MOTOROLA
intelligence everywhere™

digital dna™

MC68HC908GT16
MC68HC908GT8

Technical Data

M68HC08
Microcontrollers

WWW.MOTOROLA.COM/SEMICONDUCTORS

MC68HC908GT16
MC68HC908GT8
Technical Data

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://www.motorola.com/semiconductors/

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

Revision History

Date	Revision Level	Description	Page Number(s)
March, 2002	N/A	Original release	N/A
May, 2002	1	**7.3 Features** — Corrected third bulleted item to reflect ±4 percent variability	111
		Figure 15-1. Forced Monitor Mode (Low) — Reworked for clarity	211
		Figure 15-2. Forced Monitor Mode (High) — Reworked for clarity	211
		Figure 15-3. Standard Monitor Mode — Reworked for clarity	212
		Table 15-1. Monitor Mode Signal Requirements and Options — Reworked for clarity	214
		Figure 16-4. Port A I/O Circuit — Reworked to correct pullup resistor	231
		Figure 16-11. Port C I/O Circuit — Reworked to correct pullup resistor	238
		Figure 16-15. Port D I/O Circuit — Reworked to correct pullup resistor	243
June, 2002	2	**Figure 2-2. Control, Status, and Data Registers** — Corrected ESCI arbiter data register (SCIADAT) to reflect read-only status	50
		Figure 18-18. ESCI Arbiter Control Register (SCIACTL) — Corrected address location designator from $0018 to $000A	290
		Figure 18-19. ESCI Arbiter Data Register (SCIADAT) — Corrected address location designator from $0019 to $000B	292

List of Sections

Table of Contents

Section 3. Low-Power Modes

Section 4. Resets and Interrupts

Section 5. Analog-to-Digital Converter (ADC)

Section 6. Break Module (BRK)

Section 7. Internal Clock Generator (ICG) Module

Section 8. Configuration Register (CONFIG)

Section 9. Computer Operating Properly (COP) Module

Section 10. Central Processor Unit (CPU)

Section 11. FLASH Memory

Section 12. External Interrupt (IRQ)

Section 13. Keyboard Interrupt Module (KBI)

Section 14. Low-Voltage Inhibit (LVI)

Section 15. Monitor ROM (MON)

Section 16. Input/Output (I/O) Ports

Section 17. Random-Access Memory (RAM)

Section 18. Enhanced Serial Communications Interface (ESCI) Module

Section 19. System Integration Module (SIM)

Section 20. Serial Peripheral Interface Module (SPI)

Section 21. Timebase Module (TBM)

Section 22. Timer Interface Module (TIM)

Section 23. Electrical Specifications

MC68HC908GT16 • MC68HC908GT8 — Rev. 2   Technical Data

Table of Contents

Section 24. Mechanical Specifications

Section 25. Ordering Information

List of Figures

List of Figures

List of Figures

Figure	Title	Page

Figure	Title	Page

List of Figures

List of Tables

List of Tables

Section 1. General Description

1.1 Contents

1.2 Introduction

The MC68HC908GT16 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.

All references to the MC68HC908GT16 in this data book apply equally to the MC68HC908GT8, unless otherwise stated.

1.3 Features

For convenience, features have been organized to reflect:

- Standard features of the MC68HC908GT16/MC68HC908GT8

- Features of the CPU08

1.3.1 Standard Features of the MC68HC908GT16/MC68HC908GT8

- High-performance M68HC08 architecture optimized for C-compilers

- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families

- 8-MHz internal bus frequency

- Internal oscillator requiring no external components:
    - Software selectable bus frequencies
    - ±25 percent accuracy with trim capability to ±4 percent
    - Clock monitor
    - Option to allow use of external clock source or external crystal/ceramic resonator

- FLASH program memory security[1]

---

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

- 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 optional reset and selectable trip points for 3.0-V and 5.0-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)

- 16 Kbytes of on-chip 100k cycle write/erase capable FLASH memory (8 Kbytes on MC68HC908GT8)

- 512 bytes of on-chip random-access memory (RAM)

- 720 bytes of flash programming routines ROM

- Serial peripheral interface module (SPI)

- Serial communications interface module (SCI)

- Two 16-bit, 2-channel timer interface modules (TIM1 and TIM2) with selectable input capture, output compare, and pulse-width modulation (PWM) capability on each channel

- 8-channel, 8-bit successive approximation analog-to-digital converter (ADC)

- BREAK module (BRK) to allow single breakpoint setting during in-circuit debugging

- Internal pullups on $\overline{\text{IRQ}}$ and $\overline{\text{RST}}$ to reduce customer system cost

- Up to 36 general-purpose input/output (I/O) pins, including:
  - 28 shared-function I/O pins
  - Six or eight dedicated I/O pins, depending on package choice
- 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/10-mA source capability on all port pins
- Higher current 20-mA sink/source capability on PTC0–PTC4
- Timebase module 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 32-kHz crystal or internal oscillator
- User selection of having the oscillator enabled or disabled during stop mode
- 8-bit keyboard wakeup port
- Available packages:
  - 42-pin shrink dual in-line package (SDIP)
  - 44-pin quad flat pack (QFP)
- Specific features of the MC68HC908GT16 in 42-pin SDIP are:
  - Port C is only 5 bits: PTC0–PTC4
  - Port D is 8 bits: PTD0–PTD7; dual 2-channel TIM modules
- Specific features of the MC68HC908GT16 in 44-pin QFP are:
  - Port C is 7 bits: PTC0–PTC6
  - Port D is 8 bits: PTD0–PTD7; dual 2-channel TIM modules

1.3.2 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

1.4 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908GT16/MC68HC908GT8. Text in parentheses within a module block indicates the module name. Text in parentheses next to a signal indicates the module which uses the signal.

† Ports are software configurable with pullup device if input port.
‡ Higher current drive port pins
* Pin contains integrated pullup device
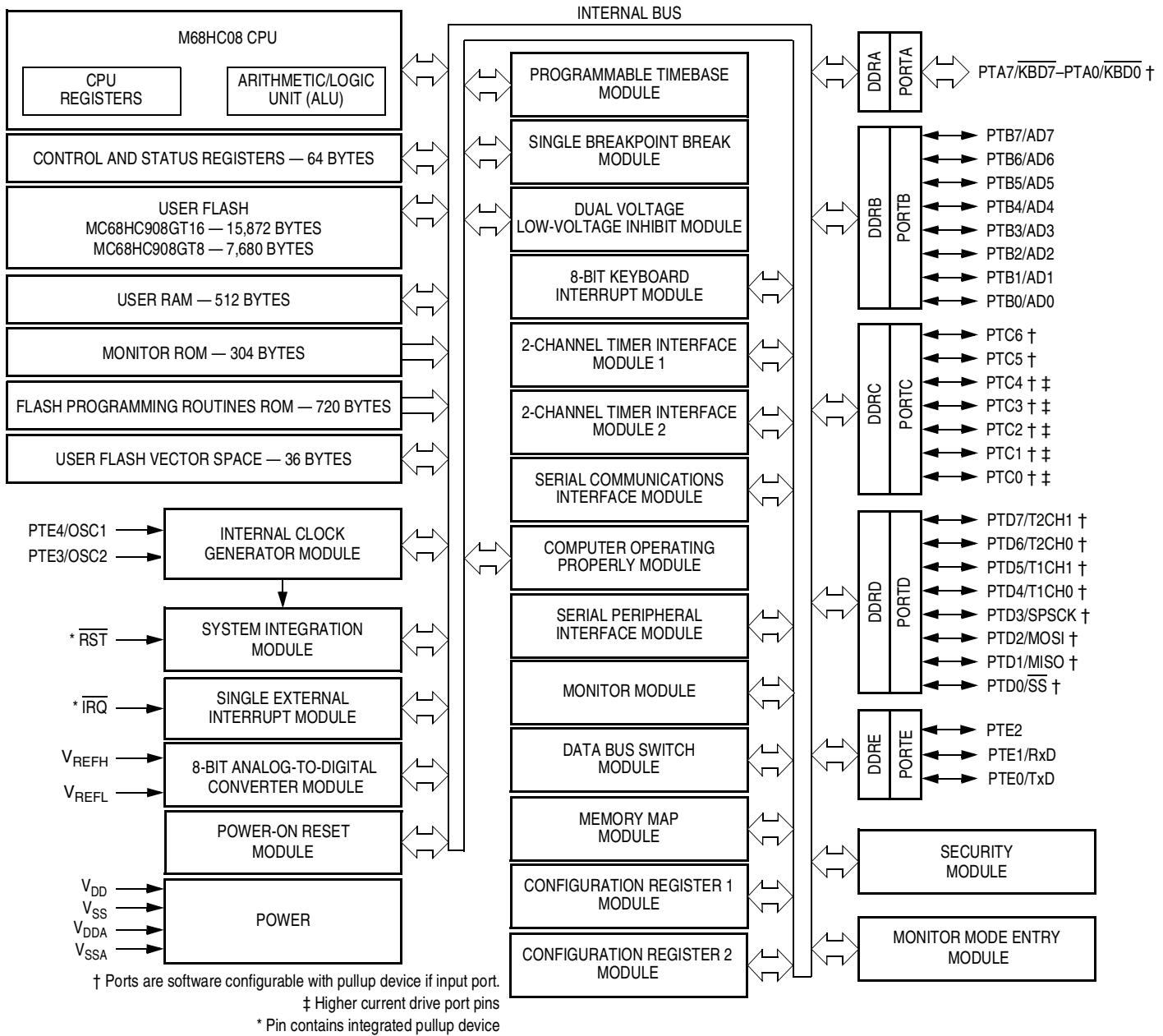
Figure 1-1. MCU Block Diagram

1.5 Pin Assignments
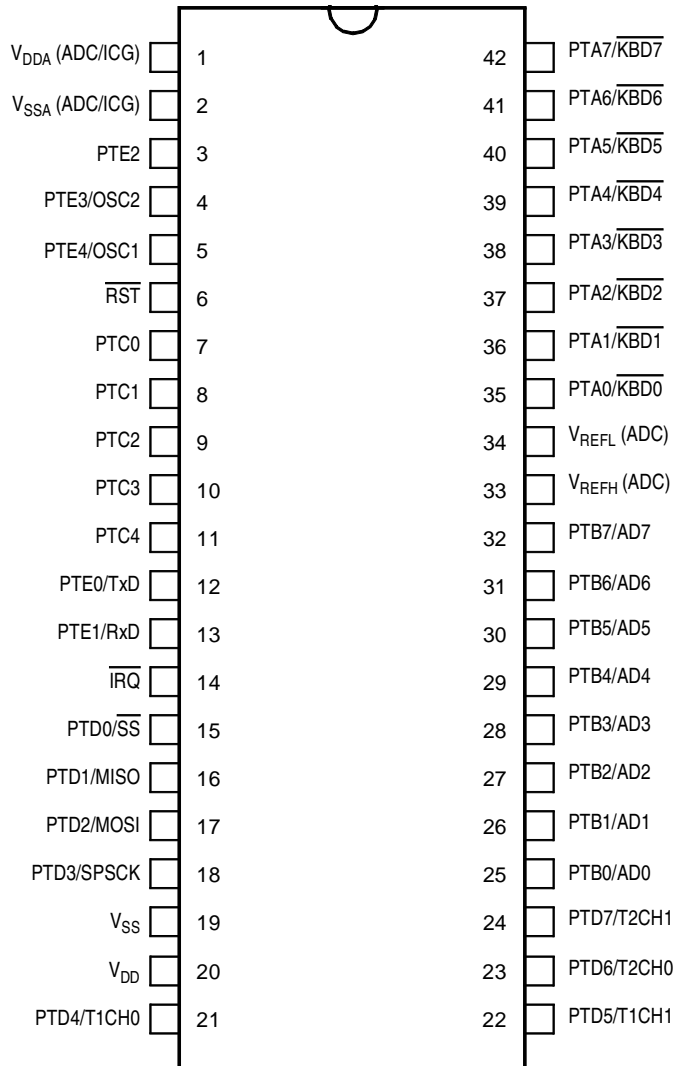
Pins Not Available on 42-Pin Package	Internal Connection
PTC5	Connected to ground
PTC6	Connected to ground

Figure 1-2. 42-Pin SDIP Pin Assignments
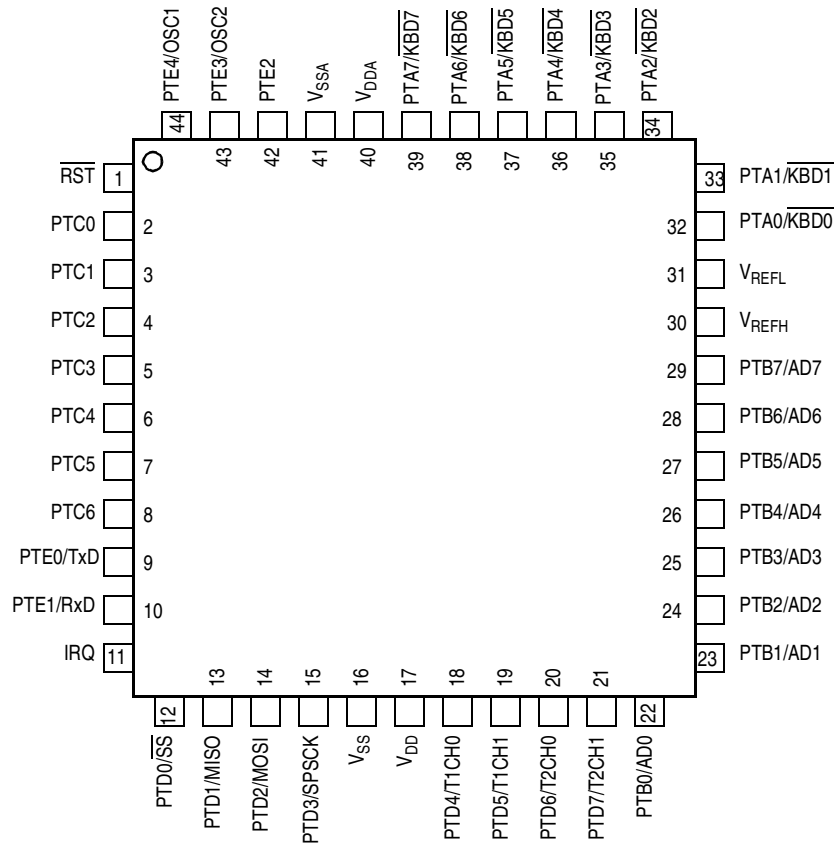
Figure 1-3. 44-Pin QFP Pin Assignments

1.6 Pin Functions

Descriptions of the pin functions are provided here.

1.6.1 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.
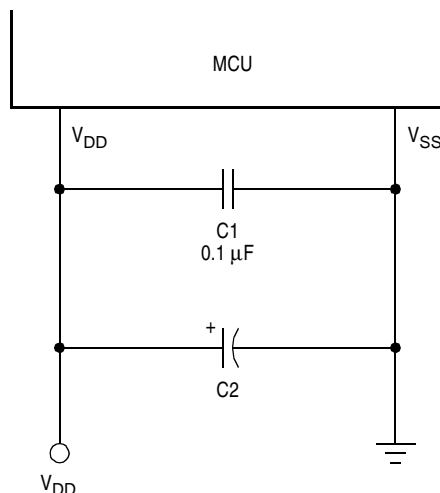
Note: Component values shown represent typical applications.

Figure 1-4. Power Supply Bypassing

1.6.2 Oscillator Pins (PTE4/OSC1 and PTE3/OSC2)

PTE4/OSC1 and PTE3/OSC2 are general-purpose, bidirectional I/O port pins. These pins can also be programmed to be the connections for an external crystal, resonator or clock circuit. See **Section 7. Internal Clock Generator (ICG) Module**.

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

A logic 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 **Section 19. System Integration Module (SIM)**.

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

$\overline{\text{IRQ}}$ is an asynchronous external interrupt pin. This pin contains an internal pullup resistor. See **Section 12. External Interrupt (IRQ)**.

1.6.5 ADC and ICG Power Supply Pins (V$_{DDA}$ and V$_{SSA}$)

V$_{DDA}$ and V$_{SSA}$ are the power supply pins for the analog-to-digital converter (ADC) and the internal clock generator (ICG). Connect the V$_{DDA}$ pin to the same voltage potential as V$_{DD}$, and the V$_{SSA}$ pin to the same voltage potential as V$_{SS}$. Decoupling of these pins should be as per the digital supply. See **Section 5. Analog-to-Digital Converter (ADC)** and **Section 7. Internal Clock Generator (ICG) Module**.

1.6.6 ADC Reference Pins (V$_{REFH}$ and V$_{REFL}$)

V$_{REFH}$ and V$_{REFL}$ are the reference voltage pins for the analog-to-digital converter (ADC). V$_{REFH}$ is the high reference supply for the ADC and should be filtered. V$_{REFH}$ can be connected to the same voltage potential as the analog supply pin, V$_{DDA}$, or to an external precision voltage reference. V$_{REFL}$ is the low reference supply for the ADC and should be externally filtered. While V$_{REFL}$ can be connected to an external reference, it is practical to connect it to the same voltage potential as the analog supply pin V$_{SSA}$. See **Section 5. Analog-to-Digital Converter (ADC)**.

1.6.7 Port A Input/Output (I/O) Pins (PTA7/$\overline{KBD7}$–PTA0/$\overline{KBD0}$)

PTA7–PTA0 are general-purpose, bidirectional I/O port pins. Any or all of the port A pins can be programmed to serve as keyboard interrupt pins. See **Section 16. Input/Output (I/O) Ports** and **Section 13. 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.

1.6.8 Port B I/O Pins (PTB7/AD7–PTB0/AD0)

PTB7–PTB0 are general-purpose, bidirectional I/O port pins that can also be used for analog-to-digital converter (ADC) inputs. See **Section 16. Input/Output (I/O) Ports** and **Section 5. Analog-to-Digital Converter (ADC)**.

1.6.9 Port C I/O Pins (PTC6–PTC0)

PTC6–PTC0 are general-purpose, bidirectional I/O port pins. PTC0–PTC4 have higher current sink/source capability. PTC5 and PTC6 are only available on the 44-pin QFP package.

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. See **Section 16. Input/Output (I/O) Ports**.

1.6.10 Port D I/O Pins (PTD7/T2CH1–PTD0/$\overline{\text{SS}}$)

PTD7–PTD0 are special-function, bidirectional I/O port pins. PTD0–PTD3 can be programmed to be serial peripheral interface (SPI) pins, while PTD4–PTD7 can be individually programmed to be timer interface module (TIM1 and TIM2) pins. See **Section 22. Timer Interface Module (TIM)**, **Section 20. Serial Peripheral Interface Module (SPI)**, and **Section 16. 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.

1.6.11 Port E I/O Pins (PTE4–PTE2, PTE1/RxD, and PTE0/TxD)

PTE0–PTE4 are general-purpose, bidirectional I/O port pins. PTE0–PTE1 can also be programmed to be serial communications interface (SCI) pins. See **Section 18. Enhanced Serial Communications Interface (ESCI) Module** and **Section 16. Input/Output (I/O) Ports**.

PTE3 and PTE4 can also be programmed to be clock or oscillator pins. See **Section 8. Configuration Register (CONFIG)** and **Section 16. 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 MC68HC908GT16 • MC68HC908GT8 do not require termination, termination is recommended to reduce the possibility of static damage.*

Section 2. Memory Map

2.1 Contents

2.2 Introduction

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

- User FLASH memory:

    – MC68HC908GT16 — 15,872 bytes

    – MC68HC908GT8 — 7,680 bytes

- 512 bytes of random-access memory (RAM)

- 720 bytes of FLASH programming routines read-only memory (ROM)

- 36 bytes of user-defined vectors

- 304 bytes of monitor ROM

2.3 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.

2.4 Reserved Memory Locations

Accessing a reserved location can have unpredictable effects on 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.

2.5 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; SIM break status register, SBSR
- $FE01; SIM reset status register, SRSR
- $FE02; reserved, SUBAR
- $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
- $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
- $FF80; ICG user trim register 5V ICGTR5
- $FF81; ICG user trim register 3V ICGTR3
- $FFFF; COP control register, COPCTL

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

Address	Contents
$0000 ↓ $003F	I/O REGISTERS 64 BYTES
$0040 ↓ $023F	RAM 512 BYTES
$0240 ↓ $1B4F	UNIMPLEMENTED 6416 BYTES
$1B50 ↓ $1E1F	FLASH PROGRAMMING ROUTINES ROM 720 BYTES
$1E20 ↓ $BFFF	UNIMPLEMENTED 41,440 BYTES
$C000 ↓ $FDFF	FLASH MEMORY MC68HC908GT16 15,872 BYTES

Right column:

UNIMPLEMENTED[1]	$C000 ↓ $DFFF
FLASH MEMORY MC68HC908GT8 7,680 BYTES	$E000 ↓ $FDFF

1. Inadvertent access to these locations will not cause an illegal address reset.

Address	Register
$FE00	SIM BREAK STATUS REGISTER (SBSR)
$FE01	SIM RESET STATUS REGISTER (SRSR)
$FE02	RESERVED (SUBAR)
$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
$FE08	FLASH CONTROL REGISTER (FLCR)
$FE09	BREAK ADDRESS REGISTER HIGH (BRKH)

Figure 2-1. Memory Map

MC68HC908GT16 • MC68HC908GT8 — Rev. 2   Technical Data

$FE0A	BREAK ADDRESS REGISTER LOW (BRKL)
$FE0B	BREAK STATUS AND CONTROL REGISTER (BRKSCR)
$FE0C	LVI STATUS REGISTER (LVISR)
$FE0D ↓ $FE0F	UNIMPLEMENTED 3 BYTES
$FE10 ↓ $FE1F	UNIMPLEMENTED 16 BYTES RESERVED FOR COMPATIBILITY WITH MONITOR CODE FOR A-FAMILY PART
$FE20 ↓ $FF4F	MONITOR ROM 304 BYTES
$FF50 ↓ $FF7D	UNIMPLEMENTED 46 BYTES
$FF7E	FLASH BLOCK PROTECT REGISTER (FLBPR)
$FF7F	UNIMPLEMENTED 1 BYTE
$FF80	ICG USER TRIM REGISTER 5V (ICGTR5)
$FF81	ICG USER TRIM REGISTER 3V (ICGTR3)
$FF82 ↓ $FFDB	UNIMPLEMENTED 90 BYTES
$FFDC ↓ $FFFF[2]	FLASH VECTORS 36 BYTES

2. $FFF6–$FFFD reserved for eight 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 229.	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:				Unaffected by reset				
$0001	Port B Data Register (PTB) See page 233.	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:				Unaffected by reset				
$0002	Port C Data Register (PTC) See page 236.	Read: Write:	0	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
		Reset:				Unaffected by reset				
$0003	Port D Data Register (PTD) See page 240.	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:				Unaffected by reset				
$0004	Data Direction Register A (DDRA) See page 230.	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB) See page 234.	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC) See page 237.	Read: Write:	0	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD) See page 242.	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE) See page 245.	Read: Write:	0	0	0	PTE4	PTE3	PTE2	PTE1	PTE0
		Reset:				Unaffected by reset				
$0009	ESCI Prescaler Register (SCPSC) See page 286.	Read: Write:	PDS2	PDS1	PDS0	PSSB4	PSSB3	PSSB2	PSSB1	PSSB0
		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 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000A	ESCI Arbiter Control Register (SCIACTL) See page 290.	Read:	AM1	ALOST	AM0	ACLK	AFIN	ARUN	AOVFL	ARD8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000B	ESCI Arbiter Data Register (SCIADAT) See page 292.	Read:	ARD7	ARD6	ARD5	ARD4	ARD3	ARD2	ARD1	ARD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000C	Data Direction Register E (DDRE) See page 246.	Read:	0	0	0	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000D	Port A Input Pullup Enable Register (PTAPUE) See page 232.	Read:	PTAPUE7	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000E	Port C Input Pullup Enable Register (PTCPUE) See page 239.	Read:	0	PTCPUE6	PTCPUE5	PTCPUE4	PTCPUE3	PTCPUE2	PTCPUE1	PTCPUE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000F	Port D Input Pullup Enable Register (PTDPUE) See page 244.	Read:	PTDPUE7	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0010	SPI Control Register (SPCR) See page 345.	Read:	SPRIE	DMAS	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 347.	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 350.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:				Unaffected by reset				
$0013	ESCI Control Register 1 (SCC1) See page 272.	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		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 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0014	ESCI Control Register 2 (SCC2) See page 275.	Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0
$0015	ESCI Control Register 3 (SCC3) See page 278.	Read: Write:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
		Reset:	U	U	0	0	0	0	0	0
$0016	ESCI Status Register 1 (SCS1) See page 279.	Read: Write:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Reset:	1	1	0	0	0	0	0	0
$0017	ESCI Status Register 2 (SCS2) See page 283.	Read: Write:							BKF	RPF
		Reset:	0	0	0	0	0	0	0	0
$0018	ESCI Data Register (SCDR) See page 284.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$0019	ESCI Baud Rate Register (SCBR) See page 284.	Read: Write:			SCP1	SCP0	R	SCR2	SCR1	SCR0
		Reset:	0	0	0	0	0	0	0	0
$001A	Keyboard Status and Control Register (INTKBSCR) See page 201.	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 202.	Read: Write:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
		Reset:	0	0	0	0	0	0	0	0
$001C	Timebase Module Control Register (TBCR) See page 353.	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 193.	Read:	0	0	0	0	IRQF1	0	IMASK1	MODE1
		Write:						ACK1		
		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 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001E	Configuration Register 2 (CONFIG2)† See page 148.	Read:	R	0	EXT-XTALEN	EXT-SLOW	EXT-CLKEN	0	OSCEN-INSTOP	R
		Write:	R		EXT-XTALEN	EXT-SLOW	EXT-CLKEN		OSCEN-INSTOP	R
		Reset:	0	0	0	0	0	0	0	0
$001F	Configuration Register 1 (CONFIG1)† See page 148.	Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVI5OR3†	SSREC	STOP	COPD
		Write:								
		Reset:	0	0	0	0	0	0	0	0

†One-time writable register after each reset, except LVI5OR3 bit. LVI5OR3 bit is only reset via POR (power-on reset).

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	Timer 1 Status and Control Register (T1SC) See page 371.	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 374.	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 374.	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 375.	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 375.	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 376.	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 380.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:				Indeterminate after reset				

= Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0027	Timer 1 Channel 0 Register Low (T1CH0L) See page 380.	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 376.	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 380.	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 380.	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 371.	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 374.	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 374.	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 375.	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 375.	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 376.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0

☐ = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 5 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0031	Timer 2 Channel 0 Register High (T2CH0H) See page 380.	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 380.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:				Indeterminate after reset				
$0033	Timer 2 Channel 1 Status and Control Register (T2SC1) See page 376.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0034	Timer 2 Channel 1 Register High (T2CH1H) See page 380.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:				Indeterminate after reset				
$0035	Timer 2 Channel 1 Register Low (T2CH1L) See page 380.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:				Indeterminate after reset				
$0036	ICG Control Register (ICGCR) See page 141.	Read:	CMIE	CMF	CMON	CS	ICGON	ICGS	ECGON	ECGS
		Write:		0						
		Reset:	0	0	0	0	1	0	0	0
$0037	ICG Multiplier Register (ICGMR) See page 143.	Read:		N6	$\overline{N5}$	N4	N3	N2	N1	N0
		Write:								
		Reset:	0	0	0	1	0	1	0	1
$0038	ICG Trim Register (ICGTR) See page 144.	Read:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
		Write:								
		Reset:	1	0	0	0	0	0	0	0
$0039	ICG Divider Control Register (ICGDVR) See page 144.	Read:					DDIV3	DDIV2	DDIV1	DDIV0
		Write:								
		Reset:	0	0	0	0	U	U	U	U
$003A	ICG DCO Stage Control Register (ICGDSR) See page 145.	Read:	DSTG7	DSTG6	DSTG5	DSTG4	DSTG3	DSTG2	DSTG1	DSTG0
		Write:	R	R	R	R	R	R	R	R
		Reset:	U	U	U	U	U	U	U	U

　　= Unimplemented　　R = Reserved　　U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 6 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003B	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:	Indeterminate after reset							
$003C	ADC Status and Control Register (ADSCR) See page 95.	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$003D	ADC Data Register (ADR) See page 98.	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003E	ADC Clock Register (ADCLK) See page 98.	Read:	ADIV2	ADIV1	ADIV0	ADICLK	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003F	Unimplemented	Read:								
		Write:								
		Reset:								
$FE00	SIM Break Status Register (SBSR) See page 107.	Read:	R	R	R	R	R	R	SBSW	R
		Write:							NOTE	
		Reset:	0	0	0	0	0	0	0	0

Note: Writing a logic 0 clears SBSW.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE01	SIM Reset Status Register (SRSR) See page 76.	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
$FE02	SIM Upper Byte Address Register (SUBAR)	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE03	SIM Break Flag Control Register (SBFCR) See page 108.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							

  = Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 7 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE04	Interrupt Status Register 1 (INT1) See page 86.	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 86.	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 87.	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 179.	Read:	0	0	0	0	HVEN	MASS	ERASE	PGM
		Write:					HVEN	MASS	ERASE	PGM
		Reset:	0	0	0	0	0	0	0	0
$FE09	Break Address Register High (BRKH) See page 106.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Break Address Register Low (BRKL) See page 106.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$FE0B	Break Status and Control Register (BRKSCR) See page 105.	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:	BRKE	BRKA						
		Reset:	0	0	0	0	0	0	0	0
$FE0C	LVI Status Register (LVISR) See page 207.	Read:	LVIOUT	0	0	0	0	0	0	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 8 of 9)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FF7E	FLASH Block Protect Register (FLBPR)[†] See page 185.	Read: Write:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
		Reset:	U	U	U	U	U	U	U	U
$FF80	ICG User Trim Register 5V (ICGTR5)[†] See page 187.	Read: Write:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
		Reset:	U	U	U	U	U	U	U	U
$FF81	ICG User Trim Register 3V (ICGTR3)[†] See page 187.	Read: Write:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
		Reset:	U	U	U	U	U	U	U	U
$FFFF	COP Control Register (COPCTL) See page 157.	Read:	Low byte of reset vector							
		Write:	Writing clears COP counter (any value)							
		Reset:	Unaffected by reset							

[†]Non-volatile FLASH register

= Unimplemented   R = Reserved   U = Unaffected

Figure 2-2. Control, Status, and Data Registers (Sheet 9 of 9)

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	TIM2 Channel 1 Vector (High)
		$FFEF	TIM2 Channel 1 Vector (Low)
	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	ICG Vector (High)
		$FFF9	ICG Vector (Low)
	IF1	$FFFA	$\overline{\text{IRQ}}$ Vector (High)
		$FFFB	$\overline{\text{IRQ}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
	—	$FFFE	Reset Vector (High)
Highest		$FFFF	Reset Vector (Low)

Section 3. Low-Power Modes

3.1 Contents

3.2 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.

3.2.1 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 LVI module and/or the timebase module through bits in the
CONFIG1 register. (See **Section 8. Configuration Register
(CONFIG)**.)

3.2.2 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 OSCENINSTOP bit in the CONFIG2 register is at a logic 0. (See **Section 8. Configuration Register (CONFIG)**.)

3.3 Analog-to-Digital Converter (ADC)

3.3.1 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.

3.3.2 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.

3.4 Break Module (BRK)

3.4.1 Wait Mode

If enabled, 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.

3.4.2 Stop Mode

The break module is inactive in stop mode. A break interrupt causes exit from stop mode and sets the SBSW bit in the break status register. The STOP instruction does not affect break module register states.

3.5 Central Processor Unit (CPU)

3.5.1 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

3.5.2 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.

3.6 Internal Clock Generator Module (ICG)

3.6.1 Wait Mode

The internal clock generator (ICG) module remains active in wait mode. If enabled, the ICG interrupt to the CPU can bring the MCU out of wait mode.

In some applications, low power-consumption is desired in wait mode and a high-frequency clock is not needed. In these applications, reduce power consumption by either selecting a low-frequency external clock and turn the internal clock generator off or reduce the bus frequency by minimizing the ICG multiplier factor (N) before executing the WAIT instruction.

3.6.2 Stop Mode

The value of the oscillator enable in stop (OSCENINSTOP) bit in the CONFIG2 register determines the behavior of the ICG in stop mode. If OSCENINSTOP is low, the ICG is disabled in stop and, upon execution of the STOP instruction, all ICG activity will cease and the output clocks (CGMXCLK, CGMOUT, COPCLK, and TBMCLK) will be held low. Power consumption will be minimal.

If OSCENINSTOP is high, the ICG is enabled in stop and activity will continue. This is useful if the timebase module (TBM) is required to bring the MCU out of stop mode. ICG interrupts will not bring the MCU out of stop mode in this case.

During stop mode, if OSCENINSTOP is low, several functions in the ICG are affected. The stable bits (ECGS and ICGS) are cleared, which will enable the external clock stabilization divider upon recovery. The clock monitor is disabled (CMON = 0) which will also clear the clock monitor interrupt enable (CMIE) and clock monitor flag (CMF) bits. The CS, ICGON, ECGON, N, TRIM, DDIV, and DSTG bits are unaffected.

3.7 Computer Operating Properly Module (COP)

3.7.1 Wait Mode

The computer operating properly (COP) module remains active in wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter in a CPU interrupt routine.

3.7.2 Stop Mode

Stop mode turns off the COPCLK 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.

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.

3.8 External Interrupt Module (IRQ)

3.8.1 Wait Mode

The external interrupt (IRQ) module remains active in wait mode. Clearing the IMASK1 bit in the IRQ status and control register enables $\overline{IRQ}$ CPU interrupt requests to bring the MCU out of wait mode.

3.8.2 Stop Mode

The IRQ module remains active in stop mode. Clearing the IMASK1 bit in the IRQ status and control register enables $\overline{IRQ}$ CPU interrupt requests to bring the MCU out of stop mode.

3.9 Keyboard Interrupt Module (KBI)

3.9.1 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.

3.9.2 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.

3.10 Low-Voltage Inhibit Module (LVI)

3.10.1 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.

3.10.2 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.

3.11 Enhanced Serial Communications Interface Module (SCI)

3.11.1 Wait Mode

The enhanced serial communications interface (ESCI), or SCI module for short, 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.

3.11.2 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.

3.12 Serial Peripheral Interface Module (SPI)

3.12.1 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.

3.12.2 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.

3.13 Timer Interface Module (TIM1 and TIM2)

3.13.1 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.

3.13.2 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.

3.14 Timebase Module (TBM)

3.14.1 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.

3.14.2 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 OSCENINSTOP 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.

3.15 Exiting Wait Mode

These events restart the CPU clock and load the program counter with the reset vector or with an interrupt vector:

- External reset — A logic 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 ($\overline{\text{IRQ}}$ pin) loads the program counter with the contents of locations: $FFFA and $FFFB; $\overline{\text{IRQ}}$ pin.

- Break interrupt — A break interrupt loads the program counter with the contents of $FFFC and $FFFD.

- Computer operating properly module (COP) reset — A timeout of the COP counter resets the MCU and loads the program counter with the contents of $FFFE and $FFFF.

- Low-voltage inhibit module (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.

- Internal Clock Generator module (ICG) interrupt — A CPU interrupt request from the ICG loads the program counter with the contents of $FFF8 and $FFF9.

- Keyboard module (KBI) interrupt — A CPU interrupt request from the KBI module loads the program counter with the contents of $FFE0 and $FFE1.

- Timer 1 interface module (TIM1) interrupt — A CPU interrupt request from the TIM1 loads the program counter with the contents of:
  - $FFF2 and $FFF3; TIM1 overflow
  - $FFF4 and $FFF5; TIM1 channel 1
  - $FFF6 and $FFF7; TIM1 channel 0

- Timer 2 interface module (TIM2) interrupt — A CPU interrupt request from the TIM2 loads the program counter with the contents of:
  - $FFEC and $FFED; TIM2 overflow
  - $FFEE and $FFEF; TIM2 channel 1
  - $FFF0 and $FFF1; TIM2 channel 0

- Serial peripheral interface module (SPI) interrupt — A CPU interrupt request from the SPI loads the program counter with the contents of:
  - $FFE8 and $FFE9; SPI transmitter
  - $FFEA and $FFEB; SPI receiver

- Serial communications interface module (SCI) interrupt — A CPU interrupt request from the SCI loads the program counter with the contents of:
  - $FFE2 and $FFE3; SCI transmitter
  - $FFE4 and $FFE5; SCI receiver
  - $FFE6 and $FFE7; SCI receiver error

- Analog-to-digital converter module (ADC) interrupt — A CPU interrupt request from the ADC loads the program counter with the contents of: $FFDE and $FFDF; ADC conversion complete.

- Timebase module (TBM) interrupt — A CPU interrupt request from the TBM loads the program counter with the contents of: $FFDC and $FFDD; TBM interrupt.

3.16 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 logic 0 on the $\overline{RST}$ 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; $\overline{IRQ}$ pin
  – $FFE0 and $FFE1; keyboard interrupt pins

- Low-voltage inhibit (LVI) reset — A power supply voltage below the LVI_{TRIPF} voltage resets the MCU and loads the program counter with the contents of locations $FFFE and $FFFF.

- Break interrupt — A break interrupt loads the program counter with the contents of locations $FFFC and $FFFD.

- 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.*

Section 4. Resets and Interrupts

4.1 Contents

4.2 Introduction

Resets and interrupts are responses to exceptional events during program execution. A reset re-initializes the MCU to its startup condition. An interrupt vectors the program counter to a service routine.

4.3 Resets

A reset immediately returns the MCU to a known startup condition and begins program execution from a user-defined memory location.

4.3.1 Effects

A reset:

- Immediately stops the operation of the instruction being executed

- Initializes certain control and status bits

- Loads the program counter with a user-defined reset vector address from locations $FFFE and $FFFF

- Selects CGMXCLK divided by four as the bus clock

4.3.2 External Reset

A logic 0 applied to the $\overline{RST}$ pin for a time, t_{IRL}, generates an external reset. An external reset sets the PIN bit in the SIM reset status register.

4.3.3 Internal Reset

Sources:

- Power-on reset (POR)

- Computer operating properly (COP)

- Low-power reset circuits

- Illegal opcode

- Illegal address

All internal reset sources pull the $\overline{\text{RST}}$ pin low for 32 CGMXCLK cycles to allow resetting of external devices. The MCU is held in reset for an additional 32 CGMXCLK cycles after releasing the $\overline{\text{RST}}$ pin.
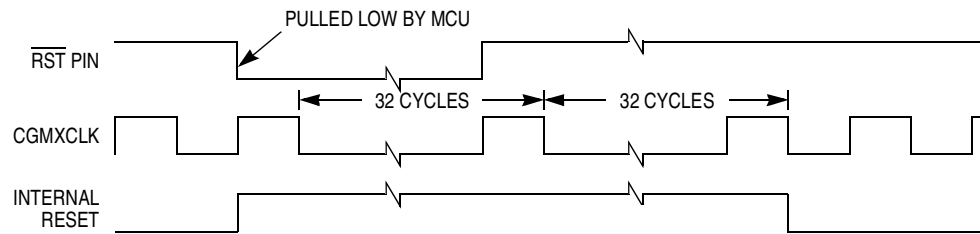
Figure 4-1. Internal Reset Timing

4.3.3.1 Power-On Reset (POR)

A power-on reset (POR) is an internal reset caused by a positive transition on the V_{DD} pin. V_{DD} at the POR must go completely to 0 V to reset the MCU. This distinguishes between a reset and a POR. The POR is not a brown-out detector, low-voltage detector, or glitch detector.

A power-on reset:

- Holds the clocks to the CPU and modules inactive for an oscillator stabilization delay of 4096 CGMXCLK cycles

- Drives the $\overline{\text{RST}}$ pin low during the oscillator stabilization delay

- Releases the RST pin 32 CGMXCLK cycles after the oscillator stabilization delay

- Releases the CPU to begin the reset vector sequence 64 CGMXCLK cycles after the oscillator stabilization delay

- Sets the POR and LP bits in the SIM reset status register and clears all other bits in the register

4.3.3.2 Computer Operating Properly (COP) Reset

A COP reset is an internal reset caused by an overflow of the COP counter. A COP reset sets the COP bit in the system integration module (SIM) reset status register.

---
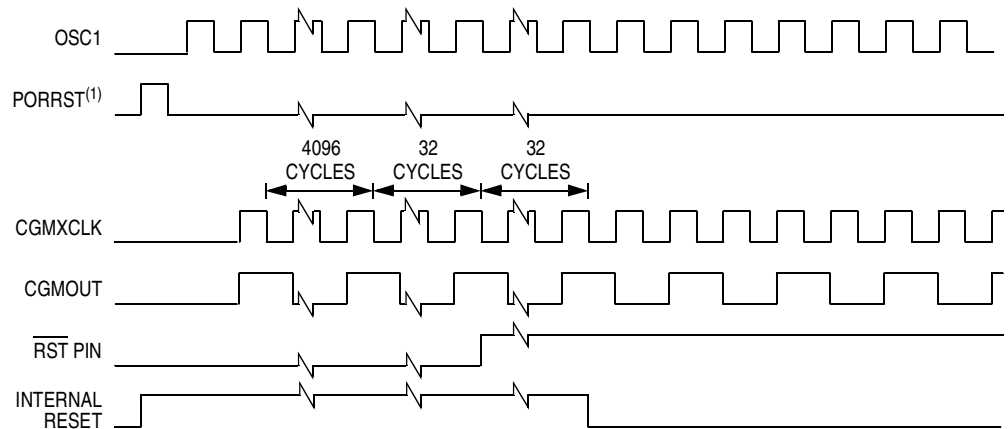
1. PORRST is an internally generated power-on reset pulse.

Figure 4-2. Power-On Reset Recovery

To clear the COP counter and prevent a COP reset, write any value to the COP control register at location $FFFF.

4.3.3.3 Low-Voltage Inhibit Reset

A low-voltage inhibit (LVI) reset is an internal reset caused by a drop in the power supply voltage to the LVI_{TRIPF} voltage.

An LVI reset:

- Holds the clocks to the CPU and modules inactive for an oscillator stabilization delay of 4096 CGMXCLK cycles after the power supply voltage rises to the LVI_{TRIPR} voltage

- Drives the $\overline{RST}$ pin low for as long as V_{DD} is below the LVI_{TRIPR} voltage and during the oscillator stabilization delay

- Releases the $\overline{RST}$ pin 32 CGMXCLK cycles after the oscillator stabilization delay

- Releases the CPU to begin the reset vector sequence 64 CGMXCLK cycles after the oscillator stabilization delay

- Sets the LVI bit in the SIM reset status register

4.3.3.4 Illegal Opcode Reset

An illegal opcode reset is an internal reset caused by an opcode that is not in the instruction set. An illegal opcode reset sets the ILOP bit in the SIM reset status register.

If the stop enable bit, STOP, in the mask option register is a logic 0, the STOP instruction causes an illegal opcode reset.

4.3.3.5 Illegal Address Reset

An illegal address reset is an internal reset caused by opcode fetch from an unmapped address. An illegal address reset sets the ILAD bit in the SIM reset status register.

A data fetch from an unmapped address does not generate a reset.

4.3.4 SIM Reset Status Register

This read-only register contains flags to show reset sources. All flag bits are automatically cleared following a read of the register. Reset service can read the SIM reset status register to clear the register after power-on reset and to determine the source of any subsequent reset.

The register is initialized on power-up as shown with the POR bit set and all other bits cleared. During a POR or any other internal reset, the $\overline{RST}$ pin is pulled low. After the pin is released, it will be sampled 32 CGMXCLK cycles later. If the pin is not above a V_{IH} at that time, then the PIN bit in the SRSR may be set in addition to whatever other bits are set.

NOTE: *Only a read of the SIM reset status register clears all reset flags. After multiple resets from different sources without reading the register, multiple flags remain set.*

Address: $FE01

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
Write:								
POR:	1	0	0	0	0	0	0	0

= Unimplemented

Figure 4-3. SIM Reset Status Register (SRSR)

POR — Power-On Reset Flag
   1 = Power-on reset since last read of SRSR
   0 = Read of SRSR since last power-on reset

PIN — External Reset Flag
   1 = External reset via $\overline{\text{RST}}$ pin since last read of SRSR
   0 = POR or read of SRSR since last external reset

COP — Computer Operating Properly Reset Bit
   1 = Last reset caused by timeout of COP counter
   0 = POR or read of SRSR since any reset

ILOP — Illegal Opcode Reset Bit
   1 = Last reset caused by an illegal opcode
   0 = POR or read of SRSR since any reset

ILAD — Illegal Address Reset Bit
   1 = Last reset caused by an opcode fetch from an illegal address
   0 = POR or read of SRSR since any reset

MODRST — Monitor Mode Entry Module Reset Bit
   1 = Last reset caused by forced monitor mode entry.
   0 = POR or read of SRSR since any reset

LVI — Low-Voltage Inhibit Reset Bit
   1 = Last reset caused by low-power supply voltage
   0 = POR or read of SRSR since any reset

4.4 Interrupts

An interrupt temporarily changes the sequence of program execution to respond to a particular event. An interrupt does not stop the operation of the instruction being executed, but begins when the current instruction completes its operation.

4.4.1 Effects

An interrupt:

- Saves the CPU registers on the stack. At the end of the interrupt, the RTI instruction recovers the CPU registers from the stack so that normal processing can resume.

- Sets the interrupt mask (I bit) to prevent additional interrupts. Once an interrupt is latched, no other interrupt can take precedence, regardless of its priority.

- Loads the program counter with a user-defined vector address

*High byte of index register is not stacked.
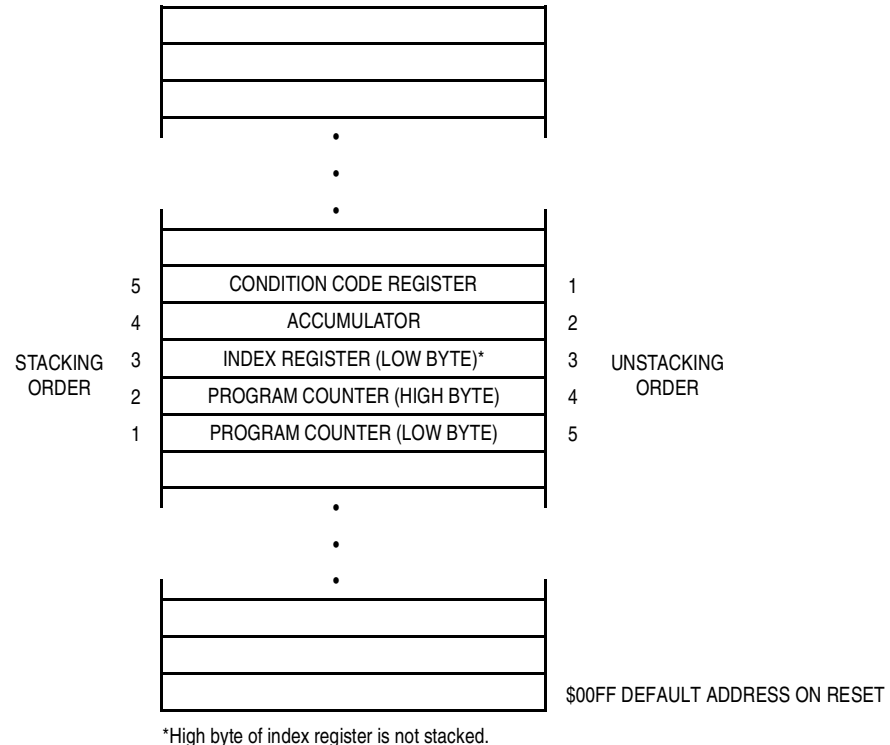
Figure 4-4. Interrupt Stacking Order

After every instruction, the CPU checks all pending interrupts if the I bit is not set. If more than one interrupt is pending when an instruction is done, the highest priority interrupt is serviced first. In the example shown in **Figure 4-5**, if an interrupt is pending upon exit from the interrupt service routine, the pending interrupt is serviced before the LDA instruction is executed.

```
CLI ───────────────────────────────────┐
                                        ├─ BACKGROUND
     ──► LDA   #$FF ─────────────────────┘  ROUTINE

  ──► INT1   PSHH ──────────────┐
                ↓               ├─ INT1 INTERRUPT SERVICE ROUTINE
             PULH               │
             RTI ───────────────┘

  ──► INT2   PSHH ──────────────┐
                ↓               ├─ INT2 INTERRUPT SERVICE ROUTINE
             PULH               │
             RTI ───────────────┘
```
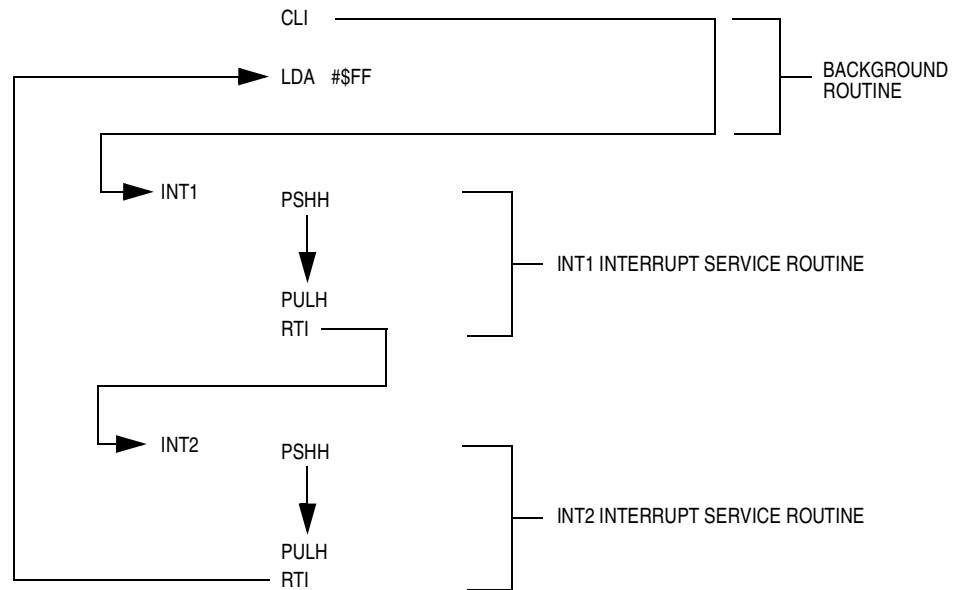
Figure 4-5. **Interrupt Recognition Example**

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, save the H register and then restore it prior to exiting the routine.*

4.4.2 Sources

The sources in **Table 4-1** can generate CPU interrupt requests.
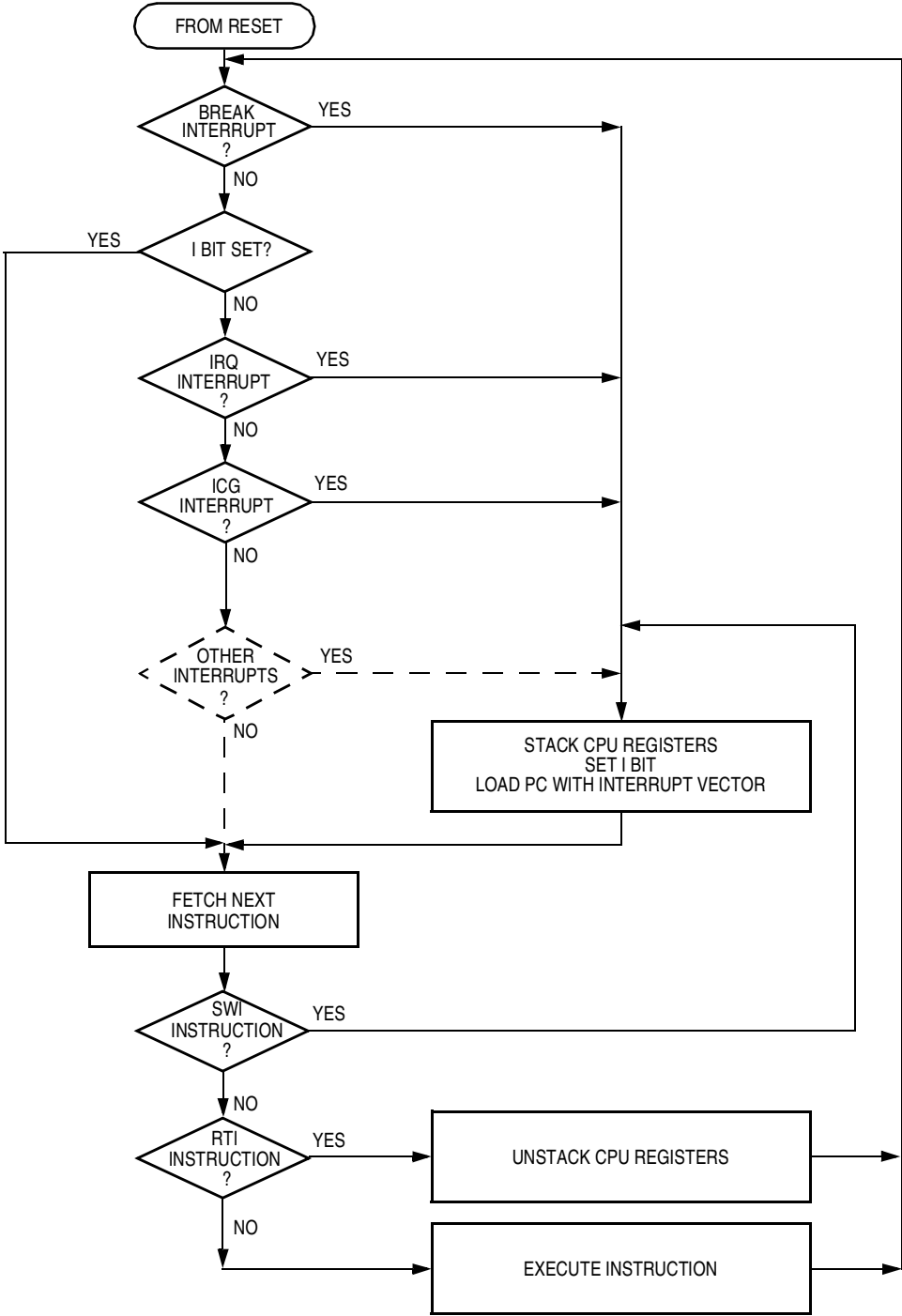
Figure 4-6. Interrupt Processing

Table 4-1. Interrupt Sources

Source	Flag	Mask[1]	INT Register Flag	Priority[2]	Vector Address
Reset	None	None	None	0	$FFFE–$FFFF
SWI instruction	None	None	None	0	$FFFC–$FFFD
IRQ pin	IRQF	IMASK1	IF1	1	$FFFA–$FFFB
ICG clock monitor	CMF	CMIE	IF2	2	$FFF8–$FFF9
TIM1 channel 0	CH0F	CH0IE	IF3	3	$FFF6–$FFF7
TIM1 channel 1	CH1F	CH1IE	IF4	4	$FFF4–$FFF5
TIM1 overflow	TOF	TOIE	IF5	5	$FFF2–$FFF3
TIM2 channel 0	CH0F	CH0IE	IF6	6	$FFF0–$FFF1
TIM2 channel 1	CH1F	CH1IE	IF7	7	$FFEE–$FFEF
TIM2 overflow	TOF	TOIE	IF8	8	$FFEC–$FFED
SPI receiver full	SPRF	SPRIE	IF9	9	$FFEA–$FFEB
SPI overflow	OVRF	ERRIE			
SPI mode fault	MODF	ERRIE			
SPI transmitter empty	SPTE	SPTIE	IF10	10	$FFE8–$FFE9
SCI receiver overrun	OR	ORIE	IF11	11	$FFE6–$FFE7
SCI noise fag	NF	NEIE			
SCI framing error	FE	FEIE			
SCI parity error	PE	PEIE			
SCI receiver full	SCRF	SCRIE	IF12	12	$FFE4–$FFE5
SCI input idle	IDLE	ILIE			
SCI transmitter empty	SCTE	SCTIE	IF13	13	$FFE2–$FFE3
SCI transmission complete	TC	TCIE			
Keyboard pin	KEYF	IMASKK	IF14	14	$FFE0–$FFE1
ADC conversion complete	COCO	AIEN	IF15	15	$FFDE–$FFDF
Timebase	TBIF	TBIE	IF16	16	$FFDC–$FFDD

1. The I bit in the condition code register is a global mask for all interrupt sources except the SWI instruction.
2. 0 = highest priority

4.4.2.1 Software Interrupt (SWI) Instruction

The software interrupt instruction (SWI) causes a non-maskable interrupt.

