_{OH} = –2.0 mA
$V_{OH} > V_{DD}$ –1.5 V @ I_{OH} = –10.0 mA

Figure 19-1. Typical High-Side Driver Characteristics — Port PTA3–PTA0 (V_{DD} = 4.5 Vdc)



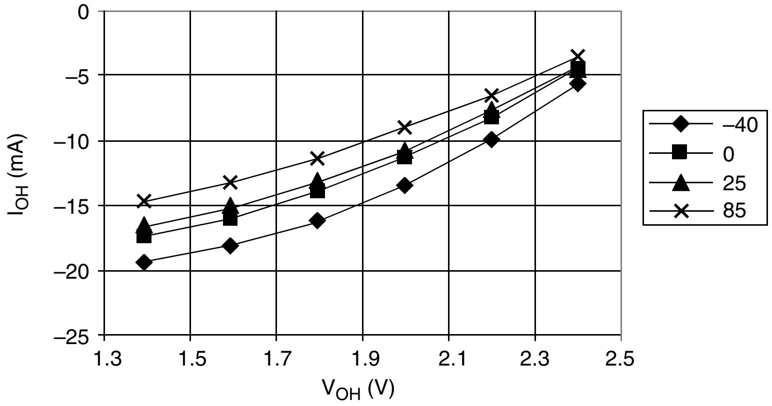
$V_{OH} > V_{DD}$ –0.3 V @ I_{OH} = –0.6 mA
$V_{OH} > V_{DD}$ –1.0 V @ I_{OH} = –4.0 mA

Figure 19-2. Typical High-Side Driver Characteristics — Port PTA3–PTA0 (V_{DD} = 2.7 Vdc)



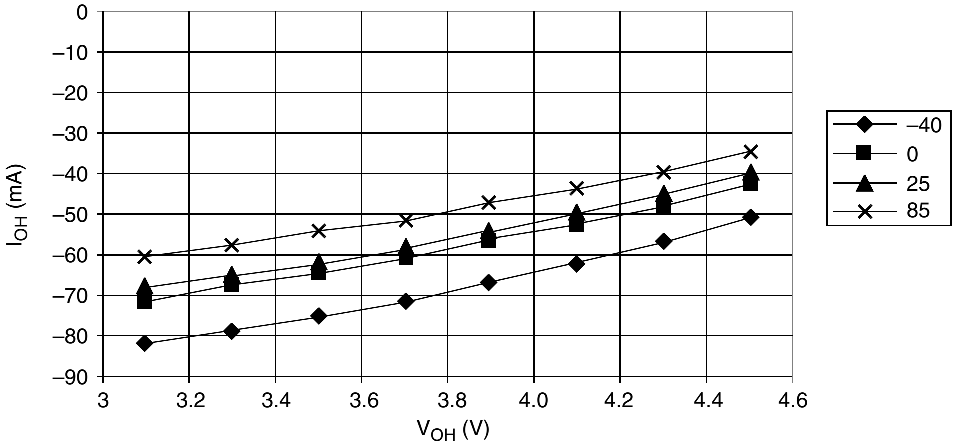
$V_{OH} > V_{DD}$ –0.8 V @ I_{OH} = –10.0 mA

Figure 19-3. Typical High-Side Driver Characteristics — Port PTC1–PTC0 (V_{DD} = 4.5 Vdc)



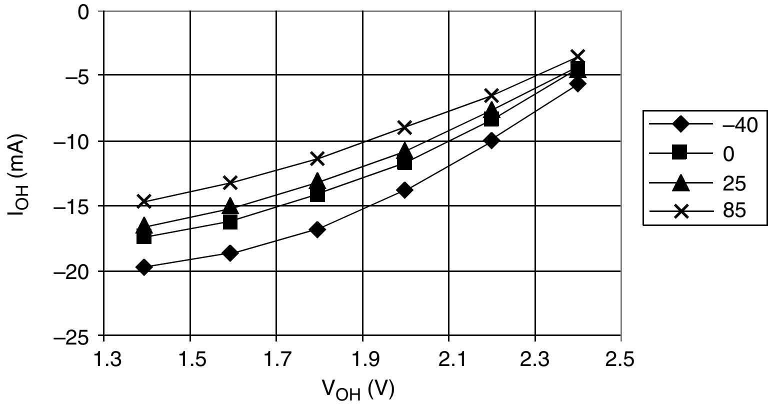
V_{OH} > V_{DD} –0.5 V @ I_{OH} = –4.0 mA

Figure 19-4. Typical High-Side Driver Characteristics — Port PTC1–PTC0 (V_{DD} = 2.7 Vdc)



V_{OH} > V_{DD} –0.8 V @ I_{OH} = –2.0 mA
V_{OH} > V_{DD} –1.5 V @ I_{OH} = –10.0 mA

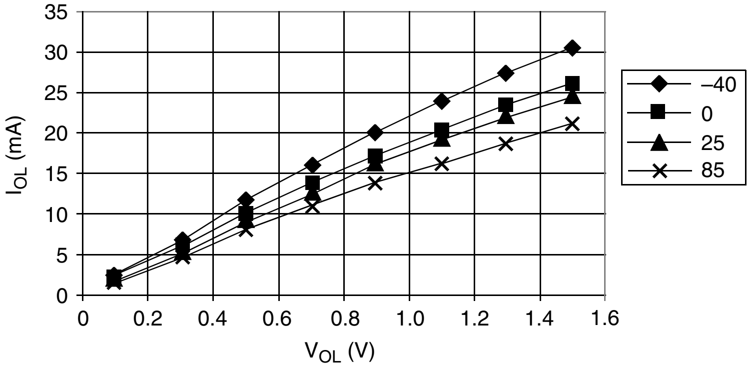
Figure 19-5. Typical High-Side Driver Characteristics — Ports PTB5–PTB0, PTD6–PTD0, and PTE1–PTE0 (V_{DD} = 5.5 Vdc)



V_{OH} > V_{DD} –0.3 V @ I_{OH} = –0.6 mA
V_{OH} > V_{DD} –1.0 V @ I_{OH} = –4.0 mA

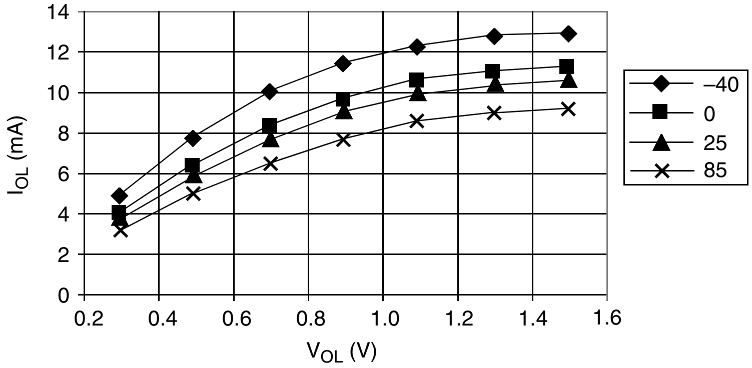
Figure 19-6. Typical High-Side Driver Characteristics — Ports PTB5–PTB0, PTD6–PTD0, and PTE1–PTE0 (V_{DD} = 2.7 Vdc)