NOTE: *A software interrupt pushes PC onto the stack. An SWI does **not** push PC – 1, as a hardware interrupt does.*

4.4.2.2 Break Interrupt

The break module causes the CPU to execute an SWI instruction at a software-programmable break point.

4.4.2.3 $\overline{IRQ}$ Pin

A logic 0 on the $\overline{IRQ1}$ pin latches an external interrupt request.

4.4.2.4 Internal Clock Generator (ICG)

The ICG can generate a CPU interrupt request every time the selected internal or external clock becomes inactive. When the clock monitor CMON bit is set and the currently selected clock becomes inactive, the clock monitor interrupt flag CMF is set. The clock monitor interrupt enable bit (CMIE) enables ICG CPU interrupt requests. CMIE, CMF, and CMON are in the ICGCR control register.

4.4.2.5 Timer Interface Module 1 (TIM1)

TIM1 CPU interrupt sources:

- TIM1 overflow flag (TOF) — The TOF bit is set when the TIM1 counter value rolls over to $0000 after matching the value in the TIM1 counter modulo registers. The TIM1 overflow interrupt enable bit, TOIE, enables TIM1 overflow CPU interrupt requests. TOF and TOIE are in the TIM1 status and control register.

- TIM1 channel flags (CH1F–CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. The channel x interrupt enable bit, CHxIE, enables channel x TIM1 CPU interrupt requests. CHxF and CHxIE are in the TIM1 channel x status and control register.

4.4.2.6 Timer Interface Module 2 (TIM2)

TIM2 CPU interrupt sources:

- TIM2 overflow flag (TOF) — The TOF bit is set when the TIM2 counter value rolls over to $0000 after matching the value in the TIM2 counter modulo registers. The TIM2 overflow interrupt enable bit, TOIE, enables TIM2 overflow CPU interrupt requests. TOF and TOIE are in the TIM2 status and control register.

- TIM2 channel flags (CH1F–CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. The channel x interrupt enable bit, CHxIE, enables channel x TIM2 CPU interrupt requests. CHxF and CHxIE are in the TIM2 channel x status and control register.

4.4.2.7 Serial Peripheral Interface (SPI)

SPI CPU interrupt sources:

- SPI receiver full bit (SPRF) — The SPRF bit is set every time a byte transfers from the shift register to the receive data register. The SPI receiver interrupt enable bit, SPRIE, enables SPRF CPU interrupt requests. SPRF is in the SPI status and control register and SPRIE is in the SPI control register.

- SPI transmitter empty (SPTE) — The SPTE bit is set every time a byte transfers from the transmit data register to the shift register. The SPI transmit interrupt enable bit, SPTIE, enables SPTE CPU interrupt requests. SPTE is in the SPI status and control register and SPTIE is in the SPI control register.

- Mode fault bit (MODF) — The MODF bit is set in a slave SPI if the $\overline{SS}$ pin goes high during a transmission with the mode fault enable bit (MODFEN) set. In a master SPI, the MODF bit is set if the $\overline{SS}$ pin goes low at any time with the MODFEN bit set. The error interrupt enable bit, ERRIE, enables MODF CPU interrupt requests. MODF, MODFEN, and ERRIE are in the SPI status and control register.

- Overflow bit (OVRF) — The OVRF bit is set if software does not read the byte in the receive data register before the next full byte enters the shift register. The error interrupt enable bit, ERRIE, enables OVRF CPU interrupt requests. OVRF and ERRIE are in the SPI status and control register.

4.4.2.8 Serial Communications Interface (SCI)

SCI CPU interrupt sources:

- SCI transmitter empty bit (SCTE) — SCTE is set when the SCI data register transfers a character to the transmit shift register. The SCI transmit interrupt enable bit, SCTIE, enables transmitter CPU interrupt requests. SCTE is in SCI status register 1. SCTIE is in SCI control register 2.

- Transmission complete bit (TC) — TC is set when the transmit shift register and the SCI data register are empty and no break or idle character has been generated. The transmission complete interrupt enable bit, TCIE, enables transmitter CPU interrupt requests. TC is in SCI status register 1. TCIE is in SCI control register 2.

- SCI receiver full bit (SCRF) — SCRF is set when the receive shift register transfers a character to the SCI data register. The SCI receive interrupt enable bit, SCRIE, enables receiver CPU interrupts. SCRF is in SCI status register 1. SCRIE is in SCI control register 2.

- Idle input bit (IDLE) — IDLE is set when 10 or 11 consecutive logic 1s shift in from the RxD pin. The idle line interrupt enable bit, ILIE, enables IDLE CPU interrupt requests. IDLE is in SCI status register 1. ILIE is in SCI control register 2.

- Receiver overrun bit (OR) — OR is set when the receive shift register shifts in a new character before the previous character was read from the SCI data register. The overrun interrupt enable bit, ORIE, enables OR to generate SCI error CPU interrupt requests. OR is in SCI status register 1. ORIE is in SCI control register 3.

- Noise flag (NF) — NF 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, enables NF to generate SCI error CPU interrupt requests. NF is in SCI status register 1. NEIE is in SCI control register 3.

- Framing error bit (FE) — FE is set when a logic 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, enables FE to generate SCI error CPU interrupt requests. FE is in SCI status register 1. FEIE is in SCI control register 3.

- Parity error bit (PE) — PE is set when the SCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, enables PE to generate SCI error CPU interrupt requests. PE is in SCI status register 1. PEIE is in SCI control register 3.

4.4.2.9 $\overline{KBD0}$–$\overline{KBD7}$ Pins

A logic 0 on a keyboard interrupt pin latches an external interrupt request.

4.4.2.10 Analog-to-Digital Converter (ADC)

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

4.4.2.11 Timebase Module (TBM)

The timebase module can interrupt the CPU on a regular basis with a rate defined by 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.

Interrupts must be acknowledged by writing a logic 1 to the TACK bit.

Technical Data   MC68HC908GT16 • MC68HC908GT8 — Rev. 2

84   Resets and Interrupts   MOTOROLA

4.4.3 Interrupt Status Registers

The flags in the interrupt status registers identify maskable interrupt sources. **Table 4-2** summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

Table 4-2. Interrupt Source Flags

Interrupt Source	Interrupt Status Register Flag
Reset	—
SWI instruction	—
$\overline{\text{IRQ}}$ pin	IF1
ICG clock monitor	IF2
TIM1 channel 0	IF3
TIM1 channel 1	IF4
TIM1 overflow	IF5
TIM2 channel 0	IF6
TIM2 channel 1	IF7
TIM2 overflow	IF8
SPI receive	IF9
SPI transmit	IF10
SCI error	IF11
SCI receive	IF12
SCI transmit	IF13
Keyboard	IF14
ADC conversion complete	IF15
Timebase	IF16

4.4.3.1 Interrupt Status Register 1

Address: $FE04

	Bit 7	6	5	4	3	2	1	Bit 0
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

R	= Reserved

Figure 4-7. Interrupt Status Register 1 (INT1)

IF6–IF1 — Interrupt Flags 6–1

These flags indicate the presence of interrupt requests from the sources shown in **Table 4-2**.
  1 = Interrupt request present
  0 = No interrupt request present

Bit 1 and Bit 0 — Always read 0

4.4.3.2 Interrupt Status Register 2

Address: $FE05

	Bit 7	6	5	4	3	2	1	Bit 0
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

R	= Reserved

Figure 4-8. Interrupt Status Register 2 (INT2)

IF14–IF7 — Interrupt Flags 14–7

These flags indicate the presence of interrupt requests from the sources shown in **Table 4-2**.
  1 = Interrupt request present
  0 = No interrupt request present

4.4.3.3 Interrupt Status Register 3

Address: $FE06

	Bit 7	6	5	4	3	2	1	Bit 0
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

R	= Reserved

Figure 4-9. Interrupt Status Register 3 (INT3)

IF16–IF15 — Interrupt Flags 16–15

 This flag indicates the presence of an interrupt request from the source shown in **Table 4-2**.
  1 = Interrupt request present
  0 = No interrupt request present

Bits 7–2 — Always read 0

Section 5. Analog-to-Digital Converter (ADC)

5.1 Contents

5.2 Introduction

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

5.3 Features

Features of the ADC module include:

- Eight 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

5.4 Functional Description

The ADC provides eight pins for sampling external sources at pins PTB7/AD7–PTB0/AD0. An analog multiplexer allows the single ADC converter to select one of eight 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 5-1**.
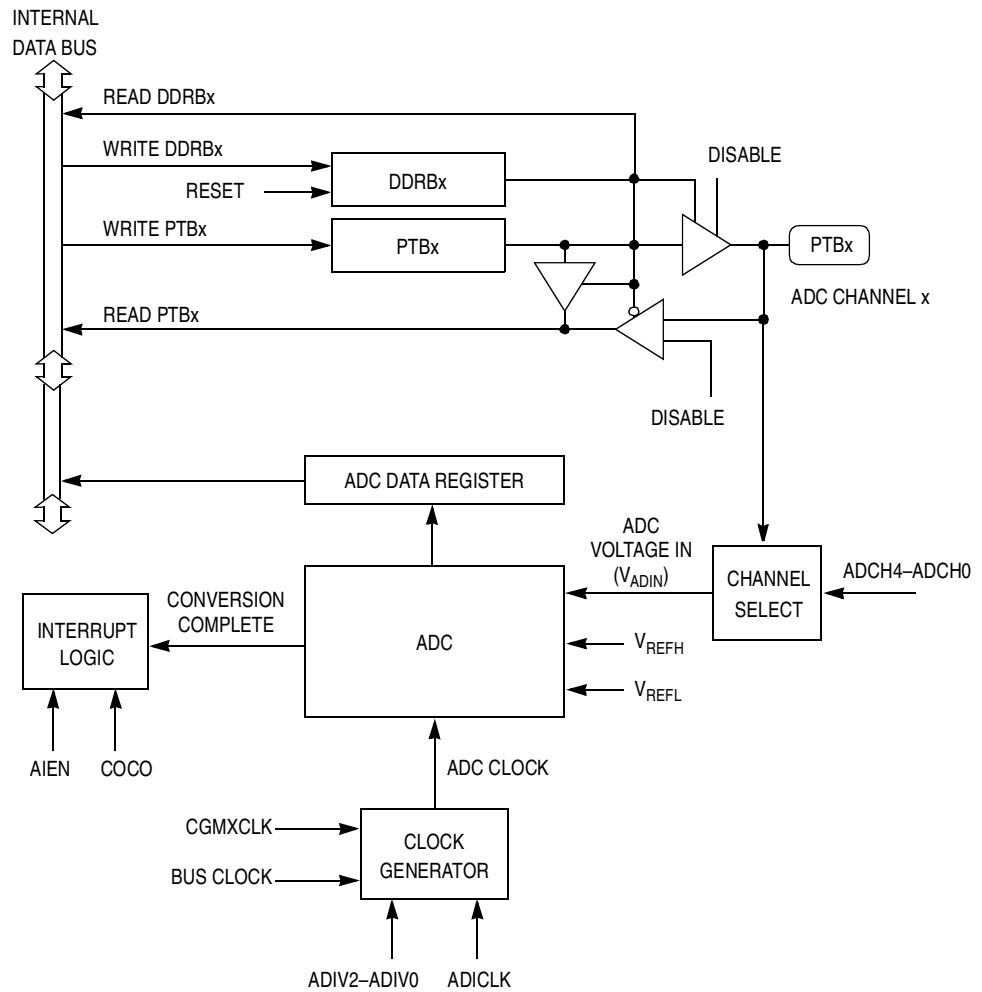
Figure 5-1. ADC Block Diagram

5.4.1 ADC Port I/O Pins

PTB7/AD7–PTB0/AD0 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 logic 0.

5.4.2 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}.*

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.

5.4.3 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.

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

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

5.4.4 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 write of the ADC status and control register or 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.

5.4.5 Accuracy and Precision

The conversion process is monotonic and has no missing codes.

5.5 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 logic 0. The COCO bit is not used as a conversion complete flag when interrupts are enabled.

5.6 Low-Power Modes

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

5.6.1 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.

5.6.2 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.

5.7 I/O Signals

The ADC module has eight pins shared with port B, PTB7/AD7–PTB0/AD0.

5.7.1 ADC Analog Power Pin (V$_{DDA}$)

The ADC analog portion uses V$_{DDA}$ as its power pin. Connect the V$_{DDA}$ pin to the same voltage potential as V$_{DD}$. External filtering may be necessary to ensure clean V$_{DDA}$ for good results.

NOTE: *For maximum noise immunity, route V$_{DDA}$ carefully and place bypass capacitors as close as possible to the package.*

5.7.2 ADC Analog Ground Pin (V$_{SSA}$)

The ADC analog portion uses V$_{SSA}$ as its ground pin. Connect the V$_{SSA}$ pin to the same voltage potential as V$_{SS}$.

NOTE: *Route V$_{SSA}$ cleanly to avoid any offset errors.*

5.7.3 ADC Voltage Reference High Pin (V$_{REFH}$)

The ADC analog portion uses V$_{REFH}$ as its upper voltage reference pin. The V$_{REFH}$ pin can be connected to an external precision reference or to the same voltage potential as V$_{DDA}$. External filtering is often necessary to ensure a clean V$_{REFH}$ for good results. Any noise present on this pin will be reflected and possibly magnified in A/D conversion values.

NOTE: *For maximum noise immunity, route 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.*

5.7.4 ADC Voltage Reference Low Pin (V$_{REFL}$)

The ADC analog portion uses V$_{REFL}$ as its lower voltage reference pin. The V$_{REFL}$ pin should be connected to the same voltage potential as V$_{SSA}$. External filtering is often necessary to ensure a clean V$_{REFL}$ for good results. Any noise present on this pin will be reflected and possibly magnified in A/D conversion values.

NOTE: *For maximum noise immunity, route V$_{REFL}$ carefully and, if not connected to V$_{SS}$, 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.

5.7.5 ADC Voltage In (V_{ADIN})

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

5.8 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)

5.8.1 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: Write:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
Reset:	0	0	0	1	1	1	1	1

Figure 5-2. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete

When the AIEN bit is a logic 0, the COCO is a read-only bit which is set each time a conversion is completed except in the continuous conversion mode where it is set after the first conversion. This bit is cleared whenever the ADSCR is written or whenever the ADR is read.

If the AIEN bit is a logic 1, the COCO becomes a read/write bit, which should be cleared to logic 0 for CPU to service the ADC interrupt request. Reset clears this bit.

1 = Conversion completed (AIEN = 0)
0 = Conversion not completed (AIEN = 0)/CPU interrupt (AIEN = 1)

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 eight channels, AD7–AD0, are available on this MCU. The channels are detailed in **Table 5-1**. 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 5-1**.

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 5-1**, are used to verify the operation of the ADC converter both in production test and for user applications.

Table 5-1. Mux Channel Select[1]

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	Input Select
0	0	0	0	0	PTB0/AD0
0	0	0	0	1	PTB1/KBD1
0	0	0	1	0	PTB2/KBD2
0	0	0	1	1	PTB3/KBD3
0	0	1	0	0	PTB4/KBD4
0	0	1	0	1	PTB5/KBD5
0	0	1	1	0	PTB6/KBD6
0	0	1	1	1	PTB7/AD7
0	1	0	0	0	
↓	↓	↓	↓	↓	Reserved
1	1	1	0	0	
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.

5.8.2 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
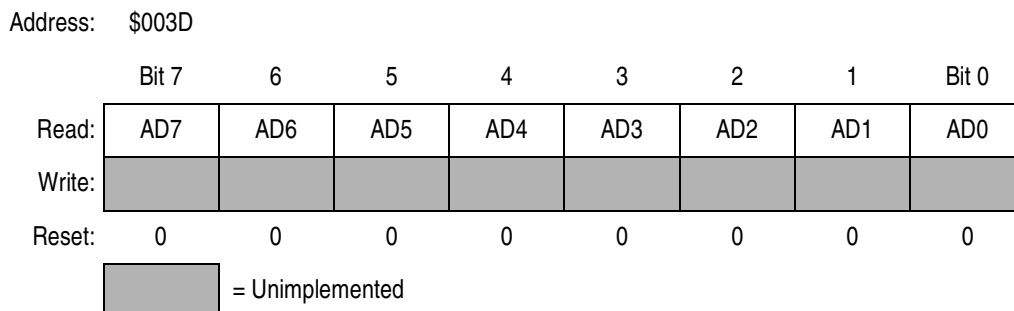
Figure 5-3. ADC Data Register (ADR)

5.8.3 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
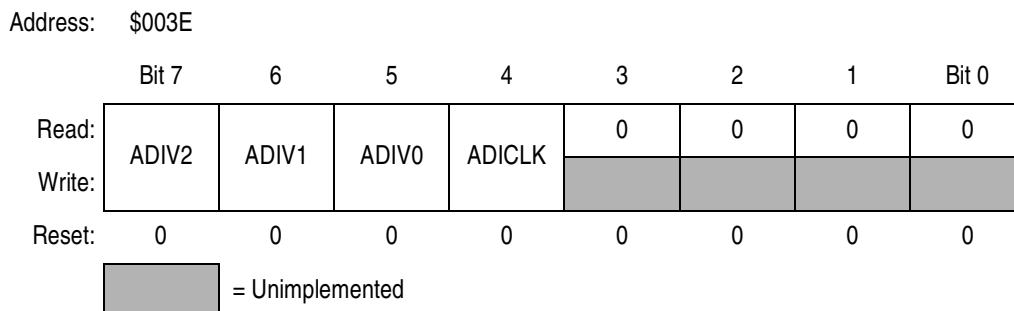
Figure 5-4. ADC Clock Register (ADCLK)

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 5-2** shows the available clock configurations. The ADC clock should be set to approximately 1 MHz.

Table 5-2. 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 outputclock (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 **23.16 ADC Characteristics**.

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

Section 6. Break Module (BRK)

6.1 Contents

6.2 Introduction

This section describes the 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.

6.3 Features

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

6.4 Functional Description

When the internal address bus matches the value written in the break address registers, the break module issues a breakpoint signal to the CPU. The CPU then loads the instruction register with a software interrupt instruction (SWI) after completion of the current CPU instruction. 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 logic 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 begins after the CPU completes its current instruction. A return-from-interrupt instruction (RTI) in the break routine ends the break interrupt and returns the microcontroller unit (MCU) to normal operation. **Figure 6-1** shows the structure of the break module.

IAB15–IAB8

IAB15–IAB0

BREAK ADDRESS REGISTER HIGH

8-BIT COMPARATOR

CONTROL → BREAK

8-BIT COMPARATOR

BREAK ADDRESS REGISTER LOW

IAB7–IAB0
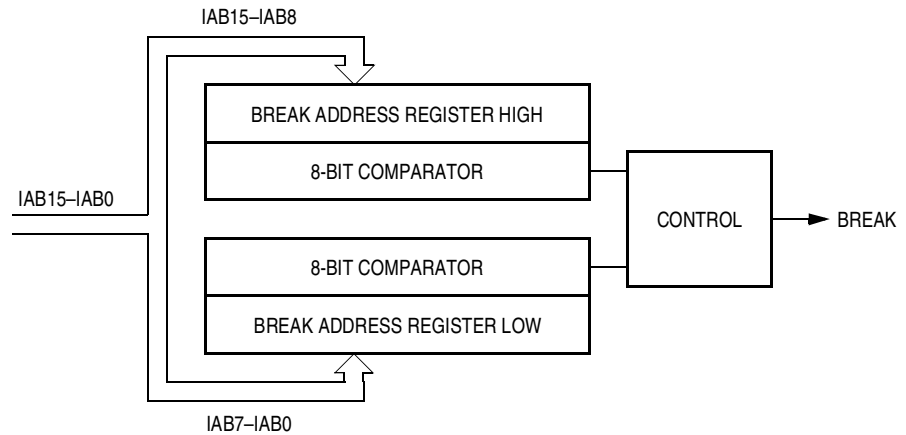
Figure 6-1. Break Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR) See page 107.	Read:	0	0	0	1	0	0	SBSW	0
		Write:	R	R	R	R	R	R	NOTE	R
		Reset:	0	0	0	1	0	0	0	0

Note: Writing a logic 0 clear SBSW.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE03	SIM Break Flag Control Register (SBFCR) See page 108.	Read: Write:	BCFE	R	R	R	R	R	R	R
		Reset:	0							
$FE09	Break Address Register High (BRKH) See page 106.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	0	0	0	0	0	0	0	0
$FE0A	Break Address Register Low (BRKL) See page 106.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	0	0	0	0	0	0	0	0
$FE0B	Break Status and Control Register (BRKSCR) See page 105.	Read:	BRKE	BRKA	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

 = Unimplemented   R = Reserved

Figure 6-2. I/O Register Summary

6.4.1 Flag Protection During Break Interrupts

The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state.

6.4.2 CPU During Break Interrupts

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 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.

6.4.3 TIM1 and TIM2 During Break Interrupts

A break interrupt stops the timer counters.

6.4.4 COP During Break Interrupts

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

6.5 Low-Power Modes

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

6.5.1 Wait Mode

If enabled, the break module is active in wait mode. In the break routine, the user can subtract one from the return address on the stack if SBSW is set. See **Section 3. Low-Power Modes**. Clear the SBSW bit by writing logic 0 to it.

6.5.2 Stop Mode

A break interrupt causes exit from stop mode and sets the SBSW bit in the break status register.

6.6 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)
- SIM break status register (SBSR)
- SIM break flag control register (SBFCR)

6.6.1 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 6-3. 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 logic 0 to bit 7. Reset clears the BRKE bit.
   1 = Breaks enabled on 16-bit address match
   0 = Breaks disabled on 16-bit address match

BRKA — Break Active Bit

This read/write status and control bit is set when a break address match occurs. Writing a logic 1 to BRKA generates a break interrupt. Clear BRKA by writing a logic 0 to it before exiting the break routine. Reset clears the BRKA bit.

    1 = (When read) Break address match
    0 = (When read) No break address match

6.6.2 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	14	13	12	11	10	9	Bit 8
Reset:	0	0	0	0	0	0	0	0
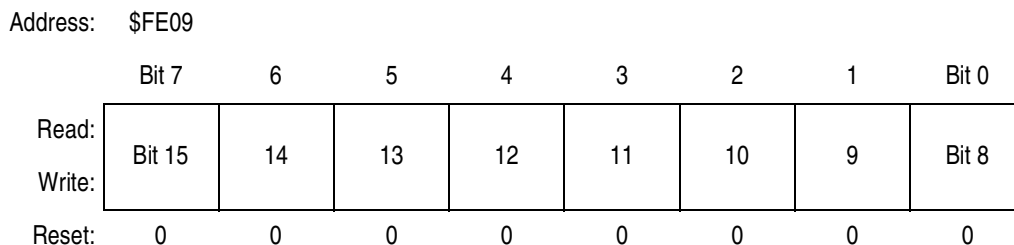
Figure 6-4. Break Address Register High (BRKH)

Address: $FE0A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0
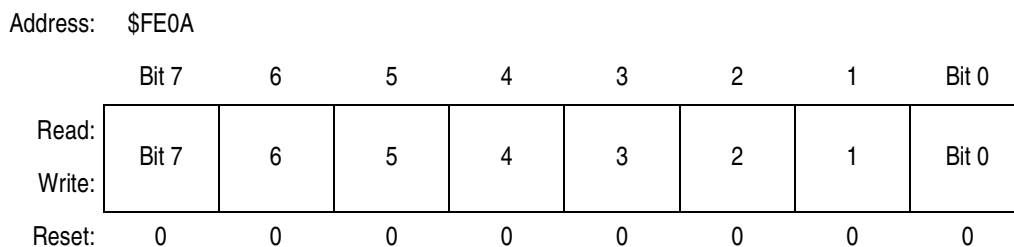
Figure 6-5. Break Address Register Low (BRKL)

6.6.3 Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from stop or wait mode. The flag is useful in applications requiring a return to stop or wait mode after exiting from a break interrupt.

Address: $FE00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	1	0	0	SBSW	0
Write:	R	R	R	R	R	R	NOTE	R
Reset:	0	0	0	1	0	0	0	0

Note: Writing a logic 0 clears SBSW.

| R | = Reserved |

Figure 6-6. SIM Break Status Register (SBSR)

SBSW — SIM Break Stop/Wait Bit

This read/write bit is set when a break interrupt causes an exit from stop or wait mode. Clear SBSW by writing a logic 0 to it. Reset clears SBSW.
  1 = Break interrupt during stop/wait mode
  0 = No break interrupt during stop/wait mode

SBSW can be read within the break interrupt routine. The user can modify the return address on the stack by subtracting 1 from it. The following code is an example.

This code works if the H register was stacked in the break interrupt routine. Execute this code at the end of the break interrupt routine.

```
    HIBYTE  EQU     5
    LOBYTE  EQU     6
;           If not SBSW, do RTI
            BRCLR  SBSW,BSR, RETURN  ;  See if wait mode or stop mode
                                     ;  was exited by break.
            TST    LOBYTE,SP         ;  If RETURNLO is not 0,
            BNE    DOLO              ;  then just decrement low byte.
            DEC    HIBYTE,SP         ;  Else deal with high byte also.
    DOLO    DEC    LOBYTE,SP         ;  Point to WAIT/STOP opcode.
    RETURN  PULH                     ;  Restore H register.
            RTI
```

6.6.4 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:	BCFE	R	R	R	R	R	R	R
Write:								
Reset:	0							

R	= Reserved

Figure 6-7. 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

Section 7. Internal Clock Generator (ICG) Module

7.1 Contents

7.2 Introduction

The internal clock generator module (ICG) is used to create a stable clock source for the microcontroller without using any external components. The ICG generates the oscillator output clock (CGMXCLK), which is used by the low-voltage inhibit (LVI) and other modules. The ICG also generates the clock generator output (CGMOUT), which is fed to the system integration module (SIM) to create the bus clocks. The bus frequency will be one-fourth the frequency of CGMXCLK and one-half the frequency of CGMOUT. Finally, the ICG generates the timebase clock (TBMCLK), which is used in the timebase module (TBM) and the computer operating properly (COP) clock (COPCLK) which is used by the COP module.

7.3 Features

The ICG has these features:

- Selectable external clock generator, either 1-pin external source or 2-pin crystal, multiplexed with port pins

- Internal clock generator with programmable frequency output in integer multiples of a nominal frequency (307.2 kHz $\pm$ 25 percent)

- Frequency adjust (trim) register to improve variability to $\pm$4 percent

- Bus clock software selectable from either internal or external clock (bus frequency range from 76.8 kHz $\pm$ 25 percent to 9.75 MHz $\pm$ 25 percent in 76.8-kHz increments

NOTE: *Do not exceed the maximum bus frequency of 8 MHz at 5.0 V and 4 MHz at 3.0 V.*

- Timebase clock automatically selected from external if external clock is available

- Clock monitor for both internal and external clocks

7.4 Functional Description

The ICG, shown in **Figure 7-1**, contains these major submodules:

- Clock enable circuit

- Internal clock generator

- External clock generator

- Clock monitor circuit

- Clock selection circuit

CS

RESET

CLOCK
SELECTION
CIRCUIT

CGMOUT

CGMXCLK

TBMCLK

COPCLK

IOFF

EOFF

CMON

CLOCK
MONITOR
CIRCUIT

ECGS

ICGS

N[6:0}

TRIM[7:0]

INTERNAL CLOCK
GENERATOR

FICGS

DDIV[3:0]

DSTG[7:0]

ICLK

IBASE

ICGEN

SIMOSCEN

OSCENINSTOP

EXTCLKEN

ECGON

ICGON

CLOCK/PIN
ENABLE
CIRCUIT

ECGEN

EXTXTALEN

EXTSLOW

EXTERNAL CLOCK
GENERATOR

ECLK

INTERNAL
TO MCU

PTE4
LOGIC

OSC1
PTE4

OSC2
PTE3

PTE3
LOGIC

EXTERNAL

| NAME | CONFIG2 REGISTER BIT |
| NAME | TOP LEVEL SIGNAL |

| NAME | REGISTER BIT |
| NAME | MODULE SIGNAL |

Figure 7-1. ICG Module Block Diagram

7.4.1 Clock Enable Circuit

The clock enable circuit is used to enable the internal clock (ICLK) or external clock (ECLK) and the port logic which is shared with the oscillator pins (OSC1 and OSC2). The clock enable circuit generates an ICG stop (ICGSTOP) signal which stops all clocks (ICLK, ECLK, and the low-frequency base clock, IBASE). ICGSTOP is set and the ICG is disabled in stop mode if the oscillator enable stop bit (OSCENINSTOP) in the CONFIG2 register is clear. The ICG clocks will be enabled in stop mode if OSCENINSTOP is high.

The internal clock enable signal (ICGEN) turns on the internal clock generator which generates ICLK. ICGEN is set (active) whenever the ICGON bit is set and the ICGSTOP signal is clear. When ICGEN is clear, ICLK and IBASE are both low.

The external clock enable signal (ECGEN) turns on the external clock generator which generates ECLK. ECGEN is set (active) whenever the ECGON bit is set and the ICGSTOP signal is clear. ECGON cannot be set unless the external clock enable (EXTCLKEN) bit in the CONFIG2 register is set. when ECGEN is clear, ECLK is low.

The port E4 enable signal (PE4EN) turns on the port E4 logic. Since port E4 is on the same pin as OSC1, this signal is only active (set) when the external clock function is not desired. Therefore, PE4EN is clear when ECGON is set. PE4EN is not gated with ICGSTOP, which means that if the ECGON bit is set, the port E4 logic will remain disabled in stop mode.

The port E3 enable signal (PE3EN) turns on the port E3 logic. Since port E3 is on the same pin as OSC2, this signal is only active (set) when 2-pin oscillator function is not desired. Therefore, PE3EN is clear when ECGON and the external crystal enable (EXTXTALEN) bit in the CONFIG2 register are both set. PE3EN is not gated with ICGSTOP, which means that if ECGON and EXTXTALEN are set, the port E3 logic will remain disabled in stop mode.

7.4.2 Internal Clock Generator

The internal clock generator, shown in **Figure 7-2**, creates a low frequency base clock (IBASE), which operates at a nominal frequency (f_{NOM}) of 307.2 kHz $\pm$ 25 percent, and an internal clock (ICLK) which is an integer multiple of IBASE. This multiple is the ICG multiplier factor (N), which is programmed in the ICG multiplier register (ICGMR). The internal clock generator is turned off and the output clocks (IBASE and ICLK) are held low when the internal clock generator enable signal (ICGEN) is clear.

The internal clock generator contains:

- A digitally controlled oscillator

- A modulo N divider

- A frequency comparator, which contains voltage and current references, a frequency to voltage converter, and comparators

- A digital loop filter
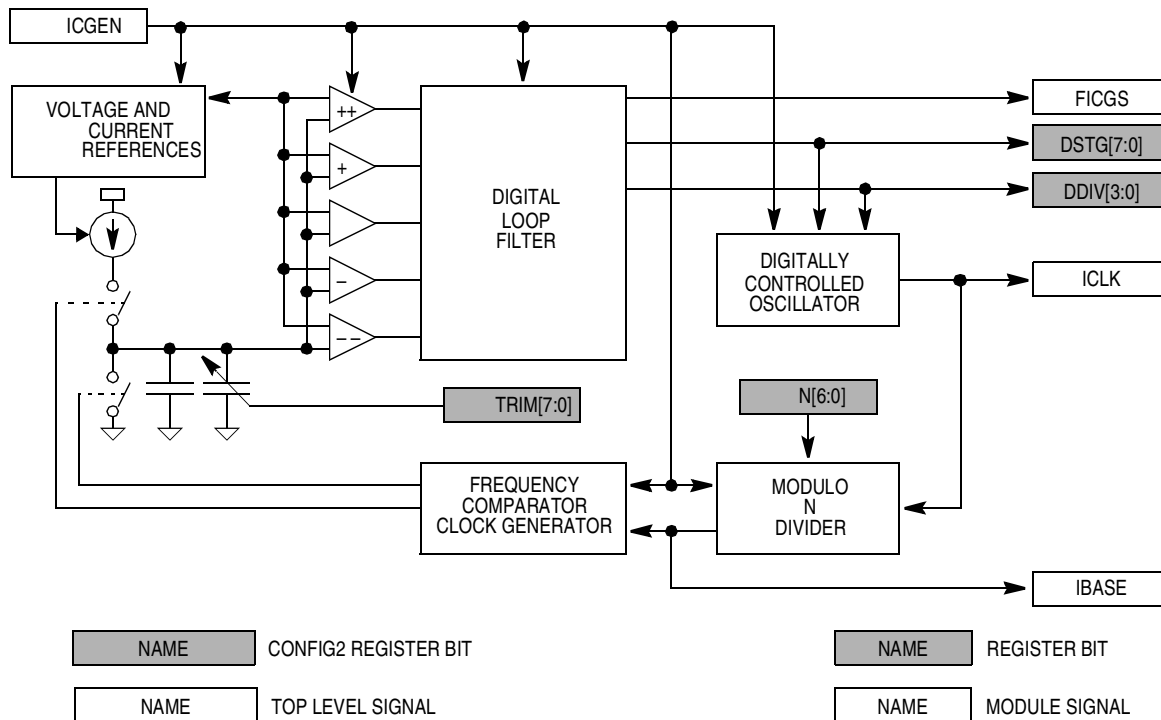
Figure 7-2. Internal Clock Generator Block Diagram

7.4.2.1 Digitally Controlled Oscillator

The digitally controlled oscillator (DCO) is an inaccurate oscillator which generates the internal clock (ICLK). The clock period of ICLK is dependent on the digital loop filter outputs (DSTG[7:0] and DDIV[3:0]). Because of only a limited number of bits in DDIV and DSTG, the precision of the output (ICLK) is restricted to a precision of approximately ±0.202 percent to ±0.368 percent when measured over several cycles (of the desired frequency). Additionally, since the propagation delays of the devices used in the DCO ring oscillator are a measurable fraction of the bus clock period, reaching the long-term precision may require alternately running faster and slower than desired, making the worst case cycle-to-cycle frequency variation ±6.45 percent to ±11.8 percent (of the desired frequency). The valid values of DDIV:DSTG range from $000 to $9FF. For more information on the quantization error in the DCO, see **7.5.4 Quantization Error in DCO Output**.

7.4.2.2 Modulo N Divider

The modulo N divider creates the low-frequency base clock (IBASE) by dividing the internal clock (ICLK) by the ICG multiplier factor (N), contained in the ICG multiplier register (ICGMR). When N is programmed to a $01 or $00, the divider is disabled and ICLK is passed through to IBASE undivided. When the internal clock generator is stable, the frequency of IBASE will be equal to the nominal frequency (f_{NOM}) of 307.2 kHz ± 25 percent.

7.4.2.3 Frequency Comparator

The frequency comparator effectively compares the low-frequency base clock (IBASE) to a nominal frequency, f_{NOM}. First, the frequency comparator converts IBASE to a voltage by charging a known capacitor with a current reference for a period dependent on IBASE. This voltage is compared to a voltage reference with comparators, whose outputs are fed to the digital loop filter. The dependence of these outputs on the capacitor size, current reference, and voltage reference causes up to ±25 percent error in f_{NOM}.

7.4.2.4 Digital Loop Filter

The digital loop filter (DLF) uses the outputs of the frequency comparator to adjust the internal clock (ICLK) clock period. The DLF generates the DCO divider control bits (DDIV[3:0]) and the DCO stage control bits (DSTG[7:0]), which are fed to the DCO. The DLF first concatenates the DDIV and DSTG registers (DDIV[3:0]:DSTG[7:0]) and then adds or subtracts a value dependent on the relative error in the low-frequency base clock's period, as shown in **Table 7-1**. In some extreme error conditions, such as operating at a V_{DD} level which is out of specification, the DLF may attempt to use a value above the maximum ($9FF) or below the minimum ($000). In both cases, the value for DDIV will be between $A and $F. In this range, the DDIV value will be interpreted the same as $9 (the slowest condition). Recovering from this condition requires subtracting (increasing frequency) in the normal fashion until the value is again below $9FF. (If the desired value is $9xx, the value may settle at $Axx through $Fxx. This is an acceptable operating condition.) If the error is less than ±5 percent, the internal clock generator's filter stable indicator (FICGS) is set, indicating relative frequency accuracy to the clock monitor.

Table 7-1. Correction Sizes from DLF to DCO

Frequency Error of IBASE Compared to f_{NOM}	DDVI[3:0]:DSTG[7:0] Correction	Current to New DDIV[3:0]:DSTG[7:0][1]		Relative Correction in DCO	
IBASE < 0.85 f_{NOM}	−32 (−$020)	Minimum	$xFF to $xDF	−2/31	−6.45%
		Maximum	$x20 to $x00	−2/19	−10.5%
0.85 f_{NOM} < IBASE IBASE < 0.95 f_{NOM}	−8 (−$008)	Minimum	$xFF to $xF7	−0.5/31	−1.61%
		Maximum	$x08 to $x00	−0.5/17.5	−2.86%
0.95 f_{NOM} < IBASE IBASE < f_{NOM}	−1 (−$001)	Minimum	$xFF to $xFE	−0.0625/31	−0.202%
		Maximum	$x01 to $x00	−0.0625/17.0625	−0.366%
f_{NOM} < IBASE IBASE < 1.05 f_{NOM}	+1 (+$001)	Minimum	$xFE to $xFF	+0.0625/30.9375	+0.202%
		Maximum	$x00 to $x01	+0.0625/17	+0.368%
1.05 f_{NOM} < IBASE IBASE < 1.15 f_{NOM}	+8 (+$008)	Minimum	$xF7 to $xFF	+0.5/30.5	+1.64%
		Maximum	$x00 to $x08	+0.5/17	+2.94%
1.15 f_{NOM} < IBASE	+32 (+$020)	Minimum	$xDF to $xFF	+2/29	+6.90%
		Maximum	$x00 to $x20	+2/17	+11.8%

1. x = Maximum error is independent of value in DDIV[3:0]. DDIV increments or decrements when an addition to DSTG[7:0] carries or borrows.

7.4.3 External Clock Generator

The ICG also provides for an external oscillator or external clock source, if desired. The external clock generator, shown in **Figure 7-3**, contains an external oscillator amplifier and an external clock input path.
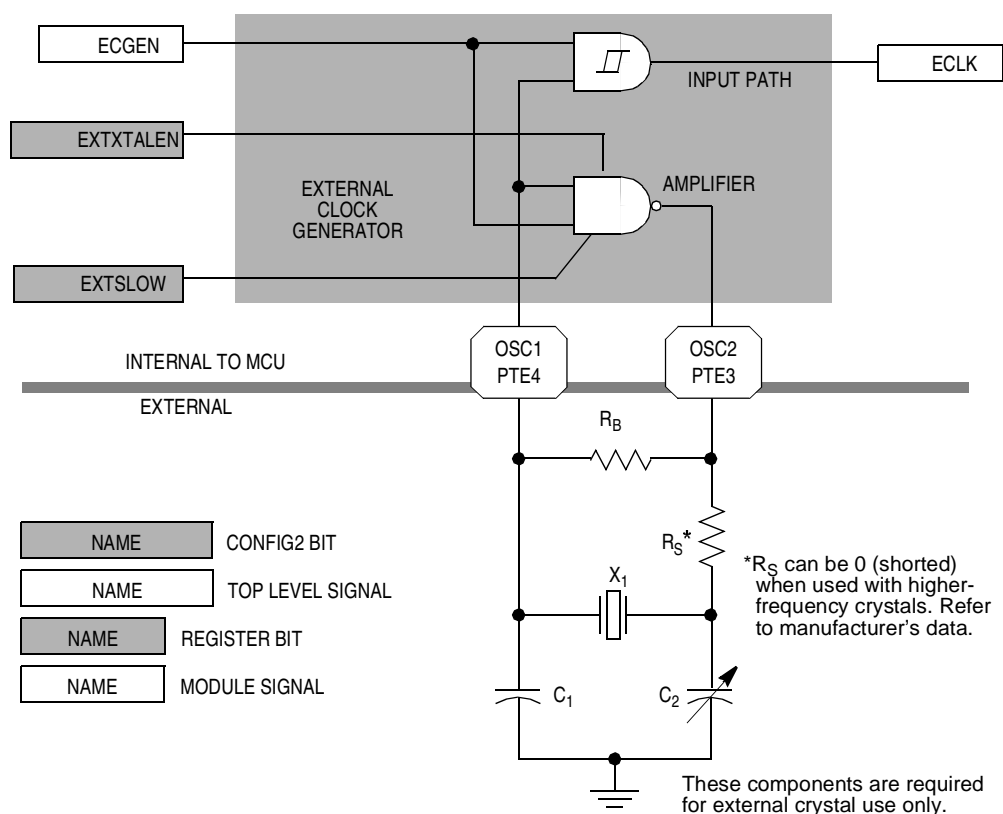
Figure 7-3. External Clock Generator Block Diagram

7.4.3.1 External Oscillator Amplifier

The external oscillator amplifier provides the gain required by an external crystal connected in a Pierce oscillator configuration. The amount of this gain is controlled by the slow external (EXTSLOW) bit in the CONFIG2 register. When EXTSLOW is set, the amplifier gain is reduced for operating low-frequency crystals (32 kHz to 100 kHz). When EXTSLOW is clear, the amplifier gain will be sufficient for 1-MHz to 8-MHz crystals. EXTSLOW must be configured correctly for the given crystal or the circuit may not operate.

The amplifier is enabled when the external clock generator enable (ECGEN) signal is set and when the external crystal enable (EXTXTALEN) bit in the CONFIG2 register is set. ECGEN is controlled by the clock enable circuit (see **7.4.1 Clock Enable Circuit**) and indicates that the external clock function is desired. When enabled, the amplifier will be connected between the PTE4/OSC1 and PTE3/OSC2 pins. Otherwise, the PTE3/OSC2 pin reverts to its port function.

In its typical configuration, the external oscillator requires five external components:

1. Crystal, X_1

2. Fixed capacitor, C_1

3. Tuning capacitor, C_2 (can also be a fixed capacitor)

4. Feedback resistor, RB

5. Series resistor, R_S (Included in **Figure 7-3** to follow strict Pierce oscillator guidelines and may not be required for all ranges of operation, especially with high frequency crystals. Refer to the crystal manufacturer's data for more information.)

7.4.3.2 External Clock Input Path

The external clock input path is the means by which the microcontroller uses an external clock source. The input to the path is the PTE4/OSC1 pin and the output is the external clock (ECLK). The path, which contains input buffering, is enabled when the external clock generator enable signal (ECGEN) is set. When not enabled, the PTE4/OSC1 pin reverts to its port function.

7.4.4 Clock Monitor Circuit

The ICG contains a clock monitor circuit which, when enabled, will continuously monitor both the external clock (ECLK) and the internal clock (ICLK) to determine if either clock source has been corrupted. The clock monitor circuit, shown in **Figure 7-4**, contains these blocks:

- Clock monitor reference generator
- Internal clock activity detector
- External clock activity detector
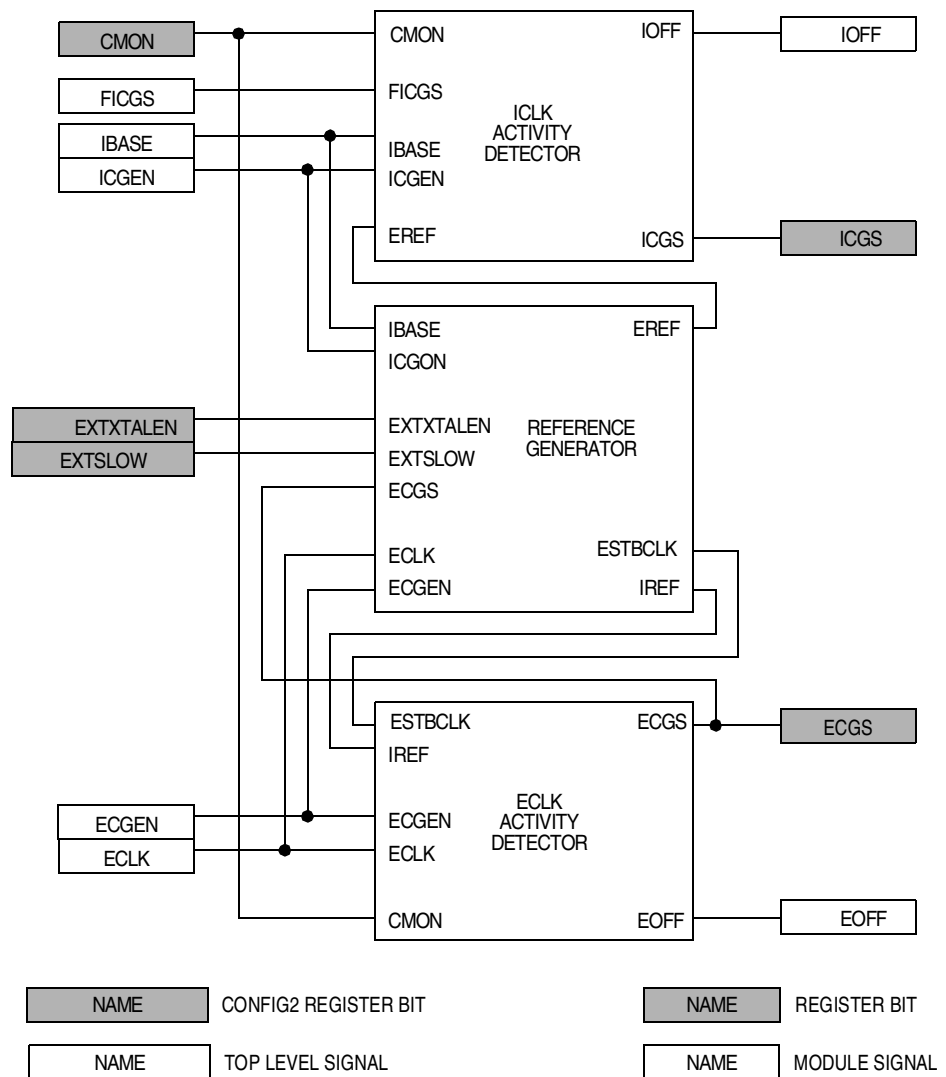
Figure 7-4. Clock Monitor Block Diagram

7.4.4.1 Clock Monitor Reference Generator

The clock monitor uses a reference based on one clock source to monitor the other clock source. The clock monitor reference generator generates the external reference clock (EREF) based on the external clock (ECLK) and the internal reference clock (IREF) based on the internal clock (ICLK). To simplify the circuit, the low-frequency base clock (IBASE) is used in place of ICLK because it always operates at or near 307.2 kHz. For proper operation, EREF must be at least twice as slow as IBASE and IREF must be at least twice as slow as ECLK.

To guarantee that IREF is slower than ECLK and EREF is slower than IBASE, one of the signals is divided down. Which signal is divided and by how much is determined by the external slow (EXTSLOW) and external crystal enable (EXTXTALEN) bits in the CONFIG2 register, according to the rules in **Table 7-2**.

NOTE: *Each signal (IBASE and ECLK) is always divided by four. A longer divider is used on either IBASE or ECLK based on the EXTSLOW bit.*

To conserve size, the long divider (divide by 4096) is also used as an external crystal stabilization divider. The divider is reset when the external clock generator is turned off or in stop mode (ECGEN is clear). When the external clock generator is first turned on, the external clock generator stable bit (ECGS) will be clear. This condition automatically selects ECLK as the input to the long divider. The external stabilization clock (ESTBCLK) will be ECLK divided by 16 when EXTXTALEN is low or 4096 when EXTXTALEN is high. This timeout allows the crystal to stabilize. The falling edge of ESTBCLK is used to set ECGS, which will set after a full 16 or 4096 cycles. When ECGS is set, the divider returns to its normal function. ESTBCLK may be generated by either IBASE or ECLK, but any clocking will only reinforce the set condition. If ECGS is cleared because the clock monitor determined that ECLK was inactive, the divider will revert to a stabilization divider. Since this will change the EREF and IREF divide ratios, it is important to turn the clock monitor off (CMON = 0) after inactivity is detected to ensure valid recovery.

7.4.4.2 Internal Clock Activity Detector

The internal clock activity detector, shown in **Figure 7-5**, looks for at least one falling edge on the low-frequency base clock (IBASE) every time the external reference (EREF) is low. Since EREF is less than half the frequency of IBASE, this should occur every time. If it does not occur two consecutive times, the internal clock inactivity indicator (IOFF) is set. IOFF will be cleared the next time there is a falling edge of IBASE while EREF is low.

The internal clock stable bit (ICGS) is also generated in the internal clock activity detector. ICGS is set when the internal clock generator's filter stable signal (FICGS) indicates that IBASE is within about 5 percent of the target 307.2 kHz $\pm$ 25 percent for two consecutive measurements. ICGS is cleared when FICGS is clear, the internal clock generator is turned off or is in stop mode (ICGEN is clear), or when IOFF is set.

Figure 7-5. Internal Clock Activity Detector

7.4.4.3 External Clock Activity Detector

The external clock activity detector, shown in **Figure 7-6**, looks for at least one falling edge on the external clock (ECLK) every time the internal reference (IREF) is low. Since IREF is less than half the frequency of ECLK, this should occur every time. If it does not occur two consecutive times, the external clock inactivity indicator (EOFF) is set. EOFF will be cleared the next time there is a falling edge of ECLK while IREF is low.