19.10 Output Low-Voltage Characteristics



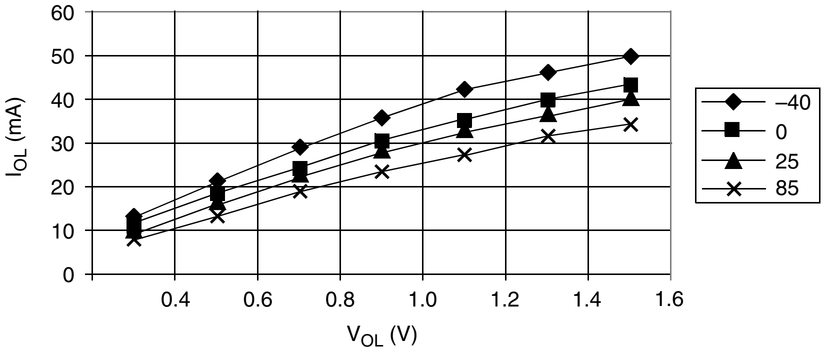
$V_{OL} < 0.4$ V @ $I_{OL} = 1.6$ mA
$V_{OL} < 1.5$ V @ $I_{OL} = 10.0$ mA

Figure 19-7. Typical Low-Side Driver Characteristics — Port PTA3–PTA0 ($V_{DD} = 5.5$ Vdc)



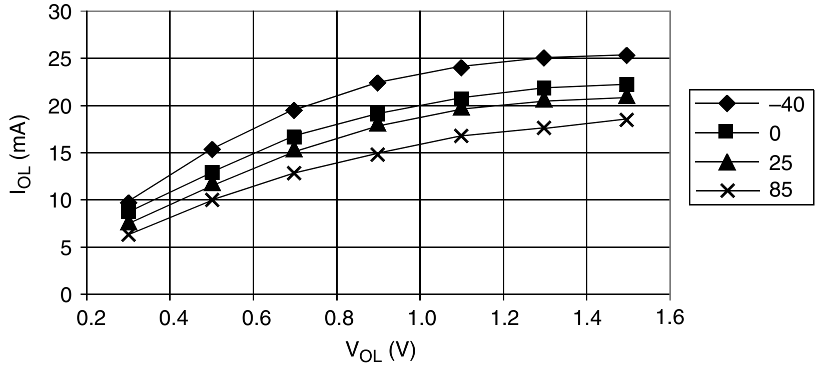
$V_{OL} < 0.3$ V @ $I_{OL} = 0.5$ mA
$V_{OL} < 1.0$ V @ $I_{OL} = 6.0$ mA

Figure 19-8. Typical Low-Side Driver Characteristics — Port PTA3–PTA0 ($V_{DD} = 2.7$ Vdc)



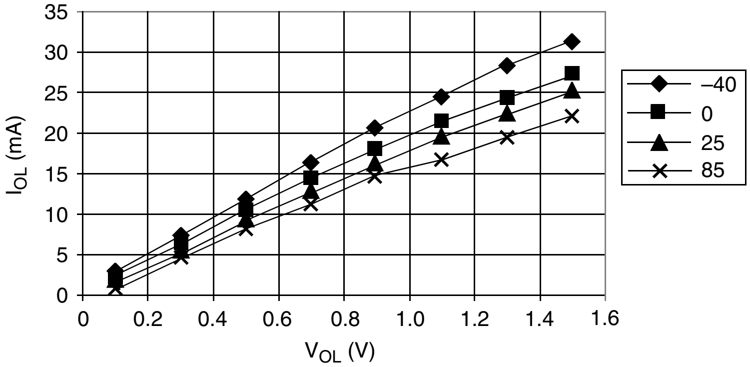
$V_{OL} < 1.0$ V @ $I_{OL} = 15$ mA

Figure 19-9. Typical Low-Side Driver Characteristics — Port PTC1–PTC0 ($V_{DD} = 4.5$ Vdc)



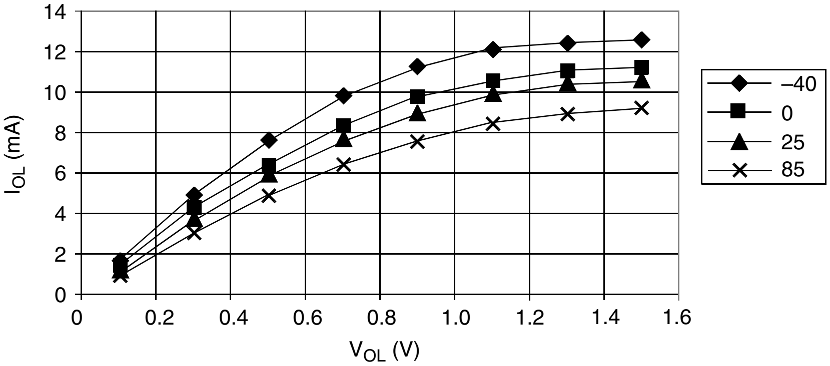
$V_{OL} < 0.8$ V @ $I_{OL} = 10$ mA

Figure 19-10. Typical Low-Side Driver Characteristics — Port PTC1–PTC0 (V_{DD} = 2.7 Vdc)



$V_{OL} < 0.4$ V @ $I_{OL} = 1.6$ mA
$V_{OL} < 1.5$ V @ $I_{OL} = 10.0$ mA

Figure 19-11. Typical Low-Side Driver Characteristics — Ports PTB5–PTB0, PTD6–PTD0, and PTE1–PTE0 (V_{DD} = 5.5 Vdc)



$V_{OL} < 0.3$ V @ $I_{OL} = 0.5$ mA
$V_{OL} < 1.0$ V @ $I_{OL} = 6.0$ mA

Figure 19-12. Typical Low-Side Driver Characteristics — Ports PTB5–PTB0, PTD6–PTD0, and PTE1–PTE0 (V_{DD} = 2.7 Vdc)

19.11 Typical Supply Currents

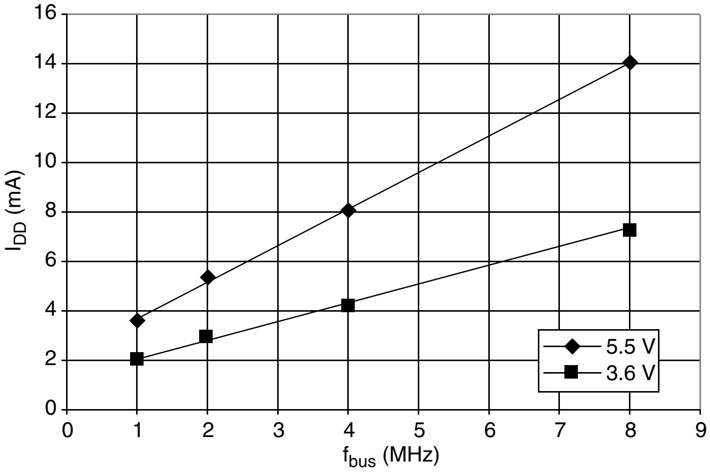


Figure 19-13. Typical Operating I_{DD}, with All Modules Turned On (–40 °C to 125 °C)

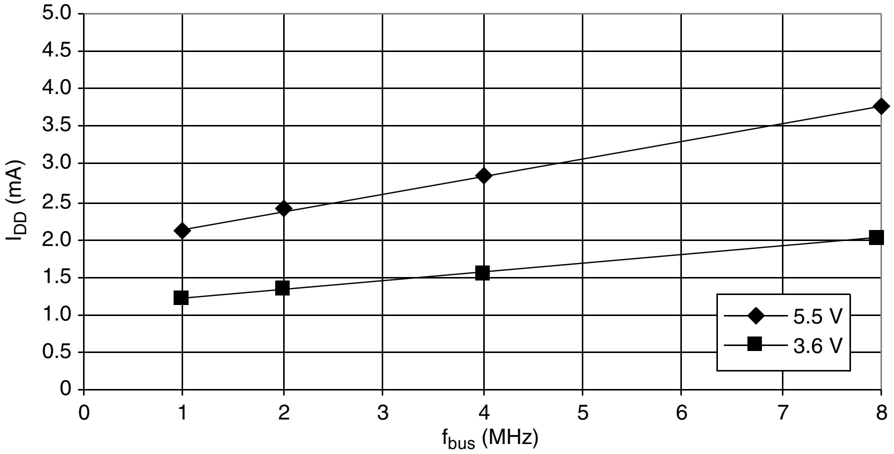


Figure 19-14. Typical Wait Mode I_{DD}, with all Modules Disabled (–40 °C to 125 °C)

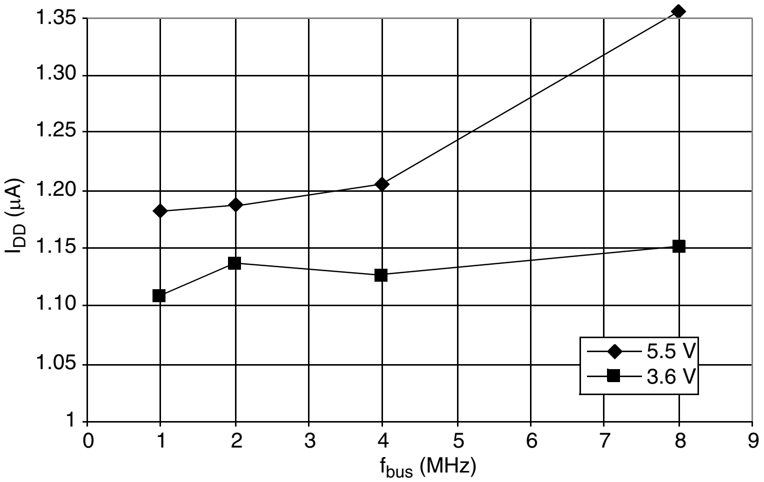


Figure 19-15. Typical Stop Mode I_{DD}, with all Modules Disabled (–40 °C to 125 °C)

19.12 ADC Characteristics

Characteristic[1]	Symbol	Min	Max	Unit	Comments
Supply voltage	V_{DDAD}	2.7 (V_{DD} min)	5.5 (V_{DD} max)	V	V_{DDAD} should be tied to the same potential as V_{DD} via separate traces.
Input voltages	V_{ADIN}	0	V_{DDAD}	V	$V_{ADIN} <= V_{REFH}$
Resolution	B_{AD}	8	8	Bits	
Absolute accuracy (V_{REFL} = 0 V, $V_{DDAD} = V_{REFH}$ = 5 V ± 10%)	A_{AD}	––	± 1	LSB	Includes quantization
ADC internal clock	f_{ADIC}	0.5	1.048	MHz	$t_{AIC} = 1/f_{ADIC}$, tested only at 1 MHz
Conversion range	R_{AD}	V_{REFL}	V_{REFH}	V	$V_{REFH} = V_{DDAD}$ $V_{REFL} = V_{SSAD}$
Power-up time	t_{ADPU}	16		t_{AIC} cycles	
Conversion time	t_{ADC}	16	17	t_{AIC} cycles	
Sample time[2]	t_{ADS}	5	—	t_{AIC} cycles	
Zero input reading[3]	Z_{ADI}	00	01	Hex	$V_{IN} = V_{REFL}$
Full-scale reading[3]	F_{ADI}	FE	FF	Hex	$V_{IN} = V_{REFH}$
Input capacitance	C_{ADI}	—	(20) 8	pF	Not tested
Input leakage[4] Port B	—	—	± 1	μA	

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, V_{DDAD} = 5.0 Vdc ± 10%, V_{SSAD} = 0 Vdc, V_{REFH} = 5.0 Vdc ± 10%, V_{REFL} = 0
2. Source impedances greater than 10 kΩ adversely affect internal RC charging time during input sampling.
3. Zero-input/full-scale reading requires sufficient decoupling measures for accurate conversions.
4. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

19.13 5.0 V SPI Characteristics

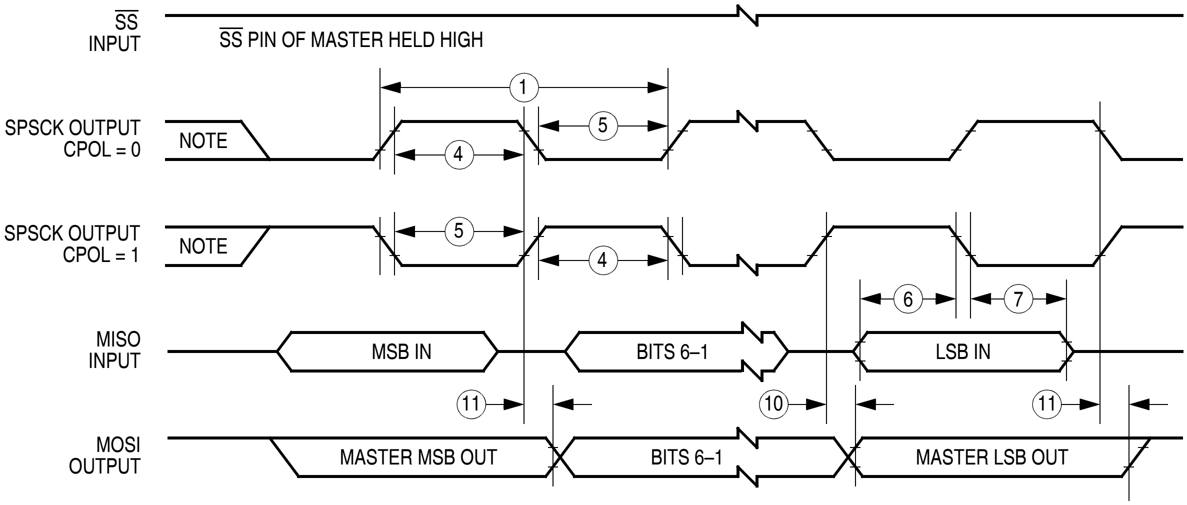
Diagram Number[1]	Characteristic[2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	$f_{OP(M)}$ $f_{OP(S)}$	$f_{OP}/128$ DC	$f_{OP}/2$ f_{OP}	MHz MHz
1	Cycle time Master Slave	$t_{CYC(M)}$ $t_{CYC(S)}$	2 1	128 —	t_{cyc} t_{cyc}
2	Enable lead time	$t_{Lead(S)}$	1	—	t_{cyc}
3	Enable lag time	$t_{Lag(S)}$	1	—	t_{cyc}
4	Clock (SPSCK) high time Master Slave	$t_{SCKH(M)}$ $t_{SCKH(S)}$	t_{cyc} –25 1/2 t_{cyc} –25	64 t_{cyc} —	ns ns
5	Clock (SPSCK) low time Master Slave	$t_{SCKL(M)}$ $t_{SCKL(S)}$	t_{cyc} –25 1/2 t_{cyc} –25	64 t_{cyc} —	ns ns
6	Data setup time (inputs) Master Slave	$t_{SU(M)}$ $t_{SU(S)}$	30 30	— —	ns ns
7	Data hold time (inputs) Master Slave	$t_{H(M)}$ $t_{H(S)}$	30 30	— —	ns ns
8	Access time, slave[3] CPHA = 0 CPHA = 1	$t_{A(CP0)}$ $t_{A(CP1)}$	0 0	40 40	ns ns
9	Disable time, slave[4]	$t_{DIS(S)}$	—	40	ns
10	Data valid time, after enable edge Master Slave[5]	$t_{V(M)}$ $t_{V(S)}$	— —	50 50	ns ns
11	Data hold time, outputs, after enable edge Master Slave	$t_{HO(M)}$ $t_{HO(S)}$	0 0	— —	ns ns