The external clock stable bit (ECGS) is also generated in the external clock activity detector. ECGS is set on a falling edge of the external stabilization clock (ESTBCLK). This will be 4096 ECLK cycles after the external clock generator on bit is set, or the MCU exits stop mode (ECGEN = 1) if the external crystal enable (EXTXTALEN) in the CONFIG2 register is set, or 16 cycles when EXTXTALEN is clear. ECGS is cleared when the external clock generator is turned off or in stop mode (ECGEN is clear) or when EOFF is set.
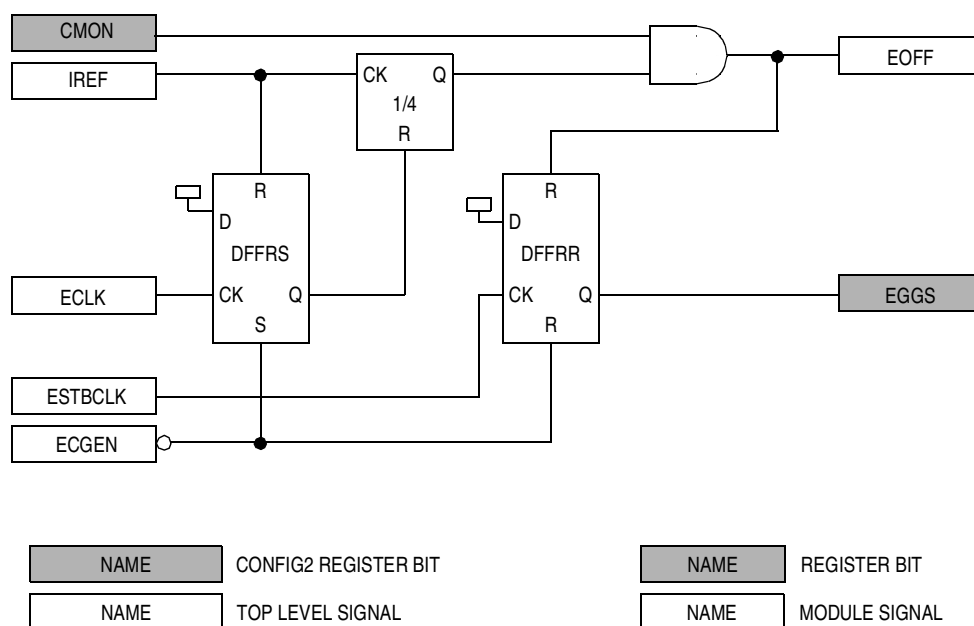
Figure 7-6. External Clock Activity Detector

7.4.5 Clock Selection Circuit

The clock selection circuit, shown in **Figure 7-7**, contains two clock switches which generate the oscillator output clock (CGMXCLK) and the timebase clock (TBMCLK) from either the internal clock (ICLK) or the external clock (ECLK). The COP clock (COPCLK) is identical to TBMCLK. The clock selection circuit also contains a divide-by-two circuit which creates the clock generator output clock (CGMOUT), which generates the bus clocks.
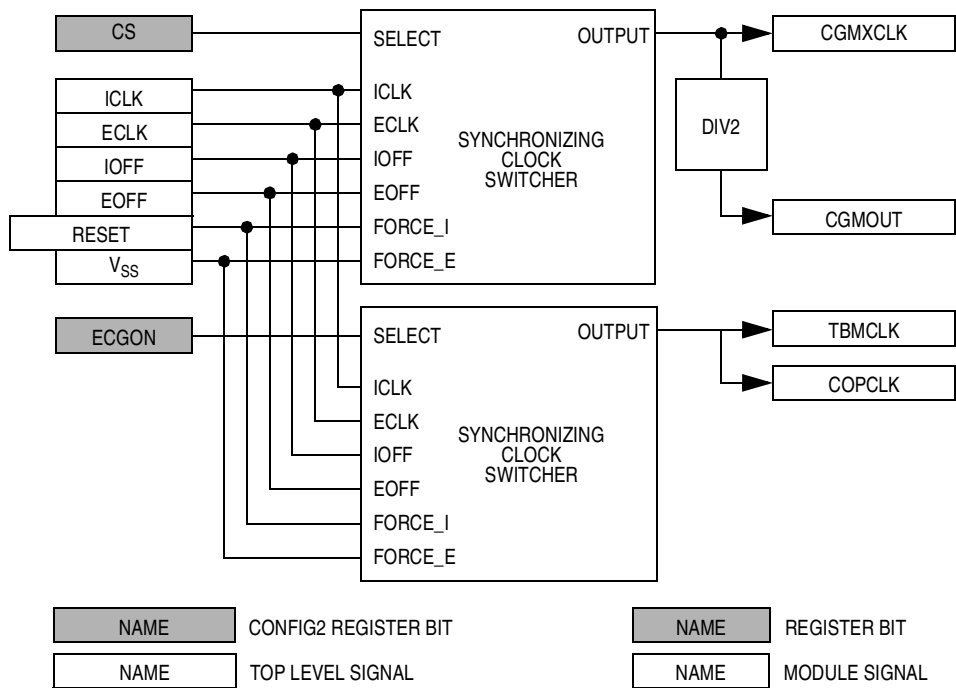
Figure 7-7. Clock Selection Circuit Block Diagram

7.4.5.1 Clock Selection Switches

The first switch creates the oscillator output clock (CGMXCLK) from either the internal clock (ICLK) or the external clock (ECLK), based on the clock select bit (CS; set selects ECLK, clear selects ICLK). When switching the CS bit, both ICLK and ECLK must be on (ICGON and ECGON set). The clock being switched to also must be stable (ICGS or ECGS set).

The second switch creates the timebase clock (TBMCLK) and the COP clock (COPCLK) from ICLK or ECLK based on the external clock on bit. When ECGON is set, the switch automatically selects the external clock, regardless of the state of the ECGS bit.

7.4.5.2 Clock Switching Circuit

To robustly switch between the internal clock (ICLK) and the external clock (ECLK), the switch assumes the clocks are completely asynchronous, so a synchronizing circuit is required to make the transition. When the select input (the clock select bit for the oscillator output clock switch or the external clock on bit for the timebase clock switch) is changed, the switch will continue to operate off the original clock for between one and two cycles as the select input is transitioned through one side of the synchronizer. Next, the output will be held low for between one and two cycles of the new clock as the select input transitions through the other side. Then the output starts switching at the new clock's frequency. This transition guarantees that no glitches will be seen on the output even though the select input may change asynchronously to the clocks. The unpredictably of the transition period is a necessary result of the asynchronicity.

The switch automatically selects ICLK during reset. When the clock monitor is on (CMON is set) and it determines one of the clock sources is inactive (as indicated by the IOFF or EOFF signals), the circuit is forced to select the active clock. There are no clocks for the inactive side of the synchronizer to properly operate, so that side is forced deselected. However, the active side will not be selected until one to two clock cycles after the IOFF or EOFF signal transitions.

7.5 Usage Notes

The ICG has several features which can provide protection to the microcontroller if properly used. Other features can greatly simplify usage of the ICG if certain techniques are employed. This section describes several possible ways to use the ICG and its features. These techniques are not the only ways to use the ICG and may not be optimum for all environments. In any case, these techniques should be used only as a template, and the user should modify them according to the application's requirements.

These notes include:

- Switching clock sources

- Enabling the clock monitor

- Using clock monitor interrupts

- Quantization error in digitally controlled oscillator (DCO) output

- Switching internal clock frequencies

- Nominal frequency settling time

- Improving frequency settling time

- Trimming frequency

7.5.1 Switching Clock Sources

Switching from one clock source to another requires both clock sources to be enabled and stable. A simple flow requires:

- Enable desired clock source

- Wait for it to become stable

- Switch clocks

- Disable previous clock source

The key point to remember in this flow is that the clock source cannot be switched (CS cannot be written) unless the desired clock is on and stable. A short assembly code example of how to employ this flow is

shown in **Figure 7-8**. This code is for illustrative purposes only and does not represent valid syntax for any particular assembler.

```
                       ;Clock Switching Code Example
                       ;This code switches from Internal to External clock
                       ;Clock Monitor and interrupts are not enabled
start  lda    #$13    ;Mask for CS, ECGON, ECGS
                       ; If switching from External to Internal, mask is $0C.
 loop    **    **     ;Other code here, such as writing the COP, since ECGS may
                       ; take some time to set
        sta    icgcr   ;Try to set CS, ECGON and clear ICGON. ICGON will not
                       ; clear until CS is set, and CS will not set until
                       ; ECGON and ECGS are set.
        cmpa   icgcr   ;Check to see if ECGS set, then CS set, then ICGON clear
        bne    loop    ;Keep looping until ICGON is clear.
```

Figure 7-8. Code Example for Switching Clock Sources

7.5.2 Enabling the Clock Monitor

Many applications require the clock monitor to determine if one of the clock sources has become inactive, so the other can be used to recover from a potentially dangerous situation. Using the clock monitor requires both clocks to be active (ECGON and ICGON both set). To enable the clock monitor, both clocks also must be stable (ECGS and ICGS both set). This is to prevent the use of the clock monitor when a clock is first turned on and potentially unstable.

Enabling the clock monitor and clock monitor interrupts requires a flow similar to this:

- Enable the alternate clock source
- Wait for both clock sources to be stable
- Switch to the desired clock source if necessary
- Enable the clock monitor
- Enable clock monitor interrupts

These events must happen in sequence. A short assembly code example of how to employ this flow is shown in **Figure 7-9**. This code is for illustrative purposes only and does not represent valid syntax for any particular assembler.

```
                          ;Clock Monitor Enabling Code Example
                          ;This code turns on both clocks, selects the desired
                          ; one, then turns on the Clock Monitor and Interrupts
start  lda    #$AF        ;Mask for CMIE, CMON, ICGON, ICGS, ECGON, ECGS
                          ; If Internal Clock desired, mask is $AF
                          ; If External Clock desired, mask is $BF
                          ; If interrupts not desired mask is $2F int; $3F ext
loop   **     **          ;Other code here, such as writing the COP, since ECGS
                          ; and ICGS may take some time to set.
       sta    icgcr       ;Try to set CMIE. CMIE wont set until CMON set; CMON
                          ; won't set until ICGON, ICGS, ECGON, ECGS set.
       brset  6,ICGCR,error ;Verify CMF is not set
       cmpa   icgcr       ;Check if ECGS set, then CMON set, then CMIE set
       bne    loop        ;Keep looping until CMIE is set.
```

Figure 7-9. Code Example for Enabling the Clock Monitor

7.5.3 Using Clock Monitor Interrupts

The clock monitor circuit can be used to recover from perilous situations such as crystal loss. To use the clock monitor effectively, these points should be observed:

- Enable the clock monitor and clock monitor interrupts.

- The first statement in the clock monitor interrupt service routine (CMISR) should be a read to the ICG control register (ICGCR) to verify that the clock monitor flag (CMF) is set. This is also the first step in clearing the CMF bit.

- The second statement in the CMISR should be a write to the ICGCR to clear the CMF bit (write the bit low). Writing the bit high will not affect it. This statement does not need to immediately follow the first, but must be contained in the CMISR.

- The third statement in the CMISR should be to clear the CMON bit. This is required to ensure proper reconfiguration of the reference dividers. This statement also must be contained in the CMISR.

- Although the clock monitor can be enabled only when both clocks are stable (ICGS is set or ECGS is set), it will remain set if one of the clocks goes unstable.

- The clock monitor only works if the external slow (EXTSLOW) bit in the CONFIG2 register is set to the correct value.

- The internal and external clocks must both be enabled and running to use the clock monitor.

- When the clock monitor detects inactivity, the inactive clock is automatically deselected and the active clock selected as the source for CGMXCLK and TBMCLK. The CMISR can use the state of the CS bit to check which clock is inactive.

- When the clock monitor detects inactivity, the application may have been subjected to extreme conditions which may have affected other circuits. The CMISR should take any appropriate precautions.

7.5.4 Quantization Error in DCO Output

The digitally controlled oscillator (DCO) is comprised of three major sub-blocks:

1. Binary weighted divider

2. Variable-delay ring oscillator

3. Ring oscillator fine-adjust circuit

Each of these blocks affects the clock period of the internal clock (ICLK). Since these blocks are controlled by the digital loop filter (DLF) outputs DDIV and DSTG, the output of the DCO can change only in quantized steps as the DLF increments or decrements its output. The following sections describe how each block will affect the output frequency.

7.5.4.1 Digitally Controlled Oscillator

The digitally controlled oscillator (DCO) is an inaccurate oscillator which generates the internal clock (ICLK), whose clock period is dependent on the digital loop filter outputs (DSTG[7:0] and DDIV[3:0]). Because of the digital nature of the DCO, the clock period of ICLK will change in quantized steps. This will create a clock period difference or quantization error (Q-ERR) from one cycle to the next. Over several cycles or for longer periods, this error is divided out until it reaches a minimum error of 0.202 percent to 0.368 percent. The dependence of this error on the DDIV[3:0] value and the number of cycles the error is measured over is shown in **Table 7-2**.

Table 7-2. Quantization Error in ICLK

DDIV[3:0]	ICLK Cycles	Bus Cycles	τ_{ICLK} Q-ERR
%0000 (min)	1	NA	6.45%–11.8%
%0000 (min)	4	1	1.61%–2.94%
%0000 (min)	≥ 32	≥ 8	0.202%–0.368%
%0001	1	NA	3.23%–5.88%
%0001	4	1	0.806%–1.47%
%0001	≥ 16	≥ 4	0.202%–0.368%
%0010	1	NA	1.61%–2.94%
%0010	4	1	0.403%–0.735%
%0010	≥ 8	≥ 2	0.202%–0.368%
%0011	1	NA	0.806%–1.47%
%0011	≥ 4	≥ 1	0.202%–0.368%
%0100	1	NA	0.403%–0.735%
%0100	≥ 2	≥ 1	0.202%–0.368%
%0101–%1001 (max)	≥ 1	≥ 1	0.202%–0.368%

7.5.4.2 Binary Weighted Divider

The binary weighted divider divides the output of the ring oscillator by a power of two, specified by the DCO divider control bits (DDIV[3:0]). DDIV maximizes at %1001 (values of %1010 through %1111 are interpreted as %1001), which corresponds to a divide by 512. When DDIV is %0000, the ring oscillator's output is divided by 1. Incrementing DDIV by one will double the period; decrementing DDIV will halve the period. The DLF cannot directly increment or decrement DDIV; DDIV is only incremented or decremented when an addition or subtraction to DSTG carries or borrows.

7.5.4.3 Variable-Delay Ring Oscillator

The variable-delay ring oscillator's period is adjustable from 17 to 31 stage delays, in increments of two, based on the upper three DCO stage control bits (DSTG[7:5]). A DSTG[7:5] of %000 corresponds to 17 stage delays; DSTG[7:5] of %111 corresponds to 31 stage delays. Adjusting the DSTG[5] bit has a 6.45 percent to 11.8 percent effect on the output frequency. This also corresponds to the size correction made when the frequency error is greater than ±15 percent. The value of the binary weighted divider does not affect the relative change in output clock period for a given change in DSTG[7:5].

7.5.4.4 Ring Oscillator Fine-Adjust Circuit

The ring oscillator fine-adjust circuit causes the ring oscillator to effectively operate at non-integer numbers of stage delays by operating at two different points for a variable number of cycles specified by the lower five DCO stage control bits (DSTG[4:0]). For example:

- When DSTG[7:5] is %011, the ring oscillator nominally operates at 23 stage delays.

- When DSTG[4:0] is %00000, the ring will always operate at 23 stage delays.

- When DSTG[4:0] is %00001, the ring will operate at 25 stage delays for one of 32 cycles and at 23 stage delays for 31 of 32 cycles.

- Likewise, when DSTG[4:0] is %11111, the ring operates at 25 stage delays for 31 of 32 cycles and at 23 stage delays for one of 32 cycles.

- When DSTG[7:5] is %111, similar results are achieved by including a variable divide-by-two, so the ring operates at 31 stages for some cycles and at 17 stage delays, with a divide-by-two for an effective 34 stage delays, for the remainder of the cycles.

Adjusting the DSTG[0] bit has a 0.202 percent to 0.368 percent effect on the output clock period. This corresponds to the minimum size correction made by the DLF, and the inherent, long-term quantization error in the output frequency.

7.5.5 Switching Internal Clock Frequencies

The frequency of the internal clock (ICLK) may need to be changed for some applications. For example, if the reset condition does not provide the correct frequency, or if the clock is slowed down for a low-power mode (or sped up after a low-power mode), the frequency must be changed by programming the internal clock multiplier factor (N). The frequency of ICLK is N times the frequency of IBASE, which is 307.2 kHz ±25 percent.

Before switching frequencies by changing the N value, the clock monitor must be disabled. This is because when N is changed, the frequency of the low-frequency base clock (IBASE) will change proportionally until the digital loop filter has corrected the error. Since the clock monitor uses IBASE, it could erroneously detect an inactive clock. The clock monitor cannot be re-enabled until the internal clock is stable again (ICGS is set).

The following flow is an example of how to change the clock frequency:

- Verify there is no clock monitor interrupt by reading the CMF bit.

- Turn off the clock monitor.

- If desired, switch to the external clock (see **7.5.1 Switching Clock Sources**).

- Change the value of N.

- Switch back to internal (see **7.5.1 Switching Clock Sources**), if desired.

- Turn on the clock monitor (see **7.5.2 Enabling the Clock Monitor**), if desired.

7.5.6 Nominal Frequency Settling Time

Because the clock period of the internal clock (ICLK) is dependent on the digital loop filter outputs (DDIV and DSTG) which cannot change instantaneously, ICLK temporarily will operate at an incorrect clock period when any operating condition changes. This happens whenever the part is reset, the ICG multiply factor (N) is changed, the ICG trim factor (TRIM) is changed, or the internal clock is enabled after inactivity (stop mode or disabled operation). The time that the ICLK takes to adjust to the correct period is known as the settling time.

Settling time depends primarily on how many corrections it takes to change the clock period and the period of each correction. Since the corrections require four periods of the low-frequency base clock ($4^*\tau_{IBASE}$), and since ICLK is N (the ICG multiply factor for the desired frequency) times faster than IBASE, each correction takes $4^*N^*\tau_{ICLK}$. The period of ICLK, however, will vary as the corrections occur.

7.5.6.1 Settling to Within 15 Percent

When the error is greater than 15 percent, the filter takes eight corrections to double or halve the clock period. Due to how the DCO increases or decreases the clock period, the total period of these eight corrections is approximately 11 times the period of the fastest correction. (If the corrections were perfectly linear, the total period would be 11.5 times the minimum period; however, the ring must be slightly nonlinear.) Therefore, the total time it takes to double or halve the clock period is $44^*N^*\tau_{ICLKFAST}$.

If the clock period needs more than doubled or halved, the same relationship applies, only for each time the clock period needs doubled, the total number of cycles doubles. That is, when transitioning from fast to slow, going from the initial speed to half speed takes $44^*N^*\tau_{ICLKFAST}$;

from half speed to quarter speed takes 88*N*$\tau_{ICLKFAST}$; going from quarter speed to eighth speed takes 176*N*$\tau_{ICLKFAST}$; and so on. This series can be expressed as $(2^x–1)$*44*N*$\tau_{ICLKFAST}$, where x is the number of times the speed needs doubled or halved. Since 2^x happens to be equal to $\tau_{ICLKSLOW}/\tau_{ICLKFAST}$, the equation reduces to 44*N*$(\tau_{ICLKSLOW}–\tau_{ICLKFAST})$.

Note that increasing speed takes much longer than decreasing speed since N is higher. This can be expressed in terms of the initial clock period (τ_1) minus the final clock period (τ_2) as such:

$$\tau_{15} = \text{abs}[44N(\tau_1 - \tau_2)]$$

7.5.6.2 Settling to Within 5 Percent

Once the clock period is within 15 percent of the desired clock period, the filter starts making smaller adjustments. When between 15 percent and 5 percent error, each correction will adjust the clock period between 1.61 percent and 2.94 percent. In this mode, a maximum of eight corrections will be required to get to less than 5 percent error. Since the clock period is relatively close to desired, each correction takes approximately the same period of time, or 4*τ_{IBASE}. At this point, the internal clock stable bit (ICGS) will be set and the clock frequency is usable, although the error will be as high as 5 percent. The total time to this point is:

$$\tau_5 = \text{abs}[44N(\tau_1 - \tau_2)] + 32\tau_{IBASE}$$

7.5.6.3 Total Settling Time

Once the clock period is within 5 percent of the desired clock period, the filter starts making minimum adjustments. In this mode, each correction will adjust the frequency between 0.202 percent and 0.368 percent. A maximum of 24 corrections will be required to get to the minimum error. Each correction takes approximately the same period of time, or 4*τ_{IBASE}. Added to the corrections for 15 percent to 5 percent, this

makes 32 corrections (128*τ_{IBASE}) to get from 15 percent to the minimum error. The total time to the minimum error is:

$$\tau_{tot} = \text{abs}[44N(\tau_1 - \tau_2)] + 128\tau_{IBASE}$$

The equations for τ_{15}, τ_5, and τ_{tot} are dependent on the actual initial and final clock periods τ_1 and τ_2, not the nominal. This means the variability in the ICLK frequency due to process, temperature, and voltage must be considered. Additionally, other process factors and noise can affect the actual tolerances of the points at which the filter changes modes. This means a worst case adjustment of up to 35 percent (ICLK clock period tolerance plus 10 percent) must be added. This adjustment can be reduced with trimming. **Table 7-3** shows some typical values for settling time.

Table 7-3. Typical Settling Time Examples

τ_1	τ_2	N	τ_{15}	τ_5	τ_{tot}
1/ (6.45 MHz)	1/ (25.8 MHz)	84	430 µs	535 µs	850 µs
1/ (25.8 MHz)	1/ (6.45 MHz)	21	107 µs	212 µs	525 µs
1/ (25.8 MHz)	1/ (307.2 kHz)	1	141 µs	246 µs	560 µs
1/ (307.2 kHz)	1/ (25.8 MHz)	84	11.9 ms	12.0 ms	12.3 ms

7.5.7 Trimming Frequency on the Internal Clock Generator

The unadjusted frequency of the low-frequency base clock (IBASE), when the comparators in the frequency comparator indicate zero error, will vary as much as ±25 percent due to process, temperature, and voltage dependencies. These dependencies are in the voltage and current references, the offset of the comparators, and the internal capacitor.

The method of changing the unadjusted operating point is by changing the size of the capacitor. This capacitor is designed with 639 equally sized units. Of that number, 384 of these units are always connected. The remaining 255 units are put in by adjusting the ICG trim factor (TRIM). The default value for TRIM is $80, or 128 units, making the default capacitor size 512. Each unit added or removed will adjust the

output frequency by about ±0.195 percent of the unadjusted frequency (adding to TRIM will decrease frequency). Therefore, the frequency of IBASE can be changed to ±25 percent of its unadjusted value, which is enough to cancel the process variability mentioned before.

The best way to trim the internal clock is to use the timer to measure the width of an input pulse on an input capture pin (this pulse must be supplied by the application and should be as long or wide as possible). Considering the prescale value of the timer and the theoretical (zero error) frequency of the bus (307.2 kHz *N/4), the error can be calculated. This error, expressed as a percentage, can be divided by 0.195 percent and the resultant factor added or subtracted from TRIM. This process should be repeated to eliminate any residual error.

7.6 Low-Power Modes

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

7.6.1 Wait Mode

The ICG remains active in wait mode. If enabled, the ICG interrupt to the CPU can bring the MCU out of wait mode.

In some applications, low power-consumption is desired in wait mode and a high-frequency clock is not needed. In these applications, reduce power consumption by either selecting a low-frequency external clock and turn the internal clock generator off or reduce the bus frequency by minimizing the ICG multiplier factor (N) before executing the WAIT instruction.

7.6.2 Stop Mode

The value of the oscillator enable in stop (OSCENINSTOP) bit in the CONFIG2 register determines the behavior of the ICG in stop mode. If OSCENINSTOP is low, the ICG is disabled in stop and, upon execution of the STOP instruction, all ICG activity will cease and the output clocks

(CGMXCLK, CGMOUT, COPCLK, and TBMCLK) will be held low. Power consumption will be minimal.

If OSCENINSTOP is high, the ICG is enabled in stop and activity will continue. This is useful if the timebase module (TBM) is required to bring the MCU out of stop mode. ICG interrupts will not bring the MCU out of stop mode in this case.

During stop mode, if OSCENINSTOP is low, several functions in the ICG are affected. The stable bits (ECGS and ICGS) are cleared, which will enable the external clock stabilization divider upon recovery. The clock monitor is disabled (CMON = 0) which will also clear the clock monitor interrupt enable (CMIE) and clock monitor flag (CMF) bits. The CS, ICGON, ECGON, N, TRIM, DDIV, and DSTG bits are unaffected.

7.7 CONFIG2 Options

Four CONFIG2 register options affect the functionality of the ICG. These options are:

1. EXTCLKEN, external clock enable

2. EXTXTALEN, external crystal enable

3. EXTSLOW, slow external clock

4. OSCENINSTOP, oscillator enable in stop

All CONFIG2 options will have a default setting. Refer to **Section 8. Configuration Register (CONFIG)** on how the CONFIG2 register is used.

7.7.1 External Clock Enable (EXTCLKEN)

External clock enable (EXTCLKEN), when set, enables the ECGON bit to be set. ECGON turns on the external clock input path through the PTE4/OSC1 pin. When EXTCLKEN is clear, ECGON cannot be set and PTE4/OSC1 will always perform the PTE4 function.

The default state for this option is clear.

7.7.2 External Crystal Enable (EXTXTALEN)

External crystal enable (EXTXTALEN), when set, will enable an amplifier to drive the PTE3/OSC2 pin from the PTE4/OSC1 pin. The amplifier will drive only if the external clock enable (EXTCLKEN) bit and the ECGON bit are also set. If EXTCLKEN or ECGON are clear, PTE3/OSC2 will perform the PTE3 function. When EXTXTALEN is clear, PTE3/OSC2 will always perform the PTE3 function.

EXTXTALEN, when set, also configures the clock monitor to expect an external clock source in the valid range of crystals (30 kHz to 100 kHz or 1 MHz to 8 MHz). When EXTXTALEN is clear, the clock monitor will expect an external clock source in the valid range for externally generated clocks when using the clock monitor (60 Hz to 32 MHz).

EXTXTALEN, when set, also configures the external clock stabilization divider in the clock monitor for a 4096 cycle timeout to allow the proper stabilization time for a crystal. When EXTXTALEN is clear, the stabilization divider is configured to 16 cycles since an external clock source does not need a startup time.

The default state for this option is clear.

7.7.3 Slow External Clock (EXTSLOW)

Slow external clock (EXTSLOW), when set, will decrease the drive strength of the oscillator amplifier, enabling low-frequency crystal operation (30 kHz–100 kHz) if properly enabled with the external clock enable (EXTCLKEN) and external crystal enable (EXTXTALEN) bits. When clear, EXTSLOW enables high-frequency crystal operation (1 MHz to 8 MHz).

EXTSLOW, when set, also configures the clock monitor to expect an external clock source that is slower than the low-frequency base clock (60 Hz to 307.2 kHz). When EXTSLOW is clear, the clock monitor will expect an external clock faster than the low-frequency base clock (307.2 kHz to 32 MHz).

The default state for this option is clear.

7.7.4 Oscillator Enable In Stop (OSCENINSTOP)

Oscillator enable in stop (OSCENINSTOP), when set, will enable the ICG to continue to generate clocks (either CGMXCLK, CGMOUT, COPCLK, or TBMCLK) in stop mode. This function is used to keep the timebase and COP running while the rest of the microcontroller stops. The clock monitor and autoswitching functions remain operative.

When OSCENINSTOP is clear, all clock generation will cease and CGMXCLK, CGMOUT, COPCLK, and TBMCLK will be forced low during stop mode. The clock monitor and autoswitching functions become inoperative.
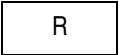
The default state for this option is clear.

7.8 Input/Output (I/O) Registers

The ICG contains five registers, summarized in **Figure 7-10**. These registers are:

1. ICG control register (ICGCR)

2. ICG multiplier register (ICGMR)

3. ICG trim register (ICGTR)

4. ICG DCO divider control register (ICGDVR)

5. ICG DCO stage control register (ICGDSR)

Several of the bits in these registers have interaction where the state of one bit may force another bit to a particular state or prevent another bit from being set or cleared. A summary of this interaction is shown in **Table 7-4**.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0036	ICG Control Register (ICGCR) See page 141.	Read:	CMIE	CMF	CMON	CS	ICGON	ICGS	ECGON	ECGS
		Write:		0*						
		Reset:	0	0	0	0	1	0	0	0

*See **7.8.1 ICG Control Register** for method of clearing the CMF bit.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0037	ICG Multiply Register (ICGMR) See page 143.	Read: Write:		N6	N5	N4	N3	N2	N1	N0
		Reset:	0	0	0	1	0	1	0	1
$0038	ICG Trim Register (ICGTR) See page 144.	Read: Write:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
		Reset:	1	0	0	0	0	0	0	0
$0039	ICG Divider Control Register (ICGDVR) See page 144.	Read:					DDIV3	DDIV2	DDIV1	DDIV0
		Write:								
		Reset:	0	0	0	0	U	U	U	U
$003A	ICG DCO Stage Control Register (ICGDSR) See page 145.	Read:	DSTG7	DSTG6	DSTG5	DSTG4	DSTG3	DSTG2	DSTG1	DSTG0
		Write:	R	R	R	R	R	R	R	R
		Reset:	U	U	U	U	U	U	U	U

= Unimplemented   R = Reserved   U = Unaffected

Figure 7-10. ICG Module I/O Register Summary

Table 7-4. ICG Module Register Bit Interaction Summary

Condition	Register Bit Results for Given Condition											
	CMIE	CMF	CMON	CS	ICGON	ICGS	ECGON	ECGS	N[6:0]	TRIM[7:0]	DDIV[3:0]	DSTG[7:0]
Reset	0	0	0	0	1	0	0	0	$15	$80	—	—
OSCENINSTOP = 0, STOP = 1	0	0	0	—	—	0	—	0	—	—	—	—
EXTCLKEN = 0	0	0	0	0	1	—	0	0	—	—	uw	uw
CMF = 1	—	(1)	1	—	1	—	1	—	uw	uw	uw	uw
CMON = 0	0	0	(0)	—	—	—	—	—	—	—	—	—
CMON = 1	—	—	(1)	—	1	—	1	—	uw	uw	uw	uw
CS = 0	—	—	—	(0)	1	—	—	—	—	—	uw	uw
CS = 1	—	—	—	(1)	—	—	1	—	—	—	—	—
ICGON = 0	0	0	0	1	(0)	0	1	—	—	—	—	—
ICGON = 1	—	—	—	—	(1)	—	—	—	—	—	uw	uw
ICGS = 0	us	—	us	uc	—	(0)	—	—	—	—	—	—
ECGON = 0	0	0	0	0	1	—	(0)	0	—	—	uw	uw
ECGS = 0	us	—	us	us	—	—	—	(0)	—	—	—	—
IOFF = 1	—	1*	(1)	1	(1)	0	(1)	—	uw	uw	uw	uw
EOFF = 1	—	1*	(1)	0	(1)	—	(1)	0	uw	uw	uw	uw
N = written	(0)	(0)	(0)	—	—	0*	—	—	—	—	—	—
TRIM = written	(0)	(0)	(0)	—	—	0*	—	—	—	—	—	—

—   Register bit is unaffected by the given condition.
0, 1   Register bit is forced clear or set (respectively) in the given condition.
0*, 1*   Register bit is temporarily forced clear or set (respectively) in the given condition.
(0), (1)   Register bit must be clear or set (respectively) for the given condition to occur.
us, uc, uw   Register bit cannot be set, cleared, or written (respectively) in the given condition.

7.8.1 ICG Control Register

The ICG control register (ICGCR) contains the control and status bits for the internal clock generator, external clock generator, and clock monitor as well as the clock select and interrupt enable bits.

Address: $0036

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CMIE	CMF	CMON	CS	ICGON	ICGS	ECGON	ECGS
Write:		0*						
Reset:	0	0	0	0	1	0	0	0

*See CMF bit description for method of clearing CMF bit.

☐ = Unimplemented

Figure 7-11. ICG Control Register (ICGCR)

CMIE — Clock Monitor Interrupt Enable Bit

This read/write bit enables clock monitor interrupts. An interrupt will occur when both CMIE and CMF are set. CMIE can be set when the CMON bit has been set for at least one cycle. CMIE is forced clear when CMON is clear or during reset.
   1 = Clock monitor interrupts enabled
   0 = Clock monitor interrupts disabled

CMF — Clock Monitor Interrupt Flag

This read-only bit is set when the clock monitor determines that either ICLK or ECLK becomes inactive and the CMON bit is set. This bit is cleared by first reading the bit while it is set, followed by writing the bit low. This bit is forced clear when CMON is clear or during reset.
   1 = Either ICLK or ECLK has become inactive.
   0 = ICLK and ECLK have not become inactive since the last read of the ICGCR, or the clock monitor is disabled.

CMON — Clock Monitor On Bit

This read/write bit enables the clock monitor. CMON can be set when both ICLK and ECLK have been on and stable for at least one bus cycle. (ICGON, ECGON, ICGS, and ECGS are all set.) CMON is

forced set when CMF is set, to avoid inadvertent clearing of CMF. CMON is forced clear when either ICGON or ECGON is clear, during stop mode with OSCENINSTOP low, or during reset.

1 = Clock monitor output enabled
0 = Clock monitor output disabled

CS — Clock Select Bit

This read/write bit determines which clock will generate the oscillator output clock (CGMXCLK). This bit can be set when ECGON and ECGS have been set for at least one bus cycle and can be cleared when ICGON and ICGS have been set for at least one bus cycle. This bit is forced set when the clock monitor determines the internal clock (ICLK) is inactive or when ICGON is clear. This bit is forced clear when the clock monitor determines that the external clock (ECLK) is inactive, when ECGON is clear, or during reset.

1 = External clock (ECLK) sources CGMXCLK
0 = Internal clock (ICLK) sources CGMXCLK

ICGON — Internal Clock Generator On Bit

This read/write bit enables the internal clock generator. ICGON can be cleared when the CS bit has been set and the CMON bit has been clear for at least one bus cycle. ICGON is forced set when the CMON bit is set, the CS bit is clear, or during reset.

1 = Internal clock generator enabled
0 = Internal clock generator disabled

ICGS — Internal Clock Generator Stable Bit

This read-only bit indicates when the internal clock generator has determined that the internal clock (ICLK) is within about 5 percent of the desired value. This bit is forced clear when the clock monitor determines the ICLK is inactive, when ICGON is clear, when the ICG multiplier register (ICGMR) is written, when the ICG TRIM register (ICGTR) is written, during stop mode with OSCENINSTOP low, or during reset.

1 = Internal clock is within 5 percent of the desired value.
0 = Internal clock may not be within 5 percent of the desired value.

ECGON — External Clock Generator On Bit

This read/write bit enables the external clock generator. ECGON can be cleared when the CS and CMON bits have been clear for at least one bus cycle. ECGON is forced set when the CMON bit or the CS bit is set. ECGON is forced clear during reset.

1 = External clock generator enabled
0 = External clock generator disabled

ECGS — External Clock Generator Stable Bit

This read-only bit indicates when at least 4096 external clock (ECLK) cycles have elapsed since the external clock generator was enabled. This is not an assurance of the stability of ECLK but is meant to provide a startup delay. This bit is forced clear when the clock monitor determines ECLK is inactive, when ECGON is clear, during stop mode with OSCENINSTOP low, or during reset.

1 = 4096 ECLK cycles have elapsed since ECGON was set.
0 = External clock is unstable, inactive, or disabled.

7.8.2 ICG Multiplier Register

Address: $0037

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:		N6	N5	N4	N3	N2	N1	N0
Reset:	0	0	0	1	0	1	0	1

    = Unimplemented
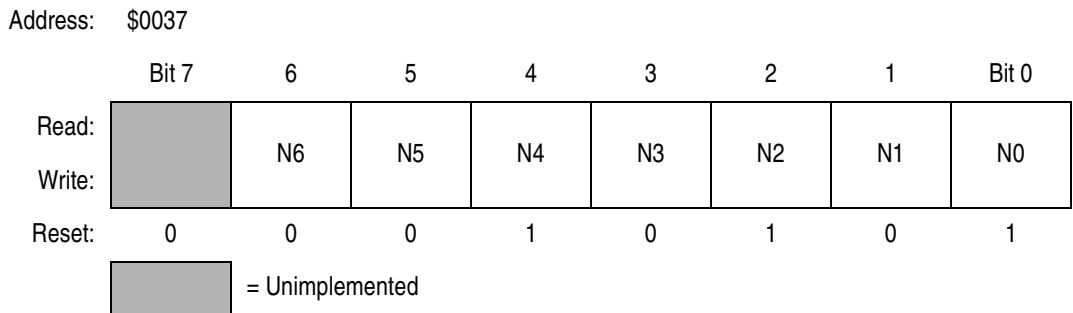
Figure 7-12. ICG Multiplier Register (ICGMR)

N6:N0 — ICG Multiplier Factor Bits

These read/write bits change the multiplier used by the internal clock generator. The internal clock (ICLK) will be:

(307.2 kHz $\pm$ 25 percent) * N

A value of $00 in this register is interpreted the same as a value of $01. This register cannot be written when the CMON bit is set. Reset sets this factor to $15 (decimal 21) for default frequency of 6.45 MHz $\pm$ 25 percent (1.613 MHz $\pm$ 25 percent bus).

7.8.3 ICG Trim Register

Address: $0038

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
Write:								
Reset:	1	0	0	0	0	0	0	0

Figure 7-13. ICG Trim Register (ICGTR)

TRIM7:TRIM0 — ICG Trim Factor Bits

These read/write bits change the size of the internal capacitor used by the internal clock generator. By testing the frequency of the internal clock and incrementing or decrementing this factor accordingly, the accuracy of the internal clock can be improved to ± 2 percent. Incrementing this register by one decreases the frequency by 0.195 percent of the unadjusted value. Decrementing this register by one increases the frequency by 0.195 percent. This register cannot be written when the CMON bit is set. Reset sets these bits to $80, centering the range of possible adjustment.

7.8.4 ICG DCO Divider Register

Address: $0039

	Bit 7	6	5	4	3	2	1	Bit 0
Read:					DDIV3	DDIV2	DDIV1	DDIV0
Write:								
Reset:	0	0	0	0	U	U	U	U

☐ = Unimplemented   U = Unaffected

Figure 7-14. ICG DCO Divider Control Register (ICGDVR)

DDIV3:DDIV0 — ICG DCO Divider Control Bits

These bits indicate the number of divide-by-twos (DDIV) that follow the digitally controlled oscillator. When ICGON is set, DDIV is controlled by the digital loop filter. The range of valid values for DDIV

is from $0 to $9. Values of $A through $F are interpreted the same as $9. Since the DCO is active during reset, reset has no effect on DSTG and the value may vary.

7.8.5 ICG DCO Stage Register

Address: $003A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DSTG7	DSTG6	DSTG5	DSTG4	DSTG3	DSTG2	DSTG1	DSTG0
Write:	R	R	R	R	R	R	R	R
Reset:	U	U	U	U	U	U	U	U

R	= Reserved	U = Unaffected

Figure 7-15. ICG DCO Stage Control Register (ICGDSR)

DSTG7:DSTG0 — ICG DCO Stage Control Bits

These bits indicate the number of stages (above the minimum) in the digitally controlled oscillator. The total number of stages is approximately equal to $1FF, so changing DSTG from $00 to $FF will approximately double the period. Incrementing DSTG will increase the period (decrease the frequency) by 0.202 percent to 0.368 percent (decrementing has the opposite effect). DSTG cannot be written when ICGON is set to prevent inadvertent frequency shifting. When ICGON is set, DSTG is controlled by the digital loop filter. Since the DCO is active during reset, reset has no effect on DSTG and the value may vary.

Section 8. Configuration Register (CONFIG)

8.1 Contents

8.2 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 ($2^{18} – 2^4$ or $2^{13} – 2^4$ COPCLK 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

- External clock, external crystal, or ICG clock source

8.3 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 8-1** and* **Figure 8-2***.*

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	EXT-XTALEN	EXT-SLOW	EXT-CLKEN	0	OSCEN-INSTOP	R
Write:								
Reset:	0	0	0	0	0	0	0	0

	= Unimplemented		R	= Reserved

Figure 8-1. 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 8-2. Configuration Register 1 (CONFIG1)

EXTXTALEN — External Crystal Enable Bit

EXTXTALEN enables the external oscillator circuits to be configured for a crystal configuration where the PTE4/OSC1 and PTE3/OSC2 pins are the connections for an external crystal.

Clearing the EXTXTALEN bit (default setting) allows the PTE3/OSC2 pin to function as a general-purpose I/O pin. Refer to **Table 8-1** for configuration options for the external source. See **Section 7. Internal Clock Generator (ICG) Module** for a more detailed description of the external clock operation.

EXTXTALEN, when set, also configures the clock monitor to expect an external clock source in the valid range of crystals (30 kHz to 100 kHz or 1 MHz to 8 MHz). When EXTXTALEN is clear, the clock

monitor will expect an external clock source in the valid range for externally generated clocks when using the clock monitor (60 Hz to 32 MHz).

EXTXTALEN, when set, also configures the external clock stabilization divider in the clock monitor for a 4096-cycle timeout to allow the proper stabilization time for a crystal. When EXTXTALEN is clear, the stabilization divider is configured to 16 cycles since an external clock source does not need a startup time.

    1 = Allows PTE3/OSC2 to be an external crystal connection.
    0 = PTE3/OSC2 functions as an I/O port pin (default).

EXTSLOW — Slow External Crystal Enable Bit

The EXTSLOW bit has two functions. It configures the ICG module for a fast (1 MHz to 8 MHz) or slow (30 kHz to 100 kHz) speed crystal. The option also configures the clock monitor operation in the ICG module to expect an external frequency higher (307.2 kHz to 32 MHz) or lower (60 Hz to 307.2 kHz) than the base frequency of the internal oscillator. See **Section 7. Internal Clock Generator (ICG) Module**.

    1 = ICG set for slow external crystal operation
    0 = ICG set for fast external crystal operation

NOTE: *This bit does not function without setting the EXTCLKEN bit also.*

EXTCLKEN — External Clock Enable Bit

EXTCLKEN enables an external clock source or crystal/ceramic resonator to be used as a clock input. Setting this bit enables PTE4/OSC1 pin to be a clock input pin. Clearing this bit (default setting) allows the PTE4/OSC1 and PTE3/OSC2 pins to function as a general-purpose input/output (I/O) pin. Refer to **Table 8-1** for configuration options for the external source. See **Section 7. Internal Clock Generator (ICG) Module** for a more detailed description of the external clock operation.

    1 = Allows PTE4/OSC1 to be an external clock connection
    0 = PTE4/OSC1 and PTE3/OSC2 function as I/O port pins
       (default).

Table 8-1. External Clock Option Settings

External Clock Configuration Bits		Pin Function		Description
EXTCLKEN	**EXTXTALEN**	**PTE4/OSC1**	**PTE3/OSC2**	**Description**
0	0	PTE4	PTE3	Default setting — external oscillator disabled
0	1	PTE4	PTE3	External oscillator disabled since EXTCLKEN not set
1	0	OSC1	PTE3	External oscillator configured for an external clock source input (square wave) on OSC1
1	1	OSC1	OSC2	External oscillator configured for an external crystal configuration on OSC1 and OSC2. System will also operate with square-wave clock source in OSC1.

OSCENINSTOP — Oscillator Enable In Stop Mode Bit

OSCENINSTOP, when set, will enable the internal clock generator module to continue to generate clocks (either internal, ICLK, or external, ECLK) in stop mode. See **Section 7. Internal Clock Generator (ICG) Module**. This function is used to keep the timebase running while the rest of the microcontroller stops. See **Section 21. Timebase Module (TBM)**. When clear, all clock generation will cease and both ICLK and ECLK will be forced low during stop mode. The default state for this option is clear, disabling the ICG in stop mode.
1 = Oscillator enabled to operate during stop mode
0 = Oscillator disabled during stop mode (default)

NOTE: *This bit has the same functionality as the OSCSTOPENB CONFIG bit in MC68HC908GP32 and MC68HC908GR8 parts.*

COPRS — COP Rate Select Bit

COPD selects the COP timeout period. Reset clears COPRS. See **Section 9. Computer Operating Properly (COP) Module**
1 = COP timeout period = $2^{13} - 2^4$ COPCLK cycles
0 = COP timeout period = $2^{18} - 2^4$ COPCLK cycles

LVISTOP — LVI Enable in Stop Mode Bit

When the LVIPWRD bit is clear, 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 **Section 14. Low-Voltage Inhibit (LVI)**.
   1 = LVI module resets disabled
   0 = LVI module resets enabled

LVIPWRD — LVI Power Disable Bit

LVIPWRD disables the LVI module. See **Section 14. Low-Voltage
Inhibit (LVI)**.
   1 = LVI module power disabled
   0 = LVI module power enabled

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

LVI5OR3 selects the voltage operating mode of the LVI module. See
Section 14. Low-Voltage Inhibit (LVI) The voltage mode selected
for the LVI should match the operating V_{DD}. See **Section 23.
Electrical Specifications** for the LVI's voltage trip points for each of
the modes.
   1 = LVI operates in 5-V mode.
   0 = LVI operates in 3-V mode.

NOTE: *The LVI5OR3 bit is cleared by a power-on reset (POR) only. Other
resets will leave this bit unaffected.*

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 selected is the internal oscillator or the external crystal and the OSCENINSTOP 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 thus the SSREC bit should not be set.

When using the LVI during normal operation but disabling during stop mode, the LVI will have an enable time of t_{en}. 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 **Section 9. Computer Operating Properly (COP) Module**.
   1 = COP module disabled
   0 = COP module enabled

Section 9. Computer Operating Properly (COP) Module

9.1 Contents

9.2 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.

9.3 Functional Description

Figure 9-1 shows the structure of the COP module.
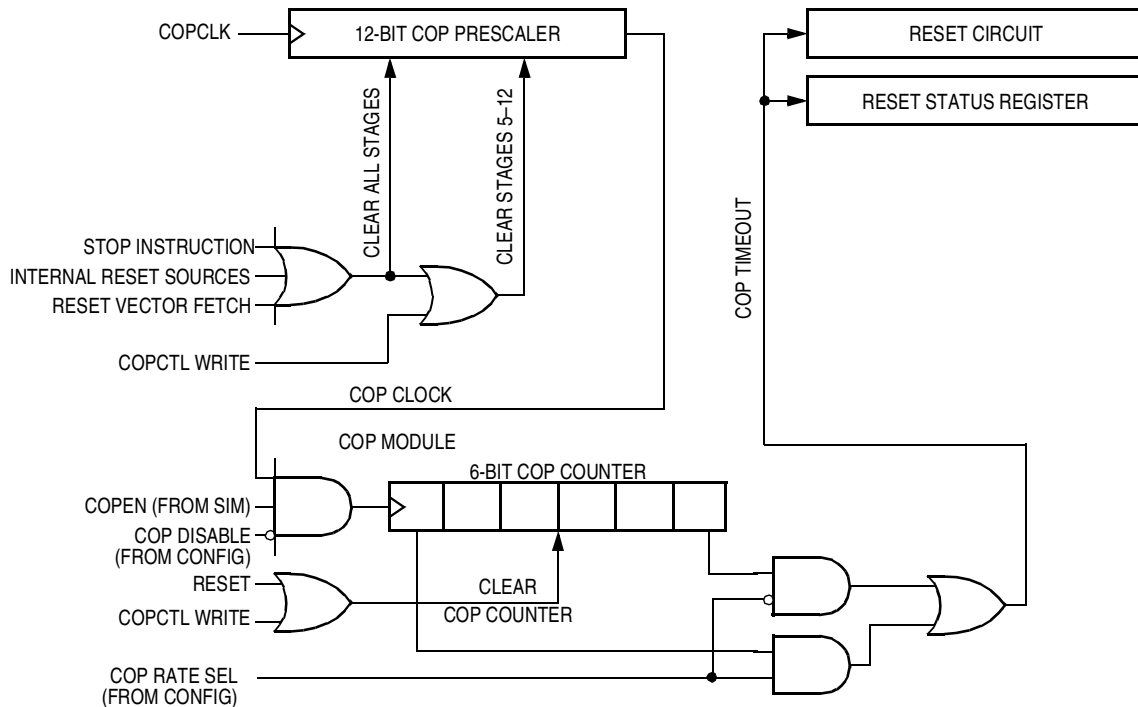
Figure 9-1. COP Block Diagram

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 $2^{18} - 2^4$ or $2^{13} - 2^4$ COPCLK cycles, depending on the state of the COP rate select bit, COPRS, in the configuration register. With a $2^{13} - 2^4$ COPCLK cycle overflow option, a 32.768-kHz crystal gives a COP timeout period of 250 ms. Writing any value to location $FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12 through 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 $\overline{RST}$ pin low for 32 COPCLK cycles and sets the COP bit in the reset status register (RSR).

In monitor mode, the COP is disabled if the $\overline{RST}$ pin or the $\overline{IRQ1}$ is held at V_{TST}. During the break state, V_{TST} on the $\overline{RST}$ 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.*

9.4 I/O Signals

The following paragraphs describe the signals shown in **Figure 9-1**.

9.4.1 COPCLK

COPCLK is a clock generated by the clock selection circuit in the internal clock generator (ICG). See **7.4.5 Clock Selection Circuit** for more details.

9.4.2 STOP Instruction

The STOP instruction clears the COP prescaler.

9.4.3 COPCTL Write

Writing any value to the COP control register (COPCTL) (see **9.5 COP Control Register**) clears the COP counter and clears bits 12 through 5 of the prescaler. Reading the COP control register returns the low byte of the reset vector.

9.4.4 Power-On Reset

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

9.4.5 Internal Reset

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

9.4.6 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the COP prescaler.

9.4.7 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the configuration register. See **Section 8. Configuration Register (CONFIG)**.

9.4.8 COPRS (COP Rate Select)

The COPRS signal reflects the state of the COP rate select bit (COPRS) in the configuration register. See **Section 8. Configuration Register (CONFIG)**.

9.5 COP Control Register

The COP control register 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							
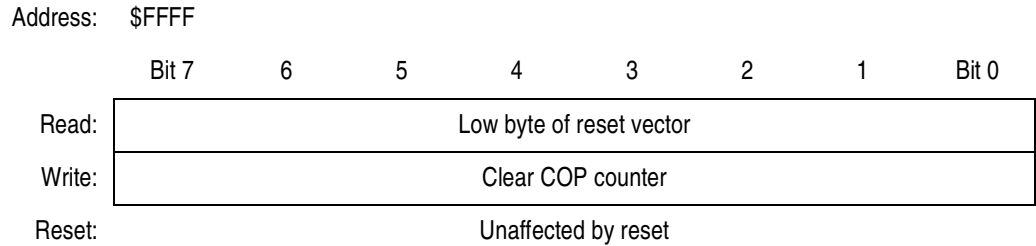
Figure 9-2. COP Control Register (COPCTL)

9.6 Interrupts

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

9.7 Monitor Mode

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

9.8 Low-Power Modes

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

9.8.1 Wait Mode

The COP remains active during wait mode. To prevent a COP reset during wait mode, periodically clear the COP counter in a CPU interrupt routine.

9.8.2 Stop Mode

Stop mode turns off the COPCLK 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 is disabled, execution of a STOP instruction results in an illegal opcode reset.

9.9 COP Module During Break Mode

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

Section 10. Central Processor Unit (CPU)

10.1 Contents

10.2 Introduction

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

10.3 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

10.4 CPU Registers

Figure 10-1 shows the five CPU registers. CPU registers are not part of the memory map.
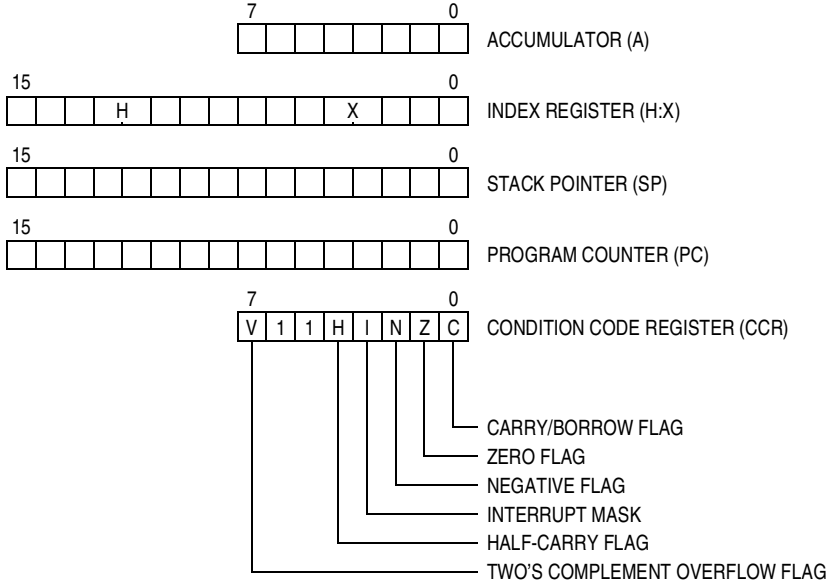
Figure 10-1. CPU Registers

10.4.1 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.
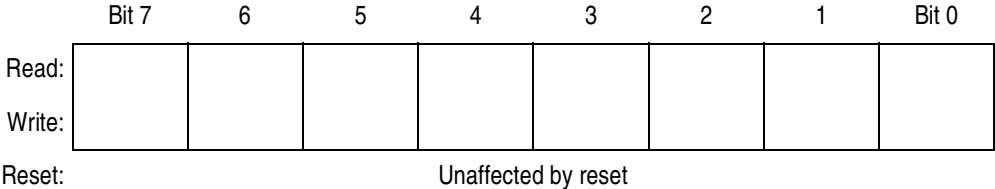
Figure 10-2. Accumulator (A)

10.4.2 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
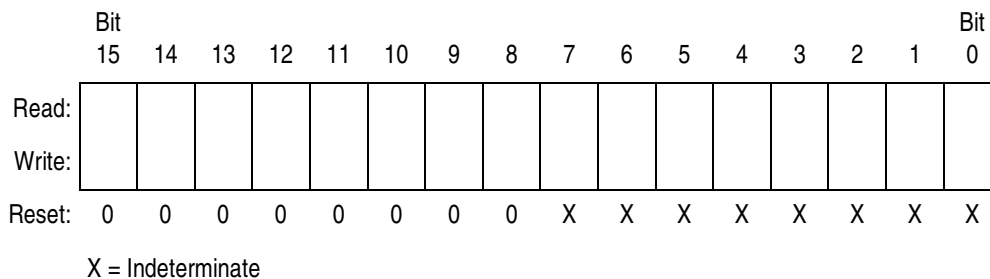
Figure 10-3. Index Register (H:X)

10.4.3 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
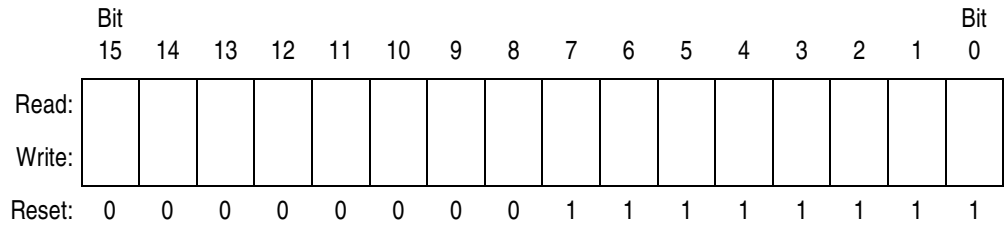
Figure 10-4. Stack Pointer (SP)

NOTE: *The location of the stack is arbitrary and may be relocated anywhere in 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.*

10.4.4 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											
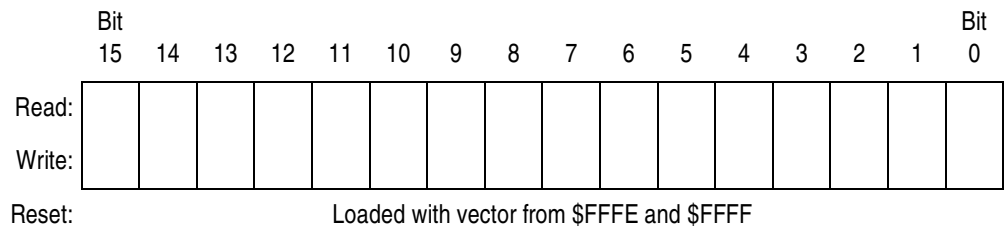
Figure 10-5. Program Counter (PC)

10.4.5 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 logic 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 10-6. 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

10.5 Arithmetic/Logic Unit (ALU)

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

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

10.6 Low-Power Modes

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

10.6.1 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

10.6.2 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.