1. Numbers refer to dimensions in Figure 19-16 and Figure 19-17.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless noted; 100 pF load on all SPI pins.
3. Time to data active from high-impedance state
4. Hold time to high-impedance state
5. With 100 pF on all SPI pins

19.14 3.0 V SPI Characteristics

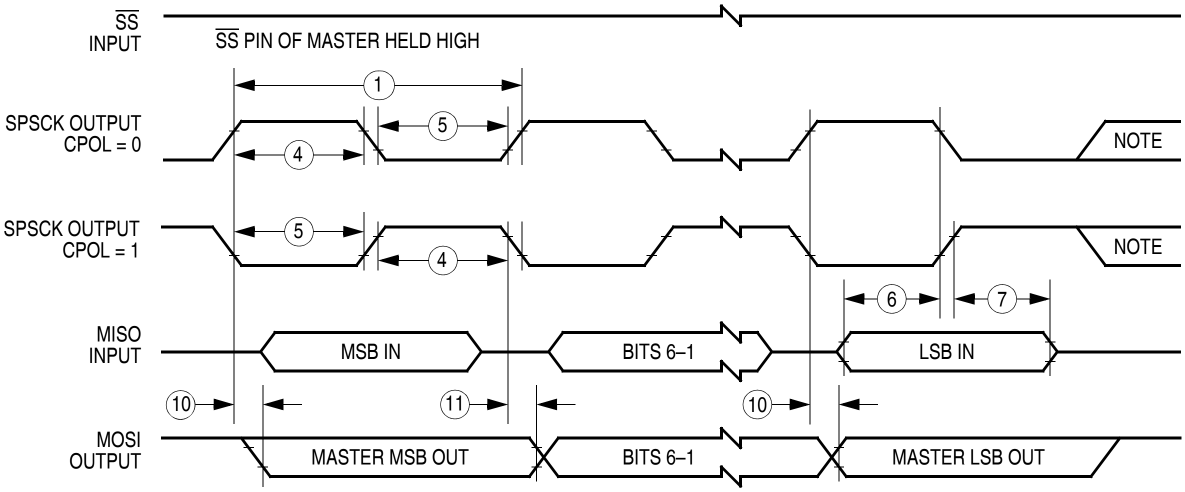
Diagram Number[1]	Characteristic[2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	$f_{OP(M)}$ $f_{OP(S)}$	$f_{OP}/128$ DC	$f_{OP}/2$ f_{OP}	MHz MHz
1	Cycle time Master Slave	$t_{CYC(M)}$ $t_{CYC(S)}$	2 1	128 —	t_{cyc} t_{cyc}
2	Enable lead time	$t_{Lead(s)}$	1	—	t_{cyc}
3	Enable lag time	$t_{Lag(s)}$	1	—	t_{cyc}
4	Clock (SPSCK) high time Master Slave	$t_{SCKH(M)}$ $t_{SCKH(S)}$	t_{cyc} −35 1/2 t_{cyc} −35	64 t_{cyc} —	ns ns
5	Clock (SPSCK) low time Master Slave	$t_{SCKL(M)}$ $t_{SCKL(S)}$	t_{cyc} −35 1/2 t_{cyc} −35	64 t_{cyc} —	ns ns
6	Data setup time (inputs) Master Slave	$t_{SU(M)}$ $t_{SU(S)}$	40 40	— —	ns ns
7	Data hold time (inputs) Master Slave	$t_{H(M)}$ $t_{H(S)}$	40 40	— —	ns ns
8	Access time, slave[3] CPHA = 0 CPHA = 1	$t_{A(CP0)}$ $t_{A(CP1)}$	0 0	50 50	ns ns
9	Disable time, slave[4]	$t_{DIS(S)}$	—	50	ns
10	Data valid time, after enable edge Master Slave[5]	$t_{V(M)}$ $t_{V(S)}$	— —	60 60	ns ns
11	Data hold time, outputs, after enable edge Master Slave	$t_{HO(M)}$ $t_{HO(S)}$	0 0	— —	ns ns

1. Numbers refer to dimensions in Figure 19-16 and Figure 19-17.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless noted; 100 pF load on all SPI pins.
3. Time to data active from high-impedance state
4. Hold time to high-impedance state
5. With 100 pF on all SPI pins



Note: This first clock edge is generated internally, but is not seen at the SPSCK pin.

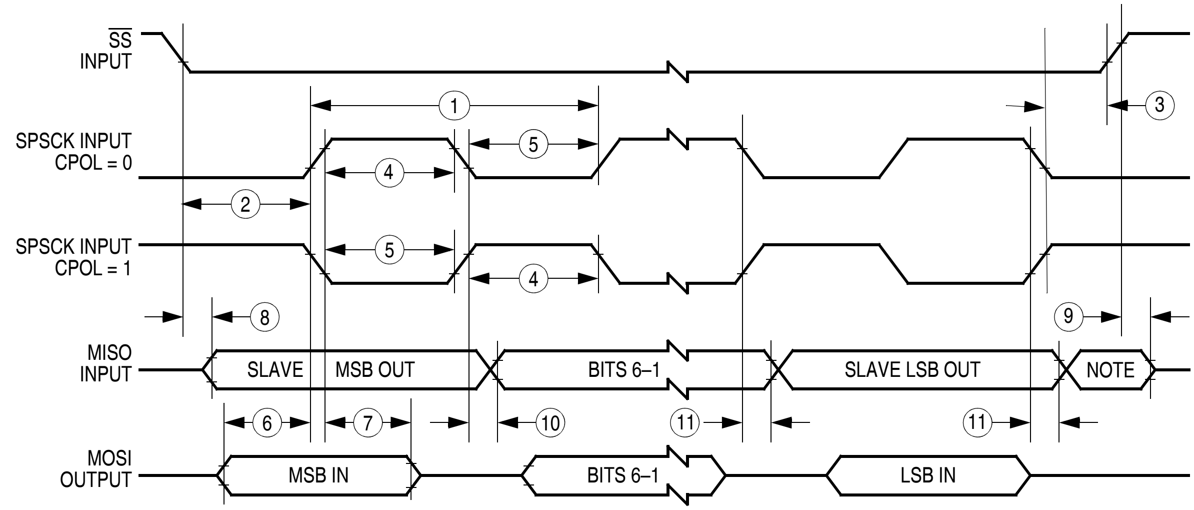
a) SPI Master Timing (CPHA = 0)



Note: This last clock edge is generated internally, but is not seen at the SPSCK pin.

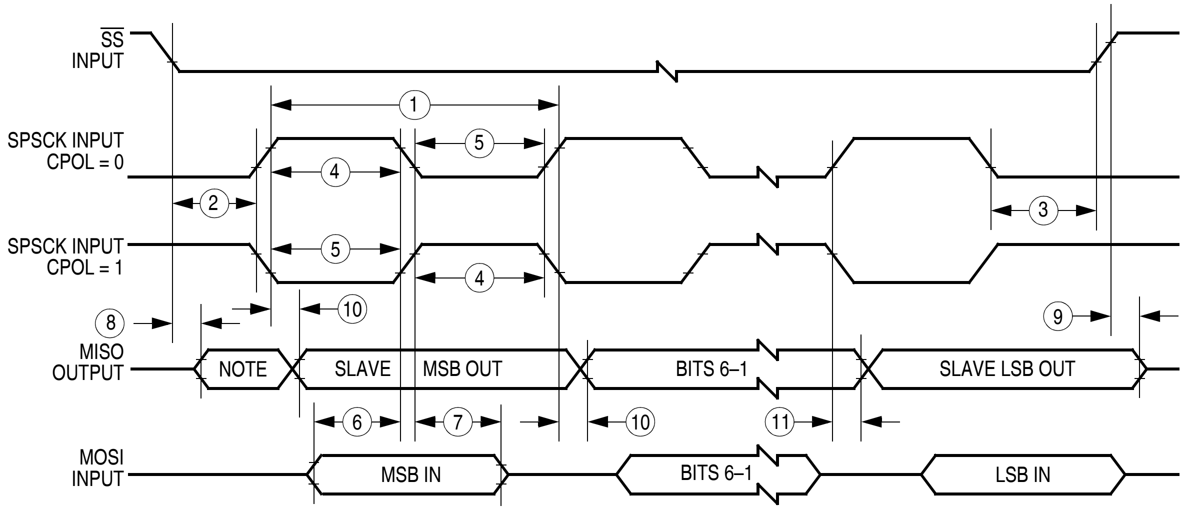
b) SPI Master Timing (CPHA = 1)