10.7 CPU During Break Interrupts

If a break module is present on the MCU, the CPU starts a break interrupt by:
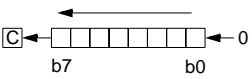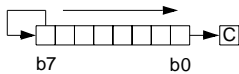
- Loading the instruction register with the SWI instruction

- 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.

10.8 Instruction Set Summary

Table 10-1. Instruction Set Summary (Sheet 1 of 8)

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

Table 10-1. Instruction Set Summary (Sheet 2 of 8)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
BCS rel	Branch if Carry Bit Set (Same as BLO)	$PC \leftarrow (PC) + 2 + rel$? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ rel	Branch if Equal	$PC \leftarrow (PC) + 2 + rel$? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE opr	Branch if Greater Than or Equal To (Signed Operands)	$PC \leftarrow (PC) + 2 + rel$? (N $\oplus$ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT opr	Branch if Greater Than (Signed Operands)	$PC \leftarrow (PC) + 2 + rel$? (Z) | (N $\oplus$ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC rel	Branch if Half Carry Bit Clear	$PC \leftarrow (PC) + 2 + rel$? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS rel	Branch if Half Carry Bit Set	$PC \leftarrow (PC) + 2 + rel$? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI rel	Branch if Higher	$PC \leftarrow (PC) + 2 + rel$? (C) | (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS rel	Branch if Higher or Same (Same as BCC)	$PC \leftarrow (PC) + 2 + rel$? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH rel	Branch if $\overline{IRQ}$ Pin High	$PC \leftarrow (PC) + 2 + rel$? $\overline{IRQ}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL rel	Branch if $\overline{IRQ}$ Pin Low	$PC \leftarrow (PC) + 2 + rel$? $\overline{IRQ}$ = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #opr BIT opr BIT opr BIT opr,X BIT opr,X BIT ,X BIT opr,SP BIT opr,SP	Bit Test	(A) & (M)	0	–	–	$\updownarrow$	$\updownarrow$	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A5 B5 C5 D5 E5 F5 9EE5 9ED5	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
BLE opr	Branch if Less Than or Equal To (Signed Operands)	$PC \leftarrow (PC) + 2 + rel$? (Z) | (N $\oplus$ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO rel	Branch if Lower (Same as BCS)	$PC \leftarrow (PC) + 2 + rel$? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS rel	Branch if Lower or Same	$PC \leftarrow (PC) + 2 + rel$? (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT opr	Branch if Less Than (Signed Operands)	$PC \leftarrow (PC) + 2 + rel$? (N $\oplus$ V) = 1	–	–	–	–	–	–	REL	91	rr	3
BMC rel	Branch if Interrupt Mask Clear	$PC \leftarrow (PC) + 2 + rel$? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI rel	Branch if Minus	$PC \leftarrow (PC) + 2 + rel$? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS rel	Branch if Interrupt Mask Set	$PC \leftarrow (PC) + 2 + rel$? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE rel	Branch if Not Equal	$PC \leftarrow (PC) + 2 + rel$? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL rel	Branch if Plus	$PC \leftarrow (PC) + 2 + rel$? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA rel	Branch Always	$PC \leftarrow (PC) + 2 + rel$	–	–	–	–	–	–	REL	20	rr	3
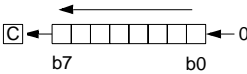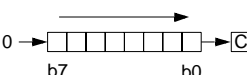
Table 10-1. Instruction Set Summary (Sheet 3 of 8)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 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) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n*,*opr*	Set Bit *n* in M	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 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* CBEQA #*opr*,*rel* CBEQX #*opr*,*rel* CBEQ *opr*,X+,*rel* CBEQ X+,*rel* CBEQ *opr*,SP,*rel*	Compare and Branch if Equal	PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 3 + rel ? (X) – (M) = $00 PC ← (PC) + 3 + rel ? (A) – (M) = $00 PC ← (PC) + 2 + rel ? (A) – (M) = $00 PC ← (PC) + 4 + rel ? (A) – (M) = $00	–	–	–	–	–	–	DIR IMM IMM IX1+ IX+ SP1	31 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 4 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask	I ← 0	–	–	0	–	–	–	INH	9A		2
CLR *opr* CLRA CLRX CLRH CLR *opr*,X CLR ,X CLR *opr*,SP	Clear	M ← $00 A ← $00 X ← $00 H ← $00 M ← $00 M ← $00 M ← $00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	3 1 1 1 3 2 4
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X CMP *opr*,SP CMP *opr*,SP	Compare A with M	(A) – (M)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A1 B1 C1 D1 E1 F1 9EE1 9ED1	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
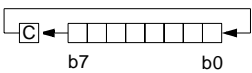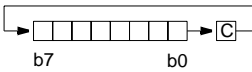
Table 10-1. Instruction Set Summary (Sheet 4 of 8)

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

Table 10-1. Instruction Set Summary (Sheet 5 of 8)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
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)$_{Destination}$ ← (M)$_{Source}$ 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

Table 10-1. Instruction Set Summary (Sheet 6 of 8)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
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
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

Table 10-1. Instruction Set Summary (Sheet 7 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
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 IRQ Pin; Stop Oscillator	I ← 0; Stop Oscillator	–	–	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
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

Table 10-1. Instruction Set Summary (Sheet 8 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
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; Stop Processor	I bit ← 0	–	–	0	–	–	–	INH	8F		1

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

10.9 Opcode Map

See **Table 10-2**.

Table 10-2. Opcode Map

Each cell format: cycles / mnemonic / bytes / addressing mode.

	Bit Manip. DIR (0)	Bit Manip. DIR (1)	Branch REL (2)	RMW DIR (3)	RMW INH (4)	RMW INH (5)	RMW IX1 (6)	RMW SP1 (9E6)	RMW IX (7)	Control INH (8)	Control INH (9)	Reg/Mem IMM (A)	Reg/Mem DIR (B)	Reg/Mem EXT (C)	Reg/Mem IX2 (D)	Reg/Mem SP2 (9ED)	Reg/Mem IX1 (E)	Reg/Mem SP1 (9EE)	Reg/Mem IX (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	5 NSA 1 INH		3 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 IMM	3 CPHX 3 IMM		4 CPHX 2 DIR	2 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	5 TST 3 DIR	1 TSTA 1 INH	1 TSTX 1 INH	3 TST 3 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		4 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	4 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

Legend:

INH	Inherent	REL	Relative
IMM	Immediate	IX	Indexed, No Offset
DIR	Direct	IX1	Indexed, 8-Bit Offset
EXT	Extended	IX2	Indexed, 16-Bit Offset
DD	Direct-Direct	IMD	Immediate-Direct
IX+D	Indexed-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

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

Section 11. FLASH Memory

11.1 Contents

11.2 Introduction

This section 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. It is recommended that the user utilize the FLASH programming routines provided in the on-chip ROM, which are described more fully in a separate Motorola application note.

11.3 Functional Description

The FLASH memory is an array of 15,872 bytes (7,680 bytes on MC68HC908GT8) with an additional 36 bytes of user vectors, one byte of block protection and two bytes of ICG user trim storage. *An erased bit reads as logic 1 and a programmed bit reads as a logic 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:

- $C000–$FDFF; user memory ($E000–$FDFF on MC68HC908GT8)

- $FE08; FLASH control register

- $FF7E; FLASH block protect register

- $FF80; ICG user trim register (ICGTR5)

- $FF81; ICG user trim register (ICGTR3)

- $FFDC–$FFFF; these locations are reserved for user-defined interrupt and reset vectors

Programming tools are available from Motorola. Contact your local Motorola representative for more information.

NOTE: *A security feature prevents viewing of the FLASH contents.[1]*

---

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

11.4 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:					HVEN	MASS	ERASE	PGM
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 11-1. 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 16Kbyte FLASH array for mass erase operation.

   1 = MASS erase operation selected
   0 = MASS erase operation unselected

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

11.5 FLASH Page Erase Operation

Use this step-by-step procedure to erase a page (64 bytes) of FLASH memory to read as logic 1:

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 address within the page address range desired.

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)

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 again in read mode.

NOTE: *While these operations must be performed in the order shown, other unrelated operations may occur between the steps.*

11.6 FLASH Mass Erase Operation

Use this step-by-step procedure to erase entire FLASH memory to read as logic 1:

1. Set both the ERASE bit, and the MASS bit in the FLASH control register.

2. Read from 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 bit.

8. Wait for a time, t_{NVHL} (minimum 100 µs)

9. Clear the HVEN bit.

10. After a time, t_{RCV} (minimum 1 µs), the memory can be accessed again in read mode.

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

---

1. When in monitor mode, with security sequence failed (see **15.5 Security**), write to the FLASH block protect register instead of any FLASH address.

11.7 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. Use this step-by-step procedure to program a row of FLASH memory (**Figure 11-2** is a flowchart representation):

NOTE: *In order to avoid program disturbs, the row must be erased before any byte on that row is 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 from 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.[1]

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} (minimum 1 μs), the memory can be accessed in read mode again.

---

1. The time between each FLASH address change, or the time between the last FLASH address programmed to clearing PGM bit, must not exceed the maximum programming time, t_{PROG} maximum.

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

NOTE: *Programming and erasing of FLASH locations cannot be performed by code being executed from the FLASH memory. While these operations must be performed in the order shown, other unrelated operations may occur between the steps. Do not exceed t_{PROG} maximum. See* **23.20 Memory Characteristics**.

11.8 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 **11.8.1 FLASH Block Protect Register**. Once the FLBPR is programmed with a value other than $FF, any erase or program of the FLBPR or the protected block of FLASH memory is prohibited. The FLBPR itself can be erased or programmed only with an external voltage, V_{TST}, present on the $\overline{IRQ}$ pin. This voltage also allows entry from reset into the monitor mode.

Algorithm for programming
a row (32 bytes) of FLASH memory

1 — SET PGM BIT

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}

5 — SET HVEN BIT

6 — WAIT FOR A TIME, t_{PGS}

7 — WRITE DATA TO THE FLASH ADDRESS TO BE PROGRAMMED

8 — WAIT FOR A TIME, t_{PROG}

COMPLETED PROGRAMMING THIS ROW? — Y / N

10 — CLEAR PGM BIT

11 — WAIT FOR A TIME, t_{NVH}

12 — CLEAR HVEN BIT

13 — WAIT FOR A TIME, t_{RCV}

END OF PROGRAMMING

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 11-2. FLASH Programming Flowchart

11.8.1 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.
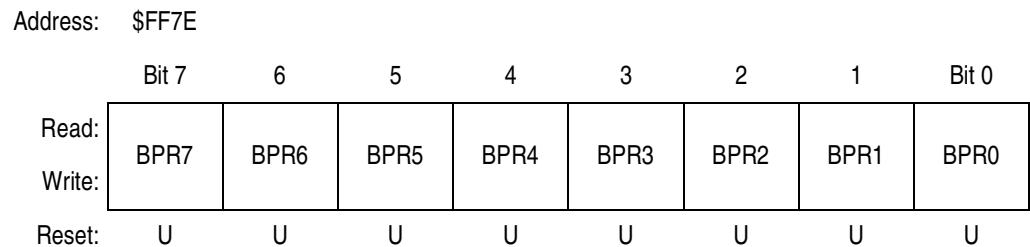
Address: $FF7E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	BPR7	BPR6	BPR5	BPR4	BPR3	BPR2	BPR1	BPR0
Reset:	U	U	U	U	U	U	U	U

U = Unaffected by reset. Initial value from factory is 1.
Write to this register is by a programming sequence to the FLASH memory.

Figure 11-3. 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 logic 1s and bits [5:0] are logic 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.

16-BIT MEMORY ADDRESS

START ADDRESS OF FLASH
BLOCK PROTECT
| 1 | 1 | FLBPR VALUE | 0 | 0 | 0 | 0 | 0 | 0 |
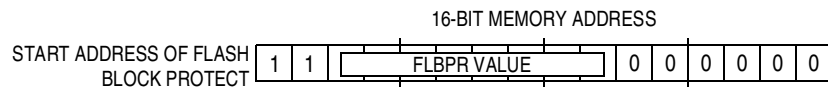
Figure 11-4. FLASH Block Protect Start Address

Examples of protect start address:

BPR[7:0]	Start of Address of Protect Range
$00	The entire FLASH memory is protected.
$01 (**0000 0001**)	$C040 (11**00 0000 01**00 0000)
$02 (**0000 0010**)	$C080 (11**00 0000 10**00 0000)
and so on...	
$FE (**1111 1110**)	$FF80 (11**11 1111 10**00 0000)
$FF	The entire FLASH memory is not protected.

Note: The end address of the protected range is always $FFFF.

11.8.2 ICG User Trim Registers (ICGTR5 and ICGTR3)

The ICG user trim register are two normal bytes of FLASH memory which are allocated for the user to store copies of the ICG trim register (ICGTR) value. ICGTR5 is allocated for storage of the trim value when a 5-V supply is used, ICGTR3 for storage of the trim value when a 3-V supply is used. Representative trim values are programmed into these locations by Motorola but they may be erased and reprogrammed by the user at any time.

Storage and retrieval of data in these registers is not automatic and must be performed programmatically. Typically, these locations are programmed by the user during an in-system calibration procedure and one of them, depending on the application supply voltage, is subsequently used by the user's initialization code to configure the ICG each time following a reset.

ICGTR5 is used by the MC68HC908GT16 monitor ROM program during its initialization sequence if monitor mode was entered while clocking from the ICG. If the contents of ICGTR5 are not $FF then the contents are copied to ICGTR.

NOTE: *The contents of ICGTR3 are not utilized by the monitor ROM program.*

Address: ICGTR5, $FF80 and ICGTR3, $FF81

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
Write:								
Reset:	U	U	U	U	U	U	U	U

U = Unaffected by reset. Initial value from factory is 1.
Write to this register is by a programming sequence to the FLASH memory.

Figure 11-5. ICG User Trim Registers (ICGTR5 and ICGTR3)

TRIM[7:0] — ICG Trim Factor Bits

These bits are copied by the monitor ROM program following a reset, if monitor mode was entered while clocking from the ICG and may be copied by the user's initialization code to the ICG trim register (ICGTR).

11.9 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.

11.10 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.

FLASH Memory

Section 12. External Interrupt (IRQ)

12.1 Contents

12.2 Introduction

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

12.3 Features

Features of the IRQ module include:

- A dedicated external interrupt pin ($\overline{\text{IRQ1}}$)

- IRQ interrupt control bits

- Hysteresis buffer

- Programmable edge-only or edge and level interrupt sensitivity

- Automatic interrupt acknowledge

- Internal pullup resistor

12.4 Functional Description

A logic 0 applied to the external interrupt pin can latch a central processor unit (CPU) interrupt request. **Figure 12-1** shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ1}}$ 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 logic 1 to the ACK bit clears the IRQ latch.

- Reset — A reset automatically clears the interrupt latch.
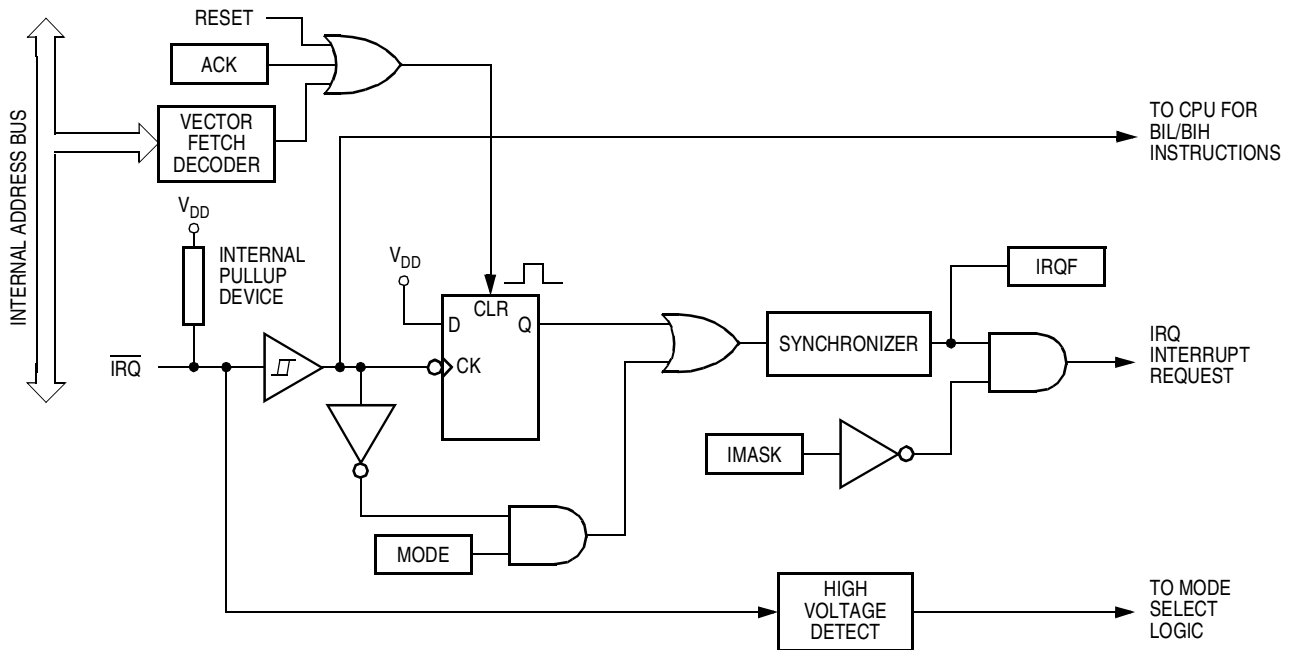
Figure 12-1. IRQ Module Block Diagram

The external interrupt pin is falling-edge triggered 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 IRQ1 pin.

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

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

- Vector fetch or software clear
- Return of the interrupt pin to logic 1

The vector fetch or software clear may occur before or after the interrupt pin returns to logic 1. 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 mask 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 193.	Read:	0	0	0	0	IRQF	0	IMASK	MODE
		Write:						ACK		
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 12-2. IRQ I/O Register Summary

12.5 $\overline{IRQ1}$ Pin

A logic 0 on the $\overline{IRQ1}$ 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 $\overline{IRQ1}$ 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 logic 1 to the ACK bit in the interrupt status and control register (INTSCR). The ACK bit is useful in applications that poll the $\overline{IRQ1}$ 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 $\overline{IRQ1}$ pin. A falling edge that occurs after writing to the ACK bit 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 $\overline{IRQ1}$ pin to logic 1 — As long as the $\overline{IRQ1}$ pin is at logic 0, IRQ remains active.

The vector fetch or software clear and the return of the $\overline{IRQ1}$ pin to logic 1 may occur in any order. The interrupt request remains pending as long as the $\overline{IRQ1}$ pin is at logic 0. 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 $\overline{IRQ1}$ 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 $\overline{IRQ1}$ pin.

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

12.6 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 **Section 6. Break Module (BRK)**.

To allow software to clear the IRQ latch during a break interrupt, write a logic 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 logic 0 to the BCFE bit. With BCFE at logic 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.

12.7 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 $\overline{IRQ1}$ interrupt pin

Address: $001D

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

☐ = Unimplemented

Figure 12-3. 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 logic 1 to this write-only bit clears the IRQ latch. ACK always reads as logic 0. Reset clears ACK.

IMASK — IRQ Interrupt Mask Bit

Writing a logic 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 $\overline{\text{IRQ1}}$ pin. Reset clears MODE.
1 = $\overline{\text{IRQ1}}$ interrupt requests on falling edges and low levels
0 = $\overline{\text{IRQ1}}$ interrupt requests on falling edges only

Section 13. Keyboard Interrupt Module (KBI)

13.1 Contents

13.2 Introduction

The keyboard interrupt module (KBI) provides eight independently maskable external interrupts which are accessible via PTA0–PTA7. When a port pin is enabled for keyboard interrupt function, an internal pullup device is also enabled on the pin.

13.3 Features

Features include:

- Eight 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)

13.4 Functional Description

Writing to the KBIE7–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 logic 0 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.

Figure 13-1. Keyboard Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
	Keyboard Status	Read:	0	0	0	0	KEYF	0	IMASKK	MODEK
$001A	and Control Register (INTKBSCR)	Write:						ACKK		
	See page 201.	Reset:	0	0	0	0	0	0	0	0
	Keyboard Interrupt Enable	Read:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
$001B	Register (INTKBIER)	Write:								
	See page 202.	Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 13-2. 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 logic 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 logic 1 — As long as any enabled keyboard interrupt pin is at logic 0, the keyboard interrupt remains set.

The vector fetch or software clear and the return of all enabled keyboard interrupt pins to logic 1 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 logic 0.

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 logic 0 for software to read the pin.*

13.5 Keyboard Initialization

When a keyboard interrupt pin is enabled, it takes time for the internal pullup to reach a logic 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 logic 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.

13.6 Low-Power Modes

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

13.6.1 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.

13.6.2 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.

13.7 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 SIM break flag control register (SBFCR) enables software to clear status bits during the break state.
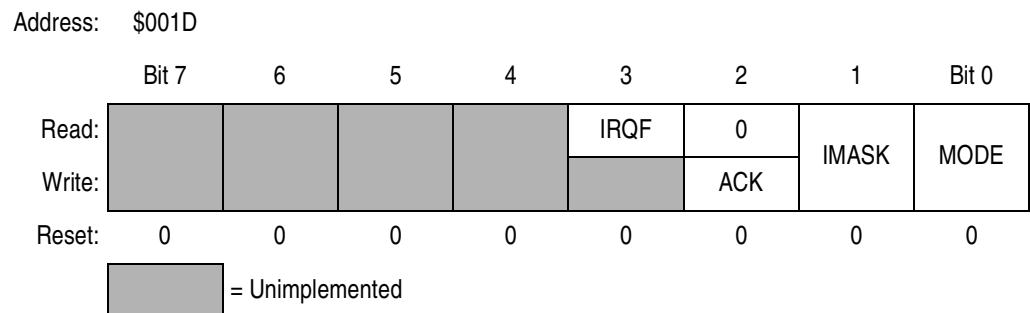
To allow software to clear the keyboard interrupt latch during a break interrupt, write a logic 1 to the BCFE bit. If a latch is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect the latch during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 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 **13.8.1 Keyboard Status and Control Register**.

13.8 I/O Registers

These registers control and monitor operation of the keyboard module:

- Keyboard status and control register (INTKBSCR)

- Keyboard interrupt enable register (INTKBIER)

13.8.1 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 13-3. Keyboard Status and Control Register (INTKBSCR)

Bits 7–4 — Not used

These read-only bits always read as logic 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 logic 1 to this write-only bit clears the keyboard interrupt request. ACKK always reads as logic 0. Reset clears ACKK.

IMASKK — Keyboard Interrupt Mask Bit

Writing a logic 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

13.8.2 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:	KBIE7	KBIE6	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
Reset:	0	0	0	0	0	0	0	0

Figure 13-4. Keyboard Interrupt Enable Register (INTKBIER)

KBIE7–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

Section 14. Low-Voltage Inhibit (LVI)

14.1 Contents

14.2 Introduction

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

14.3 Features

Features of the LVI module include:

- Programmable LVI reset
- Selectable LVI trip voltage
- Programmable stop mode operation

14.4 Functional Description

Figure 14-1 shows the structure of the LVI module. The LVI is enabled out of reset. The LVI module contains a bandgap reference circuit and comparator. Clearing the LVI power disable bit, LVIPWRD, enables the LVI to monitor V_{DD} voltage. Clearing the LVI reset disable bit, LVIRSTD, enables the LVI module to generate a reset when V_{DD} falls below a voltage, V_{TRIPF}. Setting the LVI enable in stop mode bit, LVISTOP, enables the LVI to operate in stop mode. Setting the LVI 5-V or 3-V trip point bit, LVI5OR3, enables the trip point voltage, V_{TRIPF}, to be configured for 5-V operation. Clearing the LVI5OR3 bit enables the trip point voltage, V_{TRIPF}, to be configured for 3-V operation. The actual trip points are shown in **Section 23. Electrical Specifications**.

NOTE: *After a power-on reset (POR) the LVI's default mode of operation is 3 V. If a 5-V system is used, the user must set the LVI5OR3 bit to raise the trip point to 5-V operation. Note that this must be done after every power-on reset since the default will revert back to 3-V mode after each power-on reset. If the V_{DD} supply is below the 5-V mode trip voltage but above the 3-V mode trip voltage when POR is released, the part will operate because V_{TRIPF} defaults to 3-V mode after a POR. So, in a 5-V system care must be taken to ensure that V_{DD} is above the 5-V mode trip voltage after POR is released.*

NOTE: *If the user requires 5-V mode and sets the LVI5OR3 bit after a power-on reset while the V_{DD} supply is not above the V_{TRIPR} for 5-V mode, the microcontroller unit (MCU) will immediately go into reset. The LVI in this case will hold the part in reset until either V_{DD} goes above the rising 5-V trip point, V_{TRIPR}, which will release reset or V_{DD} decreases to*

approximately 0 V which will re-trigger the power-on reset and reset the trip point to 3-V operation.

LVISTOP, LVIPWRD, LVI5OR3, and LVIRSTD are in the configuration register (CONFIG1). See **Figure 8-2. 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 **19.4.2.5 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 $\overline{RST}$ pin low to provide low-voltage protection to external peripheral devices.
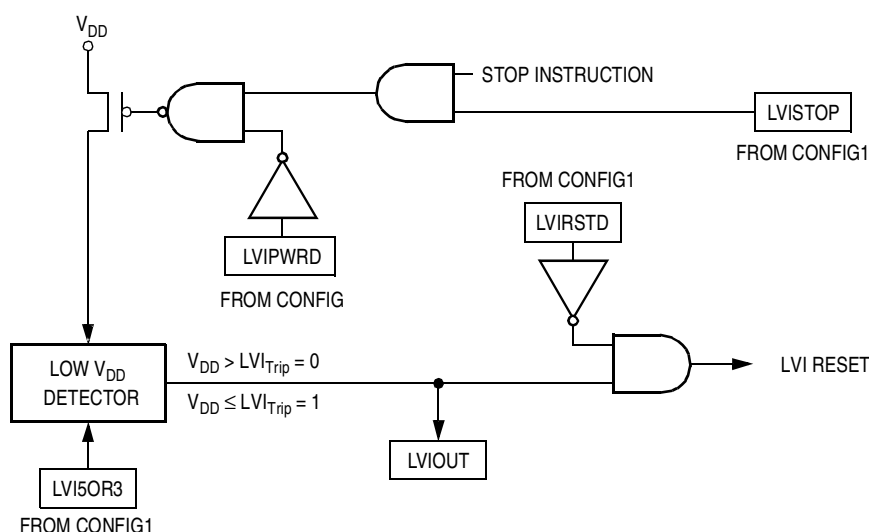
Figure 14-1. LVI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
	LVI Status Register (LVISR) See page 207.	Read:	LVIOUT	0	0	0	0	0	0	0
$FE0C		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 14-2. LVI I/O Register Summary

14.4.1 Polled LVI Operation

In applications that can operate at V_{DD} levels below the V_{TRIPF} level, software can monitor V_{DD} by polling the LVIOUT bit. In the configuration register, the LVIPWRD bit must be at logic 0 to enable the LVI module, and the LVIRSTD bit must be at logic 1 to disable LVI resets.

14.4.2 Forced Reset Operation

In applications that require V_{DD} to remain above the V_{TRIPF} level, enabling LVI resets allows the LVI module to reset the MCU when V_{DD} falls below the V_{TRIPF} level. In the configuration register, the LVIPWRD and LVIRSTD bits must be at logic 0 to enable the LVI module and to enable LVI resets.

14.4.3 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}.

14.4.4 LVI Trip Selection

The LVI5OR3 bit in the configuration register selects whether the LVI is configured for 5-V or 3-V protection.

NOTE: *The microcontroller is guaranteed to operate at a minimum supply voltage. The trip point (V_{TRIPF} [5 V] or V_{TRIPF} [3 V]) may be lower than this. (See **Section 23. Electrical Specifications** for the actual trip point voltages.)*

14.5 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 14-3. 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 14-1**). Reset clears the LVIOUT bit.

Table 14-1. LVIOUT Bit Indication

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

14.6 LVI Interrupts

The LVI module does not generate interrupt requests.

14.7 Low-Power Modes

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

14.7.1 Wait Mode

If enabled, the 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.

14.7.2 Stop Mode

If enabled in stop mode (LVISTOP 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.

Section 15. Monitor ROM (MON)

15.1 Contents

15.2 Introduction

This section describes the monitor ROM (MON) and the monitor mode entry methods. The monitor ROM allows complete testing of the 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.

15.3 Features

Features of the monitor ROM include:

- Normal user-mode pin functionality

- One pin dedicated to serial communication between monitor read-only memory (ROM) and host computer

- Standard mark/space non-return-to-zero (NRZ) communication with host computer

- Execution of code in random-access memory (RAM) or FLASH

- FLASH memory security feature[1]

- FLASH memory programming interface

- External 4.92MHz or 9.83MHz clock used to generate internal frequency of 2.4576 MHz

- Optional ICG mode of operation (no external clock or high voltage)

- 304 bytes monitor ROM code size ($FE20 to $FF4F)

- Monitor mode entry without high voltage, V_{TST}, if reset vector is blank ($FFFE and $FFFF contain $FF)

- Standard monitor mode entry if high voltage is applied to $\overline{IRQ}$

15.4 Functional Description

The monitor ROM receives and executes commands from a host computer. **Figure 15-1**, **Figure 15-2**, and **Figure 15-3** 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.

---

1. No security feature is absolutely secure. However, Motorola's strategy is to make reading or copying the FLASH difficult for unauthorized users.
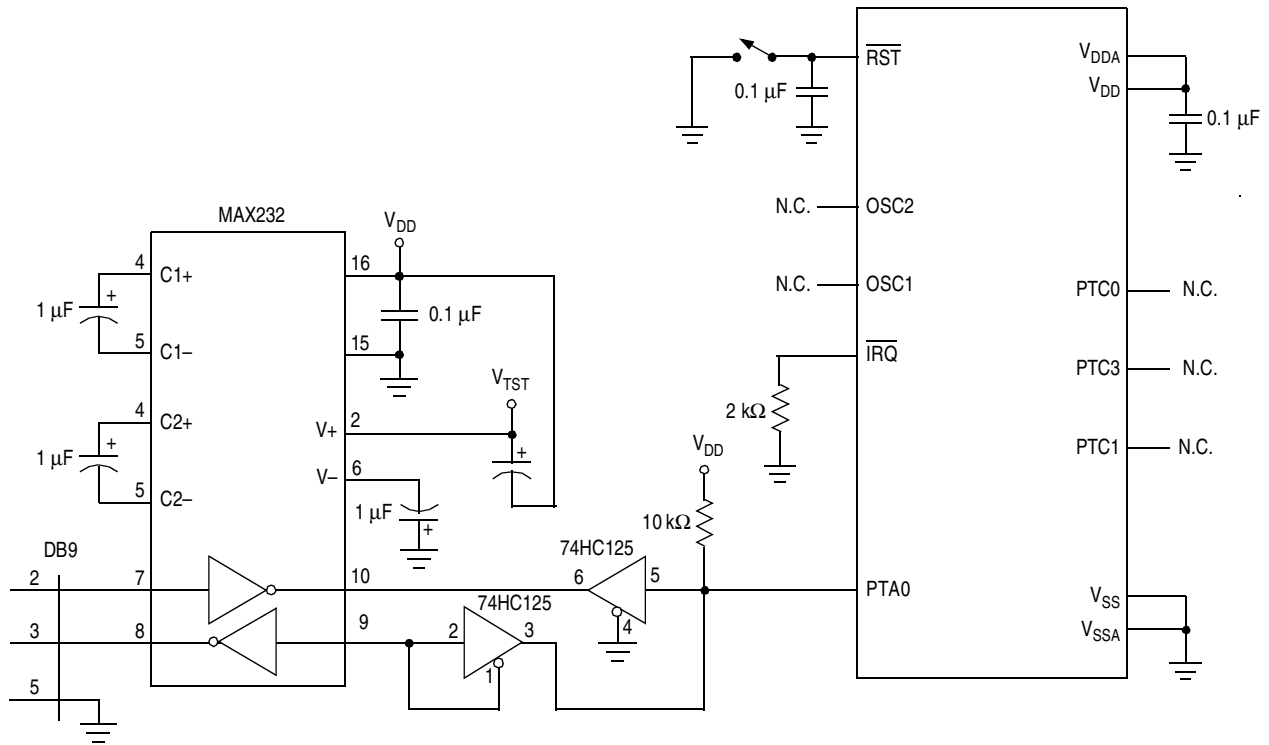
Figure 15-1. Forced Monitor Mode (Low)

Figure 15-2. Forced Monitor Mode (High)

MC68HC908GT16 • MC68HC908GT8 — Rev. 2   Technical Data
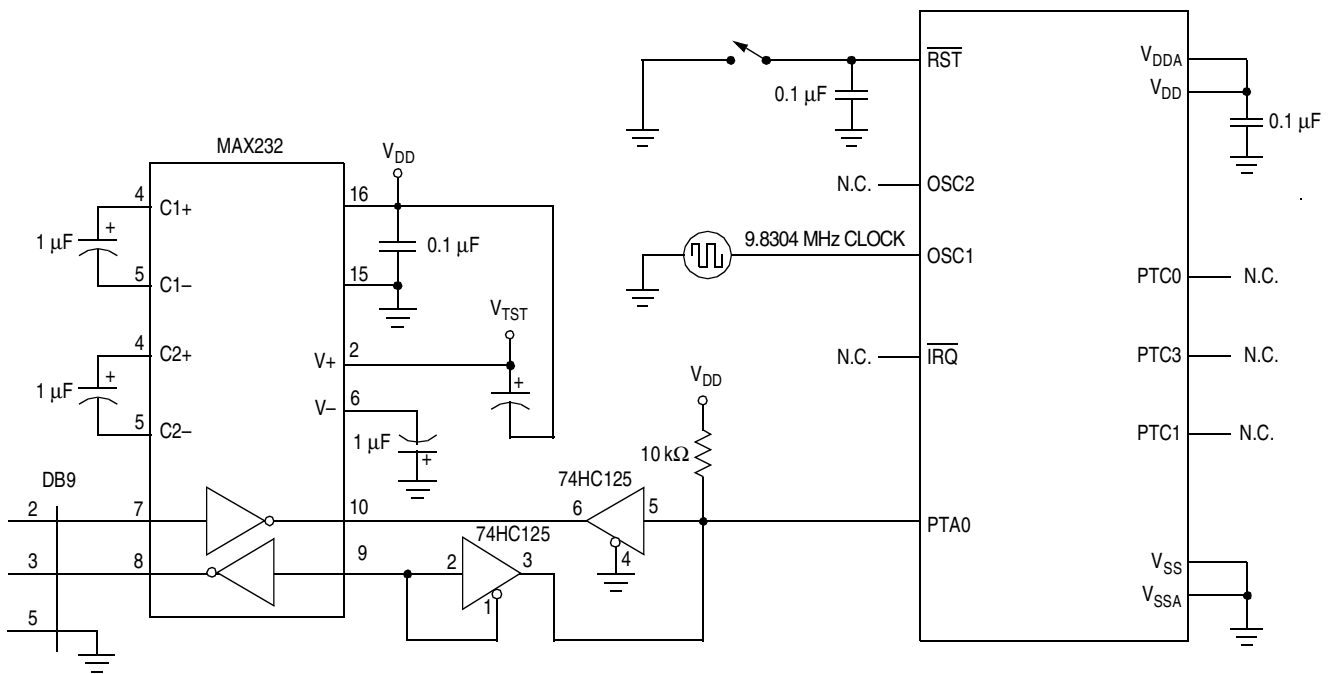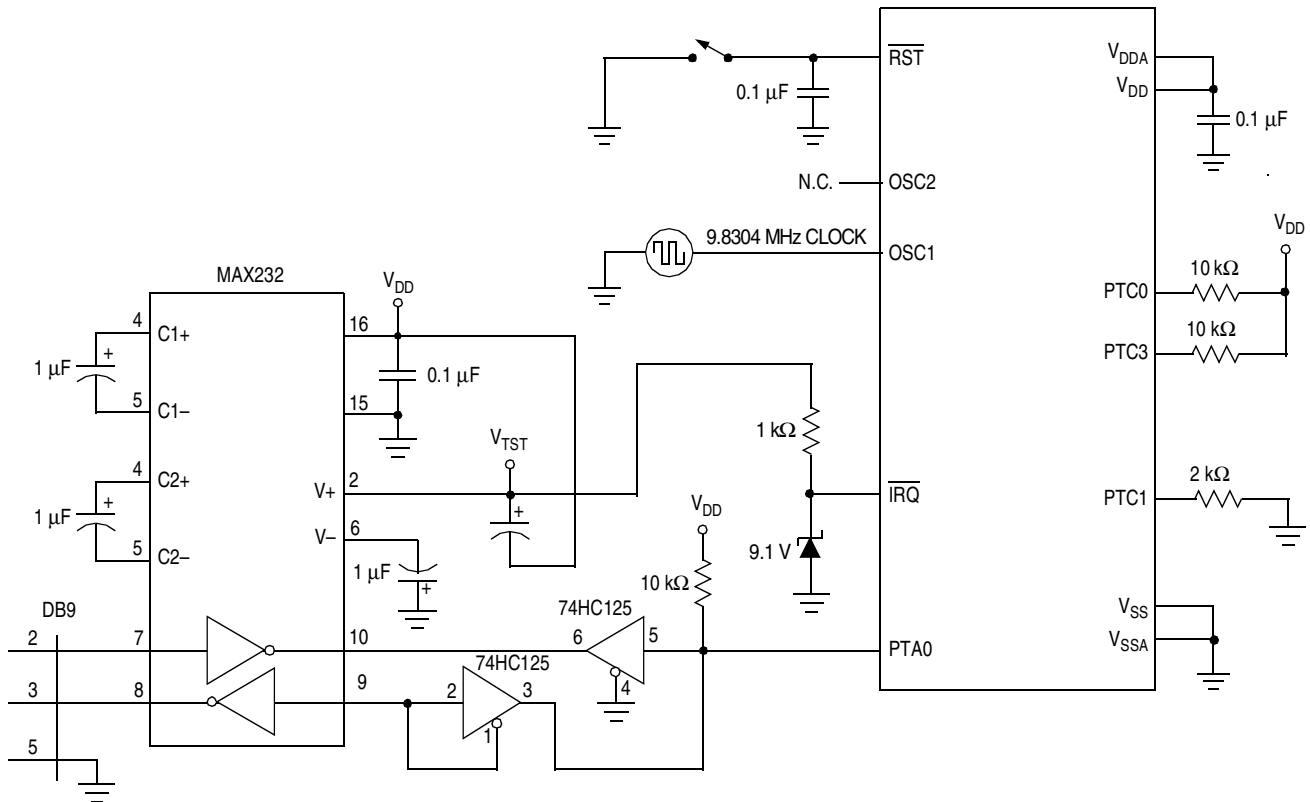
Figure 15-3. Standard Monitor Mode

The monitor code has been updated from previous versions of the monitor code to allow the ICG to generate the internal clock. This option, which is selected when $\overline{\text{IRQ}}$ is held low out of reset, is intended to support serial communication/ programming at 9600 baud in monitor mode by using the ICG, and the ICG user trim value ICGTR5 (if programmed) to generate the desired internal frequency (2.4576 MHz). If ICGTR5 is not programmed (i.e., the value is $FF) then the ICG will operate at a nominal (untrimmed) 2.45 MHz and communications will be nominally at 9600 baud but the untrimmed rate may cause difficulties with hosts which cannot automatically adjust their data rates to match.

Since this feature is enabled only when $\overline{\text{IRQ}}$ is held low out of reset, it cannot be used when the reset vector is programmed (i.e., the value is not $FFFF) because entry into monitor mode in this case requires V_{TST} on $\overline{\text{IRQ}}$.

15.4.1 Entering Monitor Mode

Table 15-1 shows the pin conditions for entering monitor mode. As specified in the table, monitor mode may 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:

1. If $FFFE and $FFFF does not contain $FF (programmed state):
   - The external clock is 4.9152 MHz with PTC3 low or 9.8304 MHz with PTC3 high
   - $\overline{IRQ} = V_{TST}$

2. If $FFFE and $FFFF contain $FF (erased state):
   - The external clock is 9.8304 MHz
   - $\overline{IRQ} = V_{DD}$ (this can be implemented through the internal $\overline{IRQ}$ pullup)

3. If $FFFE and $FFFF contain $FF (erased state):
   - $\overline{IRQ} = V_{SS}$ (ICG is selected, no external clock required)

If V_{TST} is applied to $\overline{IRQ}$ and PTC3 is low upon monitor mode entry (above condition set 1), the bus frequency is a divide-by-two of the input clock. If PTC3 is high with V_{TST} applied to $\overline{IRQ}$ upon monitor mode entry, the bus frequency will be a divide-by-four of the input clock. Holding the PTC3 pin low when entering monitor mode causes a bypass of a divide-by-two stage at the oscillator *only if V_{TST} is applied to $\overline{IRQ}$*. In this event, the CGMOUT frequency is equal to the CGMXCLK frequency, and the OSC1 input directly generates internal bus clocks. In this case, the OSC1 signal must have a 50% duty cycle at maximum bus frequency.

If entering monitor mode without high voltage on $\overline{IRQ}$ (above condition set 2 or 3, where applied voltage is either V_{DD} or V_{SS}), then all port C pin requirements and conditions, including the PTC3 frequency divisor selection, 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 POR reset. Once the part has been programmed, the traditional method of applying a voltage, V_{TST}, to $\overline{IRQ}$ must be used to enter monitor mode.*

Table 15-1. Monitor Mode Signal Requirements and Options

$\overline{IRQ}$	RESET	$FFFE/ $FFFF	ICG	PTC0	PTC1	PTC3	External Clock	CGMOUT	Bus Frequency	COP	For Serial Communication		Comment
											PTA0	Baud Rate	
X	GND	X	X	X	X	X	X	0	0	Disabled	X	0	No operation until reset goes high
V_{TST}	V_{DD}	X	OFF	1	0	0	4.9152 MHz	4.9152 MHz	2.4576 MHz	Disabled	1	9600	PTC3 determines frequency divider
V_{TST}	V_{DD}	X	OFF	1	0	1	9.8304 MHz	4.9152 MHz	2.4576 MHz	Disabled	1	9600	PTC3 determines frequency divider
V_{DD}	V_{DD}	$FF (blank)	OFF	X	X	X	9.8304 MHz	4.9152 MHz	2.4576 MHz	Disabled	1	9600	External frequency always divided by 4
GND	V_{DD}	$FF (blank)	ON	X	X	X	X	Nominal 4.91 MHz	Nominal 2.45 MHz	Disabled	1	Nominal 9600	ICG enabled
V_{DD} or GND	V_{DD}	$FF (blank)	OFF	X	X	X	X	—	—	Enabled	X	—	Enters user mode — will encounter an illegal address reset
V_{DD} or GND	V_{DD}	Not $FF (program-med)	ON	X	X	X	X	Nominal 3.2 MHz	Nominal 1.6 MHz	Enabled	X	—	Enters user mode

Note: X = don't care

The computer operating properly (COP) module is disabled in monitor mode based on these conditions:

- If monitor mode was entered as a result of the reset vector being blank (above condition set 2 or 3), the COP is always disabled regardless of the state of $\overline{\text{IRQ}}$ or $\overline{\text{RST}}$.

- If monitor mode was entered with V_{TST} on $\overline{\text{IRQ}}$ (condition set 1), then the COP is disabled as long as V_{TST} is applied to either $\overline{\text{IRQ}}$ or $\overline{\text{RST}}$.

The second 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.

Figure 15-4 shows a simplified diagram of the monitor mode entry when the reset vector is blank and just 1 x V_{DD} voltage is applied to the $\overline{\text{IRQ}}$ pin. 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.
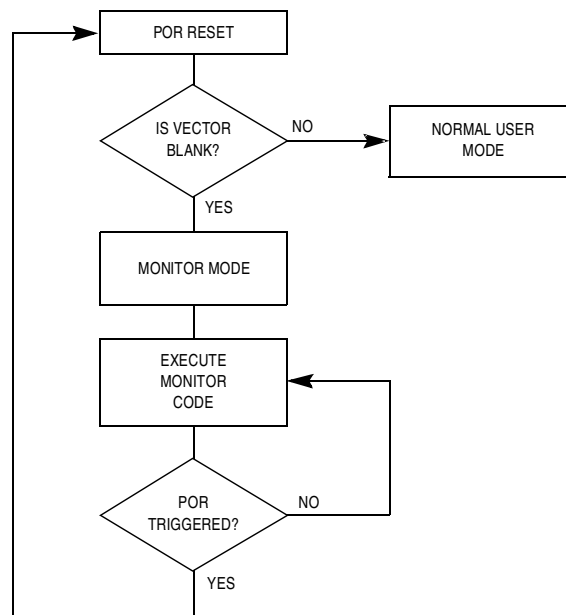
Figure 15-4. Low-Voltage Monitor Mode Entry Flowchart

Enter monitor mode with pin configuration shown in **Figure 15-3** by pulling $\overline{\text{RST}}$ low and then high. The rising edge of $\overline{\text{RST}}$ latches monitor mode. Once monitor mode is latched, the values on the specified pins can change.

Once out of reset, the MCU waits for the host to send eight security bytes (see **15.5 Security**). After the security bytes, the MCU sends a break signal (10 consecutive logic 0s) to the host, indicating that it is ready to receive a command.

NOTE: *The PTA0 pin must remain at logic 1 for 24 bus cycles after the $\overline{\text{RST}}$ pin goes high to enter monitor mode properly.*

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.

NOTE: *Exiting monitor mode after it has been initiated by having a blank reset vector requires a power-on reset (POR). Pulling $\overline{\text{RST}}$ low will not exit monitor mode in this situation.*

Table 15-2 summarizes the differences between user mode and monitor mode.

Table 15-2. 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

15.4.2 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.
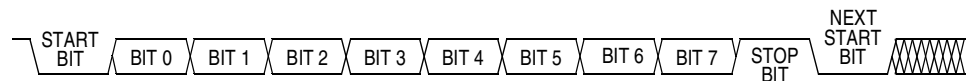
Figure 15-5. Monitor Data Format

15.4.3 Break Signal

A start bit (logic 0) followed by nine logic 0 bits is a break signal. When the monitor receives a break signal, it drives the PTA0 pin high for the duration of two bits and then echoes back the break signal.
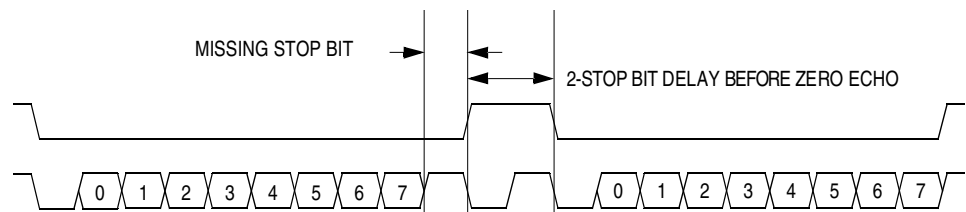
Figure 15-6. Break Transaction

15.4.4 Baud Rate

The communication baud rate is controlled by the crystal frequency and the state of the PTC3 pin (when $\overline{IRQ}$ is set to V_{TST}) upon entry into monitor mode. When PTC3 is high, the divide by ratio is 1024. If the PTC3 pin is at logic 0 upon entry into monitor mode, the divide by ratio is 512.

If monitor mode was entered with V_{DD} on $\overline{IRQ}$, then the divide by ratio is set at 1024, regardless of PTC3. If monitor mode was entered with V_{SS} on $\overline{IRQ}$, then the ICG generates 2.4576 MHz. These latter two conditions for monitor mode entry require that the reset vector is blank.

Table 15-3 lists external frequencies required to achieve a standard baud rate of 9600 bps. Other standard baud rates can be accomplished using proportionally higher or lower frequency generators. If using a crystal as the clock source, be aware of the upper frequency limit that the internal clock module can handle. See **23.8 5.0-V Control Timing** and **23.9 3.0-V Control Timing** for this limit.

Table 15-3. Monitor Baud Rate Selection

External Frequency	$\overline{IRQ}$	PTC3	Internal Frequency	Baud Rate (bps)
4.9152 MHz	V_{TST}	0	2.4576 MHz	9600
9.8304 MHz	V_{TST}	1	2.4576 MHz	9600
9.8304 MHz	V_{DD}	X	2.4576 MHz	9600

15.4.5 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.*
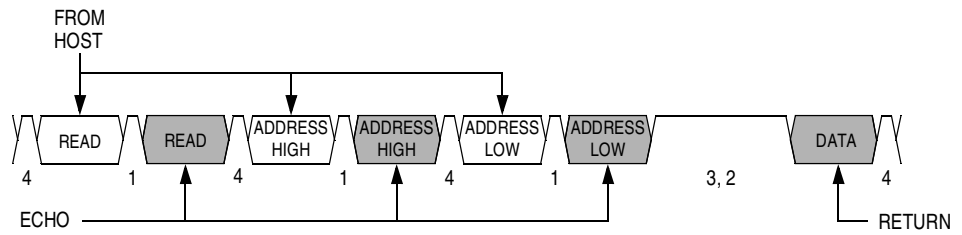
Notes:
1 = Echo delay, 2 bit times
2 = Data return delay, 2 bit times
3 = Cancel command delay, 11 bit times
4 = Wait 1 bit time before sending next byte.

Figure 15-7. Read Transaction
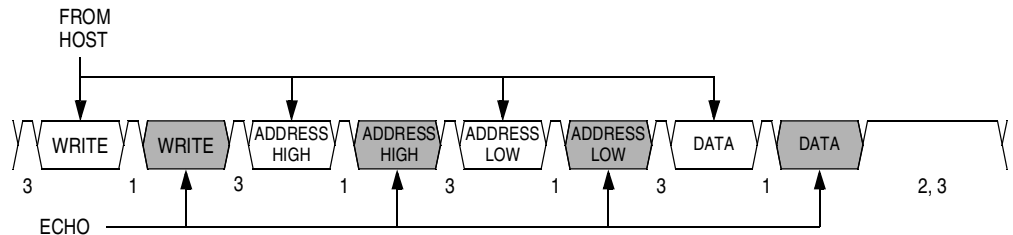
Notes:
1 = Echo delay, 2 bit times
2 = Cancel command delay, 11 bit times
3 = Wait 1 bit time before sending next byte.

Figure 15-8. Write Transaction

A brief description of each monitor mode command is given in
Table 15-4 through **Table 15-9**.

Table 15-4. 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 15-5. 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 15-6. IREAD (Indexed Read) Command

Description	Read next 2 bytes in memory from last address accessed
Operand	2-byte address in high byte:low byte order
Data Returned	Returns contents of next two addresses
Opcode	$1A
Command Sequence	

Table 15-7. 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 15-8. 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 15-9. 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.

	SP
HIGH BYTE OF INDEX REGISTER	SP + 1
CONDITION CODE REGISTER	SP + 2
ACCUMULATOR	SP + 3
LOW BYTE OF INDEX REGISTER	SP + 4
HIGH BYTE OF PROGRAM COUNTER	SP + 5
LOW BYTE OF PROGRAM COUNTER	SP + 6
	SP + 7
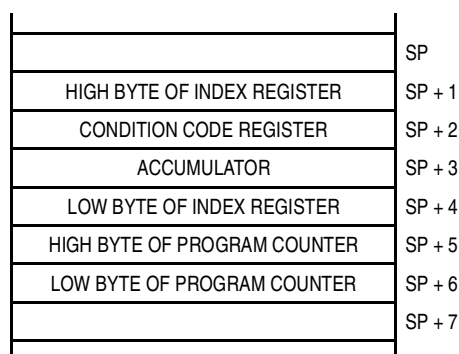
Figure 15-9. Stack Pointer at Monitor Mode Entry

15.5 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 15-10**.)

Figure 15-10. Monitor Mode Entry Timing

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.*

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).

Section 16. Input/Output (I/O) Ports

16.1 Contents

16.2 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.*

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register (PTA) See page 229.	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:				Unaffected by reset				
$0001	Port B Data Register (PTB) See page 233.	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:				Unaffected by reset				
$0002	Port C Data Register (PTC) See page 236.	Read: Write:	0	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
		Reset:				Unaffected by reset				
$0003	Port D Data Register (PTD) See page 240.	Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Reset:				Unaffected by reset				
$0004	Data Direction Register A (DDRA) See page 230.	Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0

☐ = Unimplemented

Figure 16-1. 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 234.	Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC) See page 237.	Read: Write:	0	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$0007	Data Direction Register D (DDRD) See page 242.	Read: Write:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE) See page 245.	Read: Write:	0	0	0	PTE4	PTE3	PTE2	PTE1	PTE0
		Reset:				Unaffected by reset				
$000C	Data Direction Register E (DDRE) See page 246.	Read: Write:	0	0	0	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
		Reset:	0	0	0	0	0	0	0	0
$000D	Port A Input Pullup Enable Register (PTAPUE) See page 232.	Read: Write:	PTAPUE7	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
		Reset:	0	0	0	0	0	0	0	0
$000E	Port C Input Pullup Enable Register (PTCPUE) See page 239.	Read: Write:	0	PTCPUE6	PTCPUE5	PTCPUE4	PTCPUE3	PTCPUE2	PTCPUE1	PTCPUE0
		Reset:	0	0	0	0	0	0	0	0
$000F	Port D Input Pullup Enable Register (PTDPUE) See page 244.	Read: Write:	PTDPUE7	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-1. I/O Port Register Summary (Continued)

Table 16-1. Port Control Register Bits Summary

Port	Bit	DDR	Module Control		Pin
A	0	DDRA0		KBIE0	PTA0/KBD0
	1	DDRA1		KBIE1	PTA1/KBD1
	2	DDRA2		KBIE2	PTA2/KBD2
	3	DDRA3	KBD	KBIE3	PTA3/KBD3
	4	DDRA4		KBIE4	PTA4/KBD4
	5	DDRA5		KBIE5	PTA5/KBD5
	6	DDRA6		KBIE6	PTA6/KBD6
	7	DDRA7		KBIE7	PTA7/KBD7
B	0	DDRB0			PTB0/AD0
	1	DDRB1			PTB1/AD1
	2	DDRB2			PTB2/AD2
	3	DDRB3	ADC	ADCH4–ADCH0	PTB3/AD3
	4	DDRB4			PTB4/AD4
	5	DDRB5			PTB5/AD5
	6	DDRB6			PTB6/AD6
	7	DDRB7			PTB7/AD7
C	0	DDRC0			PTC0
	1	DDRC1			PTC1
	2	DDRC2			PTC2
	3	DDRC3			PTC3
	4	DDRC4			PTC4
	5	DDRC5			PTC5
	6	DDRC6			PTC6
D	0	DDRD0			PTD0/$\overline{SS}$
	1	DDRD1	SPI	SPE	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
	7	DDRD7		ELS1B:ELS1A	PTD7/T2CH1
E	0	DDRE0	SCI	ENSCI	PTE0/TxD
	1	DDRE1			PTE1/RxD
	2	DDRE2			PTE2
	3	DDRE3	ICG	ECGON: EXTXTALEN	PTE3/OSC2
	4	DDRE4		ECGON	PTE4/OSC1

16.3 Port A

Port A is an 8-bit special-function port that shares all eight of its pins with the keyboard interrupt (KBI) module. Port A also has software configurable pullup devices if configured as an input port.

16.3.1 Port A Data Register

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

Address: $0000

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
Reset:	Unaffected by reset							
Alternate Function:	KBD7	KBD6	KBD5	KBD4	KBD3	KBD2	KBD1	KBD0
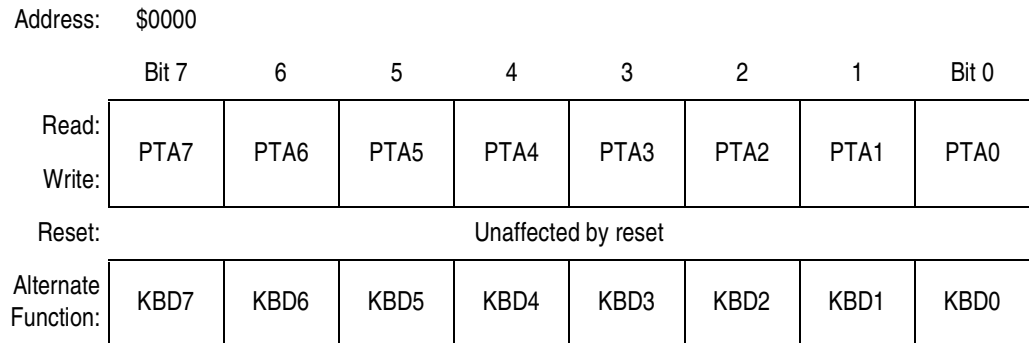
Figure 16-2. Port A Data Register (PTA)

PTA7–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.

KBD7–KBD0 — Keyboard Inputs

The keyboard interrupt enable bits, KBIE7–KBIE0, in the keyboard interrupt control register (KBICR) enable the port A pins as external interrupt pins. See **Section 13. Keyboard Interrupt Module (KBI)**.

16.3.2 Data Direction Register A

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

Address: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset:	0	0	0	0	0	0	0	0

Figure 16-3. Data Direction Register A (DDRA)

DDRA7–DDRA0 — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA7–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 16-4 shows the port A I/O logic.
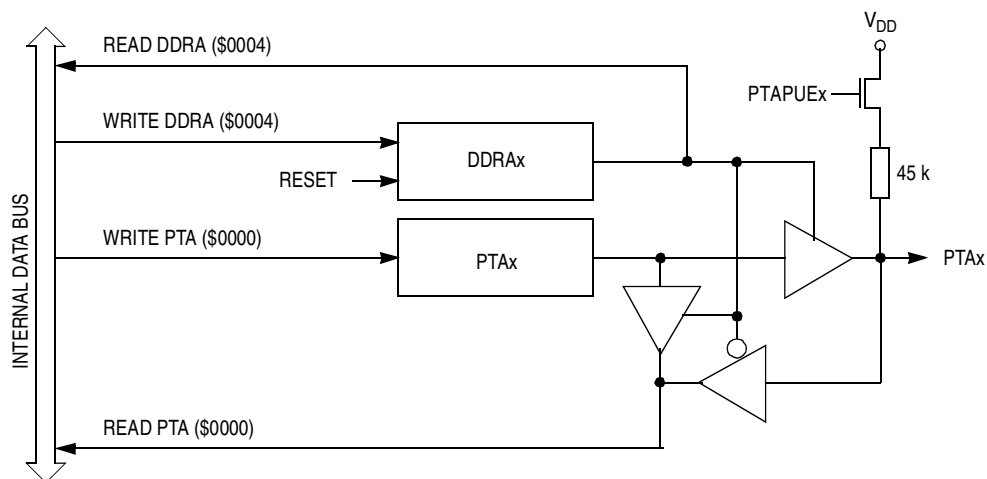
Figure 16-4. Port A I/O Circuit

When bit DDRAx is a logic 1, reading address $0000 reads the PTAx data latch. When bit DDRAx is a logic 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 16-2** summarizes the operation of the port A pins.

Table 16-2. 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]	DDRA7–DDRA0	Pin	PTA7–PTA0[3]
0	0	X	Input, Hi-Z[4]	DDRA7–DDRA0	Pin	PTA7–PTA0[3]
X	1	X	Output	DDRA7–DDRA0	PTA7–PTA0	PTA7–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

16.3.3 Port A Input Pullup Enable Register

The port A input pullup enable register (PTAPUE) contains a software configurable pullup device for each of the eight 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: $0004

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTAPUE7	PTAPUE6	PTAPUE5	PTAPUE4	PTAPUE3	PTAPUE2	PTAPUE1	PTAPUE0
Reset:	0	0	0	0	0	0	0	0

Figure 16-5. Port A Input Pullup Enable Register (PTAPUE)

PTAPUE7–PTAPUE0 — Port A Input Pullup Enable Bits

These writable 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

16.4 Port B

Port B is an 8-bit special-function port that shares all eight of its pins with the analog-to-digital converter (ADC) module.

16.4.1 Port B Data Register

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

Address: $0001

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
Reset:	Unaffected by reset							
Alternate Function:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0

Figure 16-6. Port B Data Register (PTB)

PTB7–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.

AD7–AD0 — Analog-to-Digital Input Bits

AD7–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 AD7–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.*

16.4.2 Data Direction Register B

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

Address: $0005

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Reset:	0	0	0	0	0	0	0	0
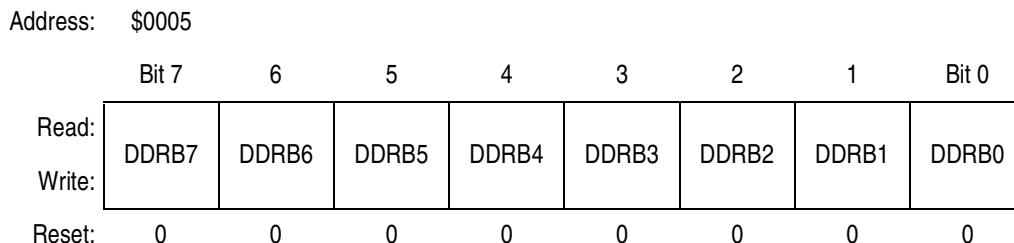
Figure 16-7. Data Direction Register B (DDRB)

DDRB7–DDRB0 — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB7–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.*

Figure 16-8 shows the port B I/O logic.
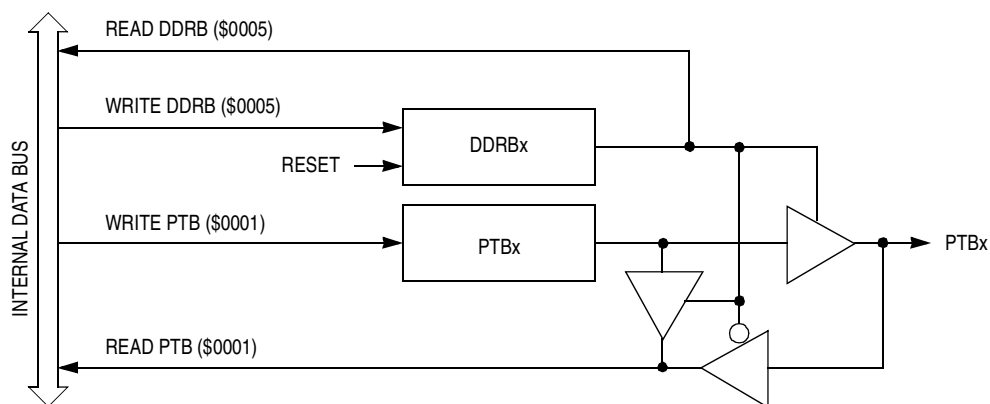
Figure 16-8. Port B I/O Circuit

When bit DDRBx is a logic 1, reading address $0001 reads the PTBx data latch. When bit DDRBx is a logic 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 16-3** summarizes the operation of the port B pins.

Table 16-3. 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]	DDRB7–DDRB0	Pin	PTB7–PTB0[3]
1	X	Output	DDRB7–DDRB0	PTB7–PTB0	PTB7–PTB0

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

16.5 Port C

Port C is a 7-bit, general-purpose bidirectional I/O port. Port C also has software configurable pullup devices if configured as an input port.

16.5.1 Port C Data Register

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

NOTE: *Bit 6 and bit 5 of PTC are not available in the 42-pin shrink dual in-line package.*

Address: $0002

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
Write:								
Reset:				Unaffected by reset				

      = Unimplemented
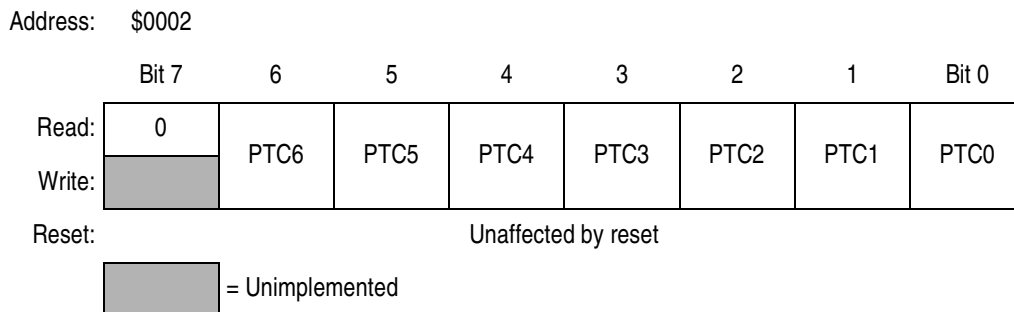
Figure 16-9. Port C Data Register (PTC)

PTC6–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.

16.5.2 Data Direction Register C

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

Address: $0006

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
Write:		DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-10. Data Direction Register C (DDRC)

DDRC6–DDRC0 — Data Direction Register C Bits

These read/write bits control port C data direction. Reset clears DDRC6–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.*

Figure 16-11 shows the port C I/O logic.

NOTE: *For those devices packaged in a 42-pin shrink dual in-line package, PTC5 and PTC6 are connected to ground internally. DDRC5 and DDRC6 should be set to a 0 to configure PTC5 and PTC6 as inputs.*
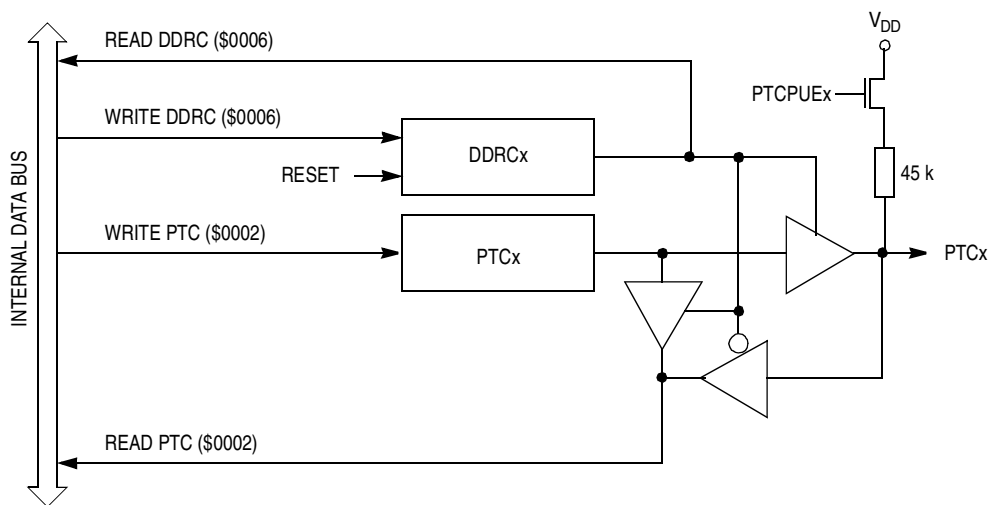
Figure 16-11. Port C I/O Circuit

When bit DDRCx is a logic 1, reading address $0002 reads the PTCx data latch. When bit DDRCx is a logic 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 16-4** summarizes the operation of the port C pins.

Table 16-4. 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]	DDRC6–DDRC0	Pin	PTC6–PTC0[3]
0	0	X	Input, Hi-Z[4]	DDRC6–DDRC0	Pin	PTC6–PTC0[3]
X	1	X	Output	DDRC6–DDRC0	PTC6–PTC0	PTC6–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

16.5.3 Port C Input Pullup Enable Register

The port C input pullup enable register (PTCPUE) contains a software configurable pullup device for each of the seven 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	PTCPUE6	PTCPUE5	PTCPUE4	PTCPUE3	PTCPUE2	PTCPUE1	PTCPUE0
Write:		PTCPUE6	PTCPUE5	PTCPUE4	PTCPUE3	PTCPUE2	PTCPUE1	PTCPUE0
Reset:	0	0	0	0	0	0	0	0

☐ = Unimplemented
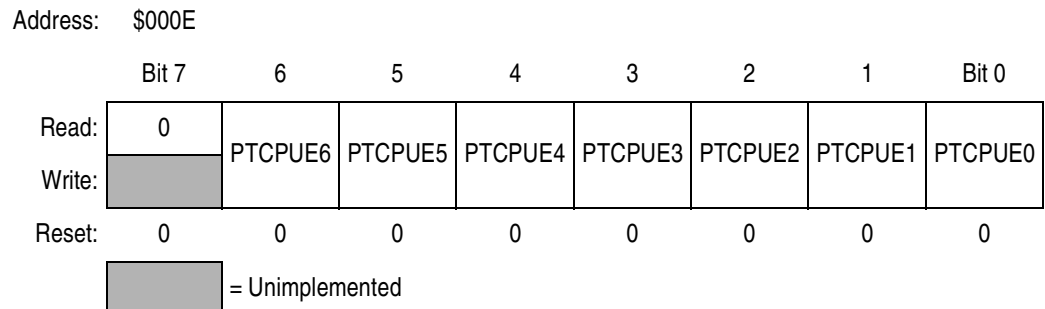
Figure 16-12. Port C Input Pullup Enable Register (PTCPUE)

PTCPUE6–PTCPUE0 — Port C Input Pullup Enable Bits

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

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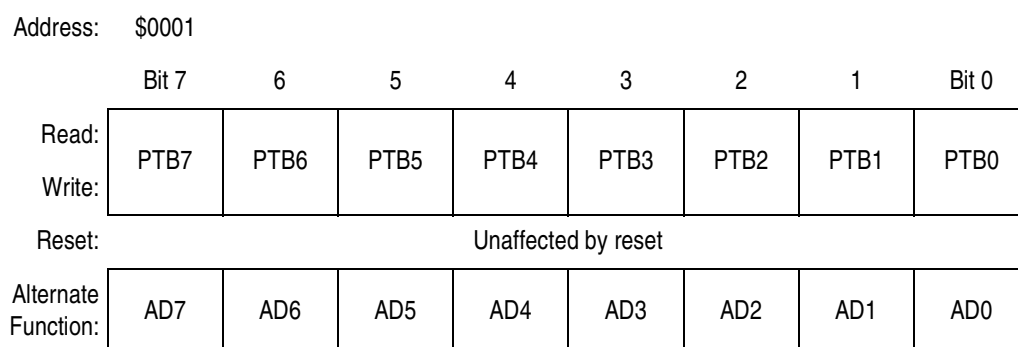0 = Corresponding port C pin internal pullup disconnected

16.6 Port D

Port D is an 8-bit special-function port that shares four of its pins with the serial peripheral interface (SPI) module and four 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.

16.6.1 Port D Data Register

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

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Reset:				Unaffected by reset				
Alternate Function:	T2CH1	T2CH0	$\overline{\text{T1CH1}}$	T1CH0	SPSCK	$\overline{\text{MOSI}}$	MISO	$\overline{\text{SS}}$

Figure 16-13. Port D Data Register (PTD)

PTD7–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.

T2CH1 and T2CH0 — Timer 2 Channel I/O Bits

The PTD7/T2CH1–PTD6/T2CH0 pins are the TIM2 input capture/output compare pins. The edge/level select bits, ELSxB:ELSxA, determine whether the PTD7/T2CH1–PTD6/T2CH0 pins are timer channel I/O pins or general-purpose I/O pins. See **Section 22. Timer Interface Module (TIM)**.

T1CH1 and T1CH0 — Timer 1 Channel I/O Bits

The PTD7/T1CH1–PTD6/T1CH0 pins are the TIM1 input capture/output compare pins. The edge/level select bits, ELSxB and ELSxA, determine whether the PTD7/T1CH1–PTD6/T1CH0 pins are timer channel I/O pins or general-purpose I/O pins. See **Section 22. Timer Interface Module (TIM)**.

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/$\overline{SS}$ 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 16-5**.

$\overline{SS}$ — Slave Select

The PTD0/$\overline{SS}$ 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/$\overline{SS}$ 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/$\overline{SS}$ pin.

16.6.2 Data Direction Register D

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

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 16-14. Data Direction Register D (DDRD)

DDRD7−DDRD0 — Data Direction Register D Bits

These read/write bits control port D data direction. Reset clears DDRD7−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 16-15 shows the port D I/O logic.
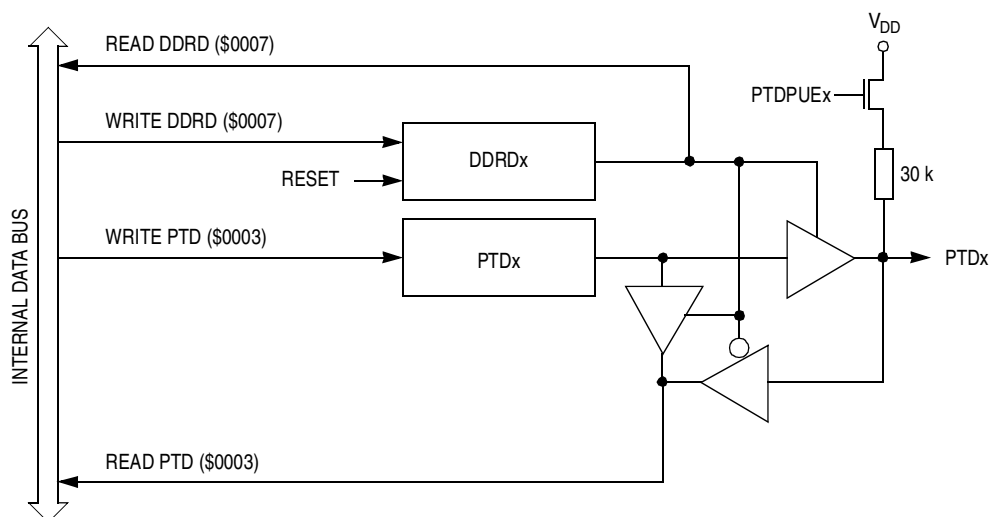
Figure 16-15. Port D I/O Circuit

When bit DDRDx is a logic 1, reading address $0003 reads the PTDx
data latch. When bit DDRDx is a logic 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 16-5** summarizes
the operation of the port D pins.

Table 16-5. 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]	DDRD7–DDRD0	Pin	PTD7–PTD0[3]
0	0	X	Input, Hi-Z[4]	DDRD7–DDRD0	Pin	PTD7–PTD0[3]
X	1	X	Output	DDRD7–DDRD0	PTD7–PTD0	PTD7–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 imp[edance

16.6.3 Port D Input Pullup Enable Register

The port D input pullup enable register (PTDPUE) contains a software configurable pullup device for each of the eight 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:	PTDPUE7	PTDPUE6	PTDPUE5	PTDPUE4	PTDPUE3	PTDPUE2	PTDPUE1	PTDPUE0
Write:								
Reset:	0	0	0	0	0	0	0	0

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

PTDPUE7–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

16.7 Port E

Port E is a 5-bit special-function port that shares two of its pins with the serial communications interface (SCI) module and two of its pins with the internal clock generator (ICG).

16.7.1 Port E Data Register

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

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	PTE4	PTE3	PTE2	PTE1	PTE0
Write:								
Reset:				Unaffected by reset				
Alternate Function:				OSC1	OSC2		RxD	TxD

                    = Unimplemented
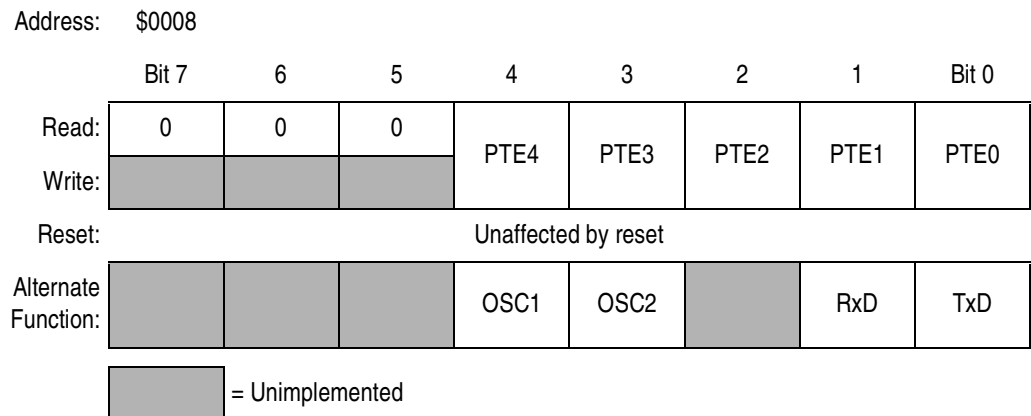
Figure 16-17. Port E Data Register (PTE)

PTE4-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 Edata.

NOTE: *Data direction register E (DDRE) does not affect the data direction of port E pins that are being used by the SCI 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 16-6.*

OSC2 and OSC1 — OSC2 and OSC1 Bits

Under software control, PTE4 and PTE3 can be configured as external clock inputs and outputs. PTE3 will become an output clock, OSC2, if selected in the configuration registers and enabled in the

ICG registers. PTE4 will become an external input clock source, OSC1, if selected in the configuration registers and enabled in the ICG registers. See **Section 7. Internal Clock Generator (ICG) Module** and **Section 9. Computer Operating Properly (COP) Module**. While configured as oscillator pins, writes have no effect and reads return undefined values.

RxD — SCI Receive Data Input

The PTE1/RxD pin is the receive data input for the SCI module. When the enable SCI bit, ENSCI, is clear, the SCI module is disabled, and the PTE1/RxD pin is available for general-purpose I/O. See **Section 18. Enhanced Serial Communications Interface (ESCI) Module**.

TxD — SCI Transmit Data Output

The PTE0/TxD pin is the transmit data output for the SCI module. When the enable SCI bit, ENSCI, is clear, the SCI module is disabled, and the PTE0/TxD pin is available for general-purpose I/O. See **Section 18. Enhanced Serial Communications Interface (ESCI) Module**.

16.7.2 Data Direction Register E

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

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
Write:				DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
Reset:	0	0	0	0	0	0	0	0

	= Unimplemented

Figure 16-18. Data Direction Register E (DDRE)

DDRE4–DDRE0 — Data Direction Register E Bits

These read/write bits control port E data direction. Reset clears DDRE4–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 16-19 shows the port E I/O logic.
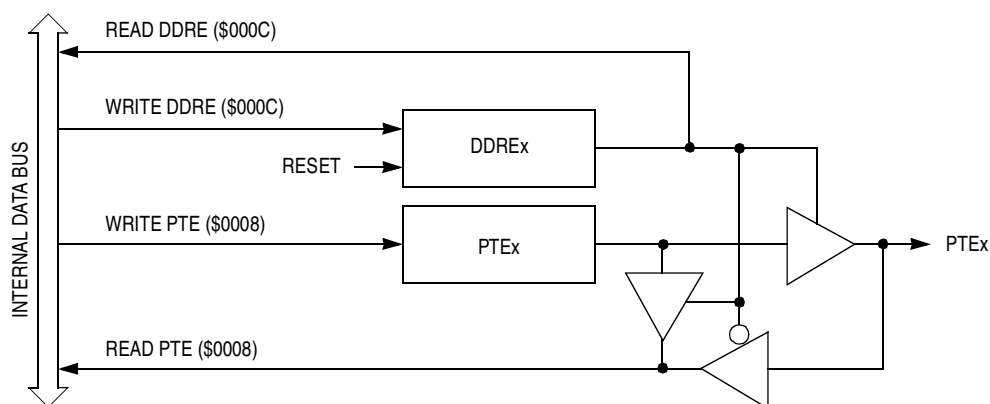
Figure 16-19. Port E I/O Circuit

When bit DDREx is a logic 1, reading address $0008 reads the PTEx data latch. When bit DDREx is a logic 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 16-6** summarizes the operation of the port E pins.

Table 16-6. 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]	DDRE4–DDRE0	Pin	PTE4–PTE0[3]
1	X	Output	DDRE4–DDRE0	PTE4–PTE0	PTE4–PTE0

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

Section 17. Random-Access Memory (RAM)

17.1 Contents

17.2 Introduction

This section describes the 512 bytes of RAM (random-access memory).

17.3 Functional Description

Addresses $0040 through $023F 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.*

Section 18. Enhanced Serial Communications Interface (ESCI) Module

18.1 Contents

18.2 Introduction

The enhanced serial communications interface (ESCI) module allows asynchronous communications with peripheral devices and other microcontroller units (MCU).

18.3 Features

Features include:

- Full-duplex operation

- Standard mark/space non-return-to-zero (NRZ) format

- Programmable baud rates

- Programmable 8-bit or 9-bit character length

- Separately enabled transmitter and receiver

- Separate receiver and transmitter central processor unit (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

18.4 Pin Name Conventions

The generic names of the ESCI input/output (I/O) pins are:

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

ESCI I/O lines are implemented by sharing parallel I/O port pins. The full name of an ESCI input or output reflects the name of the shared port pin. **Table 18-1** shows the full names and the generic names of the ESCI I/O pins. The generic pin names appear in the text of this section.

Table 18-1. Pin Name Conventions

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

18.5 Functional Description

Figure 18-1 shows the structure of the ESCI module. The ESCI allows full-duplex, asynchronous, NRZ serial communication between the MCU and remote devices, including other MCUs. The transmitter and receiver of the ESCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the ESCI, writes the data to be transmitted, and processes received data.

For reference, a summary of the ESCI module input/output registers is provided in **Figure 18-2**.

Figure 18-1. ESCI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0009	ESCI Prescaler Register (SCPSC) See page 286.	Read: Write:	PDS2	PDS1	PDS0	PSSB4	PSSB3	PSSB2	PSSB1	PSSB0
		Reset:	0	0	0	0	0	0	0	0
$000A	ESCI Arbiter Control Register (SCIACTL) See page 290.	Read: Write:	AM1	ALOST	AM0	ACLK	AFIN	ARUN	AROVFL	ARD8
		Reset:	0	0	0	0	0	0	0	0
$000B	ESCI Arbiter Data Register (SCIADAT) See page 292.	Read: Write:	ARD7	ARD6	ARD5	ARD4	ARD3	ARD2	ARD1	ARD0
		Reset:	0	0	0	0	0	0	0	0
$0013	ESCI Control Register 1 (SCC1) See page 272.	Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Reset:	0	0	0	0	0	0	0	0
$0014	ESCI Control Register 2 (SCC2) See page 275.	Read: Write:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Reset:	0	0	0	0	0	0	0	0
$0015	ESCI Control Register 3 (SCC3) See page 278.	Read: Write:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
		Reset:	U	0	0	0	0	0	0	0
$0016	ESCI Status Register 1 (SCS1) See page 279.	Read: Write:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Reset:	1	1	0	0	0	0	0	0
$0017	ESCI Status Register 2 (SCS2) See page 283.	Read: Write:	0	0	0	0	0	0	BKF	RPF
		Reset:	0	0	0	0	0	0	0	0
$0018	ESCI Data Register (SCDR) See page 284.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:				Unaffected by reset				
$0019	ESCI Baud Rate Register (SCBR) See page 284.	Read: Write:	R	LINR	SCP1	SCP0	R	SCR2	SCR1	SCR0
		Reset:	0	0	0	0	0	0	0	0

	= Unimplemented	R	= Reserved

Figure 18-2. ESCI I/O Register Summary

18.5.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in **Figure 18-3**.
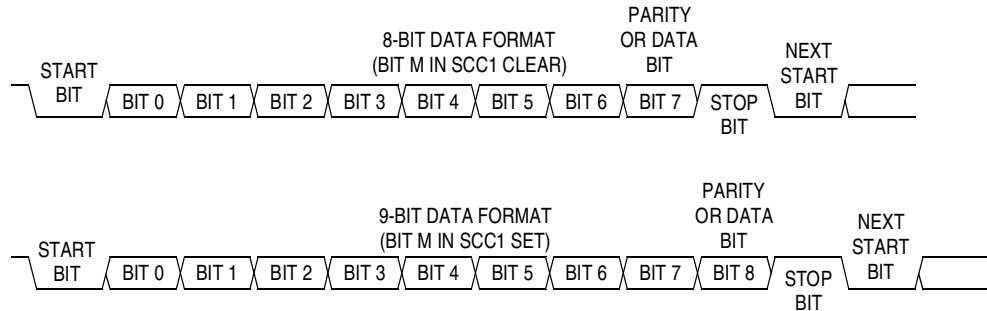
Figure 18-3. SCI Data Formats

18.5.2 Transmitter

Figure 18-4 shows the structure of the SCI transmitter and the registers are summarized in **Figure 18-2**.

18.5.2.1 Character Length

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

18.5.2.2 Character Transmission

During an ESCI transmission, the transmit shift register shifts a character out to the TxD pin. The ESCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register.
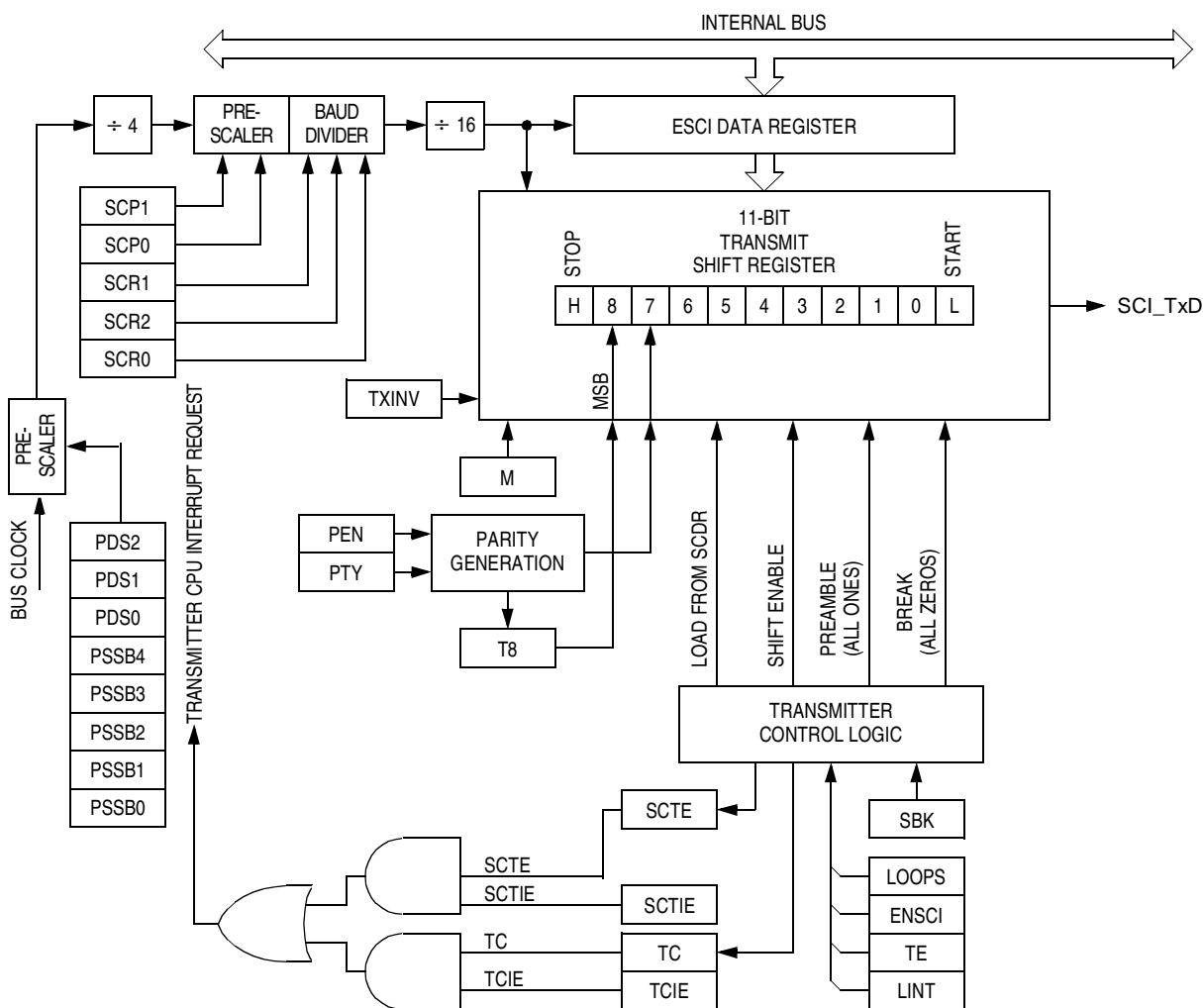
Figure 18-4. ESCI Transmitter

To initiate an ESCI transmission:

1. Enable the ESCI by writing a logic 1 to the enable ESCI bit (ENSCI) in ESCI control register 1 (SCC1).

2. Enable the transmitter by writing a logic 1 to the transmitter enable bit (TE) in ESCI control register 2 (SCC2).

3. Clear the ESCI transmitter empty bit (SCTE) by first reading ESCI status register 1 (SCS1) and then writing to the SCDR. For 9-bit data, also write the T8 bit in SCC3.

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 logic 1s. After the preamble shifts out, control logic transfers the SCDR data into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit (LSB) position of the transmit shift register. A logic 1 stop bit goes into the most significant bit (MSB) position.

The ESCI 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 ESCI 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 TxD pin goes to the idle condition, logic 1. If at any time software clears the ENSCI bit in ESCI control register 1 (SCC1), the transmitter and receiver relinquish control of the port E pins.

18.5.2.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCC2 loads the transmit shift register with a break character. For TXINV = 0 (output not inverted), a transmitted break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCC1 and the LINR bits in SCBR. As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

When LINR is cleared in SCBR, the ESCI recognizes a break character when a start bit is followed by eight or nine logic 0 data bits and a logic 0 where the stop bit should be, resulting in a total of 10 or 11 consecutive logic 0 data bits. When LINR is set in SCBR, the ESCI recognizes a break character when a start bit is followed by 9 or 10 logic 0 data bits and a logic 0 where the stop bit should be, resulting in a total of 11 or 12 consecutive logic 0 data bits.

Receiving a break character has these effects on ESCI registers:

- Sets the framing error bit (FE) in SCS1

- Sets the ESCI receiver full bit (SCRF) in SCS1

- Clears the ESCI 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

18.5.2.4 Idle Characters

For TXINV = 0 (output not inverted), a transmitted idle character contains all logic 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 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 logic 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. A good time to toggle the TE bit for a queued idle character is when the SCTE bit becomes set and just before writing the next byte to the SCDR.*

18.5.2.5 Inversion of Transmitted Output

The transmit inversion bit (TXINV) in ESCI 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 at logic 1. See **18.9.1 ESCI Control Register 1**.

18.5.2.6 Transmitter Interrupts

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

- ESCI 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 ESCI 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.

18.5.3 Receiver

Figure 18-5 shows the structure of the ESCI receiver. The receiver I/O registers are summarized in **Figure 18-2**.

18.5.3.1 Character Length

The receiver can accommodate either 8-bit or 9-bit data. The state of the M bit in ESCI control register 1 (SCC1) determines character length. When receiving 9-bit data, bit R8 in ESCI control register 3 (SCC3) is the ninth bit (bit 8). When receiving 8-bit data, bit R8 is a copy of the eighth bit (bit 7).

18.5.3.2 Character Reception

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

Figure 18-5. ESCI Receiver Block Diagram

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

18.5.3.3 Data Sampling

The receiver samples the 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 these times (see **Figure 18-6**):

- After every start bit

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

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.
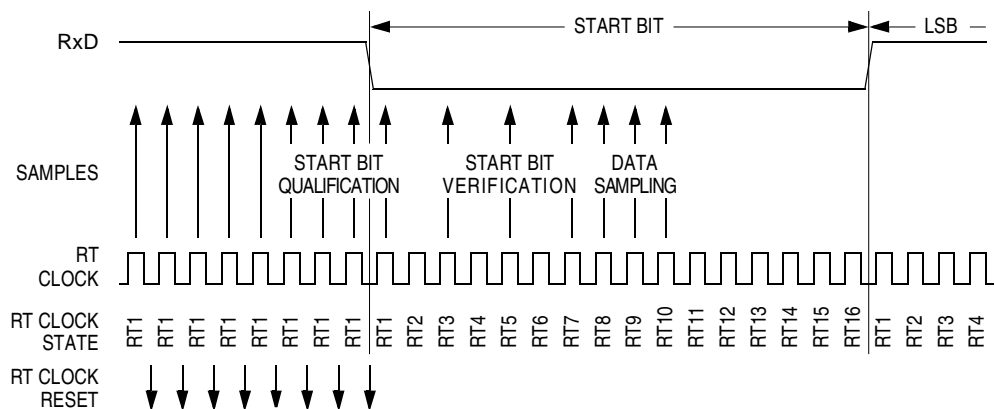
Figure 18-6. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. **Table 18-2** summarizes the results of the start bit verification samples.

Table 18-2. 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

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 18-3** summarizes the results of the data bit samples.

Table 18-3. 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 logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit.*

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. **Table 18-4** summarizes the results of the stop bit samples.

Table 18-4. 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

18.5.3.4 Framing Errors

If the data recovery logic does not detect a logic 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.

18.5.3.5 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 18-7 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.
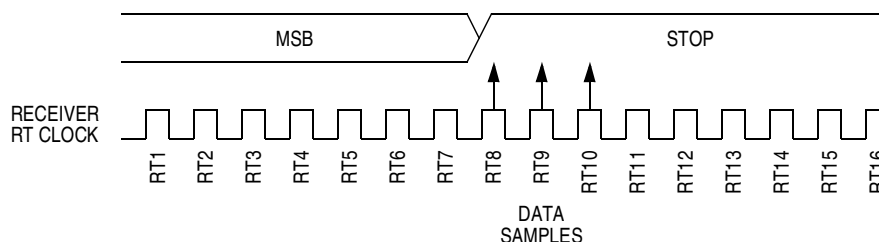
Figure 18-7. Slow Data

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

With the misaligned character shown in **Figure 18-7**, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 9 bit times $\times$ 16 RT cycles + 3 RT cycles = 147 RT cycles.

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 $\times$ 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in **Figure 18-7**, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 10 bit times $\times$ 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 18-8 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.
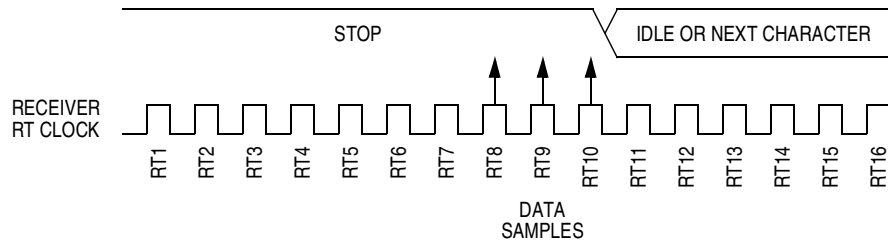
Figure 18-8. Fast Data

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

With the misaligned character shown in **Figure 18-8**, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times $\times$ 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

$$\left| \frac{154 - 160}{154} \right| \times 100 = 3.90\%.$$

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

With the misaligned character shown in **Figure 18-8**, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 11 bit times $\times$ 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:

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

18.5.3.6 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 RxD pin can bring the receiver out of the standby state:

1. Address mark — An address mark is a logic 1 in the MSB 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 ESCI 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.

2. Idle input line condition — When the WAKE bit is clear, an idle character on the 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 ESCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting logic 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 will cause the receiver to wake up.*

18.5.3.7 Receiver Interrupts

These sources can generate CPU interrupt requests from the ESCI receiver:

- ESCI 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 ESCI 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 logic 1s shifted in from the RxD pin. The idle line interrupt enable bit, ILIE, in SCC2 enables the IDLE bit to generate CPU interrupt requests.

18.5.3.8 Error Interrupts

These 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 ESCI error CPU interrupt requests.

- Noise flag (NF) — The NF bit is set when the ESCI 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 ESCI error CPU interrupt requests.

- Framing error (FE) — The FE bit in SCS1 is set when a logic 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate ESCI error CPU interrupt requests.

- Parity error (PE) — The PE bit in SCS1 is set when the ESCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate ESCI error CPU interrupt requests.

18.6 Low-Power Modes

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

18.6.1 Wait Mode

The ESCI module remains active in wait mode. Any enabled CPU interrupt request from the ESCI module can bring the MCU out of wait mode.

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

18.6.2 Stop Mode

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

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

18.7 ESCI During Break Module Interrupts

The BCFE bit in the break flag control register (SBFCR) enables software to clear status bits during the break state. See **Section 6. Break Module (BRK)**.

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 status bits 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 I/O registers during the break state without affecting status bits. Some status bits have a two-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 logic 0. After the break, doing the second step clears the status bit.

18.8 I/O Signals

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

- PTE0/TxD — transmit data
- PTE1/RxD — receive data

18.8.1 PTE0/TxD (Transmit Data)

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

18.8.2 PTE1/RxD (Receive Data)

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

18.9 I/O Registers

These I/O registers control and monitor ESCI operation:

- ESCI control register 1, SCC1
- ESCI control register 2, SCC2
- ESCI control register 3, SCC3
- ESCI status register 1, SCS1
- ESCI status register 2, SCS2

- ESCI data register, SCDR
- ESCI baud rate register, SCBR
- ESCI prescaler register, SCPSC
- ESCI arbiter control register, SCIACTL
- ESCI arbiter data register, SCIADAT

18.9.1 ESCI Control Register 1

ESCI control register 1 (SCC1):
- Enables loop mode operation
- Enables the ESCI
- Controls output polarity
- Controls character length
- Controls ESCI 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 18-9. ESCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation. In loop mode the RxD pin is disconnected from the ESCI, 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 ESCI Bit

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

    1 = ESCI enabled
    0 = ESCI 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 ESCI characters are eight or nine bits long (See **Table 18-5**).The ninth bit can serve as a receiver wakeup signal or as a parity bit. Reset clears the M bit.

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

Table 18-5. Character Format Selection

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

WAKE — Wakeup Condition Bit

This read/write bit determines which condition wakes up the ESCI: a logic 1 (address mark) in the MSB position of a received character or an idle condition on the 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 ESCI starts counting logic 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. 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 ESCI parity function (see **Table 18-5**). When enabled, the parity function inserts a parity bit in the MSB position (see **Table 18-3**). Reset clears the PEN bit.
  1 = Parity function enabled
  0 = Parity function disabled

PTY — Parity Bit

This read/write bit determines whether the ESCI generates and checks for odd parity or even parity (see **Table 18-5**). 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.*

18.9.2 ESCI Control Register 2

ESCI control register 2 (SCC2):

- Enables these CPU interrupt requests:
  - SCTE bit to generate transmitter CPU interrupt requests
  - TC bit to generate transmitter CPU interrupt requests
  - SCRF bit to generate receiver CPU interrupt requests
  - IDLE bit to generate receiver CPU interrupt requests
- Enables the transmitter
- Enables the receiver
- Enables ESCI wakeup
- Transmits ESCI 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 18-10. ESCI Control Register 2 (SCC2)

SCTIE — ESCI Transmit Interrupt Enable Bit

This read/write bit enables the SCTE bit to generate ESCI transmitter CPU interrupt requests. Setting the SCTIE bit in SCC2 enables the SCTE bit to generate 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 ESCI 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 — ESCI Receive Interrupt Enable Bit

This read/write bit enables the SCRF bit to generate ESCI receiver CPU interrupt requests. Setting the SCRIE bit in SCC2 enables the SCRF bit to generate 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 ESCI 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 logic 1s from the transmit shift register to the TxD pin. If software clears the TE bit, the transmitter completes any transmission in progress before the TxD returns to the idle condition (logic 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 ESCI bit (ENSCI) is clear. ENSCI is in ESCI 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 ESCI bit (ENSCI) is clear. ENSCI is in ESCI 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 logic 1. The logic 1 after the break character guarantees recognition of a valid start bit. If SBK remains set, the transmitter continuously transmits break characters with no logic 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 ESCI to send a break character instead of a preamble.*

18.9.3 ESCI Control Register 3

ESCI control register 3 (SCC3):

- Stores the ninth ESCI data bit received and the ninth ESCI data bit to be transmitted.

- Enables these interrupts:
    - Receiver overrun
    - Noise error
    - Framing error
    - Parity error

Address: $0015

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R8	T8	R	R	ORIE	NEIE	FEIE	PEIE
Write:		T8	R	R	ORIE	NEIE	FEIE	PEIE
Reset:	U	0	0	0	0	0	0	0

    = Unimplemented   R = Reserved   U = Unaffected
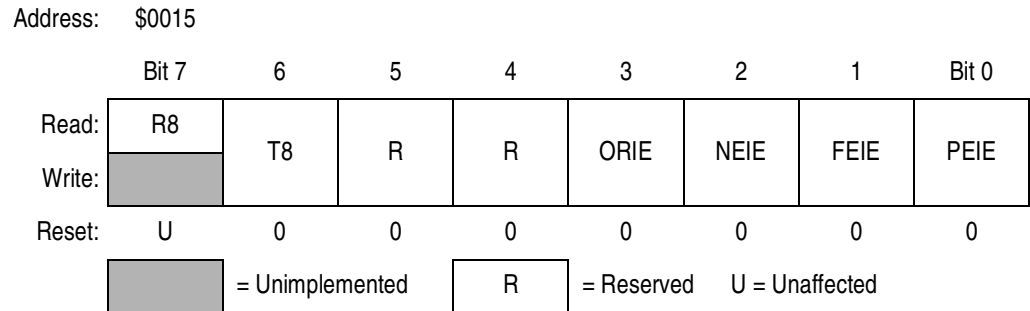
Figure 18-11. ESCI Control Register 3 (SCC3)

R8 — Received Bit 8

  When the ESCI 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 ESCI 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 ESCI 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 clears the T8 bit.

ORIE — Receiver Overrun Interrupt Enable Bit

  This read/write bit enables ESCI error CPU interrupt requests generated by the receiver overrun bit, OR. Reset clears ORIE.
    1 = ESCI error CPU interrupt requests from OR bit enabled
    0 = ESCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

  This read/write bit enables ESCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.
    1 = ESCI error CPU interrupt requests from NE bit enabled
    0 = ESCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables ESCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.
1 = ESCI error CPU interrupt requests from FE bit enabled
0 = ESCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables ESCI receiver CPU interrupt requests generated by the parity error bit, PE. Reset clears PEIE.
1 = ESCI error CPU interrupt requests from PE bit enabled
0 = ESCI error CPU interrupt requests from PE bit disabled

18.9.4 ESCI Status Register 1

ESCI 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 18-12. ESCI Status Register 1 (SCS1)

SCTE — ESCI 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 ESCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an ESCI 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 ESCI transmitter CPU interrupt request if the TCIE bit in SCC2 is also set. TC is cleared automatically 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 — ESCI Receiver Full Bit

This clearable, read-only bit is set when the data in the receive shift register transfers to the ESCI data register. SCRF can generate an ESCI receiver CPU interrupt request. When the SCRIE bit in SCC2 is set the 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 logic 1s appear on the receiver input. IDLE generates an ESCI error 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 ESCI 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 18-13** 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.