Figure 19-16. SPI Master Timing



Note: Not defined but normally MSB of character just received

a) SPI Slave Timing (CPHA = 0)



Note: Not defined but normally LSB of character previously transmitted

b) SPI Slave Timing (CPHA = 1)

Figure 19-17. SPI Slave Timing

19.15 Clock Generation Module Characteristics

19.15.1 CGM Component Specifications

Characteristic	Symbol	Min	Typ	Max	Unit
Crystal reference frequency(1)	f_{XCLK}	1	4	8	MHz
Crystal load capacitance(2)	C_L	—	20	—	pF
Crystal fixed capacitance(2)	C_1	—	$2 \times C_L$	—	pF
Crystal tuning capacitance(2)	C_2	—	$2 \times C_L$	—	pF
Feedback bias resistor	R_B	0.5	1	10	MΩ
Series resistor	R_S	—	0	—	Ω

1. Fundamental mode crystals only
2. Consult crystal manufacturer's data.

19.15.2 CGM Electrical Specifications

Description	Symbol	Min	Typ	Max	Unit
Operating voltage	V_{DD}	2.7	—	5.5	V
Operating temperature	T_A	–40	25	125	°C
Crystal reference frequency	f_{RCLK}	1	4	8	MHz
Range nominal multiplier	f_{NOM}	—	38.4	—	kHz
VCO center-of-range frequency	f_{VRS}	4	—	40	MHz
VCO range linear range multiplier	L	1	—	255	
VCO power-of-two range multiplier	2^E	1	—	4	
VCO multiply factor	N	1	—	4095	
VCO prescale multiplier	2^P	1	1	8	
Reference divider factor	R	1	1	15	
VCO operating frequency	f_{VCLK}	4	—	40	MHz
Bus operating frequency (5-V operation)	f_{BUS}	—	—	8.2	MHz
Bus frequency (3-V operation)	f_{BUS}	—	—	4.1	MHz
Manual acquisition time	t_{Lock}	—	—	5	ms
Automatic lock time	t_{Lock}	—	—	5	ms
PLL jitter(1)	f_J	0	—	f_{RCLK} x 0.025% x 2^P N/4	Hz
External clock input frequency PLL disabled	f_{OSC}	dc	—	32.8 M	Hz
External clock input frequency PLL enabled	f_{OSC}	1	4	8	MHz

1. Deviation of average bus frequency over 2 ms. N = VCO multiplier.

19.16 Timer Interface Module Characteristics

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	t_{TIH}, t_{TIL}	1	—	t_{cyc}

19.17 Memory Characteristics

Characteristic	Symbol	Min	Typ	Max	Unit
RAM data retention voltage	V_{RDR}	1.3	—	—	V
FLASH program bus clock frequency	—	1	—	—	MHz
FLASH read bus clock frequency	f_{Read}(1)	—	—	8.2	MHz
FLASH page erase time Limited endurance (<1 K cycles) Maximum endurance (>1 K cycles)	t_{Erase}	 0.9 3.6	 1 4	 1.1 5.5	ms
FLASH mass erase time	t_{MErase}	4	—	—	ms
FLASH PGM/ERASE to HVEN setup time	t_{NVS}	10	—	—	µs
FLASH high-voltage hold time	t_{NVH}	5	—	—	µs
FLASH high-voltage hold time (mass erase)	t_{NVHL}	100	—	—	µs
FLASH program hold time	t_{PGS}	5	—	—	µs
FLASH program time	t_{PROG}	30	—	40	µs
FLASH return to read time	t_{RCV}(2)	1	—	—	µs
FLASH cumulative program hv period	t_{HV}(3)	—	—	4	ms
FLASH endurance(4)	—	10 k	100 k	—	Cycles
FLASH data retention time(5)	—	15	100	—	Years

1. f_{Read} is defined as the frequency range for which the FLASH memory can be read.
2. t_{RCV} is defined as the time it needs before the FLASH can be read after turning off the high voltage charge pump, by clearing HVEN to 0.
3. t_{HV} is defined as the cumulative high voltage programming time to the same row before next erase.
   t_{HV} must satisfy this condition: $t_{NVS} + t_{NVH} + t_{PGS} + (t_{PROG} \times 32) \leq t_{HV}$ maximum.
4. Typical endurance was evaluated for this product family. For additional information on how Freescale defines *Typical Endurance*, please refer to Engineering Bulletin EB619.
5. Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how Freescale defines *Typical Data Retention*, please refer to Engineering Bulletin EB618.

Chapter 20
Ordering Information and Mechanical Specifications

20.1 Introduction

This section provides ordering information for the MC68HC908GR8B along with the dimensions for:

- 32-pin low-profile quad flat pack package (case 873A)
- 28-pin dual in-line package (case 710)
- 28-pin small outline package (case 751F)

The following figures show the latest package drawings at the time of this publication. To make sure that you have the latest package specifications, contact your local Freescale Semiconductor Sales Office.

20.2 MC Order Numbers

Table 20-1. MC Order Numbers

MC Order Number	Operating Temperature Range	Package
MC68HC908GR8BCFA	–40°C to +85°C	32-pin low-profile quad flat package (LQFP)
MC68HC908GR8BVFA	–40°C to +105°C	
MC68HC908GR8BMFA	–40°C to +125°C	
MC68HC908GR8BCP	–40°C to +85°C	28-pin dual in-ine package (PDIP)
MC68HC908GR8BCDW	–40°C to +85°C	28-pin small outout package (SOIC

Temperature designators:
C = –40°C to +85°C
V = –40°C to +105°C
M = –40°C to +125°C