NF — Receiver Noise Flag Bit

This clearable, read-only bit is set when the ESCI detects noise on the RxD pin. NF generates an NF 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

NORMAL FLAG CLEARING SEQUENCE

DELAYED FLAG CLEARING SEQUENCE
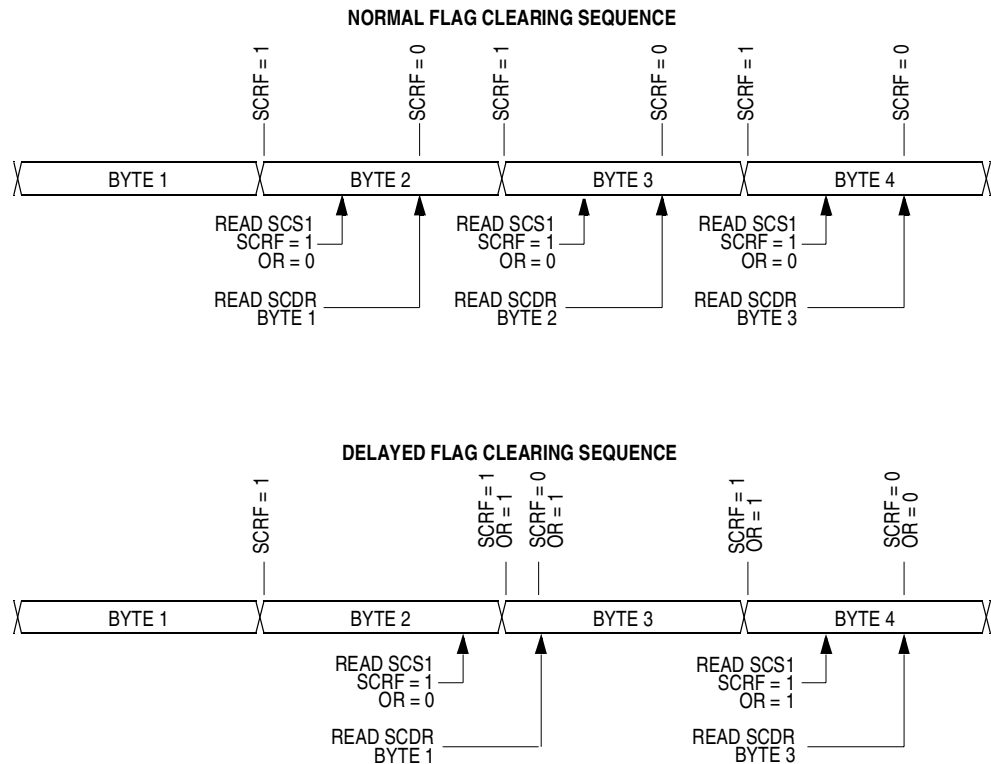
Figure 18-13. Flag Clearing Sequence

FE — Receiver Framing Error Bit

This clearable, read-only bit is set when a logic 0 is accepted as the stop bit. FE generates an ESCI 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 ESCI detects a parity error in incoming data. PE generates a PE 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

18.9.5 ESCI Status Register 2

ESCI status register 2 (SCS2) contains flags to signal these conditions:

- Break character detected
- Incoming data

Address: $0017

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	BKF	RPF
Write:								
Reset:	0	0	0	0	0	0	0	0

                    = Unimplemented

Figure 18-14. ESCI Status Register 2 (SCS2)

BKF — Break Flag Bit

This clearable, read-only bit is set when the ESCI detects a break character on the 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 logic 1s again appear on the 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 logic 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 ESCI module or entering stop mode can show whether a reception is in progress.

   1 = Reception in progress
   0 = No reception in progress

18.9.6 ESCI Data Register

The ESCI 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 ESCI 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 18-15. ESCI Data Register (SCDR)

R7/T7:R0/T0 — Receive/Transmit Data Bits

Reading address $0018 accesses the read-only received data bits, R7:R0. Writing to address $0018 writes the data to be transmitted, T7:T0. Reset has no effect on the ESCI data register.

NOTE: *Do not use read-modify-write instructions on the ESCI data register.*

18.9.7 ESCI Baud Rate Register

The ESCI baud rate register (SCBR) together with the ESCI prescaler register selects the baud rate for both the receiver and the transmitter.

NOTE: *There are two prescalers available to adjust the baud rate. One in the ESCI baud rate register and one in the ESCI prescaler register.*

Address: $0019

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	R	LINR	SCP1	SCP0	R	SCR2	SCR1	SCR0
Reset:	0	0	0	0	0	0	0	0

    = Unimplemented   R = Reserved

Figure 18-16. ESCI Baud Rate Register (SCBR)

LINR — LIN Receiver Bits

This read/write bit selects the enhanced ESCI features for the local interconnect network (LIN) protocol as shown in **Table 18-6**. Reset clears LINR.

Table 18-6. ESCI LIN Control Bits

LINR	M	Functionality
0	X	Normal ESCI functionality
1	0	11-bit break detect enabled for LIN receiver
1	1	12-bit break detect enabled for LIN receiver

SCP1 and SCP0 — ESCI Baud Rate Register Prescaler Bits

These read/write bits select the baud rate register prescaler divisor as shown in **Table 18-7**. Reset clears SCP1 and SCP0.

Table 18-7. ESCI Baud Rate Prescaling

SCP[1:0]	Baud Rate Register Prescaler Divisor (BPD)
0 0	1
0 1	3
1 0	4
1 1	13

SCR2–SCR0 — ESCI Baud Rate Select Bits

These read/write bits select the ESCI baud rate divisor as shown in **Table 18-8**. Reset clears SCR2–SCR0.

Table 18-8. ESCI Baud Rate Selection

SCR[2:1:0]	Baud Rate Divisor (BD)
0 0 0	1
0 0 1	2
0 1 0	4
0 1 1	8
1 0 0	16
1 0 1	32
1 1 0	64
1 1 1	128

18.9.8 ESCI Prescaler Register

The ESCI prescaler register (SCPSC) together with the ESCI baud rate register selects the baud rate for both the receiver and the transmitter.

NOTE: *There are two prescalers available to adjust the baud rate. One in the ESCI baud rate register and one in the ESCI prescaler register.*

Address: $0017

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PDS2	PDS1	PDS0	PSSB4	PSSB3	PSSB2	PSSB1	PSSB0
Reset:	0	0	0	0	0	0	0	0

Figure 18-17. ESCI Prescaler Register (SCPSC)

PDS2–PDS0 — Prescaler Divisor Select Bits

These read/write bits select the prescaler divisor as shown in **Table 18-9**. Reset clears PDS2–PDS0.

NOTE: *The setting of '000' will bypass this prescaler. It is not recommended to bypass the prescaler while ENSCI is set, because the switching is not glitch free.*

Table 18-9. ESCI Prescaler Division Ratio

PS[2:1:0]	Prescaler Divisor (PD)
0 0 0	Bypass this prescaler
0 0 1	2
0 1 0	3
0 1 1	4
1 0 0	5
1 0 1	6
1 1 0	7
1 1 1	8

PSSB4–PSSB0 — Clock Insertion Select Bits

These read/write bits select the number of clocks inserted in each 32 output cycle frame to achieve more timing resolution on the **average** prescaler frequency as shown in **Table 18-10**. Reset clears PSSB4–PSSB0.

Table 18-10. ESCI Prescaler Divisor Fine Adjust

PSSB[4:3:2:1:0]	Prescaler Divisor Fine Adjust (PDFA)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875
1 0 0 0 0	16/32 = 0.5
1 0 0 0 1	17/32 = 0.53125
1 0 0 1 0	18/32 = 0.5625
1 0 0 1 1	19/32 = 0.59375
1 0 1 0 0	20/32 = 0.625
1 0 1 0 1	21/32 = 0.65625
1 0 1 1 0	22/32 = 0.6875

Table 18-10. ESCI Prescaler Divisor Fine Adjust (Continued)

PSSB[4:3:2:1:0]	Prescaler Divisor Fine Adjust (PDFA)
1 0 1 1 1	23/32 = 0.71875
1 1 0 0 0	24/32 = 0.75
1 1 0 0 1	25/32 = 0.78125
1 1 0 1 0	26/32 = 0.8125
1 1 0 1 1	27/32 = 0.84375
1 1 1 0 0	28/32 = 0.875
1 1 1 0 1	29/32 = 0.90625
1 1 1 1 0	30/32 = 0.9375
1 1 1 1 1	31/32 = 0.96875

Use the following formula to calculate the ESCI baud rate:

$$\text{Baud rate} = \frac{f_{Bus}}{64 \times BPD \times BD \times (PD + PDFA)}$$

where:

f_{Bus} = Bus frequency

BPD = Baud rate register prescaler divisor

BD = Baud rate divisor

PD = Prescaler divisor

PDFA = Prescaler divisor fine adjust

Table 18-11 shows the ESCI baud rates that can be generated with a 4.9152-MHz bus frequency.

Table 18-11. ESCI Baud Rate Selection Examples

PS[2:1:0]	PSSB[4:3:2:1:0]	SCP[1:0]	Prescaler Divisor (BPD)	SCR[2:1:0]	Baud Rate Divisor (BD)	Baud Rate (f_{Bus}= 4.9152 MHz)
0 0 0	X X X X X	0 0	1	0 0 0	1	76,800
1 1 1	0 0 0 0 0	0 0	1	0 0 0	1	9600
1 1 1	0 0 0 0 1	0 0	1	0 0 0	1	9562.65
1 1 1	0 0 0 1 0	0 0	1	0 0 0	1	9525.58
1 1 1	1 1 1 1 1	0 0	1	0 0 0	1	8563.07
0 0 0	X X X X X	0 0	1	0 0 1	2	38,400
0 0 0	X X X X X	0 0	1	0 1 0	4	19,200
0 0 0	X X X X X	0 0	1	0 1 1	8	9600
0 0 0	X X X X X	0 0	1	1 0 0	16	4800
0 0 0	X X X X X	0 0	1	1 0 1	32	2400
0 0 0	X X X X X	0 0	1	1 1 0	64	1200
0 0 0	X X X X X	0 0	1	1 1 1	128	600
0 0 0	X X X X X	0 1	3	0 0 0	1	25,600
0 0 0	X X X X X	0 1	3	0 0 1	2	12,800
0 0 0	X X X X X	0 1	3	0 1 0	4	6400
0 0 0	X X X X X	0 1	3	0 1 1	8	3200
0 0 0	X X X X X	0 1	3	1 0 0	16	1600
0 0 0	X X X X X	0 1	3	1 0 1	32	800
0 0 0	X X X X X	0 1	3	1 1 0	64	400
0 0 0	X X X X X	0 1	3	1 1 1	128	200
0 0 0	X X X X X	1 0	4	0 0 0	1	19,200
0 0 0	X X X X X	1 0	4	0 0 1	2	9600
0 0 0	X X X X X	1 0	4	0 1 0	4	4800
0 0 0	X X X X X	1 0	4	0 1 1	8	2400
0 0 0	X X X X X	1 0	4	1 0 0	16	1200
0 0 0	X X X X X	1 0	4	1 0 1	32	600
0 0 0	X X X X X	1 0	4	1 1 0	64	300
0 0 0	X X X X X	1 0	4	1 1 1	128	150
0 0 0	X X X X X	1 1	13	0 0 0	1	5908
0 0 0	X X X X X	1 1	13	0 0 1	2	2954
0 0 0	X X X X X	1 1	13	0 1 0	4	1477
0 0 0	X X X X X	1 1	13	0 1 1	8	739
0 0 0	X X X X X	1 1	13	1 0 0	16	369
0 0 0	X X X X X	1 1	13	1 0 1	32	185
0 0 0	X X X X X	1 1	13	1 1 0	64	92
0 0 0	X X X X X	1 1	13	1 1 1	128	46

18.10 ESCI Arbiter

The ESCI module comprises an arbiter module designed to support software for communication tasks as bus arbitration, baud rate recovery and break time detection. The arbiter module consists of an 9-bit counter with 1-bit overflow and control logic. The CPU can control operation mode via the ESCI arbiter control register (SCIACTL).

18.10.1 ESCI Arbiter Control Register

Address: $000A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AM1	ALOST	AM0	ACLK	AFIN	ARUN	AROVFL	ARD8
Write:								
Reset:	0	0	0	0	0	0	0	0

☐ = Unimplemented
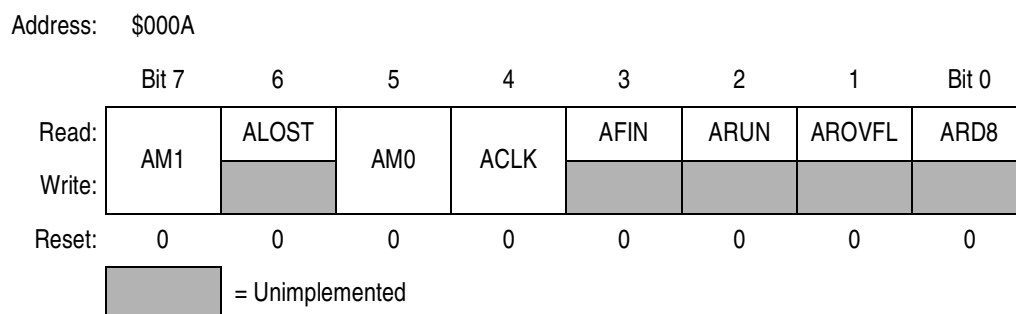
Figure 18-18. ESCI Arbiter Control Register (SCIACTL)

AM1 and AM0 — Arbiter Mode Select Bits

These read/write bits select the mode of the arbiter module as shown in **Table 18-12**. Reset clears AM1 and AM0.

Table 18-12. ESCI Arbiter Selectable Modes

AM[1:0]	ESCI Arbiter Mode
0 0	Idle / counter reset
0 1	Bit time measurement
1 0	Bus arbitration
1 1	Reserved / do not use

ALOST — Arbitration Lost Flag

This read-only bit indicates loss of arbitration. Clear ALOST by writing a logic 0 to AM1. Reset clears ALOST.

ACLK — Arbiter Counter Clock Select Bit

This read/write bit selects the arbiter counter clock source.
Reset clears ACLK.
1 = Arbiter counter is clocked with one half of the ESCI input clock
generated by the ESCI prescaler
0 = Arbiter counter is clocked with one half of the bus clock

AFIN— Arbiter Bit Time Measurement Finish Flag

This read-only bit indicates bit time measurement has finished. Clear
AFIN by writing any value to SCIACTL. Reset clears AFIN.
1 = Bit time measurement has finished
0 = Bit time measurement not yet finished

ARUN— Arbiter Counter Running Flag

This read-only bit indicates the arbiter counter is running. Reset
clears ARUN.
1 = Arbiter counter running
0 = Arbiter counter stopped

AROVFL— Arbiter Counter Overflow Bit

This read-only bit indicates an arbiter counter overflow. Clear
AROVFL by writing any value to SCIACTL. Writing logic 0s to AM1
and AM0 resets the counter keeps it in this idle state. Reset clears
AROVFL.
1 = Arbiter counter overflow has occurred
0 = No arbiter counter overflow has occurred

ARD8— Arbiter Counter MSB

This read-only bit is the MSB of the 9-bit arbiter counter. Clear ARD8
by writing any value to SCIACTL. Reset clears ARD8.

18.10.2 ESCI Arbiter Data Register

Address: $000B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ARD7	ARD6	ARD5	ARD4	ARD3	ARD2	ARD1	ARD0
Write:								
Reset:	0	0	0	0	0	0	0	0

    = Unimplemented
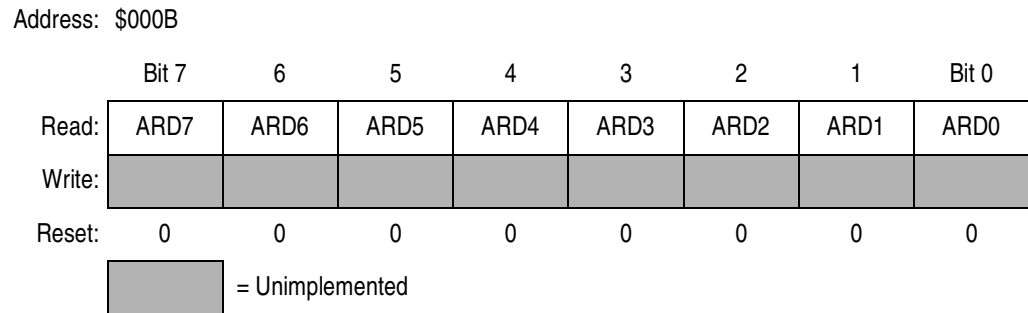
Figure 18-19. ESCI Arbiter Data Register (SCIADAT)

ARD7–ARD0 — Arbiter Least Significant Counter Bits

These read-only bits are the eight LSBs of the 9-bit arbiter counter. Clear ARD7–ARD0 by writing any value to SCIACTL. Writing logic 0s to AM1 and AM0 permanently resets the counter and keeps it in this idle state. Reset clears ARD7–ARD0.

18.10.3 Bit Time Measurement

Two bit time measurement modes, described here, are available according to the state of ACLK.

1. ACLK = 0 — The counter is clocked with one half of the bus clock. The counter is started when a falling edge on the RxD pin is detected. The counter will be stopped on the next falling edge. ARUN is set while the counter is running, AFIN is set on the second falling edge on RxD (for instance, the counter is stopped). This mode is used to recover the received baud rate. See **Figure 18-20**.

2. ACLK = 1 — The counter is clocked with one half of the ESCI input clock generated by the ESCI prescaler. The counter is started when a logic 0 is detected on RxD (see **Figure 18-21**). A logic 0 on RxD on enabling the bit time measurement with ACLK = 1 leads to immediate start of the counter (see **Figure 18-22**). The counter will be stopped on the next rising edge of RxD. This mode is used to measure the length of a received break.
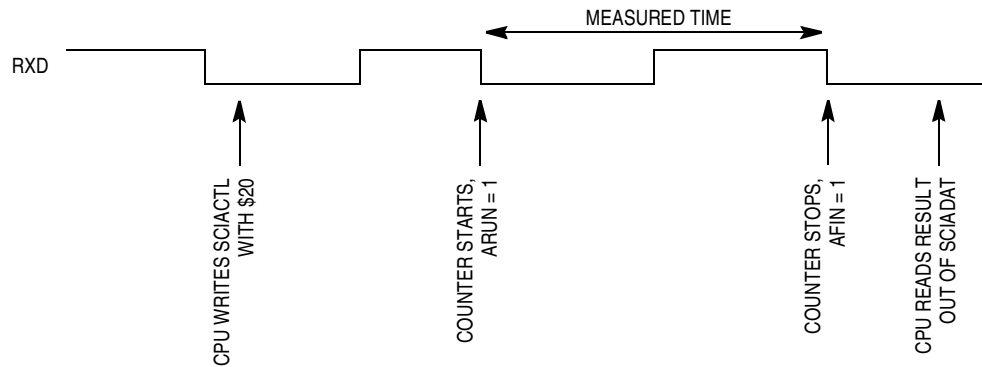
Figure 18-20. Bit Time Measurement with ACLK = 0
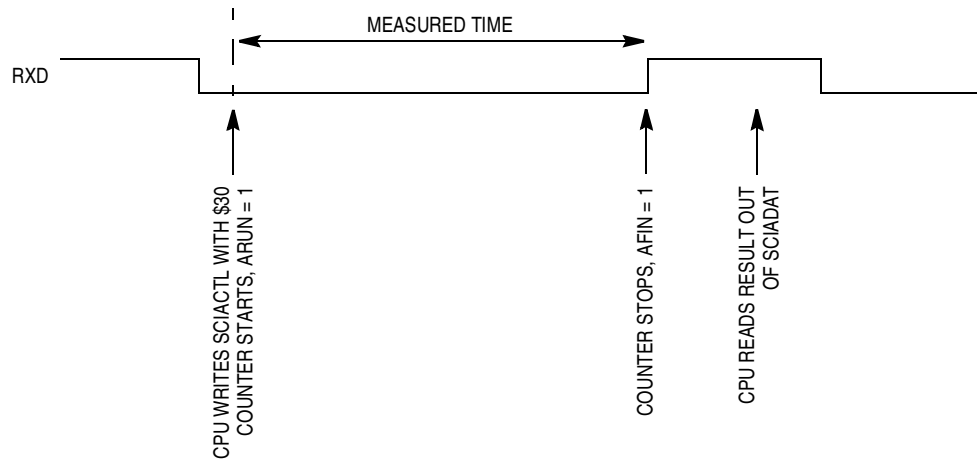
Figure 18-21. Bit Time Measurement with ACLK = 1, Scenario A
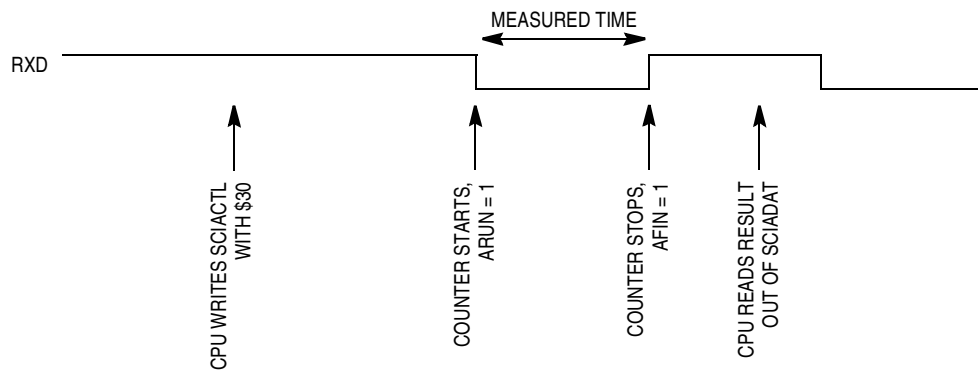
Figure 18-22. Bit Time Measurement with ACLK = 1, Scenario B

18.10.4 Arbitration Mode

If AM[1:0] is set to 10, the arbiter module operates in arbitration mode. On every rising edge of SCI_TxD (output of the ESCI module, internal chip signal), the counter is started. When the counter reaches $38 (ACLK = 0) or $08 (ACLK = 1), RxD is statically sensed. If in this case, RxD is sensed low (for example, another bus is driving the bus dominant) ALOST is set. As long as ALOST is set, the TxD pin is forced to 1, resulting in a seized transmission.

If SCI_TxD is sensed logic 0 without having sensed a logic 0 before on RxD, the counter will be reset, arbitration operation will be restarted after the next rising edge of SCI_TxD.

Section 19. System Integration Module (SIM)

19.1 Contents

19.2 Introduction

This section describes the system integration module (SIM). Together with the central processor unit (CPU), the SIM controls all microconroller unit (MCU) activities. A block diagram of the SIM is shown in **Figure 19-1**. **Table 19-1** is a summary of the SIM input/output (I/O) registers. The SIM is a system state controller that coordinates CPU and exception timing.

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 control:
    - Acknowledge timing
    - Arbitration control timing
    - Vector address generation

- CPU enable/disable timing

- Modular architecture expandable to 128 interrupt sources

Table 19-1 shows the internal signal names used in this section.
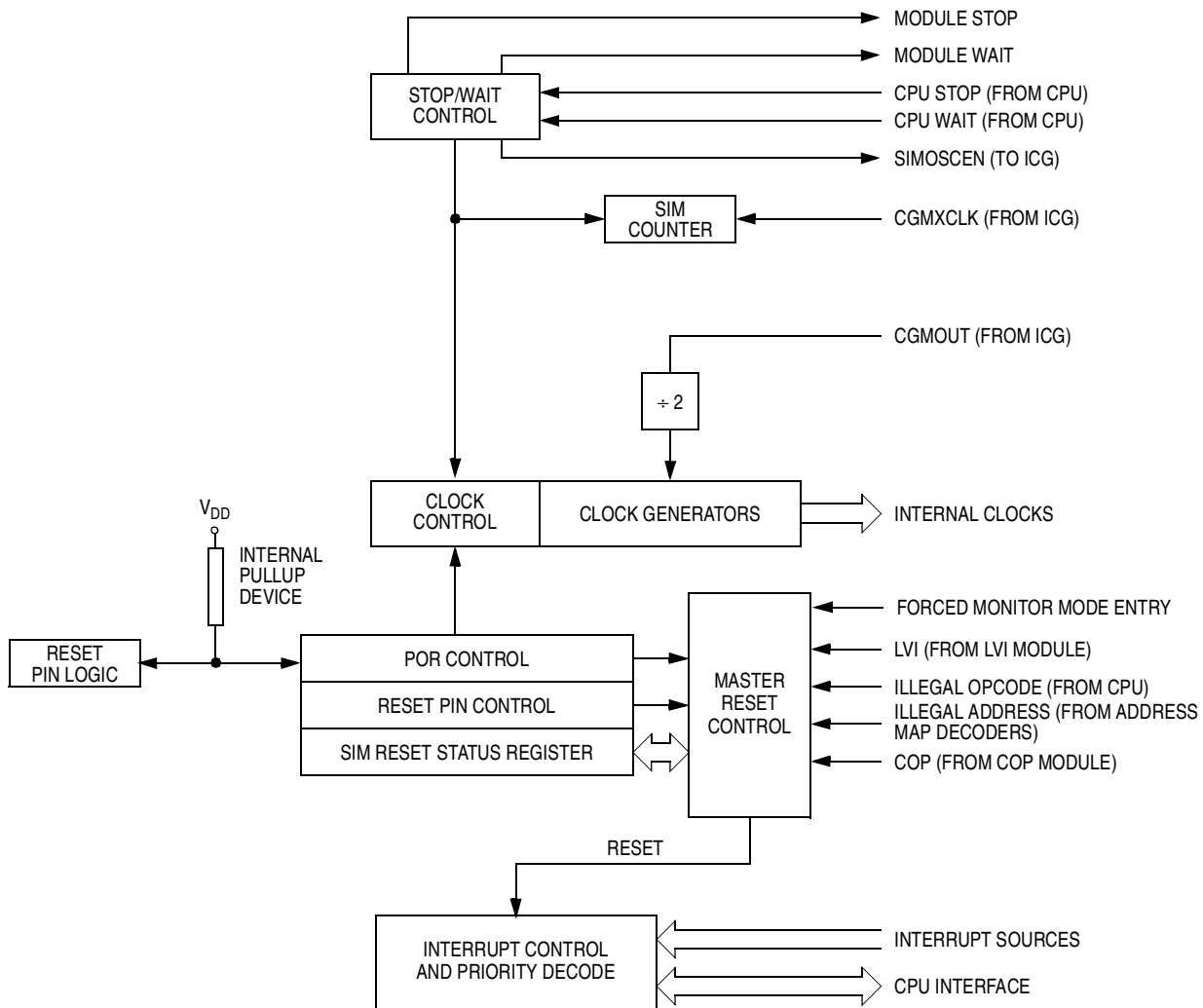
Figure 19-1. SIM Block Diagram

Table 19-1. Signal Name Conventions

Signal Name	Description
CGMXCLK	Selected clock source from internal clock generator module (ICG)
CGMOUT	Clock output from ICG 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/$\overline{\text{W}}$	Read/write signal

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE00	SIM Break Status Register (SBSR) See page 316.	Read:	R	R	R	R	R	R	SBSW	R
		Write:							NOTE	
		Reset:	0	0	0	0	0	0	0	0

Note: Writing a logic 0 clears SBSW.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE01	SIM Reset Status Register (SRSR) See page 318.	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
$FE02	SIM Upper Byte Address Register (SUBAR)	Read:	R	R	R	R	R	R	R	R
		Write:								
		Reset:								
$FE03	SIM Break Flag Control Register (SBFCR) See page 319.	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
$FE04	Interrupt Status Register 1 (INT1) See page 311.	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 311.	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 312.	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

☐ = Unimplemented

Figure 19-2. SIM I/O Register Summary

19.3 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 19-3**. This clock originates from either an external oscillator or from the internal clock generator.
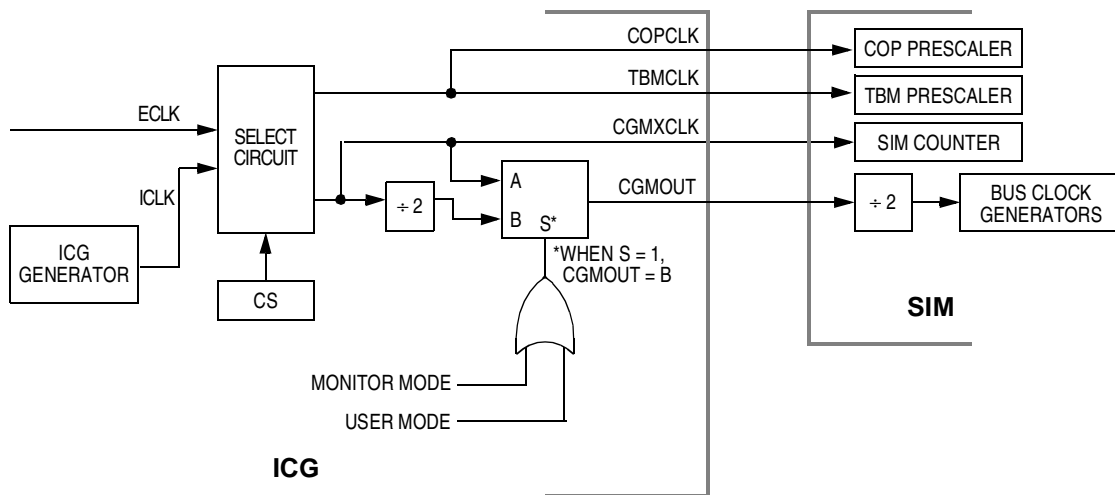
Figure 19-3. System Clock Signals

19.3.1 Bus Timing

In user mode, the internal bus frequency is the internal clock generator output (CGMXCLK) divided by four.

19.3.2 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 RST pin is driven low by the SIM during this entire period. The IBUS clocks start upon completion of the timeout.

19.3.3 Clocks in Stop Mode and Wait Mode

Upon exit from stop mode by an interrupt, break, 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 **19.7.2 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.

19.4 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 **19.5 SIM Counter**), but an external reset does not. Each of the resets sets a corresponding bit in the SIM reset status register (SRSR). See **19.8 SIM Registers**.

19.4.1 External Pin Reset

The $\overline{RST}$ pin circuit includes an internal pullup device. Pulling the asynchronous $\overline{RST}$ pin low halts all processing. The PIN bit of the SIM reset status register (SRSR) is set as long as $\overline{RST}$ 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 19-2** for details. **Figure 19-4** shows the relative timing.

Table 19-2. PIN Bit Set Timing

Reset Type	Number of Cycles Required to Set PIN
POR/LVI	4163 (4096 + 64 + 3)
All others	67 (64 + 3)
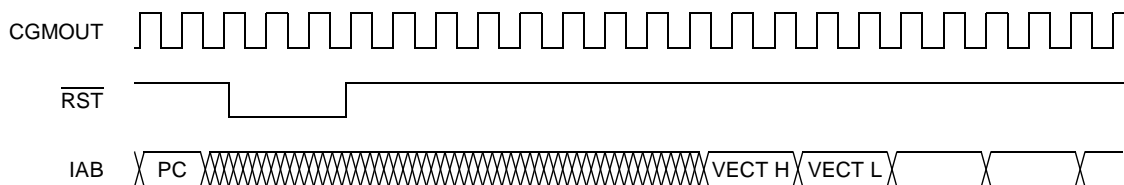
Figure 19-4. External Reset Timing

19.4.2 Active Resets from Internal Sources

All internal reset sources actively pull the $\overline{RST}$ pin low for 32 CGMXCLK cycles to allow resetting of external peripherals. The internal reset signal IRST continues to be asserted for an additional 32 cycles. See **Figure 19-5**. An internal reset can be caused by an illegal address, illegal opcode, COP timeout, LVI, or POR. See **Figure 19-6**.

NOTE: *For LVI or POR resets, the SIM cycles through 4096 + 32 CGMXCLK cycles during which the SIM forces the $\overline{RST}$ pin low. The internal reset signal then follows the sequence from the falling edge of $\overline{RST}$ shown in* **Figure 19-5**.
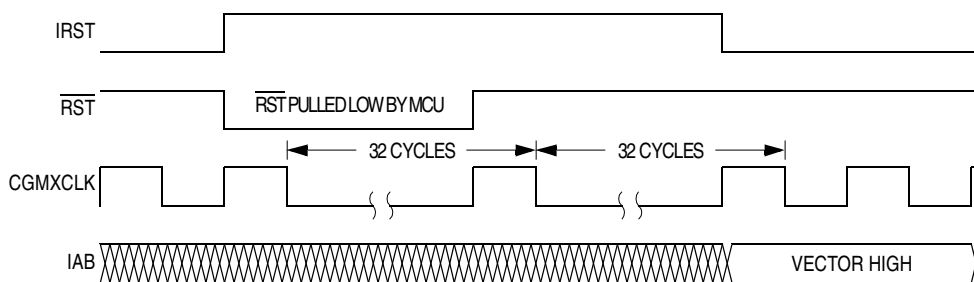
Figure 19-5. Internal Reset Timing

The COP reset is asynchronous to the bus clock.
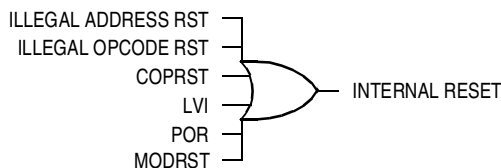
Figure 19-6. Sources of Internal Reset

The active reset feature allows the part to issue a reset to peripherals and other chips within a system built around the MCU.

19.4.2.1 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.
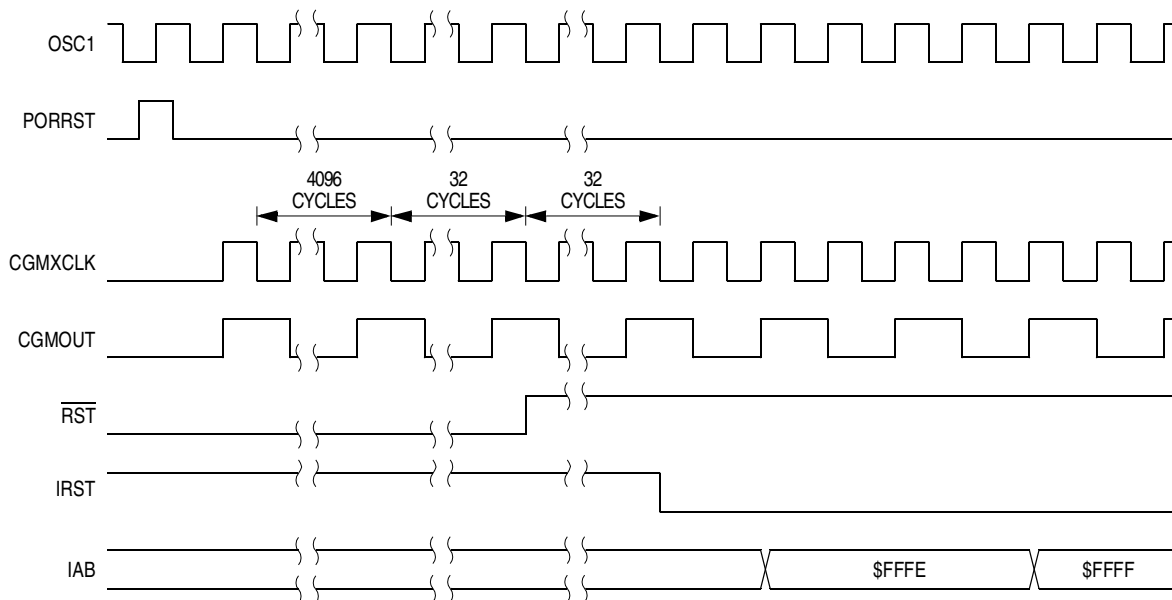
Figure 19-7. POR Recovery

19.4.2.2 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 $\overline{RST}$ pin for all internal reset sources.

The COP module is disabled if the $\overline{RST}$ pin or the $\overline{IRQ}$ pin is held at $V_{DD} + V_{HI}$ 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 $\overline{RST}$ or the $\overline{IRQ}$ pin. This prevents the COP from becoming disabled as a result of external noise. During a break state, $V_{DD} + V_{HI}$ on the $\overline{RST}$ pin disables the COP module.

19.4.2.3 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 mask option register is logic 0, the SIM treats the STOP instruction as an illegal opcode and causes an illegal opcode reset. The SIM actively pulls down the $\overline{RST}$ pin for all internal reset sources.

19.4.2.4 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 $\overline{RST}$ pin for all internal reset sources.

19.4.2.5 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 LVI_{TRIPF} voltage. The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{RST}$) 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 $\overline{RST}$ pin for all internal reset sources.

19.4.2.6 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 **15.4.1 Entering Monitor Mode**.) When MODRST gets asserted, an internal reset occurs. The SIM actively pulls down the $\overline{RST}$ pin for all internal reset sources.

19.5 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 is 13 bits long.

19.5.1 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.

19.5.2 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 mask option register. If the SSREC bit is a logic 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 canned oscillators that do not require long startup times from stop mode. External crystal applications should use the full stop recovery time, that is, with SSREC cleared.

19.5.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. See **19.7.2 Stop Mode** for details. The SIM counter is free-running after all reset states. See **19.4.2 Active Resets from Internal Sources** for counter control and internal reset recovery sequences.

19.6 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

19.6.1 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 19-8** shows interrupt entry timing. **Figure 19-9** shows 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 19-10**.
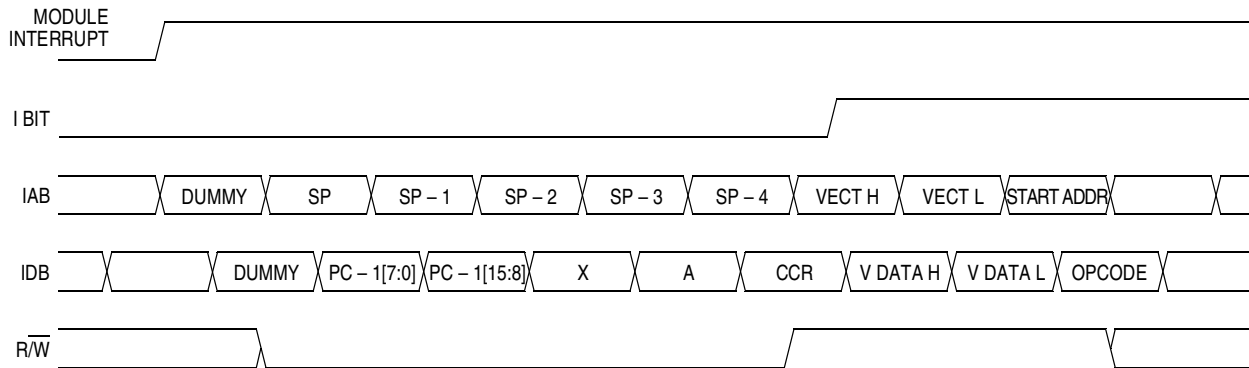
Figure 19-8. **Interrupt Entry Timing**
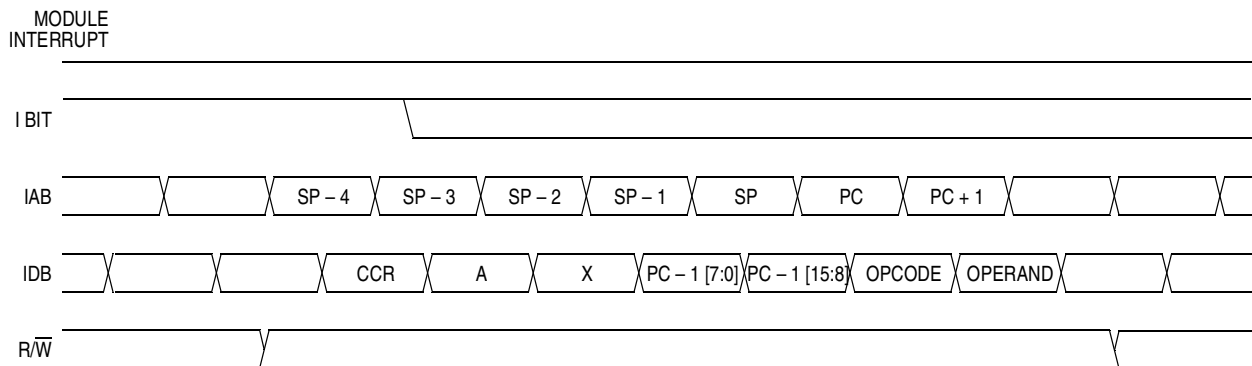
Figure 19-9. Interrupt Recovery Timing
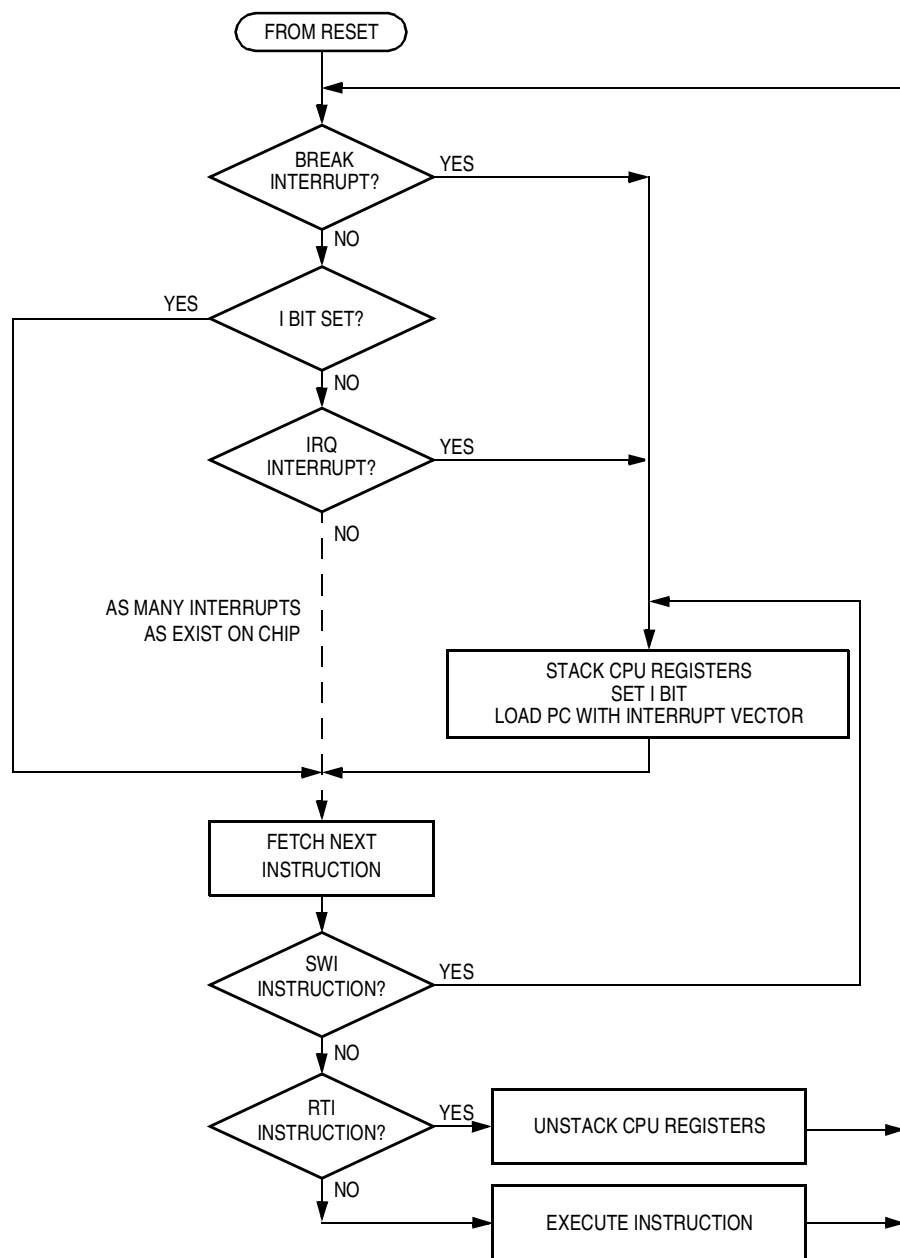
Figure 19-10. Interrupt Processing

19.6.1.1 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 19-11** 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.
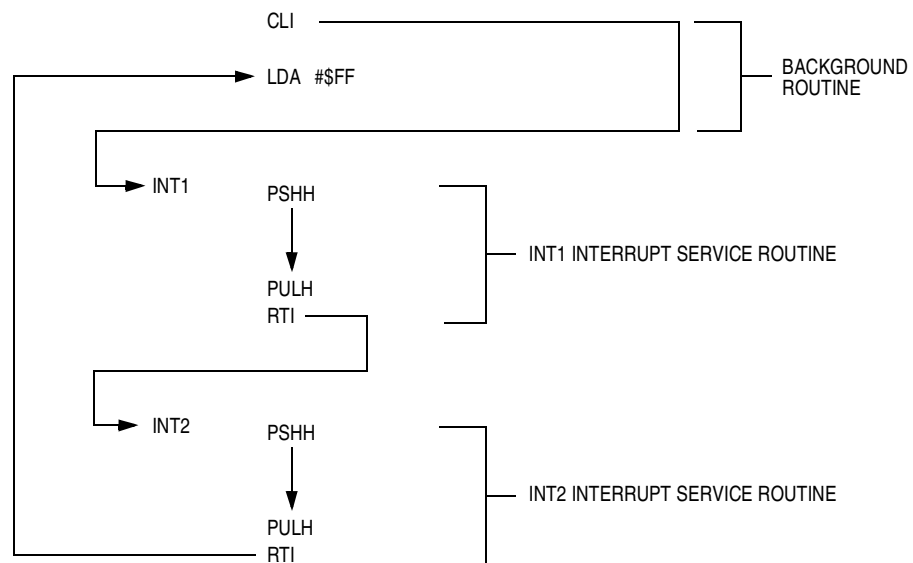
Figure 19-11. **Interrupt Recognition Example**

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.*

19.6.1.2 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.*

19.6.1.3 Interrupt Status Registers

The flags in the interrupt status registers identify maskable interrupt sources. **Table 19-3** summarizes the interrupt sources and the interrupt status register flags that they set. The interrupt status registers can be useful for debugging.

Table 19-3. Interrupt Sources

Priority	Interrupt Source	Interrupt Status Register Flag
Highest	Reset	—
	SWI instruction	—
	$\overline{\text{IRQ}}$ pin	I1
	ICG 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 19-12. 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 19-3**.
    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 19-13. 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 19-3**.
    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 19-14. Interrupt Status Register 3 (INT3)

Bits 7–2 — Always read 0

I16–I15 — Interrupt Flags 16–15

These flags indicate the presence of an interrupt request from the source shown in **Table 19-3**.
1 = Interrupt request present
0 = No interrupt request present

19.6.2 Reset

All reset sources always have equal and highest priority and cannot be arbitrated.

19.6.3 Break Interrupts

The break module can stop normal program flow at a software-programmable break point by asserting its break interrupt output (see **Section 22. Timer Interface Module (TIM)**). 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.

19.6.4 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.

19.7 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.

19.7.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. **Figure 19-15** shows the timing for wait mode entry.

A module that is active during wait mode can wake up 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. 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 mask option register is logic 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.
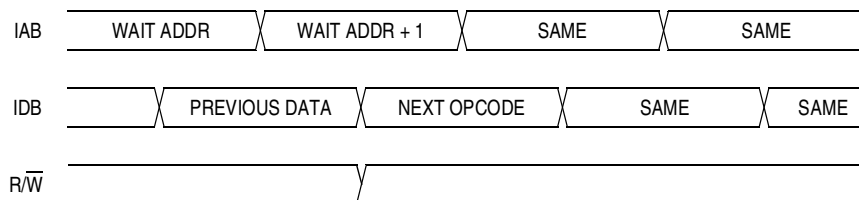
Note: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 19-15. Wait Mode Entry Timing

Figure 19-16 and **Figure 19-17** show the timing for WAIT recovery.
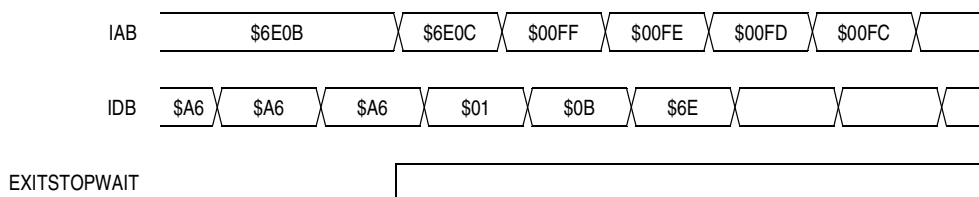
Note: EXITSTOPWAIT = $\overline{\text{RST}}$ pin, CPU interrupt, or break interrupt

Figure 19-16. Wait Recovery from Interrupt or Break
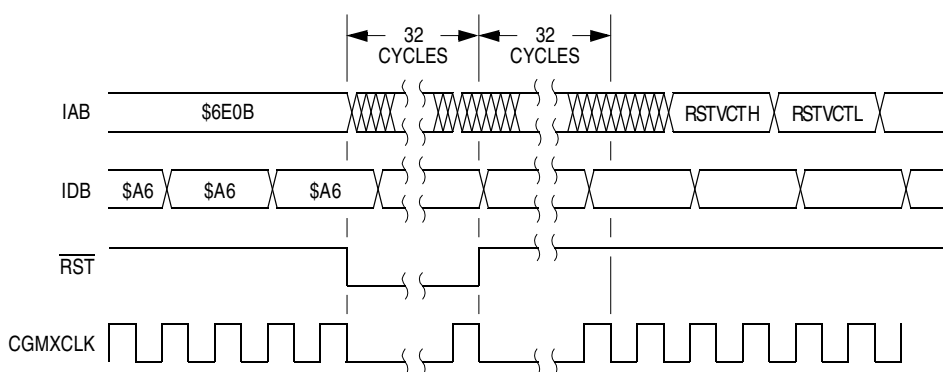
Figure 19-17. Wait Recovery from Internal Reset

19.7.2 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 or break also 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 mask option register (MOR). 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.*

A break interrupt during stop mode sets the SIM break stop/wait bit (SBSW) in the SIM break status register (SBSR).

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 19-18** shows stop mode entry timing.

NOTE: *To minimize stop current, all pins configured as inputs should be driven to a logic 1 or logic 0.*
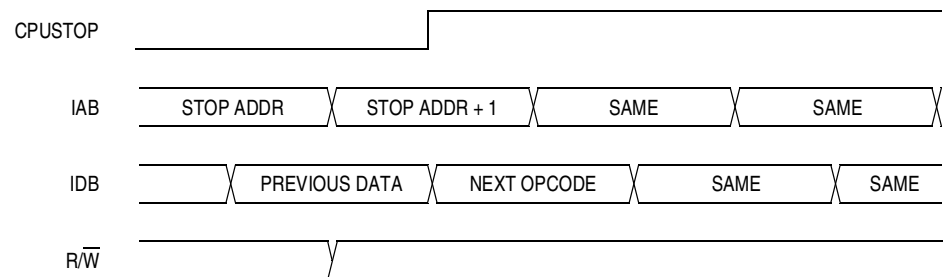
Note: Previous data can be operand data or the STOP opcode, depending on the last instruction.

Figure 19-18. Stop Mode Entry Timing
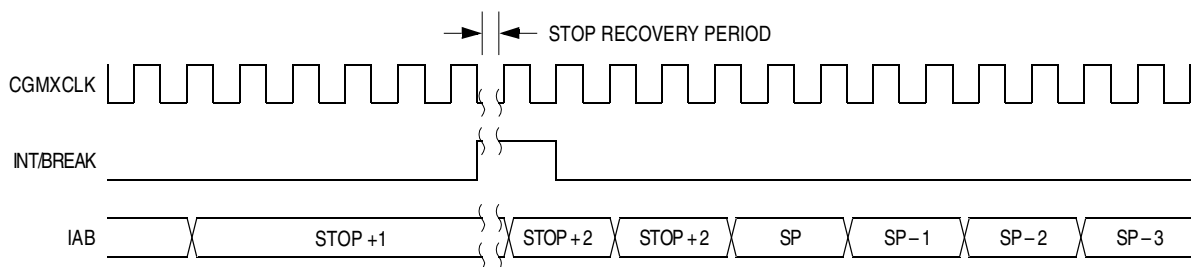
Figure 19-19. Stop Mode Recovery from Interrupt or Break

19.8 SIM Registers

The SIM has three memory-mapped registers. **Table 19-4** shows the mapping of these registers.

Table 19-4. SIM Registers

Address	Register	Access Mode
$FE00	SBSR	User
$FE01	SRSR	User
$FE03	SBFCR	User

19.8.1 SIM Break Status Register

The SIM break status register (SBSR) contains a flag to indicate that a break caused an exit from stop mode or wait 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

Note: 1. Writing a logic 0 clears SBSW.

Figure 19-20. SIM Break Status Register (SBSR)

SBSW — SIM Break Stop/Wait

This status bit is useful in applications requiring a return to wait or stop mode after exiting from a break interrupt. Clear SBSW by writing a logic 0 to it. Reset clears SBSW.

 1 = Stop mode or wait mode was exited by break interrupt.
 0 = Stop mode or wait mode was not exited by break interrupt.

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. The following code is an example of this. Writing 0 to the SBSW bit clears it.

```
;This code works if the H register has been pushed onto the stack in the break
;service routine software. This code should be executed at the end of the break
;service routine software.

 HIBYTE   EQU     5

 LOBYTE   EQU     6

;         If not SBSW, do RTI

         BRCLR   SBSW,SBSR, RETURN    ;See if wait mode or stop mode was exited by
                                      ;break.

         TST     LOBYTE,SP            ;If RETURNLO is not zero,

         BNE     DOLO                 ;then just decrement low byte.

         DEC     HIBYTE,SP            ;Else deal with high byte, too.

 DOLO    DEC     LOBYTE,SP            ;Point to WAIT/STOP opcode.

 RETURN  PULH                         ;Restore H register.
         RTI
```

19.8.2 SIM Reset Status Register

This register contains six flags that show the source of the last reset provided all previous reset status bits have been cleared. Clear the SIM reset status register by reading it. A power-on reset sets the POR bit and clears all other bits in the register.