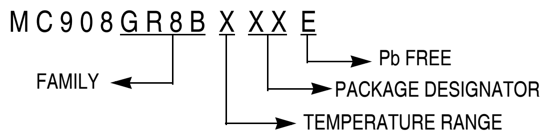
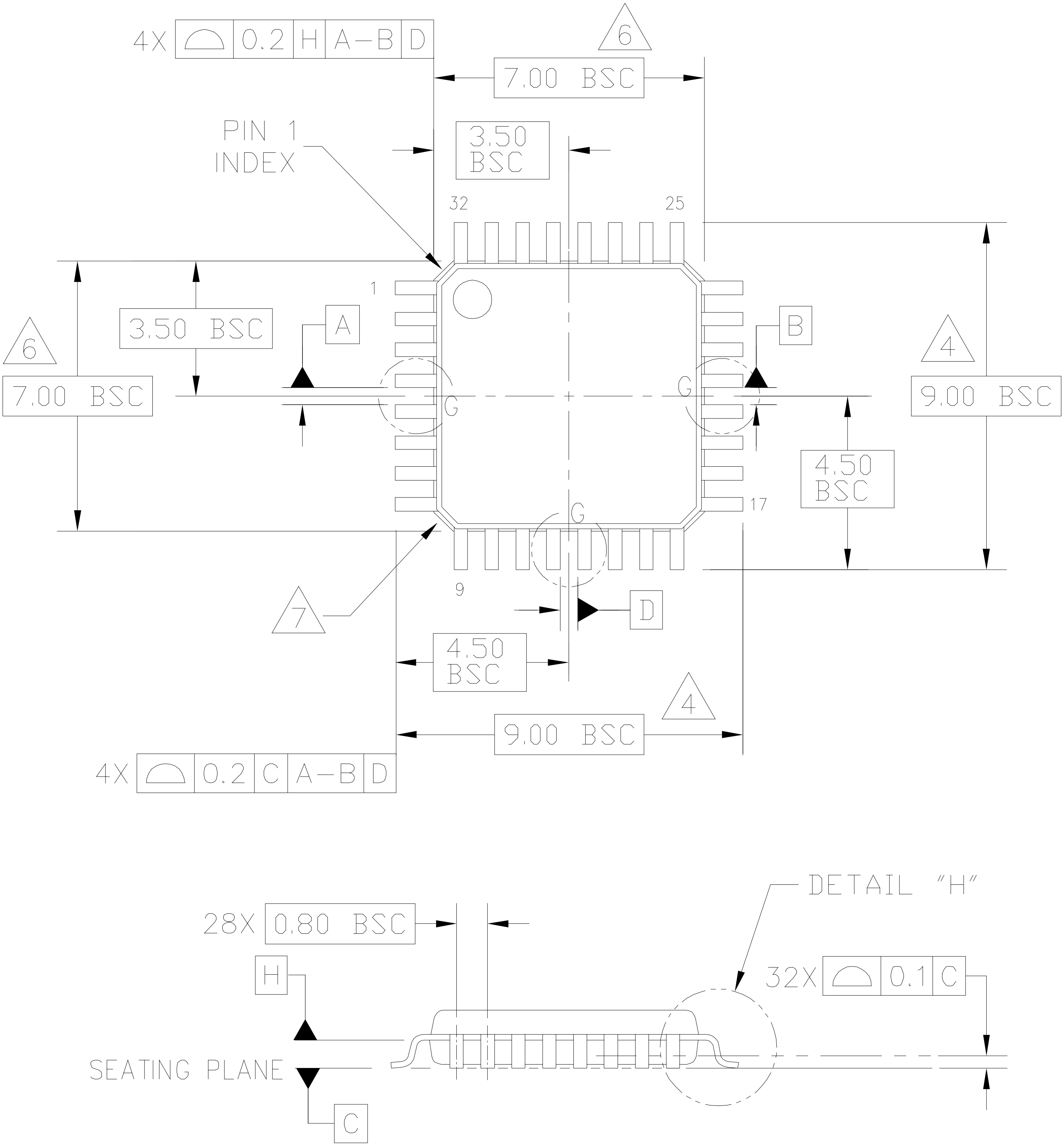


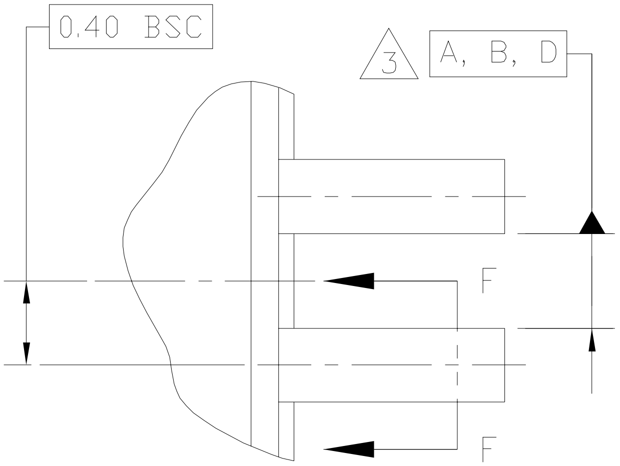
Figure 20-1. Device Numbering System

20.3 Package Dimensions

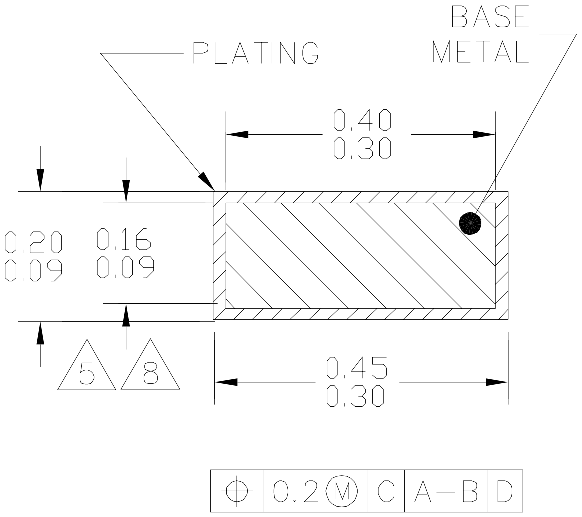
Refer to the following pages for detailed package dimensions.



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: LOW PROFILE QUAD FLAT PACK (LQFP) 32 LEAD, 0.8 PITCH (7 X 7 X 1.4)	DOCUMENT NO: 98ASH70029A CASE NUMBER: 873A-04 STANDARD: JEDEC MS-026 BBA	REV: C 01 APR 2005

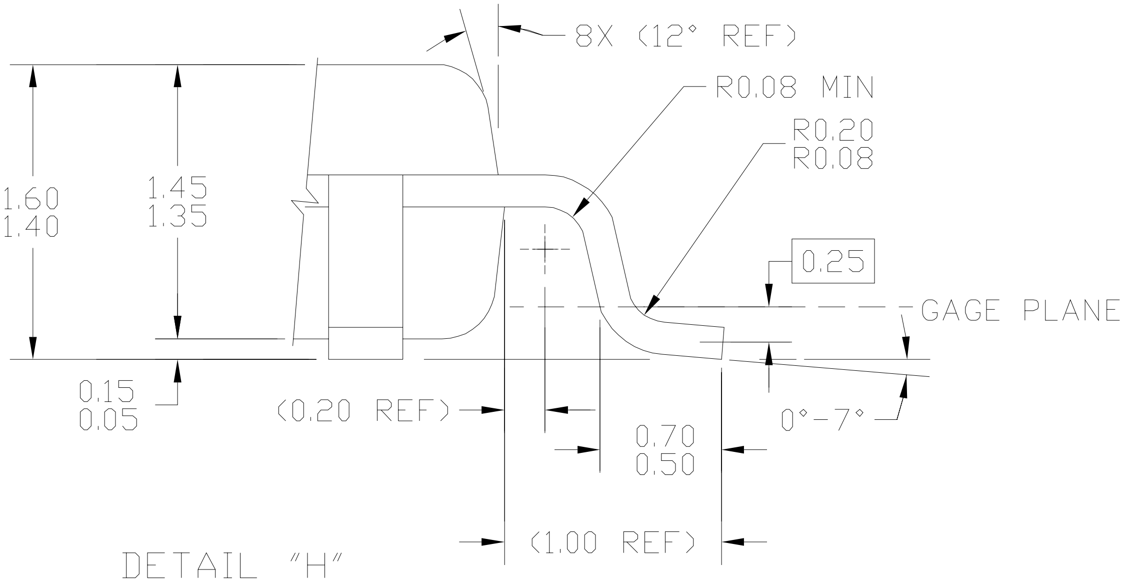


DETAIL G



SECTION F–F

ROTATED 90°CW
32 PLACES



DETAIL "H"

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: LOW PROFILE QUAD FLAT PACK (LQFP) 32 LEAD, 0.8 PITCH (7 X 7 X 1.4)	DOCUMENT NO: 98ASH70029A	REV: C
	CASE NUMBER: 873A-04	01 APR 2005
	STANDARD: JEDEC MS-026 BBA	

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5—1994.