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 19-21. 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

19.8.3 SIM Break Flag Control Register

The SIM break control register 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
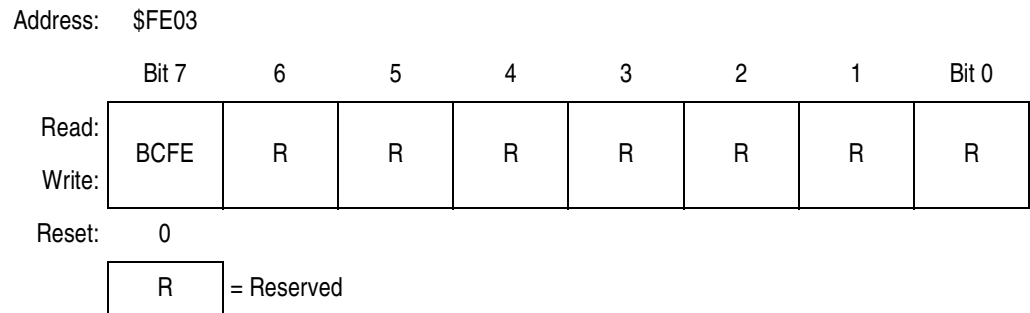
Figure 19-22. SIM Break Flag Control Register (SBFCR)

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

System Integration Module (SIM)

Section 20. Serial Peripheral Interface Module (SPI)

20.1 Contents

20.2 Introduction

This section describes the serial peripheral interface (SPI) module, which allows full-duplex, synchronous, serial communications with peripheral devices.

20.3 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^2C (inter-integrated circuit) compatibility

- I/O (input/output) port bit(s) software configurable with pullup device(s) if configured as input port bit(s)

20.4 Pin Name Conventions

The text that follows describes the SPI. The SPI I/O pin names are $\overline{SS}$ (slave select), SPSCK (SPI serial clock), CGND (clock ground), MOSI (master out slave in), and MISO (master in/slave out). The SPI shares four I/O pins with four parallel I/O ports.

The full names of the SPI I/O pins are shown in **Table 20-1**. The generic pin names appear in the text that follows.

Table 20-1. Pin Name Conventions

SPI Generic Pin Names:		MISO	MOSI	$\overline{SS}$	SPSCK	CGND
Full SPI Pin Names:	SPI	PTD1/$\overline{KBD1}$	PTD2/$\overline{KBD2}$	PTD0/KBD0	PTD3/$\overline{KBD3}$	V_{SS}

20.5 Functional Description

Figure 20-1 summarizes the SPI I/O registers and **Figure 20-2** shows the structure of the SPI module.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0010	SPI Control Register (SPCR) See page 345.	Read:	SPRIE	DMAS	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 347.	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 350.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:				Unaffected by reset				

= Unimplemented

Figure 20-1. SPI I/O Register Summary
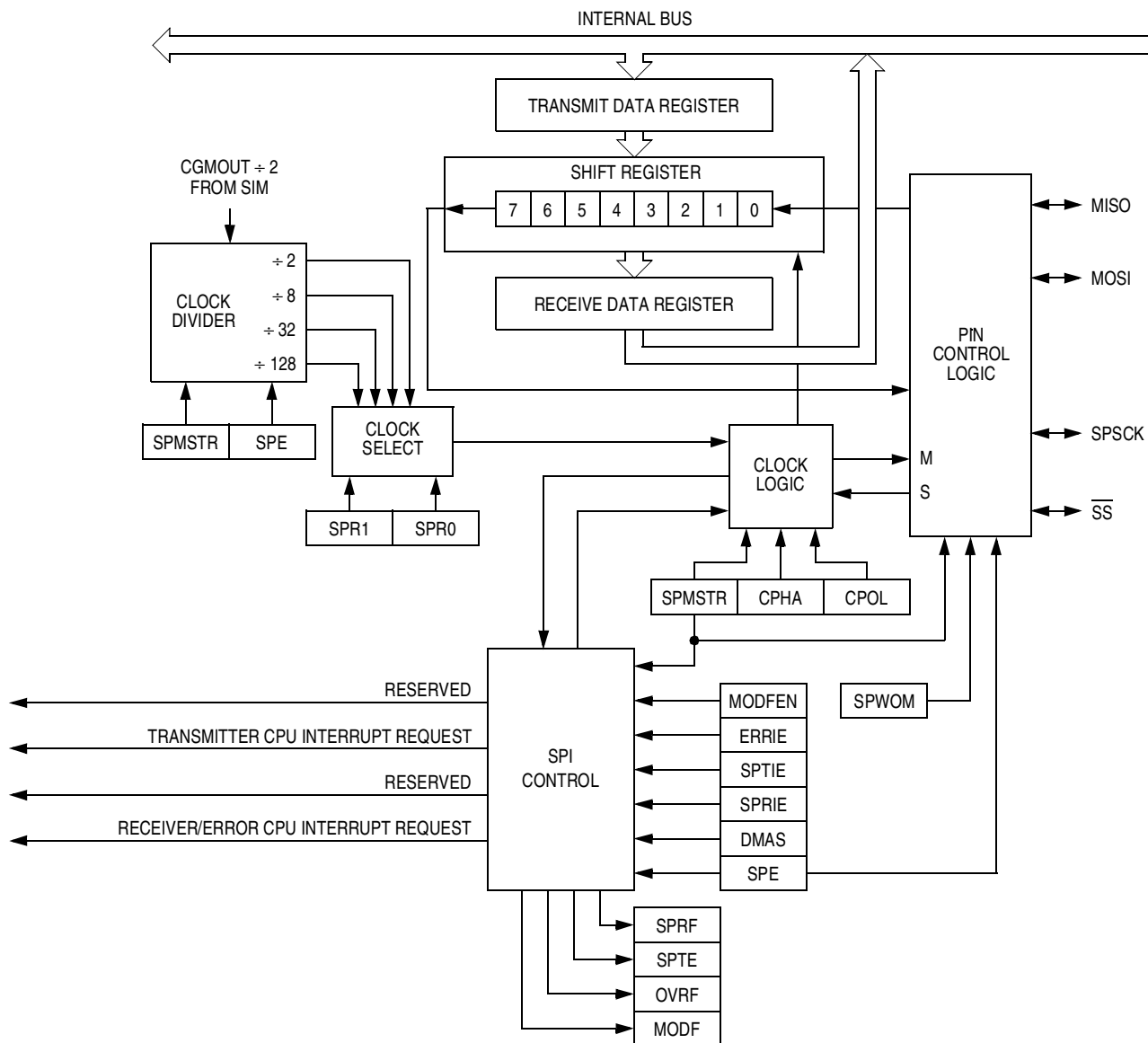
Figure 20-2. SPI Module Block Diagram

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. See **16.5.3 Port C Input Pullup Enable Register**.

The following paragraphs describe the operation of the SPI module.

20.5.1 Master Mode

The SPI operates in master mode when the SPI master bit, SPMSTR, is set.

NOTE: *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* **20.14.1 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 20-3**.
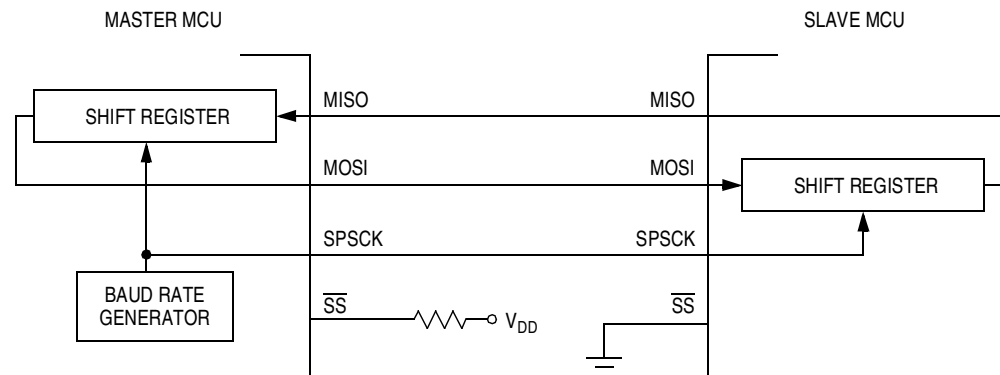
Figure 20-3. Full-Duplex Master-Slave Connections

The SPR1 and SPR0 bits control the baud rate generator and determine the speed of the shift register. (See **20.14.2 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 clears the SPTE bit.

20.5.2 Slave Mode

The SPI operates in slave mode when the SPMSTR bit 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 at logic 0. $\overline{SS}$ must remain low until the transmission is complete. See **20.8.2 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 $\overline{SS}$ starts a transmission. See **20.6 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.*

20.6 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.

20.6.1 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).*

20.6.2 Transmission Format When CPHA = 0

Figure 20-4 shows an SPI transmission in which CPHA is logic 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 $\overline{SS}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{SS}$) is at logic 0, so that only the selected slave drives to the master. The $\overline{SS}$ pin of the master is not shown but is assumed to be inactive. The $\overline{SS}$ pin of the master must be high or must be reconfigured as general-purpose I/O not affecting the SPI. (See **20.8.2 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 $\overline{SS}$ pin is used to start the slave data transmission. The slave's $\overline{SS}$ pin must be toggled back to high and then low again between each byte transmitted as shown in **Figure 20-5**.
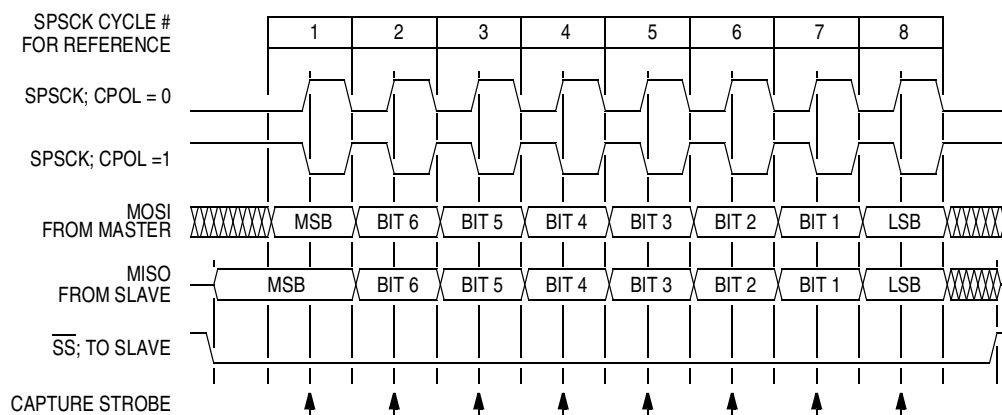
Figure 20-4. Transmission Format (CPHA = 0)
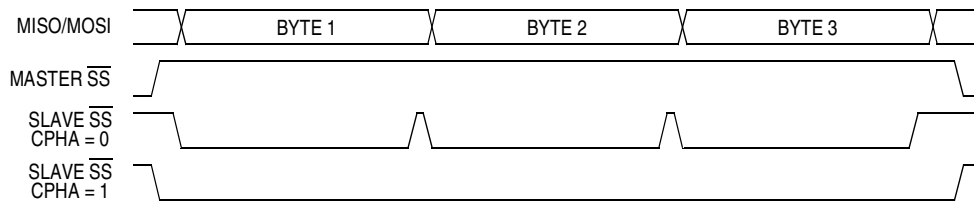
Figure 20-5. CPHA/$\overline{\text{SS}}$ Timing

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.

20.6.3 Transmission Format When CPHA = 1

Figure 20-6 shows an SPI transmission in which CPHA is logic 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 at logic 0, 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 **20.8.2 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 $\overline{\text{SS}}$ 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.
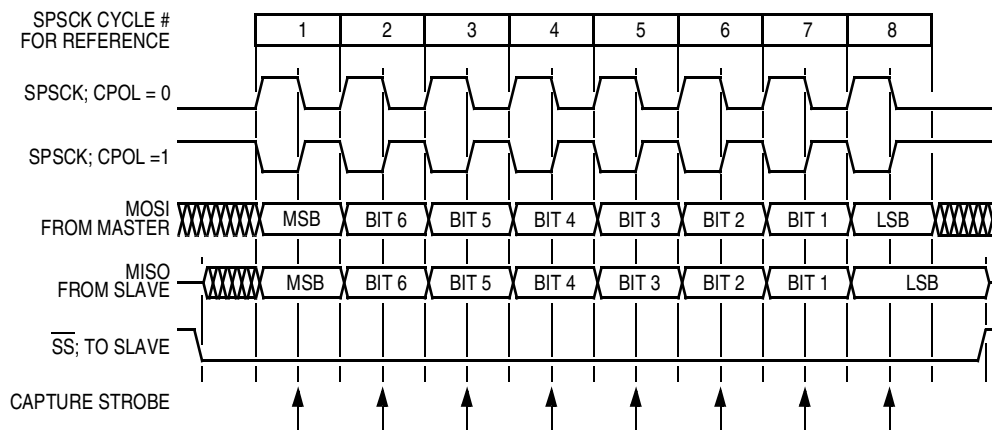
Figure 20-6. Transmission Format (CPHA = 1)

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.

20.6.4 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 20-7**.) 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. SPSCK edges occur halfway through the low time of the internal MCU clock. 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 20-7**. 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.
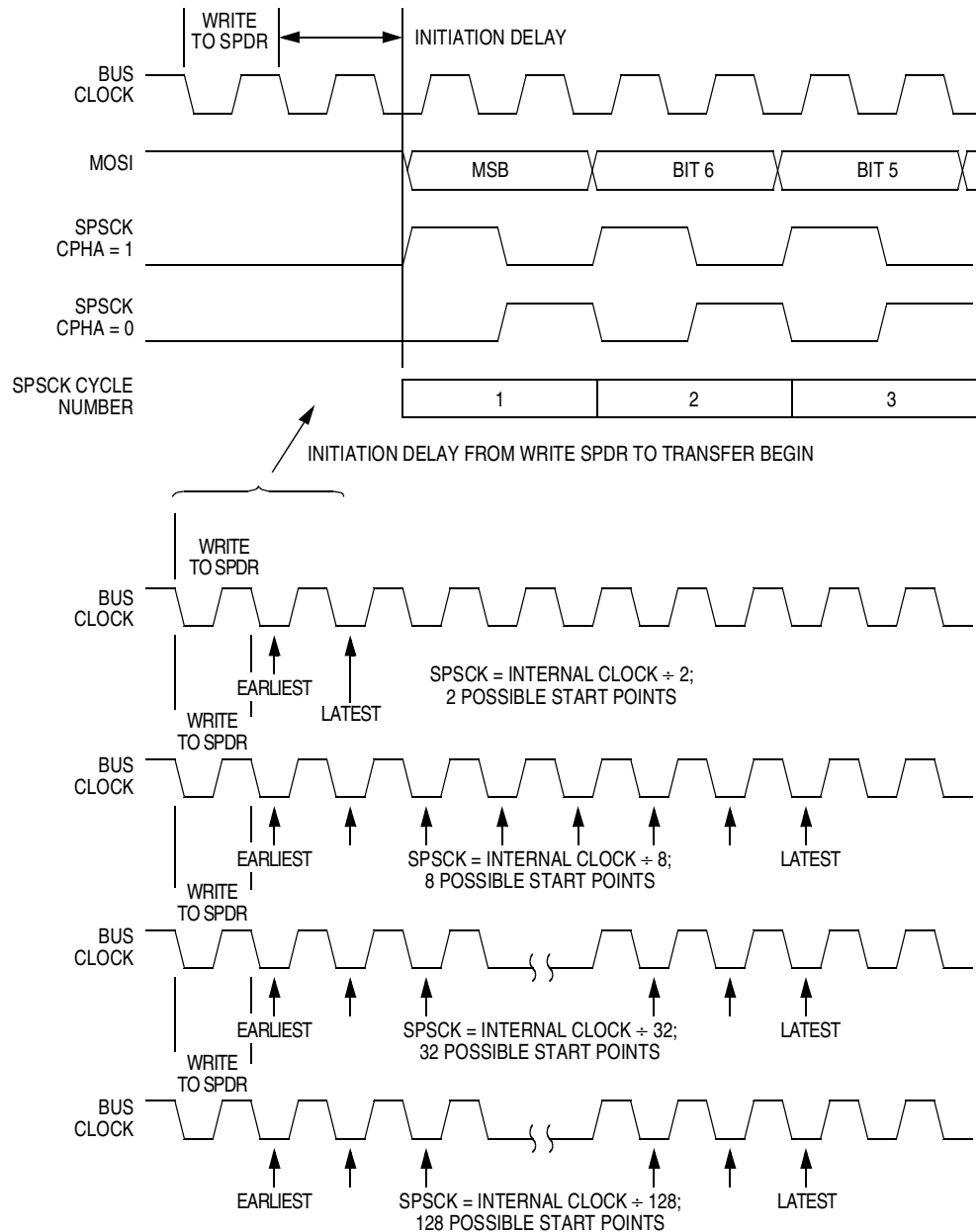
Figure 20-7. Transmission Start Delay (Master)

20.7 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 the SPTE bit is high. **Figure 20-8** 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 20-8. 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. The SPTE indicates when the next write can occur.

20.8 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.

20.8.1 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 20-4** and **Figure 20-6**.) 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 20-11**.) 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 20-9** shows how it is possible to miss an overflow. The first part of **Figure 20-9** 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.

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 BYTE 2 SETS SPRF BIT.

⑤ CPU READS SPSCR WITH SPRF BIT SET AND OVRF BIT CLEAR.

⑥ BYTE 3 SETS OVRF BIT. BYTE 3 IS LOST.

⑦ CPU READS BYTE 2 IN SPDR, CLEARING SPRF BIT, BUT NOT OVRF BIT.

⑧ BYTE 4 FAILS TO SET SPRF BIT BECAUSE OVRF BIT IS NOT CLEARED. BYTE 4 IS LOST.
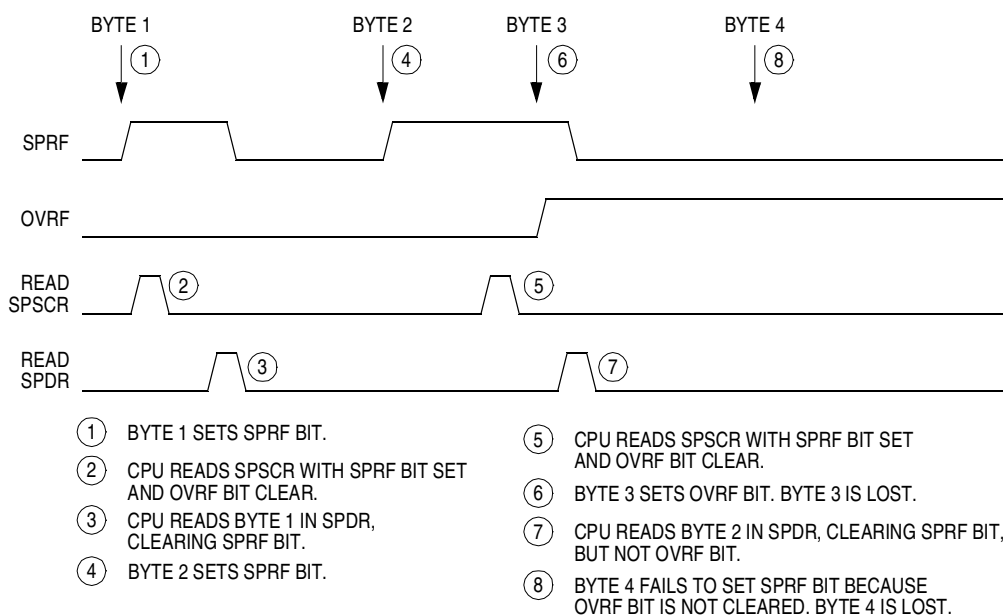
Figure 20-9. Missed Read of Overflow Condition

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 20-10** illustrates this process. Generally, to avoid this second SPSCR read, enable the OVRF to the CPU by setting the ERRIE bit.
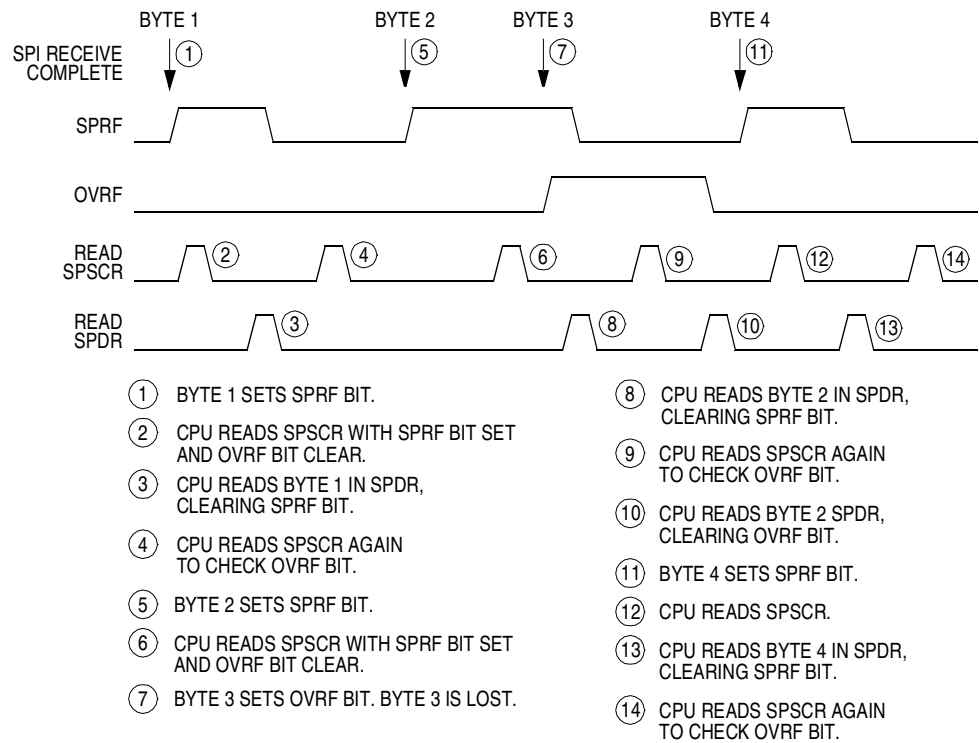
Figure 20-10. Clearing SPRF When OVRF Interrupt Is Not Enabled

20.8.2 Mode Fault Error

Setting the SPMSTR bit 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 20-11**.) 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 to logic 0. 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 **20.6 Transmission Formats**.

NOTE: *Setting the MODF flag does not clear the SPMSTR bit. The SPMSTR bit 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.*

When CPHA = 0, a MODF occurs if a slave is selected ($\overline{SS}$ is at logic 0) and later unselected ($\overline{SS}$ is at logic 1) even if no SPSCK is sent to that

slave. This happens because $\overline{SS}$ at logic 0 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), the MODF bit 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 logic 1 voltage 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.

20.9 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests.

Table 20-2. SPI Interrupts

Flag	Request

SPTE
  Transmitter empty
	SPI transmitter CPU interrupt request
	  (DMAS = 0, SPTIE = 1, SPE = 1)

SPRF
  Receiver full
	SPI receiver CPU interrupt request
	  (DMAS = 0, 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 the SPRF bit to generate receiver CPU interrupt requests, regardless of the state of the SPE bit. See **Figure 20-11**.

The error interrupt enable bit (ERRIE) enables both the MODF and OVRF bits to generate a receiver/error CPU interrupt request.

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.
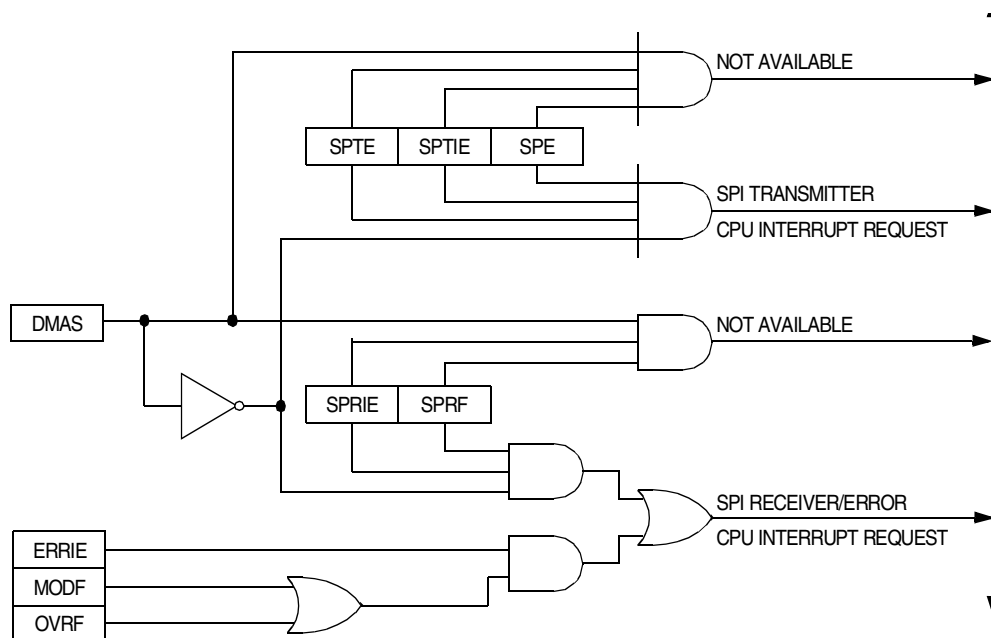
Figure 20-11. SPI Interrupt Request Generation

The following sources in the SPI status and control register can generate CPU interrupt requests:

- SPI receiver full bit (SPRF) — The SPRF bit 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) — The SPTE bit 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.

20.10 Resetting the SPI

Any system reset completely resets the SPI. Partial resets occur whenever the SPI enable bit (SPE) is low. Whenever SPE is low, 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.

20.11 Low-Power Modes

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

20.11.1 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 **20.9 Interrupts**.

20.11.2 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.

20.12 SPI 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 **Section 19. System Integration Module (SIM)**.

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 status bits 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 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 logic 0. After the break, doing the second step clears the status bit.

Since the SPTE bit cannot be cleared during a break with the BCFE bit 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 the BCFE bit cleared has no effect.

20.13 I/O Signals

The SPI module has five I/O pins and shares four of them with a parallel I/O port. They are:

- MISO — Data received
- MOSI — Data transmitted
- SPSCK — Serial clock
- $\overline{SS}$ — Slave select
- CGND — Clock ground (internally connected to V_{SS})

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}.

20.13.1 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 logic 0 and its $\overline{SS}$ pin is at logic 0. To support a multiple-slave system, a logic 1 on the $\overline{SS}$ 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.

20.13.2 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.

20.13.3 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.

20.13.4 $\overline{SS}$ (Slave Select)

The $\overline{SS}$ pin has various functions depending on the current state of the SPI. For an SPI configured as a slave, the $\overline{SS}$ is used to select a slave. For CPHA = 0, the $\overline{SS}$ is used to define the start of a transmission. (See **20.6 Transmission Formats**.) Since it is used to indicate the start of a transmission, the $\overline{SS}$ 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 20-12**.
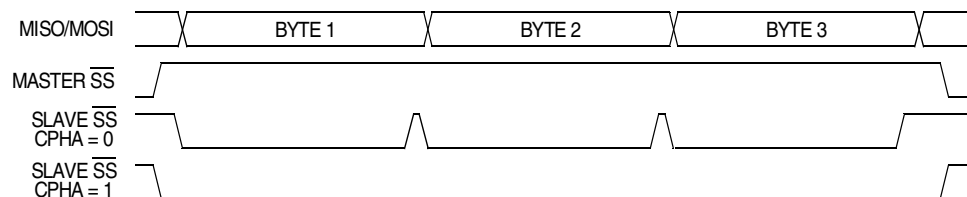
Figure 20-12. CPHA/$\overline{SS}$ Timing

When an SPI is configured as a slave, the $\overline{SS}$ 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 the $\overline{SS}$ from creating a MODF error. See **20.14.2 SPI Status and Control Register**.

NOTE: *A logic 1 voltage on the $\overline{SS}$ 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 $\overline{SS}$ input can be used in conjunction with the MODF flag to prevent multiple masters from driving MOSI and SPSCK. (See **20.8.2 Mode Fault Error**.) For the state of the $\overline{SS}$ pin to set the MODF flag, the MODFEN bit in the SPSCK register must be set. If the MODFEN bit is low for an SPI master, the $\overline{SS}$ pin can be used as a general-purpose I/O under the control of the data direction register of the shared I/O port. With MODFEN high, it 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 $\overline{SS}$ pin by configuring the appropriate pin as an input and reading the port data register. See **Table 20-3**.

Table 20-3. SPI Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	State of $\overline{SS}$ Logic
0	X[1]	X	Not enabled	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	0	X	Slave	Input-only to SPI
1	1	0	Master without MODF	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

1. X = Don't care

20.13.5 CGND (Clock Ground)

CGND is the ground return for the serial clock pin, SPSCK, and the ground for the port output buffers. It is internally connected to V_{SS} as shown in **Table 20-1**.

20.14 I/O Registers

Three registers control and monitor SPI operation:

- SPI control register (SPCR)

- SPI status and control register (SPSCR)

- SPI data register (SPDR)

20.14.1 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:	SPRIE	DMAS	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
Write:								
Reset:	0	0	1	0	1	0	0	0

= Unimplemented

Figure 20-13. 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

DMAS — DMA Select Bit

This read only bit has no effect on this version of the SPI. This bit always reads as a 0.

    0 = SPRF DMA and SPTE DMA service requests disabled
        (SPRF CPU and SPTE CPU interrupt requests enabled)

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 20-4** and **Figure 20-6**.) 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 20-4** and **Figure 20-6**.) 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 set to logic 1 between bytes. (See **Figure 20-12**.) 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 **20.10 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

20.14.2 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:						MODFEN	SPR1	SPR0
Reset:	0	0	0	0	1	0	0	0

☐ = Unimplemented

Figure 20-14. 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 the MODFEN bit 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 the MODF bit 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 or an SPTE DMA service request if the SPTIE bit in the SPI control register is set also.

NOTE: *Do not write to the SPI data register unless the SPTE bit is high.*

During an SPTE CPU interrupt, the CPU clears the SPTE bit by 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 to 1, allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag. If the SPI is enabled as a master and the MODFEN bit is low, then the $\overline{SS}$ pin is available as a general-purpose I/O.

If the MODFEN bit is set, then this pin 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 **20.13.4 SS (Slave Select)**.

If the MODFEN bit is low, 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 **20.8.2 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 20-4**. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Table 20-4. 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:

$$\text{Baud rate} = \frac{\text{CGMOUT}}{2 \times \text{BD}}$$

where:

CGMOUT = base clock output of the clock generator module (CGM)
BD = baud rate divisor

20.14.3 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 20-2**.

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 20-15. 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.*

Section 21. Timebase Module (TBM)

21.1 Contents

21.2 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 crystal clock. This TBM version uses 15 divider stages, eight of which are user selectable.

21.3 Features

Features of the TBM include:

- Software programmable 1-Hz, 4-Hz, 16-Hz, 256-Hz, 512-Hz, 1024-Hz, 2048-Hz, and 4096-Hz periodic interrupt using external 32.768-kHz crystal

- User selectable oscillator clock source enable during stop mode to allow periodic wakeup from stop

21.4 Functional Description

NOTE: *This module is designed for a 32.768-kHz oscillator.*

This module can generate a periodic interrupt by dividing the clock TBMCLK. The counter is initialized to all 0s when TBON bit is cleared. The counter, shown in **Figure 21-1**, 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.
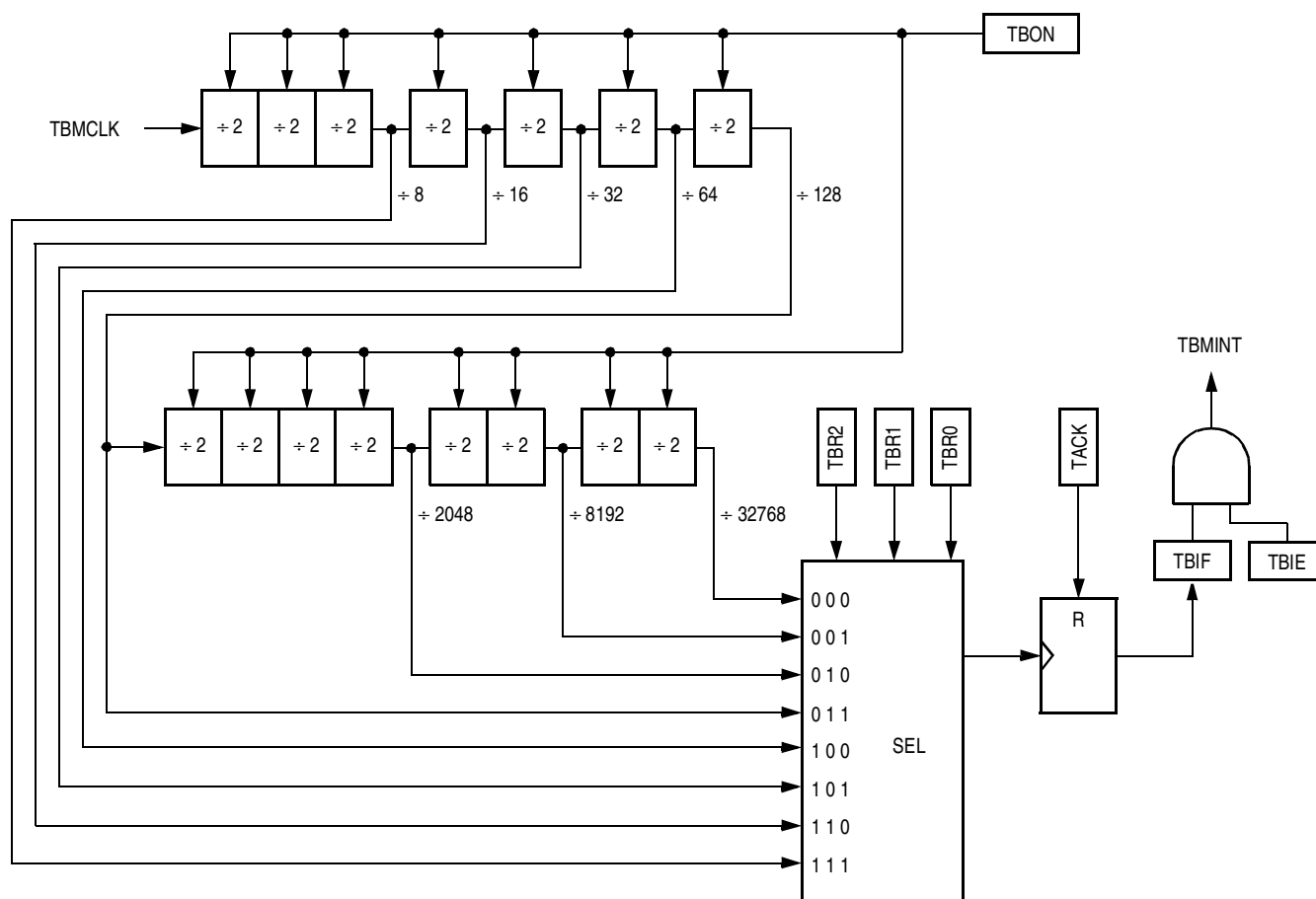
Figure 21-1. Timebase Block Diagram

21.5 Timebase Register Description

The timebase has one register, the timebase control regster (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
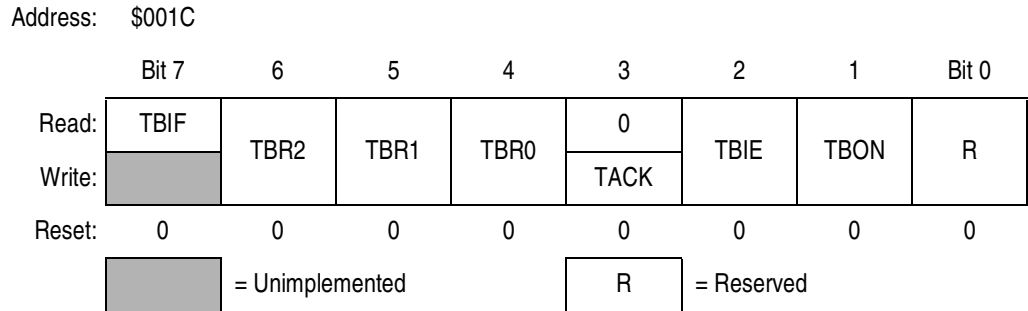
Figure 21-2. Timebase Control Register (TBCR)

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 Rate Selection

These read/write bits are used to select the rate of timebase interrupts as shown in **Table 21-1**.

Table 21-1. Timebase Rate Selection for OSC1 = 32.768 kHz

TBR2	TBR1	TBR0	Divider	Timebase Interrupt Rate	
				Hz	ms
0	0	0	32768	1	1000
0	0	1	8192	4	250
0	1	0	2048	16	62.5
0	1	1	128	256	~ 3.9
1	0	0	64	512	~2
1	0	1	32	1024	~1
1	1	0	16	2048	~0.5
1	1	1	8	4096	~0.24

NOTE: *Do not change TBR2:TBR0 bits while the timebase is enabled (TBON = 1).*

TACK — Timebase ACKnowledge

The TACK bit is a write-only bit and always reads as 0. Writing a logic 1 to this bit clears TBIF, the timebase interrupt flag bit. Writing a logic 0 to this bit has no effect.

1 = Clear timebase interrupt flag
0 = No effect

TBIE — Timebase Interrupt Enabled

This read/write bit enables the timebase interrupt when the TBIF bit becomes set. Reset clears the TBIE bit.

1 = Timebase interrupt enabled
0 = Timebase interrupt disabled

TBON — Timebase Enabled

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 enabled
0 = Timebase disabled and the counter initialized to 0s

21.6 Interrupts

The timebase module can interrupt the CPU on a regular basis with a rate defined by 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.

Interrupts must be acknowledged by writing a logic 1 to the TACK bit.

21.7 Low-Power Modes

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

21.7.1 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 enabling the WAIT instruction.

21.7.2 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 OSCSTOPEN bit in the CONFIG 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.

Section 22. Timer Interface Module (TIM)

22.1 Contents

22.2 Introduction

This section describes the timer interface (TIM) module. The TIM is a two-channel timer that provides a timing reference with input capture, output compare, and pulse-width-modulation functions. **Figure 22-1** is a block diagram of the TIM.

This particular MCU has two timer interface modules which are denoted as TIM1 and TIM2.
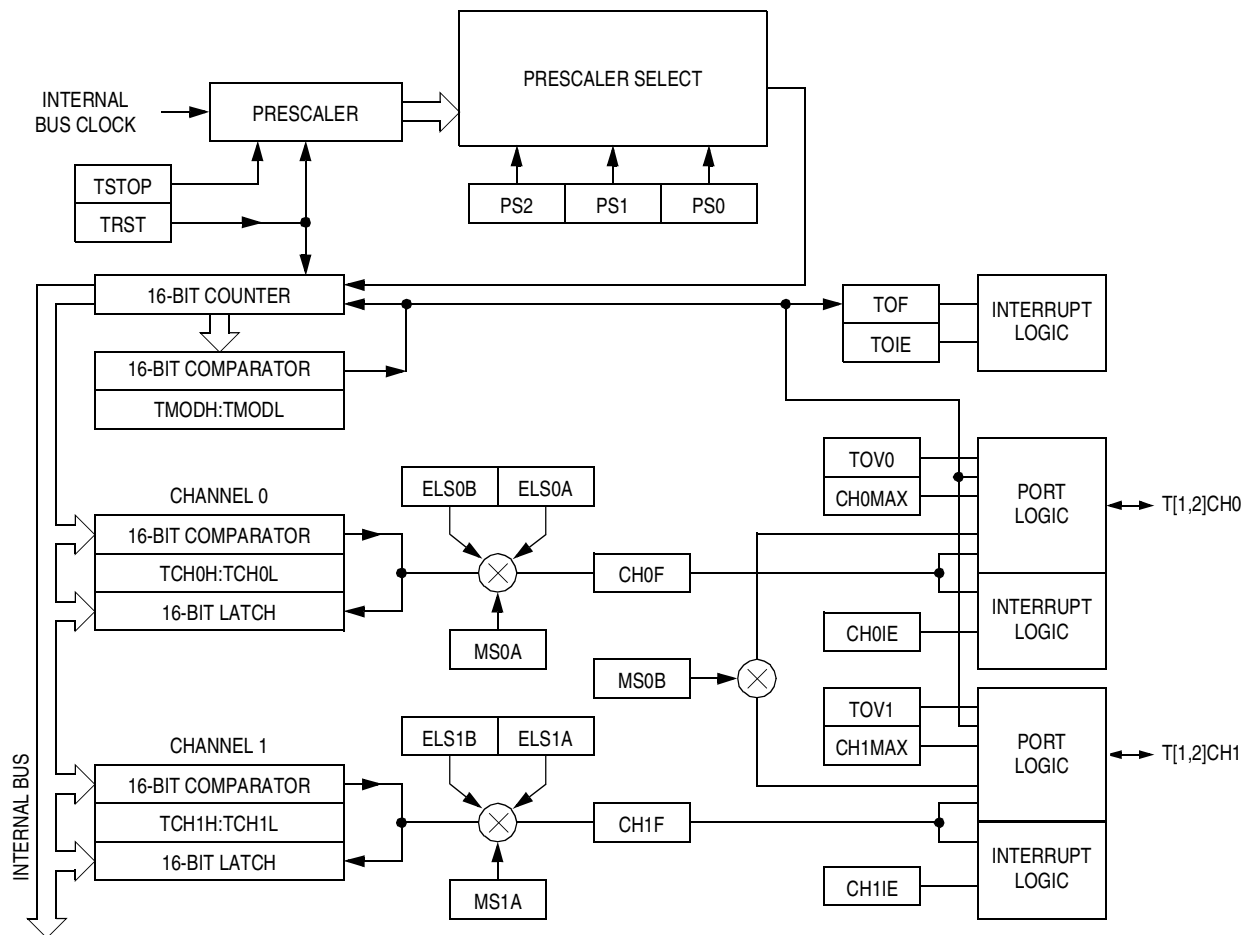
Figure 22-1. TIM Block Diagram

22.3 Features

Features of the TIM include:

- Two 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

22.4 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 four I/O pins with four port D I/O port pins. The full names of the TIM I/O pins are listed in **Table 22-1**. The generic pin names appear in the text that follows.

Table 22-1. Pin Name Conventions

TIM Generic Pin Names:		T[1,2]CH0	T[1,2]CH1
Full TIM Pin Names:	TIM1	PTD4/$\overline{\text{KBD4}}$	PTD5/$\overline{\text{KBD5}}$
	TIM2	PTD6/$\overline{\text{KBD6}}$	PTD7/KBD7

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, and TCH1 may refer to T1CH1 and T2CH1.*

22.5 Functional Description

Figure 22-1 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 **16.6.3 Port D Input Pullup Enable Register**.

Figure 22-2 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 371.	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 374.	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 374.	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 375.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	1	1	1	1	1	1	1	1

☐ = Unimplemented

Figure 22-2. TIM I/O Register Summary (Sheet 1 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0024	Timer 1 Counter Modulo Register Low (T1MODL) See page 375.	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 376.	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 380.	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 380.	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 376.	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 380.	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 380.	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 371.	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 374.	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 374.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

             = Unimplemented

Figure 22-2. TIM I/O Register Summary (Sheet 2 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$002E	Timer 2 Counter Modulo Register High (T2MODH) See page 375.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$002F	Timer 2 Counter Modulo Register Low (T2MODL) See page 375.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$0030	Timer 2 Channel 0 Status and Control Register (T2SC0) See page 376.	Read: Write:	CH0F / 0	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Reset:	0	0	0	0	0	0	0	0
$0031	Timer 2 Channel 0 Register High (T2CH0H) See page 380.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:				Indeterminate after reset				
$0032	Timer 2 Channel 0 Register Low (T2CH0L) See page 380.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:				Indeterminate after reset				
$0033	Timer 2 Channel 1 Status and Control Register (T2SC1) See page 376.	Read: Write:	CH1F / 0	CH1IE	0 / ▨	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Reset:	0	0	0	0	0	0	0	0
$0034	Timer 2 Channel 1 Register High (T2CH1H) See page 380.	Read: Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:				Indeterminate after reset				
$0035	Timer 2 Channel 1 Register Low (T2CH1L) See page 380.	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:				Indeterminate after reset				

▨ = Unimplemented

Figure 22-2. TIM I/O Register Summary (Sheet 3 of 3)

22.5.1 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.

22.5.2 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.

22.5.3 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.

22.5.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in **22.5.3 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.

22.5.3.2 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.

22.5.4 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 22-3** 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 state of the PWM pulse is logic 1. Program the TIM to set the pin if the state of the PWM pulse is logic 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 **22.10.1 TIM Status and Control Register**.
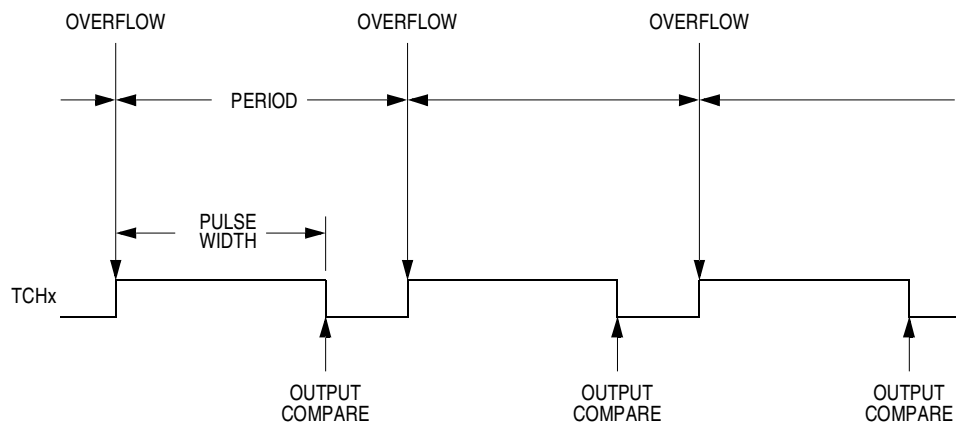
Figure 22-3. PWM Period and Pulse Width

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%.

22.5.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in **22.5.4 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.*

22.5.4.2 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.*

22.5.4.3 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.

MC68HC908GT16 • MC68HC908GT8 — Rev. 2   Technical Data

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 22-3**.)

    b. Write 1 to the toggle-on-overflow bit, TOVx.

    c. Write 1:0 (to clear output on compare) or 1:1 (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 22-3**.)

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 **22.10.4 TIM Channel Status and Control Registers**.)

22.6 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.

22.7 Low-Power Modes

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

22.7.1 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.

22.7.2 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.

22.8 TIM During Break Interrupts

A break interrupt stops the TIM counter.

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 **19.8.3 SIM Break Flag Control 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 status bits 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 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 logic 0. After the break, doing the second step clears the status bit.

22.9 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 **22.4 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.

22.10 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, TSC1)
- TIM channel registers (TCH0H:TCH0L, TCH1H:TCH1L)

22.10.1 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, $0020 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 22-4. 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 logic 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing logic 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 logic 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.*

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 logic 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 22-2** shows. Reset clears the PS[2:0] bits.

Table 22-2. 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

22.10.2 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
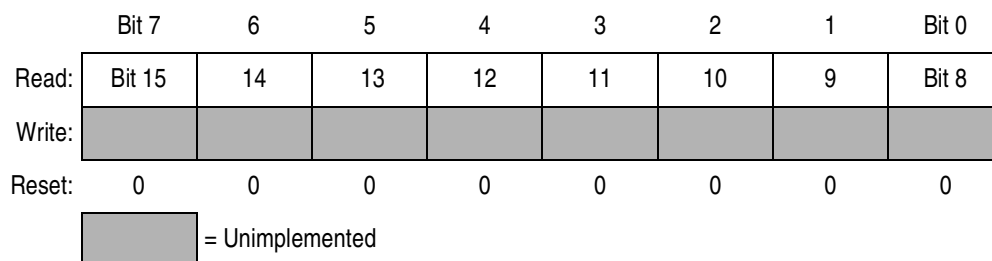
Figure 22-5. TIM Counter Registers High (TCNTH)

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
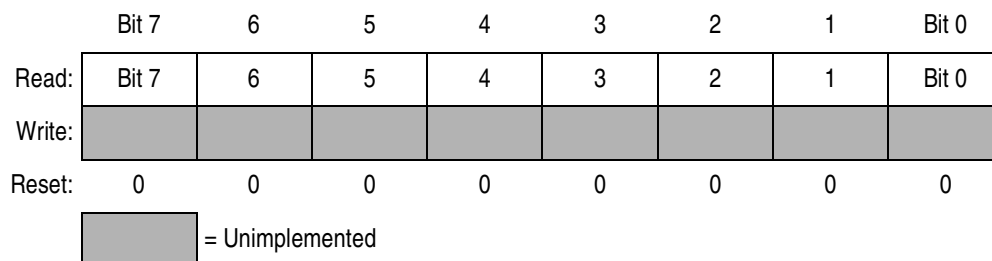
Figure 22-6. TIM Counter Registers Low (TCNTL)

22.10.3 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: Write:	Bit 15	14	13	12	11	10	9	Bit 8
Reset:	1	1	1	1	1	1	1	1
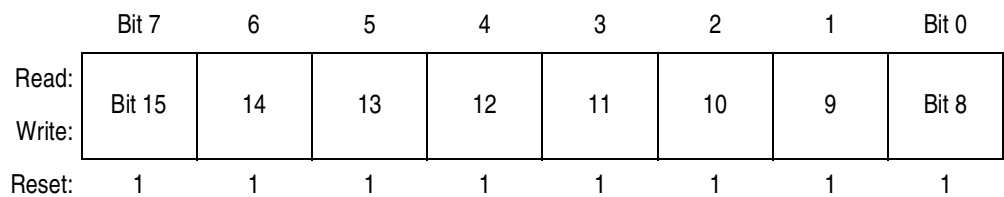
Figure 22-7. TIM Counter Modulo Register High (TMODH)

Address: T1MODL, $0024 and T2MODL, $002F

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	1	1	1	1	1	1	1	1
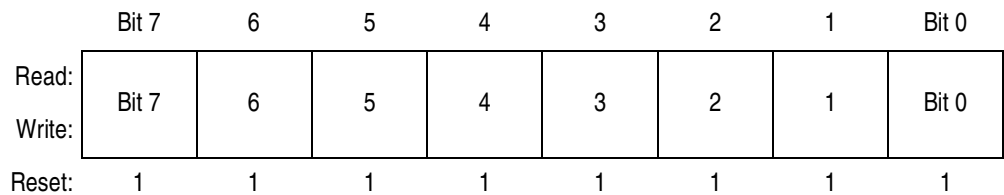
Figure 22-8. TIM Counter Modulo Register Low (TMODL)

NOTE: *Reset the TIM counter before writing to the TIM counter modulo registers.*

22.10.4 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 22-9. TIM Channel 0 Status and Control Register (TSC0)

Address: T1SC1, $0028 and T2SC1, $0033

	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

Figure 22-10. 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 logic 0 to CHxF. If another interrupt request occurs before the clearing sequence is complete, then writing logic 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 logic 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 22-3**.
     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 22-3**. 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 22-3** shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

Table 22-3. Mode, Edge, and Level Selection

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output preset	Pin under port control; initial output level high
X1	00		Pin under port control; initial output level low
00	01	Input capture	Capture on rising edge only
00	10		Capture on falling edge only
00	11		Capture on rising or falling edge
01	01	Output compare or PWM	Toggle output on compare
01	10		Clear output on compare
01	11		Set output on compare
1X	01	Buffered output compare or buffered PWM	Toggle output on compare
1X	10		Clear output on compare
1X	11		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 logic 1, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100%. As **Figure 22-11** 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.
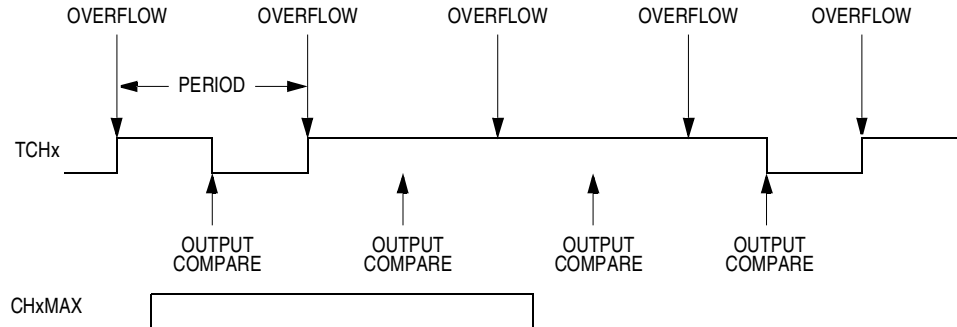
Figure 22-11. CHxMAX Latency
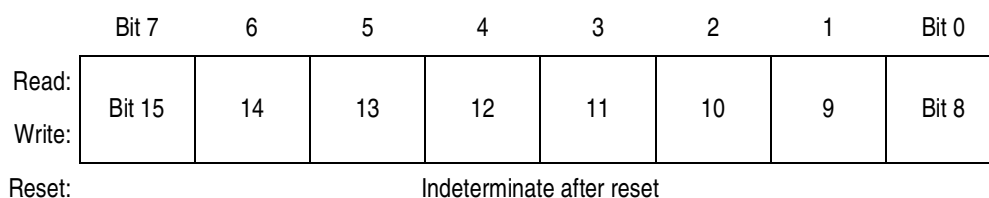
22.10.5 TIM Channel Registers

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 $\neq$ 0:0), writing to the high byte of the TIM channel x registers (TCHxH) inhibits output compares until the low byte (TCHxL) is written.

Address: T1CH0H, $0026 and T2CH0H, $0031
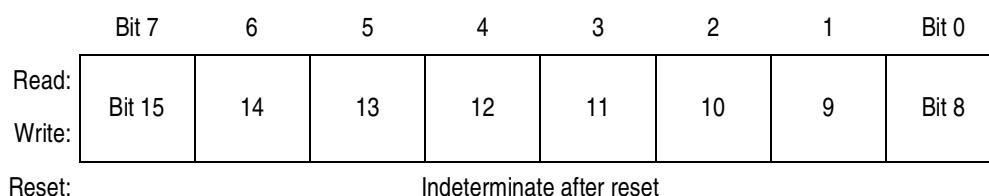
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 15	14	13	12	11	10	9	Bit 8
Write:								
Reset:				Indeterminate after reset				

Figure 22-12. TIM Channel 0 Register High (TCH0H)

Address: T1CH0L, $0027 and T2CH0L $0032

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	6	5	4	3	2	1	Bit 0
Write:								
Reset:				Indeterminate after reset				

Figure 22-13. TIM Channel 0 Register Low (TCH0L)

Address: T1CH1H, $0029 and T2CH1H, $0034

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

Figure 22-14. TIM Channel 1 Register High (TCH1H)

Address: T1CH1L, $002A and T2CH1L, $0035

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	6	5	4	3	2	1	Bit 0
Write:								
Reset:				Indeterminate after reset				
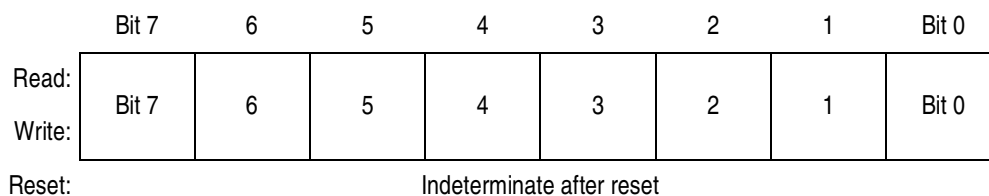
Figure 22-15. TIM Channel 1 Register Low (TCH1L)

Section 23. Electrical Specifications

23.1 Contents

23.2 Introduction

This section contains electrical and timing specifications.

23.3 Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the MCU can be exposed without permanently damaging it.

NOTE: *This device is not guaranteed to operate properly at the maximum ratings. Refer to 23.6 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 those specified below	I	± 15	mA
Maximum current for pins PTA5-PTA7, PTD4	$I_{PTA5–PTA7}$	± 20	mA
Maximum current for pins PTC0–PTC4	$I_{PTC0–PTC4}$	± 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}).*

23.4 Functional Operating Range

Characteristic	Symbol	Value	Unit
Operating temperature range	T_A	−40 to +85	°C
Operating voltage range	V_{DD}	3.0 ±10% 5.0 ±10%	V

23.5 Thermal Characteristics

Characteristic	Symbol	Value	Unit

Thermal resistance
  42-pin SDIP
  44-pin QFP
	θ_{JA}	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

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.

23.6 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} = –20.0 mA) pins PTC0–PTC4 only	V_{OH}	$V_{DD} - 1.5$	—	—	V
Maximum combined I_{OH} for port C, port E, port PTD0–PTD3	I_{OH1}	—	—	50	mA
Maximum combined I_{OH} for port PTD4–PTD7, 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} = 20mA) pins PTC0–PTC4 only	V_{OL}	—	—	1.5	V
Maximum combined I_{OL} for port C, port E, port PTD0–PTD3	I_{OL1}	—	—	50	mA
Maximum combined I_{OL} for port PTD4–PTD7, 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, $\overline{IRQ}$, $\overline{RST}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input low voltage All ports, $\overline{IRQ}$, $\overline{RST}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
V_{DD} supply current					
Run[3]		—	15	20	mA
Wait[4]		—	4	8	mA
Stop[5]	I_{DD}				
25°C		—	3	—	µA
25°C with TBM enabled[6]		—	20	—	µA
25°C with LVI and TBM enabled[6]		—	300	—	µA
–40°C to 85°C with TBM enabled[6]		—	50	—	µA
–40°C to 85°C with LVI and TBM enabled[6]		—	500	—	µA
I/O ports Hi-Z leakage current[7]	I_{IL}	—	—	±10	µA
Input current	I_{In}	—	—	±1	µA
Pullup resistors (as input only) Ports PTA7/$\overline{KBD7}$–PTA0/$\overline{KBD0}$, PTC6–PTC0, PTD7/T2CH1–PTD0/$\overline{SS}$	R_{PU}	20	45	65	kΩ

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Capacitance Ports (as input or output)	C_{Out} C_{In}	— —	— —	12 8	pF
Monitor mode entry voltage	V_{TST}	V_{DD} + 2.5	—	V_{DD} + 4.0	V
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	3.90	4.25	4.50	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	4.20	4.35	4.60	V
Low-voltage inhibit reset/recover hysteresis (V_{TRIPF} + V_{HYS} = V_{TRIPR})	V_{HYS}	—	100	—	mV
POR rearm voltage[8]	V_{POR}	0	—	100	mV
POR reset voltage[9]	V_{PORRST}	0	700	800	mV
POR rise time ramp rate[10]	R_{POR}	0.035	—	—	V/ms

1. V_{DD} = 5.0 Vdc $\pm$ 10%, V_{SS} = 0 Vdc, T_A = T_A (min) to T_A (max), 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 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 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 ICG and LVI enabled.
5. Stop I_{DD} is measured with OSC1 = V_{SS}.
6. Stop I_{DD} with TBM enabled is measured using an external square wave clock source (f_{OSC} = 32 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All inputs configured as inputs.
7. Pullups and pulldowns are disabled. Port B leakage is specified in **23.16 ADC Characteristics**.
8. Maximum is highest voltage that POR is guaranteed.
9. Maximum is highest voltage that POR is possible.
10. 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.

23.7 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} = –10.0 mA) pins PTC0–PTC4 only	V_{OH}	V_{DD} –1.0	—	—	V
Maximum combined I_{OH} for port C, port E, port PTD0–PTD3	I_{OH1}	—	—	30	mA
Maximum combined I_{OH} for port PTD4–PTD7, 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} = 5.0 mA) all I/O pins	V_{OL}	—	—	1.0	V
(I_{Load} = 10.0 mA) pins PTC0–PTC4 only	V_{OL}	—	—	1.0	V
Maximum combined I_{OL} for port C, port E, port PTD0–PTD3	I_{OL1}	—	—	30	mA
Maximum combined I_{OL} for port PTD4–PTD7, 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, $\overline{IRQ}$, $\overline{RST}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input low voltage All ports, $\overline{IRQ}$, $\overline{RST}$, OSC1	V_{IL}	V_{SS}	—	$0.3 \times V_{DD}$	V
V_{DD} supply current					
Run[3]		—	4.5	8	mA
Wait[4]		—	1.65	4	mA
Stop[5]	I_{DD}				
25°C		—	2	—	µA
25°C with TBM enabled[6]		—	12	—	µA
25°C with LVI and TBM enabled[6]		—	200	—	µA
–40°C to 85°C with TBM enabled[6]		—	30	—	µA
–40°C to 85°C with LVI and TBM enabled[6]		—	300	—	µA
I/O ports Hi-Z leakage current[7]	I_{IL}	—	—	±10	µA
Input current	I_{In}	—	—	±1	µA
Pullup resistors (as input only) Ports PTA7/$\overline{KBD7}$–PTA0/$\overline{KBD0}$, PTC6–PTC0, PTD7/T2CH1–PTD0/$\overline{SS}$	R_{PU}	20	45	65	kΩ

Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Capacitance Ports (as input or output)	C_{Out} C_{In}	— —	— —	12 8	pF
Monitor mode entry voltage	V_{TST}	V_{DD} + 2.5	—	V_{DD} + 4.0	V
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	2.45	2.60	2.70	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	2.55	2.66	2.80	V
Low-voltage inhibit reset/recover hysteresis (V_{TRIPF} + V_{HYS} = V_{TRIPR})	V_{HYS}	—	60	—	mV
POR rearm voltage[8]	V_{POR}	0	—	100	mV
POR reset voltage[9]	V_{PORRST}	0	700	800	mV
POR rise time ramp rate[10]	R_{POR}	0.02	—	—	V/ms

1. V_{DD} = 3.0 Vdc $\pm$ 10%, V_{SS} = 0 Vdc, T_A = T_A (min) to T_A (max), 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 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 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 ICG and LVI enabled.
5. Stop I_{DD} is measured with OSC1 = V_{SS}.
6. Stop I_{DD} with TBM enabled is measured using an external square wave clock source (f_{OSC} = 16 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All inputs configured as inputs.
7. Pullups and pulldowns are disabled.
8. Maximum is highest voltage that POR is guaranteed.
9. Maximum is highest voltage that POR is possible.
10. 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.

23.8 5.0-V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency	f_{OP} (f_{Bus})	—	8	MHz
Internal clock period (1/f_{OP})	t_{CYC}	122	—	ns
RESET input pulse width low[2]	t_{IRL}	50	—	ns
IRQ interrupt pulse width low[3] (edge-triggered)	t_{ILIH}	50	—	ns
IRQ interrupt pulse period	t_{ILIL}	Note 5	—	t_{CYC}

16-bit timer[4]
  Input capture pulse width
  Input capture period
	t_{TH},t_{TL}
	t_{TLTL}
		Note 5	—
			—
				ns
				t_{CYC}

1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.
2. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
3. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
4. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
5. 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}.

23.9 3.0-V Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Internal operating frequency	f_{OP} (f_{Bus})	—	4	MHz
Internal clock period (1/f_{OP})	t_{CYC}	244	—	ns
RESET input pulse width low[2]	t_{IRL}	125	—	ns
IRQ interrupt pulse width low[3] (edge-triggered)	t_{ILIH}	125	—	ns
IRQ interrupt pulse period	t_{ILIL}	Note 5	—	t_{CYC}

16-bit timer[4]
  Input capture pulse width
  Input capture period
	t_{TH},t_{TL}
	t_{TLTL}
		Note 5	—
			—
				ns
				t_{CYC}

1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.
2. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.
3. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
4. Minimum pulse width is for guaranteed interrupt. It is possible for a smaller pulse width to be recognized.
5. 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}.

23.10 Internal Oscillator Characteristics

Characteristic[1]	Symbol	Min	Typ	Max	Unit
Internal oscillator base frequency[2],[3]	f_{INTOSC}	230.4	307.2	384	kHz
Internal oscillator tolerance	f_{OSC_TOL}	−25	—	+25	%
Internal oscillator multiplier[4]	N	1	—	127	—

1. V_{DD} = 5.5 Vdc to 2.7 Vdc, V_{SS} = 0 Vdc, T_A = T_A (min) to T_A (max), unless otherwise noted
2. Internal oscillator is selectable through software for a maximum frequency. Actual frequency will be multiplier (N) x base frequency.
3. f_{Bus} = (f_{INTOSC} / 4) x N when internal clock source selected
4. Multiplier must be chosen to limit the maximum bus frequency of 4 MHz for 2.7-V operation and 8 MHz for 4.5-V operation.

23.11 External Oscillator Characteristics

Characteristic[1]	Symbol	Min	Typ	Max	Unit

External clock option[2][3]
  With ICG clock disabled
  With ICG clock enabled
    EXTSLOW = 1[4]
    EXTSLOW = 0[4]
	f_{EXTOSC}	dc[5]
		
		60
		307.2 k
			—
			
			—
			—
				32 M[6]
				
				307.2 k
				32 M[6]
					Hz

External crystal options[7][8]
  EXTSLOW = 1[4]
  EXTSLOW = 0[4]
	f_{EXTOSC}	30 k
		1 M
			—
			—
				100 k
				10 M
					Hz

Crystal load capacitance[9]	C_L	—	—	—	pF
Crystal fixed capacitance[9]	C_1	—	$2 \times C_L$	—	pF
Crystal tuning capacitance[9]	C_2	—	$2 \times C_L$	—	pF
Feedback bias resistor[9]	R_B	—	10	—	MΩ
Series resistor[9][10]	R_S	—	—	—	MΩ

1. V_{DD} = 5.5 to 2.7 Vdc, V_{SS} = 0 Vdc, T_A = T_A (min) to T_A (max), unless otherwise noted
2. Setting EXTCLKEN configuration option enables OSC1 pin for external clock square-wave input.
3. No more than 10% duty cycle deviation from 50%
4. EXTSLOW configuration option configures external oscillator for a slow speed crystal and sets the clock monitor circuits of the ICG module to expect an external clock frequency that is higher/lower than the internal oscillator base frequency, f_{INTOSC}.
5. Some modules may require a minimum frequency greater than dc for proper operation. See appropriate table for this information.
6. MCU speed derates from 32 MHz at V_{DD} = 4.5 Vdc to 16 MHz at V_{DD} = 2.7 Vdc.
7. Setting EXTCLKEN and EXTXTALEN configuration options enables OSC1 and OSC2 pins for external crystal option.
8. f_{Bus} = (f_{EXTOSC} / 4) when external clock source is selected.
9. Consult crystal vendor data sheet, see **Figure 7-3. External Clock Generator Block Diagram**.
10. Not required for high-frequency crystals

23.12 Trimmed Accuracy of the Internal Clock Generator

The unadjusted frequency of the low-frequency base clock (IBASE), when the comparators in the frequency comparator indicate zero error, can vary as much as ±25% due to process, temperature, and voltage. The trimming capability exists to compensate for process effects. The remaining variation in frequency is due to temperature, voltage, and change in target frequency (multiply register setting). These effects are designed to be minimal, however variation does occur. Better performance is seen at 3 V and lower settings of N.

23.12.1 2.7-Volt to 3.3-Volt Trimmed Internal Clock Generator Characteristics

| Characteristic[1]	Symbol	Min	Typ	Max	Unit
Absolute trimmed internal oscillator tolerance[2], [3] −40°C to 85°C	F_{abs_tol}	—	2.5	4.0	%
Variation over temperature[3], [4]	V_{ar_temp}	—	0.03	0.05	%/°C
Variation over voltage[3], [5] 25°C −40°C to 85°C	V_{ar_volt}	— —	0.5 0.7	2.0 2.0	%/V

1. These specifications concern long-term frequency variation. Each measurement is taken over a 1-ms period.
2. Absolute value of variation in ICG output frequency, trimmed at nominal V_{DD} and temperature, as temperature and V_{DD} are allowed to vary for a single given setting of N.
3. Specification is characterized but not tested.
4. Variation in ICG output frequency for a fixed N and voltage
5. Variation in ICG output frequency for a fixed N

23.12.2 4.5-Volt to 5.5-Volt Trimmed Internal Clock Generator Characteristics

Characteristic[1]	Symbol	Min	Typ	Max	Unit
Absolute trimmed internal oscillator tolerance[2], [3] −40°C to 85°C	F_{abs_tol}	—	4.0	4.0	%
Variation over temperature[3], [4]	V_{ar_temp}	—	0.05	0.08	%/°C
Variation over voltage[3], [5] 25°C −40°C to 85°C	V_{ar_volt}	— —	1.0 1.0	2.0 2.0	%/V

1. These specifications concern long-term frequency variation. Each measurement is taken over a 1-ms period.
2. Absolute value of variation in ICG output frequency, trimmed at nominal V_{DD} and temperature, as temperature and V_{DD} are allowed to vary for a single given setting of N.
3. Specification is characterized but not tested.
4. Variation in ICG output frequency for a fixed N and voltage
5. Variation in ICG output frequency for a fixed N
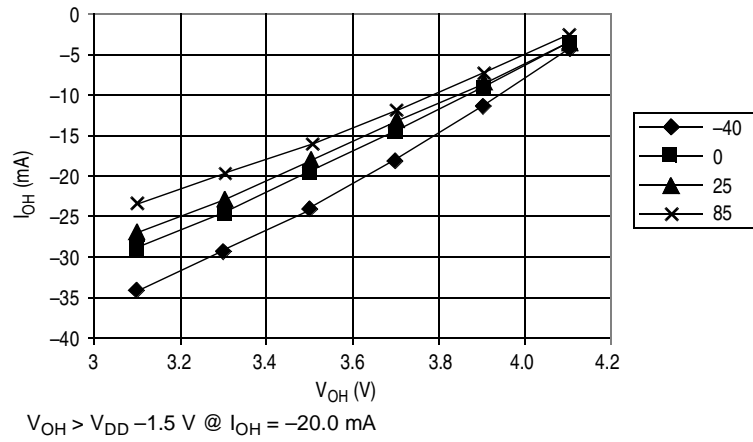
23.13 Output High-Voltage Characteristics
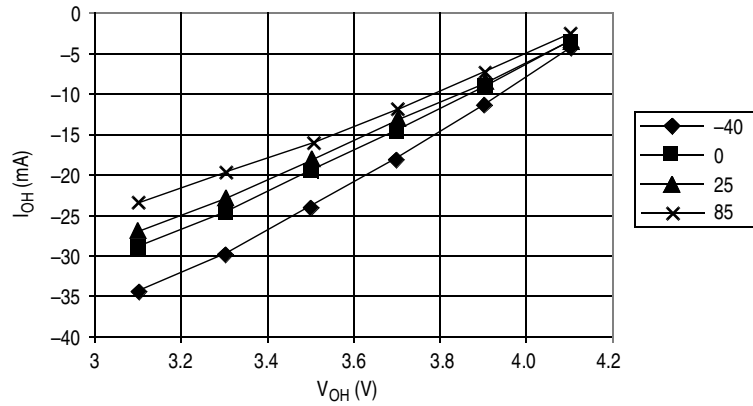
$V_{OH} > V_{DD} -0.8$ V @ $I_{OH} = -2.0$ mA
$V_{OH} > V_{DD} -1.5$ V @ $I_{OH} = -10.0$ mA

**Figure 23-1. Typical High-Side Driver Characteristics –
Port PTA7–PTA0 (V_{DD} = 4.5 Vdc)**
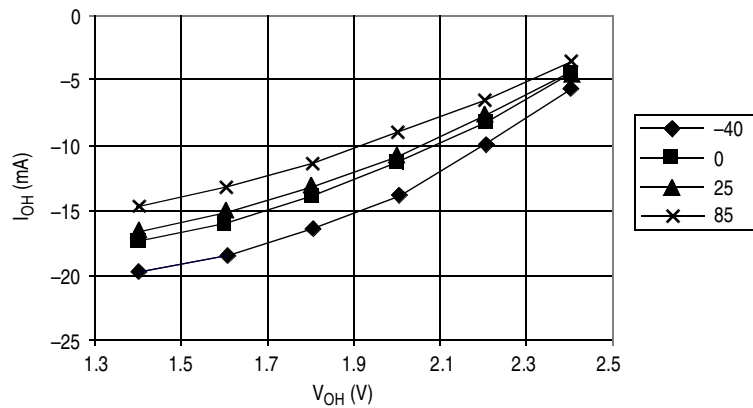
$V_{OH} > V_{DD} -0.3$ V @ $I_{OH} = -0.6$ mA
$V_{OH} > V_{DD} -1.0$ V @ $I_{OH} = -10.0$ mA

**Figure 23-2. Typical High-Side Driver Characteristics –
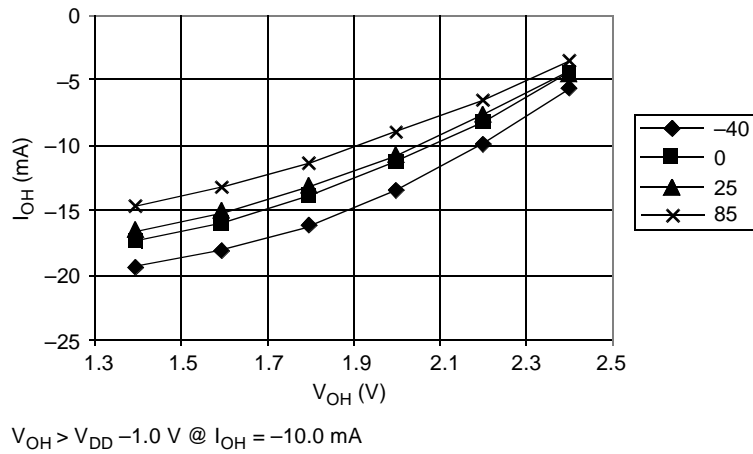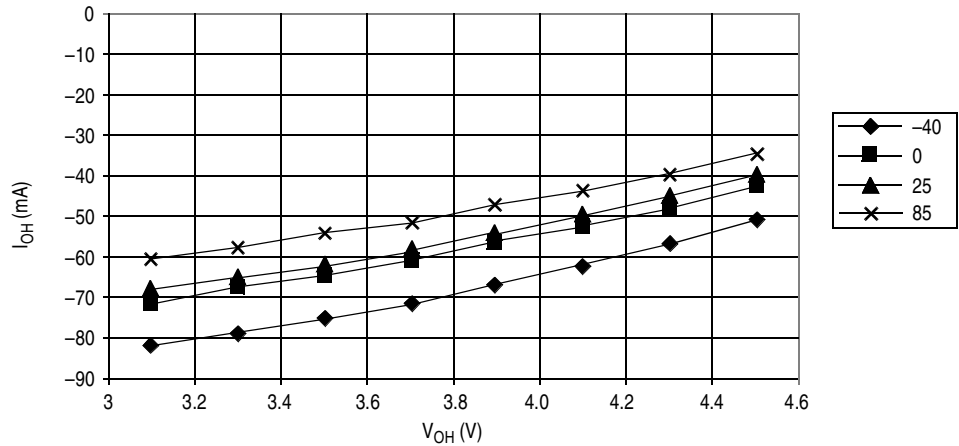Port PTA7–PTA0 (V_{DD} = 2.7 Vdc)**

MC68HC908GT16 • MC68HC908GT8 — Rev. 2   Technical Data

$V_{OH} > V_{DD} -1.5$ V @ $I_{OH} = -20.0$ mA

**Figure 23-3. Typical High-Side Driver Characteristics –
Port PTC4–PTC0 (V_{DD} = 4.5 Vdc)**

$V_{OH} > V_{DD} -1.0$ V @ $I_{OH} = -10.0$ mA

**Figure 23-4. Typical High-Side Driver Characteristics –
Port PTC4–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 23-5. Typical High-Side Driver Characteristics –
Ports PTB7–PTB0, PTC6–PTC5, PTD7–PTD0, and
PTE1–PTE0 (V_{DD} = 5.5 Vdc)**
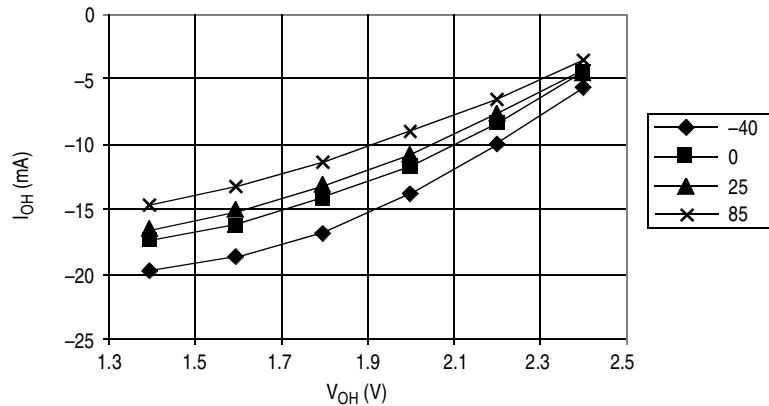
$V_{OH} > V_{DD} -0.3$ V @ $I_{OH} = -0.6$ mA
$V_{OH} > V_{DD} -1.0$ V @ $I_{OH} = -4.0$ mA

**Figure 23-6. Typical High-Side Driver Characteristics –
Ports PTB7–PTB0, PTC6–PTC5, PTD7–PTD0, and
PTE1–PTE0 (V_{DD} = 2.7 Vdc)**

23.14 Output Low-Voltage Characteristics
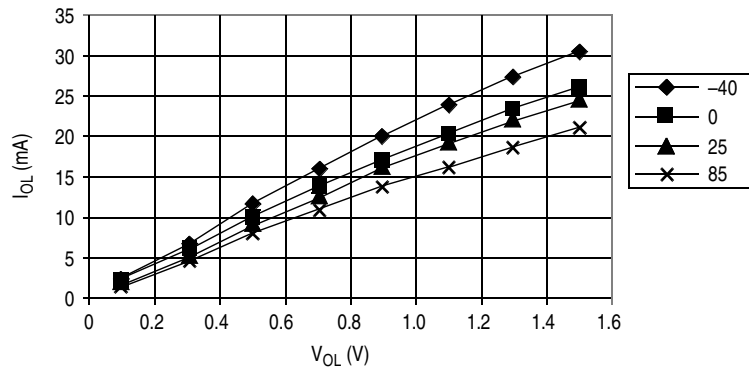
$V_{OL} < 0.4$ V @ $I_{OL} = 1.6$ mA
$V_{OL} < 1.5$ V @ $I_{OL} = 10.0$ mA

**Figure 23-7. Typical Low-Side Driver Characteristics –
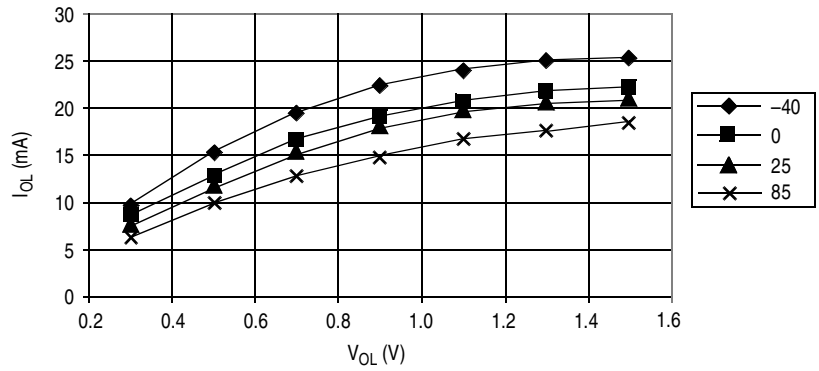Port PTA7–PTA0 (V_{DD} = 5.5 Vdc)**
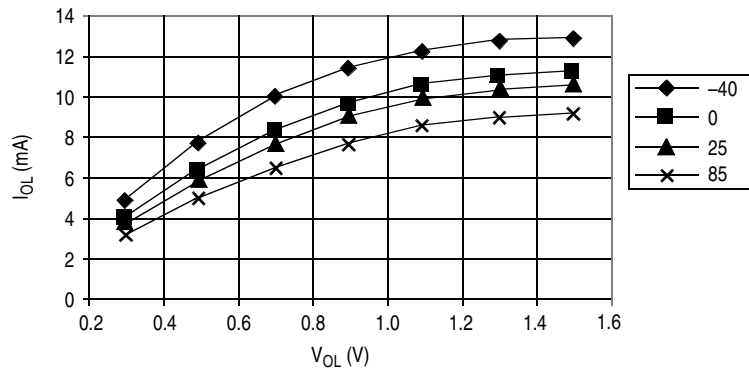
$V_{OL} < 0.3$ V @ $I_{OL} = 0.5$ mA
$V_{OL} < 1.0$ V @ $I_{OL} = 6.0$ mA

**Figure 23-8. Typical Low-Side Driver Characteristics –
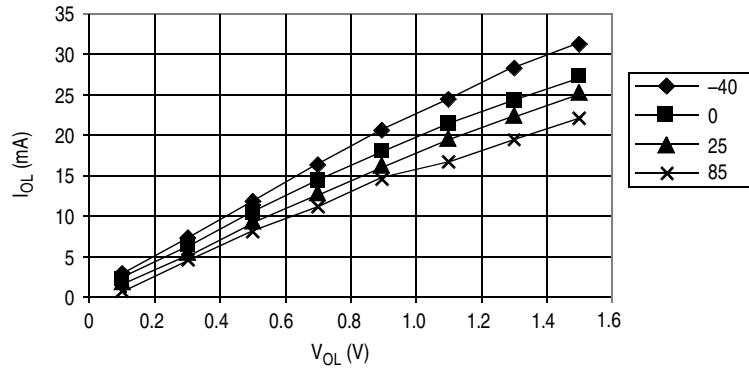Port PTA7–PTA0 (V_{DD} = 2.7 Vdc)**
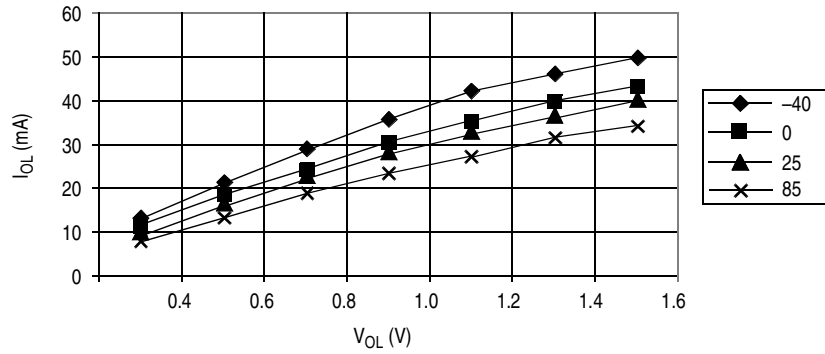
V_{OL} < 1.5 V @ I_{OL} = 20 mA

**Figure 23-9. Typical Low-Side Driver Characteristics –
Port PTC4–PTC0 (V_{DD} = 4.5 Vdc)**
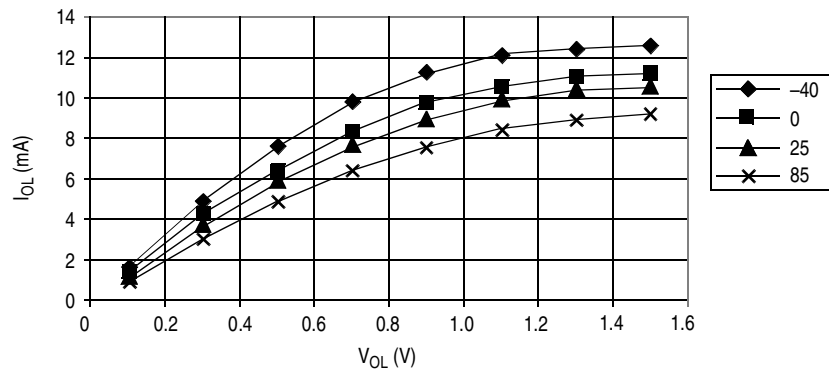

V_{OL} < 0.8 V @ I_{OL} = 10 mA

**Figure 23-10. Typical Low-Side Driver Characteristics –
Port PTC4–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
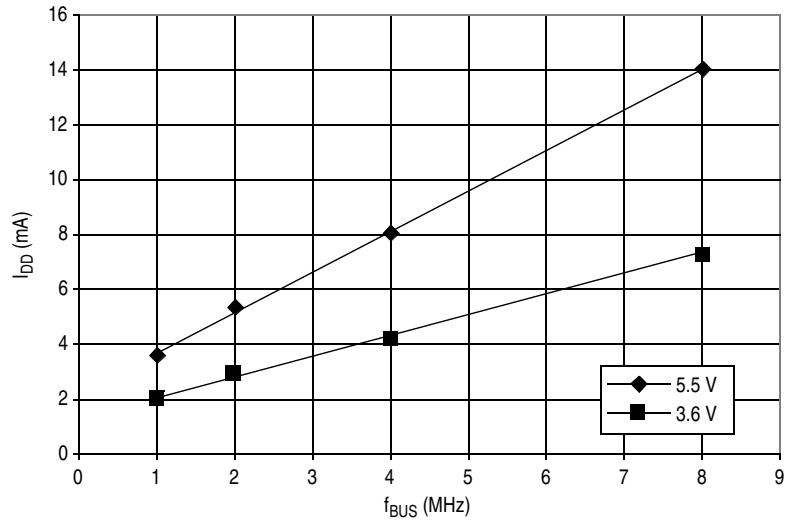
**Figure 23-11. Typical Low-Side Driver Characteristics –
Ports PTB7–PTB0, PTC6–PTC5, PTD7–PTD0, and
PTE1–PTE0 (V_{DD} = 5.5 Vdc)**

V_{OL} < 0.3 V @ I_{OL} = 0.5 mA
V_{OL} < 1.0 V @ I_{OL} = 6.0 mA

**Figure 23-12. Typical Low-Side Driver Characteristics –
Ports PTB7–PTB0, PTC6–PTC5, PTD7–PTD0, and
PTE1–PTE0 (V_{DD} = 2.7 Vdc)**

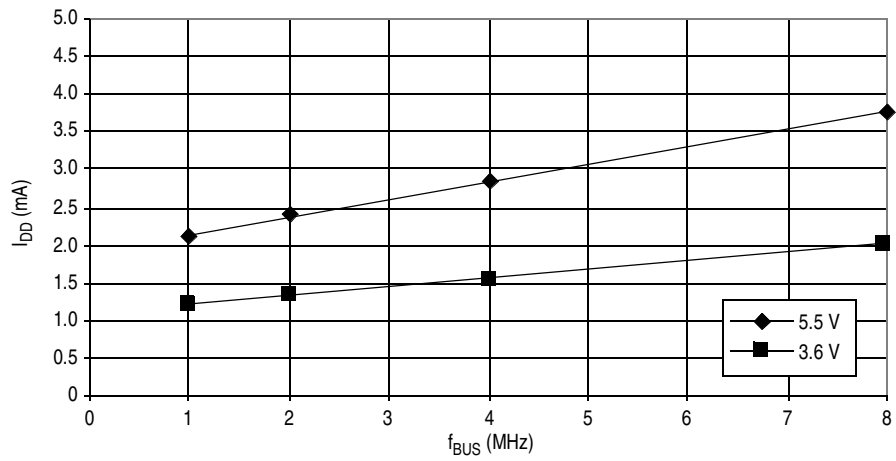## 23.15 Typical Supply Currents

**Figure 23-13. Typical Operating I$_{DD}$, with All Modules
Turned On (–40°C to 85°C)**

**Figure 23-14. Typical Wait Mode I$_{DD}$, with all Modules Disabled
(–40°C to 85°C)**

23.16 ADC Characteristics

Characteristic[1]	Symbol	Min	Max	Unit	Comments
Supply voltage	V_{DDA}	2.7 (V_{DD} min)	5.5 (V_{DD} max)	V	V_{DDA} should be tied to the same potential as V_{DD} via separate traces.
Input voltages	V_{ADIN}	0	V_{DDA}	V	
Resolution	B_{AD}	8	8	Bits	
Absolute accuracy (V_{REFL} = 0 V, V_{REFH} = V_{DDA} = 5 V $\pm$ 10%)	A_{AD}	—	$\pm$ 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	$V_{SSA} \le V_{ADIN} \le V_{DDA}$
Power-up time	t_{ADPU}	16		t_{AIC} cycles	
ADC voltage reference high	V_{REFH}	V_{SSA} - 0.1	V_{DDA} + 0.1	V	$V_{REFL} \le V_{REFH}$
ADC voltage reference low	V_{REFL}	V_{SSA} - 0.1	V_{DDA} + 0.1	V	$V_{REFL} \le V_{REFH}$
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}	—	8	pF	Not tested
Input leakage[4] Port B		—	—	$\pm$ 1	μA	

1. V_{DD} = 5.0 Vdc $\pm$ 10%, V_{SS} = 0 Vdc, V_{DDA} = 5.0 Vdc $\pm$ 10%, V_{SSA} = 0 Vdc, V_{REFH} = 5.0 Vdc $\pm$ 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.

23.17 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 23-15** and **Figure 23-16**.
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

23.18 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

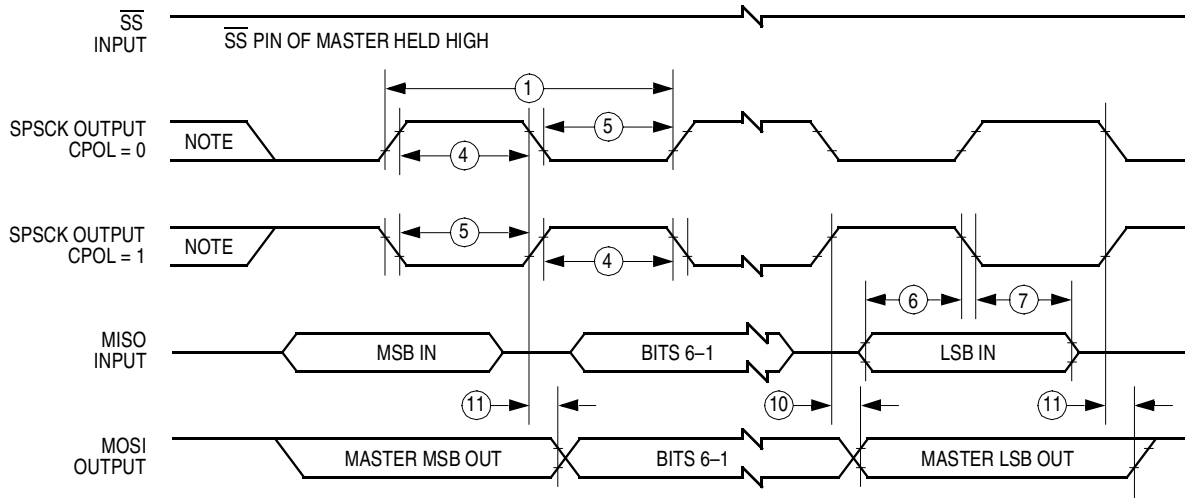1. Numbers refer to dimensions in **Figure 23-15** and **Figure 23-16.**
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless noted; 100 pF load on all SPI pins.
3. Time to data active from high-impedance state
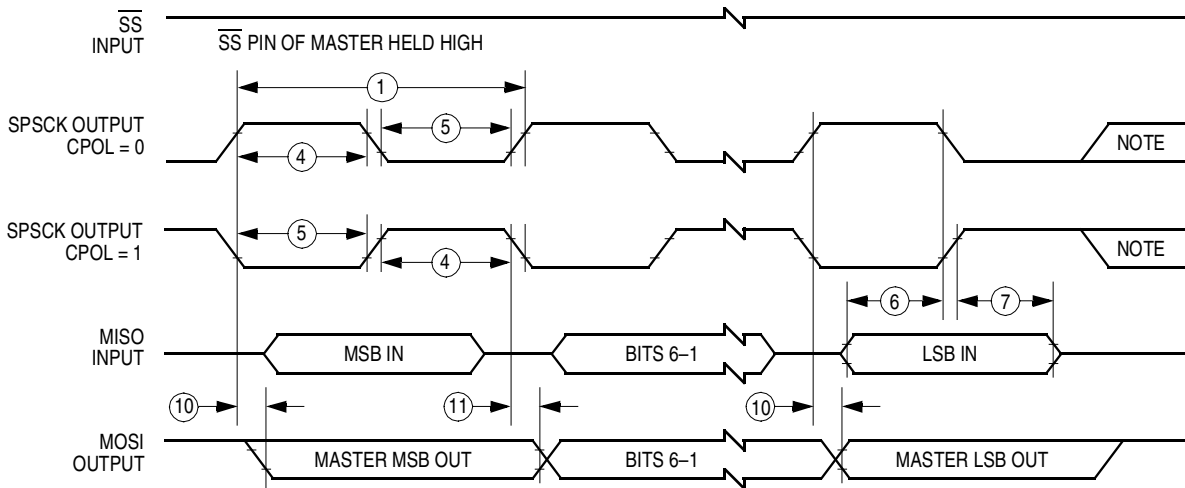4. Hold time to high-impedance state
5. With 100 pF on all SPI pins

Note: This first clock edge is generated internally, but is not seen at the SPSCK pin.

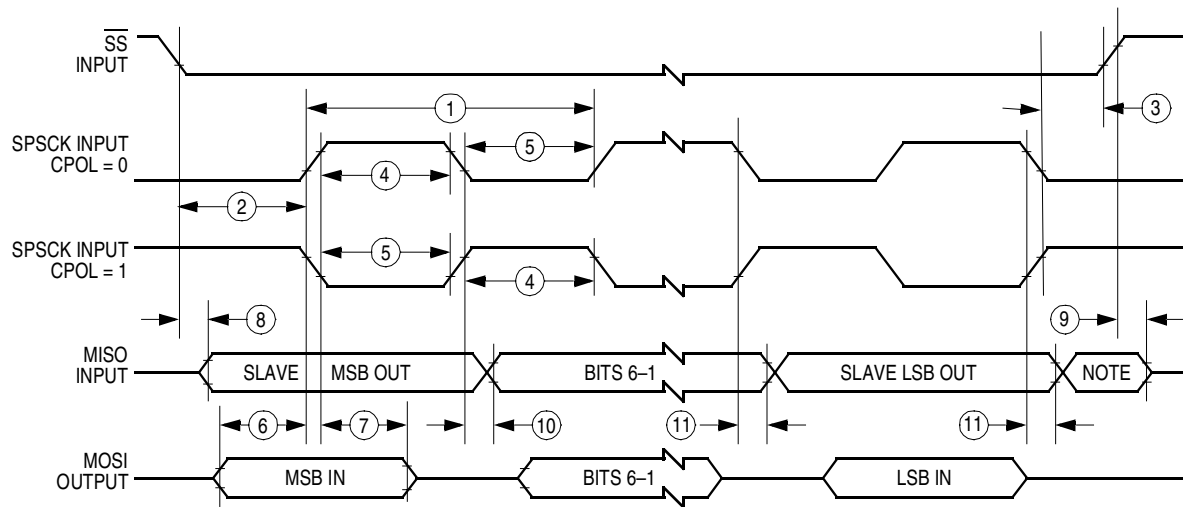**a) SPI Master Timing (CPHA = 0)**

Note: This last clock edge is generated internally, but is not seen at the SPSCK pin.
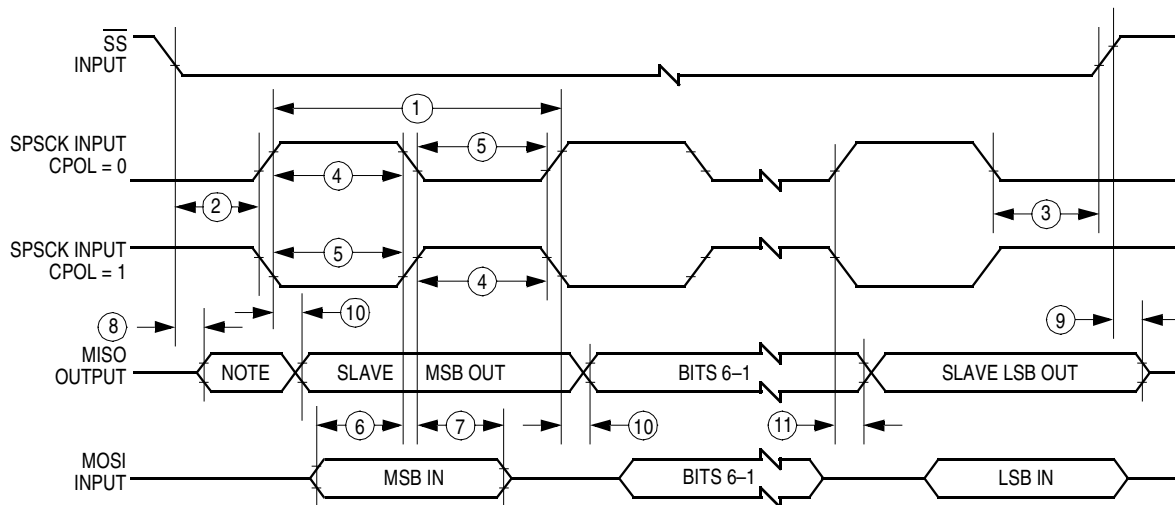
b) SPI Master Timing (CPHA = 1)

Figure 23-15. SPI Master Timing

Electrical Specifications

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 23-16. SPI Slave Timing

23.19 Timer Interface Module Characteristics

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	t_{TIH}, t_{TIL}	1	—	t_{CYC}

23.20 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]	8k	—	8M	Hz
FLASH page erase time	t_{Erase}[2]	1	—	—	ms
FLASH mass erase time	t_{MErase}[3]	4	—	—	ms
FLASH PGM/ERASE to HVEN set up 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}[4]	1	—	—	μs
FLASH cumulative program HV period	t_{HV}[5]	—	—	4	ms
FLASH row erase endurance[6]	—	10k	100k	—	Cycles
FLASH row program endurance[7]	—	10k	100k	—	Cycles
FLASH data retention time[8]	—	10	20	—	Years

1. f_{Read} is defined as the frequency range for which the FLASH memory can be read.
2. If the page erase time is longer than t_{Erase} (Min), there is no erase-disturb, but it reduces the endurance of the FLASH memory.
3. If the mass erase time is longer than t_{MErase} (Min), there is no erase-disturb, but it reduces the endurance of the FLASH memory.
4. 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 logic 0.
5. t_{HV} is defined as the cumulative high voltage programming time to the same row before next erase.
   t_{HV} must satisfy this condition: $t_{nvs} + t_{nvh} + t_{pgs} + (t_{PROG} \times 64) \leq t_{HV}$ max.
6. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase / program cycles.
7. The minimum row endurance value specifies each row of the FLASH memory is guaranteed to work for at least this many erase / program cycles.
8. The FLASH is guaranteed to retain data over the entire operating temperature range for at least the minimum time specified.

Section 24. Mechanical Specifications

## 24.1 Contents

24.2 Introduction

This section gives the dimensions for:

- 42-pin shrink dual in-line package (case 858-01)

- 44-pin plastic quad flat pack (case 824A-01)

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 Motorola sales office.

24.3 42-Pin Shrink Dual in-Line Package (SDIP)
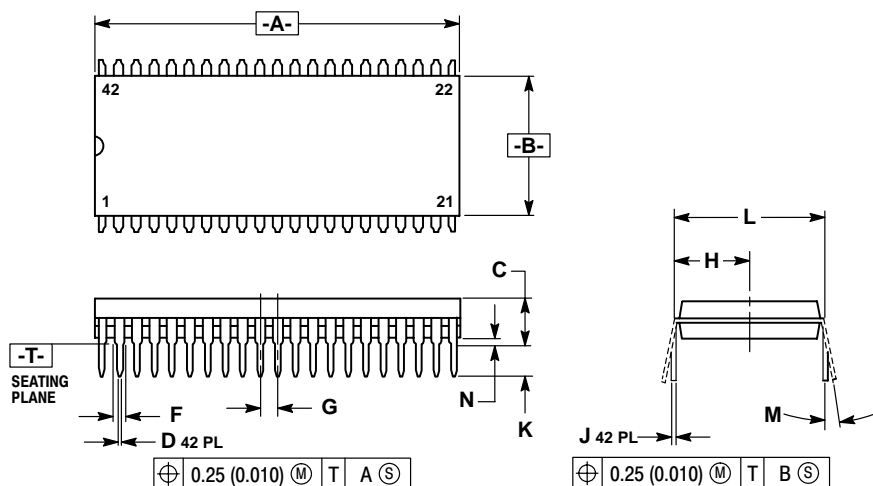
NOTES:
1. DIMENSIONS AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSIONS A AND B DO NOT INCLUDE MOLD FLASH. MAXIMUM MOLD FLASH 0.25 (0.010).

DIM	INCHES		MILLIMETERS	
	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	0.032	0.046	0.81	1.17
G	0.070 BSC		1.778 BSC	
H	0.300 BSC		7.62 BSC	
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

24.4 44-Pin Plastic Quad Flat Pack (QFP)

DETAIL A

-A-, -B-, -D-

SECTION B-B

BASE METAL
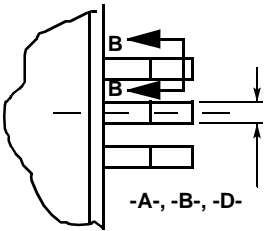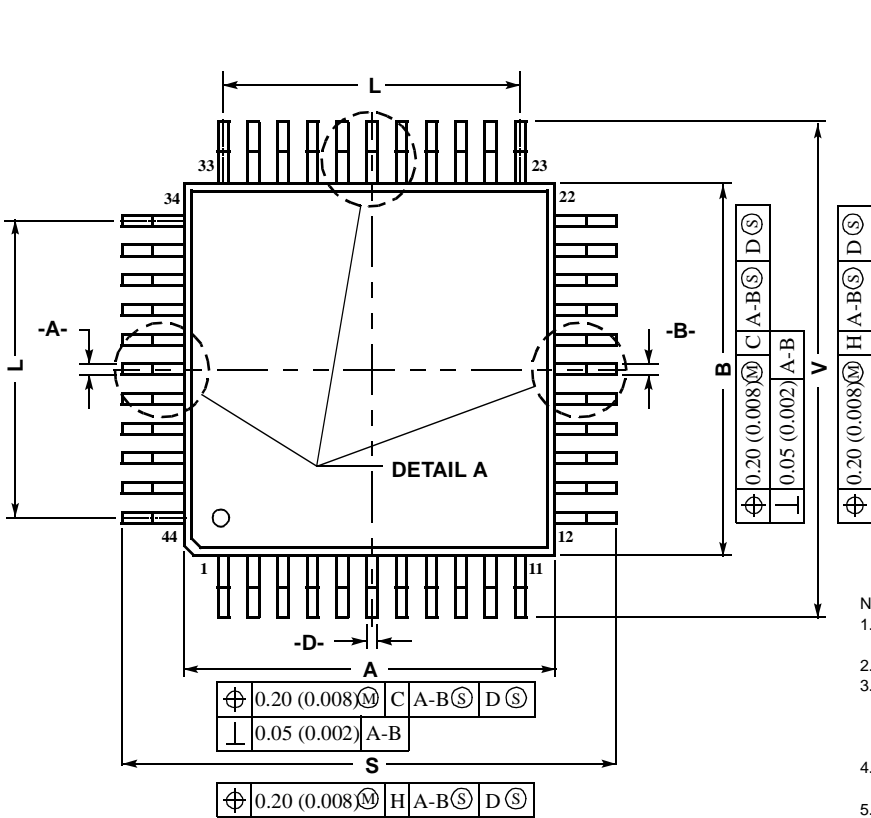
DETAIL C
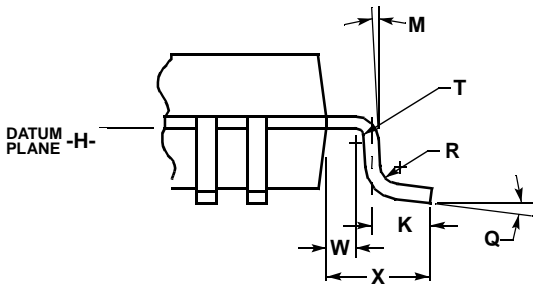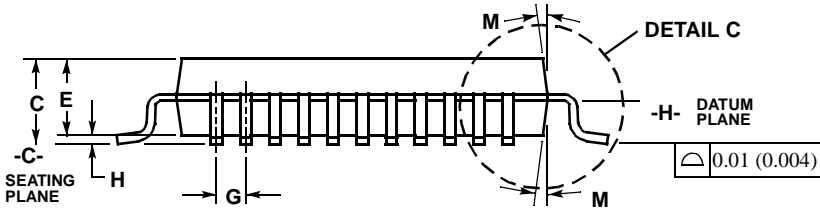
DETAIL C
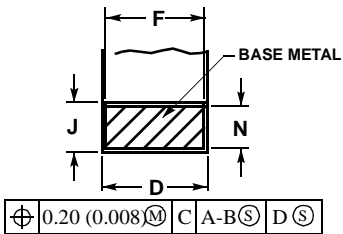
NOTES:
1. 1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. 2. CONTROLLING DIMENSION: MILLIMETER.
3. 3. DATUM PLANE -H- IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. 4. DATUMS -A-, -B- AND -D- TO BE DETERMINED AT DATUM PLANE -H-.
5. 5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE -C-.
6. 6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -H-.
7. 7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 (0.003) TOTAL IN EXCESS OF THE D DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	9.90	10.10	0.390	0.398
B	9.90	10.10	0.390	0.398
C	2.10	2.45	0.083	0.096
D	0.30	0.45	0.012	0.018
E	2.00	2.10	0.079	0.083
F	0.30	0.40	0.012	0.016
G	0.80 BSC		0.031 BSC	
H	---	0.25	---	0.010
J	0.13	0.23	0.005	0.009
K	0.65	0.95	0.026	0.037
L	8.00 REF		0.315 REF	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	12.95	13.45	0.510	0.530
T	0.13	---	0.005	---
U	0°	---	0°	---
V	12.95	13.45	0.510	0.530
W	0.40	---	0.016	---
X	1.6 REF		0.063 REF	

Mechanical Specifications

Section 25. Ordering Information

25.1 Contents

25.2 Introduction

This section contains ordering numbers for the MC68HC908GT16 and MC68HC908GT8.

25.3 MC Order Numbers

Table 25-1. MC Order Numbers

MC Order Number	Operating Temperature Range	Package
MC68HC908GT16CB	−40°C to +85°C	42-pin SDIP
MC68HC908GT16CFB	−40°C to +85°C	44-pin QFP
MC68HC908GT8CB	−40°C to +85°C	42-pin SDIP
MC68HC908GT8CFB	−40°C to +85°C	44-pin QFP

Ordering Information

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.;
Silicon Harbour Centre, 2 Dai King Street,
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://www.motorola.com/semiconductors

Information in this document is provided solely to enable system and software implementers to use Motorola 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.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 which may be provided in Motorola 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. Motorola does not convey any license under its patent rights nor the rights of others. Motorola 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 Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola 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 Motorola was negligent regarding the design or manufacture of the part.

MC68HC908GT16/D