3. DATUMS A, B, AND D TO BE DETERMINED AT DATUM PLANE H.

4. DIMENSIONS TO BE DETERMINED AT SEATING PLANE DATUM C.

5. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM DIMENSION BY MORE THAN 0.08 MM. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION: 0.07 MM.

6. DIMENSIONS DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 MM PER SIDE. DIMENSIONS ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.

7. EXACT SHAPE OF EACH CORNER IS OPTIONAL.

8. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.1 MM AND 0.25 MM FROM THE LEAD TIP.

MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE:

LOW PROFILE QUAD FLAT PACK (LQFP)
32 LEAD, 0.8 PITCH (7 X 7 X 1.4)

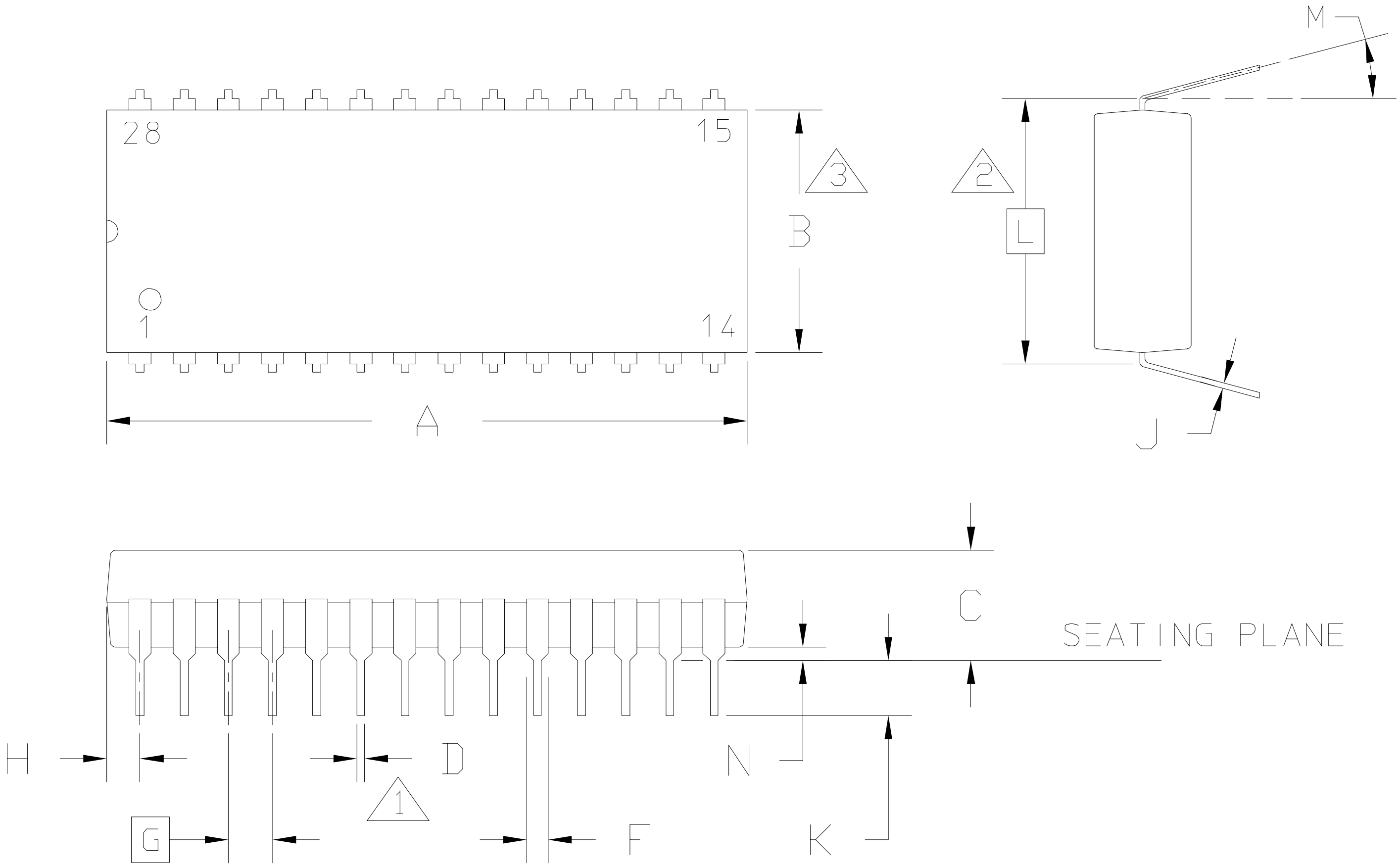
DOCUMENT NO: 98ASH70029A

REV: C

CASE NUMBER: 873A-04

01 APR 2005

STANDARD: JEDEC MS-026 BBA



MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE:

28 LD PDIP

DOCUMENT NO: 98ASB42390B

REV: C

CASE NUMBER: 710-03

05 APR 2005

STANDARD: NON-JEDEC

NOTES:

△1. POSITIONAL TOLERANCE OF LEADS, SHALL BE WITHIN 0.25 MM (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.

△2. DIMENSION TO CENTER OF LEADS WHEN FORMED PARALLEL.

△3. DIMENSION DOES NOT INCLUDE MOLD FLASH.

4. 710−02 OBSOLETE, NEW STD 710−03.

5. CONTROLLING DIMENSION: INCH

	INCH		MILLIMETER			INCH		MILLIMETER	
DIM	MIN	MAX	MIN	MAX	DIM	MIN	MAX	MIN	MAX
A	1.435	1.465	36.45	37.21					
B	0.540	0.560	13.72	14.22					
C	0.155	0.200	3.94	5.08					
D	0.014	0.022	0.36	0.56					
F	0.040	0.060	1.02	1.52					
G	0.100 BSC		2.54 BSC						
H	0.065	0.085	1.65	2.16					
J	0.008	0.015	0.20	0.38					
K	0.115	0.135	2.92	3.43					
L	0.600 BSC		15.24 BSC						
M	0°	15°	0°	15°					
N	0.020	0.040	0.51	1.02					

MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE:

28 LD PDIP

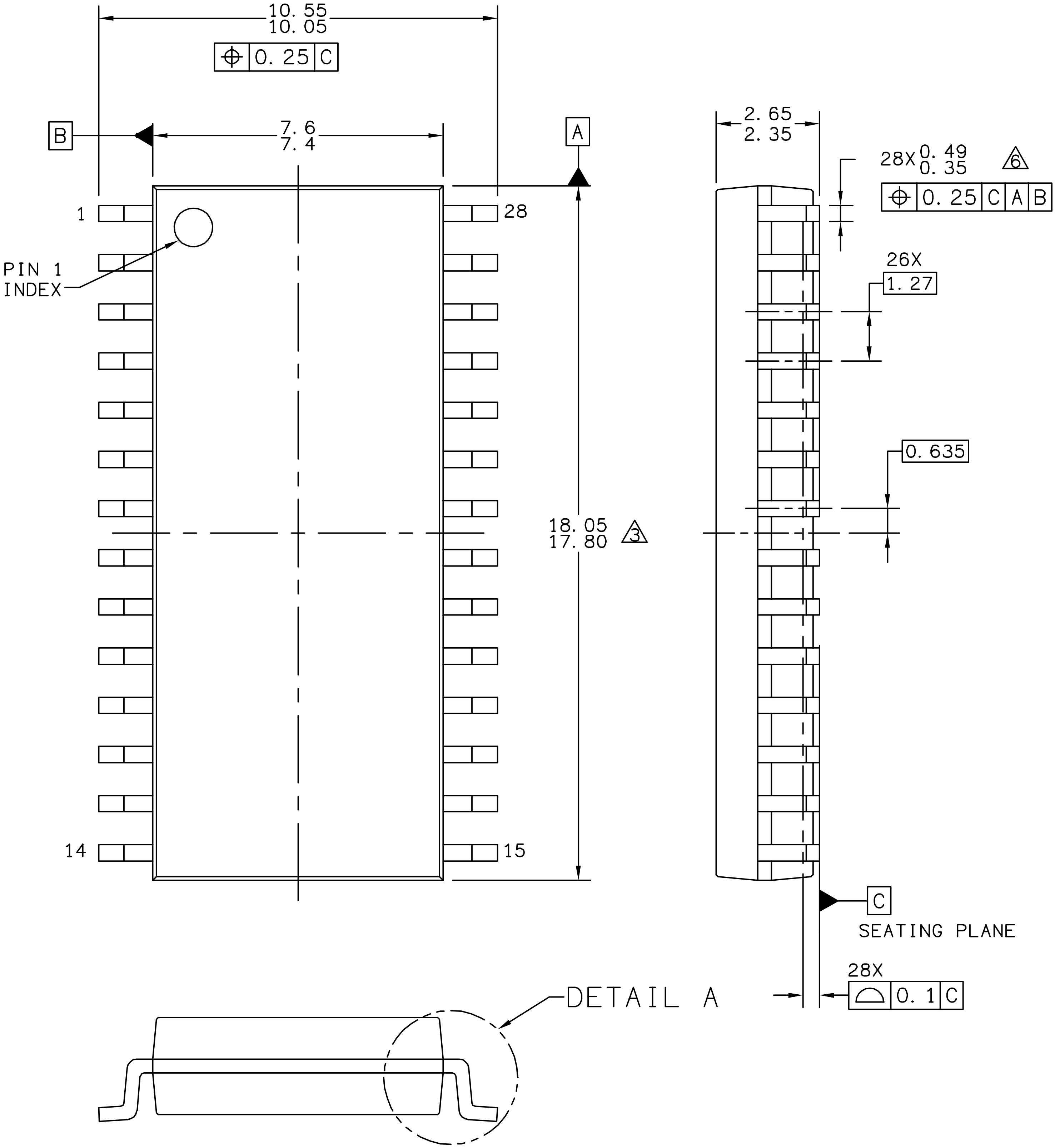
DOCUMENT NO: 98ASB42390B

REV: C

CASE NUMBER: 710-03

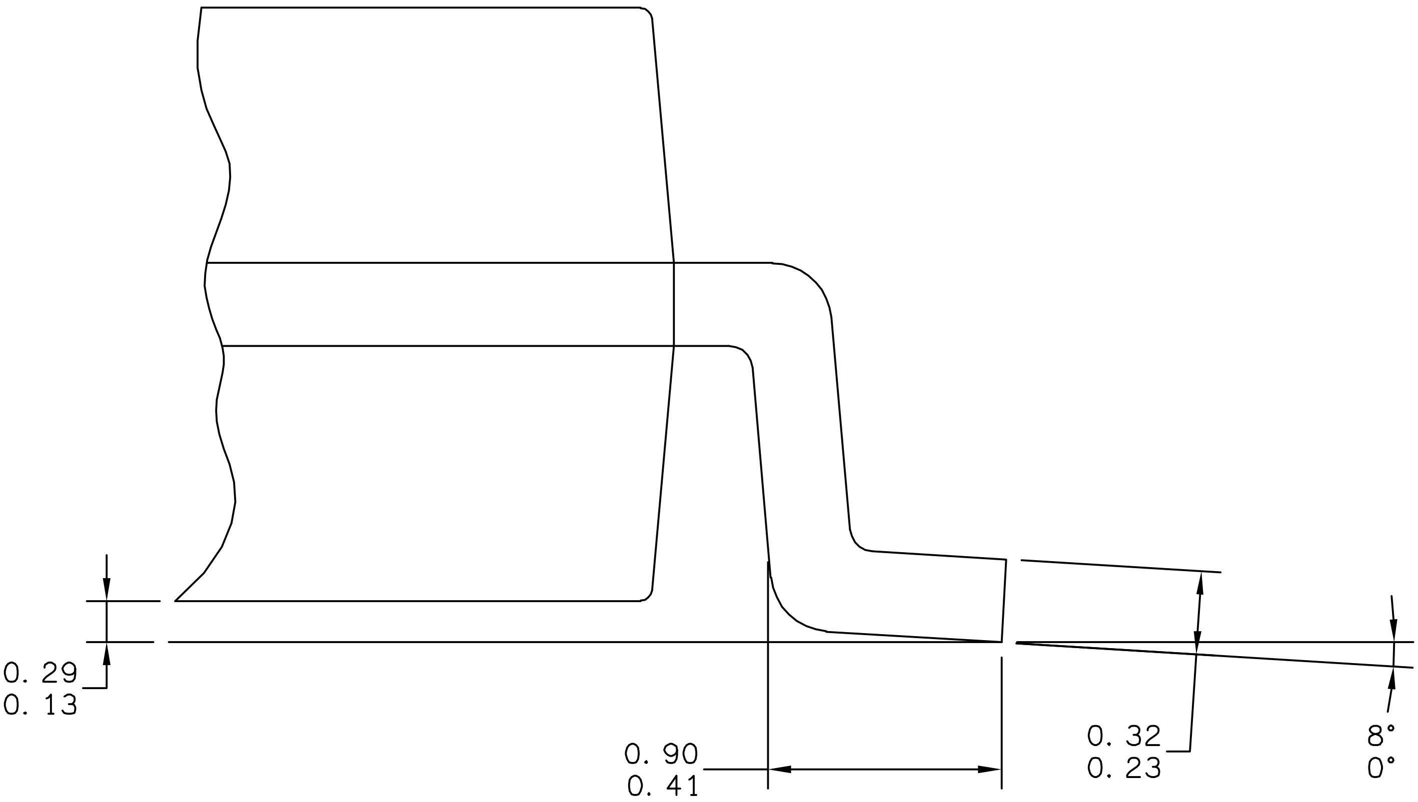
05 APR 2005

STANDARD: NON-JEDEC



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE

TITLE: SOIC, WIDE BODY, 28 LEAD CASEOUTLINE	DOCUMENT NO: 98ASB42345B	REV: G
	CASE NUMBER: 751F-05	10 MAR 2005
	STANDARD: MS-013AE	



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE

TITLE: SOIC, WIDE BODY, 28 LEAD CASEOUTLINE	DOCUMENT NO: 98ASB42345B	REV: G
	CASE NUMBER: 751F-05	10 MAR 2005
	STANDARD: MS-013AE	

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994.

△3. THIS DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. MAXIMUM MOLD PROTRUSION 0.15 PER SIDE.

4. 751F–01 THRU –04 OBSOLETE. NEW STANDARD: 751F–05

△5. THIS DIMENSION DOES NOT INCLUDE DAM BAR PROTRUSION ALLOWABLE DAM BAR PROTRUSION SHALL BE 0.13 TOTAL IN EXCESS OF THIS DIMENSION AT MAXIMUM MATERIAL CONDITION.

MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE: SOIC, WIDE BODY,
28 LEAD
CASEOUTLINE

DOCUMENT NO: 98ASB42345B

REV: G

CASE NUMBER: 751F–05

10 MAR 2005

STANDARD: MS–013AE

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to http://www.freescale.com/epp.

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

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

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.
© Freescale Semiconductor, Inc. 2005, 2006, 2007. All rights reserved.

freescale™
semiconductor

Downloaded from Elcodis.com electronic components distributor